Titre: Title:	Trustworthy Additive Explanations of Machine Learning Models through Increased Alignment
Auteur: Author:	Gabriel Laberge
Date:	2024
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Laberge, G. (2024). Trustworthy Additive Explanations of Machine Learning Models through Increased Alignment [Thèse de doctorat, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/59456/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/59456/
Directeurs de recherche: Advisors:	Foutse Khomh, & Mario Marchand
Programme: Program:	Génie informatique

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Trustworthy Additive Explanations of Machine Learning Models through Increased Alignment

GABRIEL LABERGE
Département de génie informatique et génie logiciel

Thèse présentée en vue de l'obtention du diplôme de *Philosophiæ Doctor*
Génie informatique

Septembre 2024

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Cette thèse intitulée :

Trustworthy Additive Explanations of Machine Learning Models through Increased Alignment

présentée par **Gabriel LABERGE**
en vue de l'obtention du diplôme de *Philosophiæ Doctor*
a été dûment acceptée par le jury d'examen constitué de :

Daniel ALOISE, président
Foutse KHOMH, membre et directeur de recherche
Mario MARCHAND, membre et codirecteur de recherche
Sarath Chandar ANBIL PARTHIPAN, membre
Giles HOOKER, membre externe

ACKNOWLEDGEMENTS

I wish to thank both of my supervisors, Foutse Khomh and Mario Marchand, for their constant support through-out these past four years. Both were always there to offer me guidance whenever I was lost. On the one hand, Prof Khomh knew how to push me in my critical thinking, especially when it came to considering the practical implications of my research. On the other hand, Prof. Marchand helped me on technical aspects of the mathematical theory developed during my PhD. Beyond their guidance, I am grateful that my supervisors had enough confidence to let me explore the literature at my own leisure. The eXplainable Artificial Intelligence (XAI) field still being in its infancy, it was crucial for me to take the time to ponder about the fundamental challenges faced by this field.

My sincere gratitude to the members of my jury : Daniel Aloise, Sarath Chandar, Giles Hooker, for thoroughly reviewing my work and acknowledging the important contributions within. I also thank the representant Soumaya Yacout.

In addition, I am extremely grateful to my colleague Yann Baptiste Pequignot, who was always available for a Zoom call or a casual discussion. Yann has an incredible skill: he immediately understands the issues his colleagues are facing and somehow always knows how to tackle them. As a result, he helped me overcome many of my roadblocks.

I also give my sincere thanks to Prof. Ulrich Aïvodji. We started collaborating early on during my PhD, and our joint paper was my first breakthrough. This first publication, which would not have been possible without his expertise, has launched me in an upward momentum that lasted until the end of my studies.

My honest thanks go to the members of the DEEL project and to its funding institutions, the Natural Sciences and Engineering Research Council of Canada (NSERC) and the Consortium for Research and Innovation in Aerospace in Québec (CRIAQ)

Finally, I am eternally obliged to my parents Christian Laberge, Pascale Sauvé, and my significant other Jennifer Vuong-Nguyen. Their unconditional love and support was essential in keeping me happy and motivated during this long process. The support of my parents was especially important during the first year of my PhD where, due to COVID-19 lockdowns, I was spending all days on my personal computer set up over the kitchen table. My loved one and now gym partner, Jennifer, has helped me find a work-life balance and discover a passion for physical exercise.

RÉSUMÉ

Nous vivons présentement une révolution numérique. Grâce aux avancées technologiques liées à la puissance de calcul et l'emmagasinage de données, il est désormais possible d'automatiser de nombreuses tâches. Exploiter le plein potentiel des données a demandé un changement de paradigme vers l'Apprentissage Automatique (AA). Concrètement, plutôt que de manuellement encoder la logique d'un programme, on laisse un *Algorithme d'Apprentissage* déterminer le bon programme (ou prédicteur) à partir des données collectées. La forte dépendance envers les données apporte de nouveaux risques. Par exemple, si les données sont biaisées ou non représentatives, le prédicteur appris exhibera ces biais. Puisque la plupart des prédicteurs fournis par des algorithmes d'apprentissages sont des *boites noires*, il est difficile de déterminer si les modèles ont appris les bons patrons.

Le champ de l'Intelligence Artificielle eXpliquable (XAI) a développé des mécanismes pour "expliquer" les comportements des boites noires. La promesse de ces techniques est qu'elles pourront aider un praticien à déterminer si le modèle a appris les bons patrons et qu'il ne contient pas de biais indésirables. Alors que le XAI a maturé au point de fournir une variété de techniques aux développeurs, la difficulté d'évaluer ces méthodes est devenue apparente. En effet, puisque le prédicteur est une boite-noire, il n'y a pas de *réponse de référence* pour l'explication de ces décisions. Sans *réponse de référence*, un praticien n'a pas de manière systématique de décider quelle explication est la bonne quand diverses techniques se contredisent. Pour contrer cette limitation, nombreux chercheurs développent des *métriques de qualités* sur les explications. À mon avis, cette direction de recherche n'a pas encore porté fruit, car le choix des bonnes métriques de qualités n'a pas encore été déterminé. De plus, il a été démontré expérimentalement que les métriques existantes sont inconsistantes : le choix de la meilleure explication dépend grandement de la métrique choisie.

Nous proposons une solution alternative au manque de *réponse de référence* en explicabilité. Notamment, nous proposons *d'aligner les méthodes plutôt que de les comparer*. Plus précisément, nous démontrons que toutes les techniques d'explicabilité peuvent être exprimées dans un cadre théorique uniforme. À travers ce cadre théorique, la raison des désaccords devient apparente. De plus, la théorie prédit que, quand toutes les techniques s'accordent, alors elles coïncident avec une explication sensée (celle d'un modèle additif). Ceci prouve qu'aligner les techniques d'explications est une méthodologie viable pour obtenir une *réponse de référence*. Il est finalement démontré théoriquement et empiriquement comment réduire les désaccords entre les explications, permettant d'obtenir des explications fiables.

ABSTRACT

We are currently living a numerical revolution. Given modern advancements in the computing power and data storage capabilities of computers, it is now possible to automate a variety of tasks. Exploiting the full potential of large data has required a change of paradigm to Machine Learning (ML). Concretely, instead of manually encoding the logic of a program, we let a *Learning Algorithm* decide which program (or predictor) best fits the collected data. The strong reliance on data introduces new risks. For instance, if the data is biased or non-representative, the learned predictor will also exhibit those biases. Since most predictors yielded by learning algorithms are *black-boxes*, it is hard to determine whether the model has learned the correct patterns.

The field of eXplainable Artificial Intelligence (XAI) has developed mechanisms to "explain" the behavior of black boxes. The promises of these techniques is that they will help practitioners determine whether the model has learned the correct patterns and does not exhibit undesirable biases. As XAI has matured to the point of providing a myriad of open-source tools to developers, the difficulty in comparing the various methods has become apparent. Indeed, since the model is a black-box, there is no *ground-truth* for the explanation of its decisions. Without *ground-truths*, a practitioner cannot systematically decide which explanation is correct whenever the different techniques contradict each other. To tackle this limitation, researchers have developed *quality metrics* for explanations. In my opinion, this research direction has not yet been fruitful because the choice of optimal metrics has not been determined. Even more, it was demonstrated experimentally that existing metrics are inconsistent : the choice of the best explanation depends on the metric considered.

We propose an alternative solution to the lack of *ground-truth* in explainability. Notably, we propose to *align the methods instead of comparing them.* More precisely, we demonstrate that all explainability techniques can be expressed in a unified theoretical framework. Through this framework, the root cause of disagreements becomes apparent. Moreover, the theory predicts that, when all techniques agree, they coincide with a sensible explanation (that of an additive model). This proves that aligning explanations techniques is a viable methodology for obtaining *ground-truth* explanations. We finally demonstrate theoretically and empirically how to reduce disagreements between explanations, allowing for more trustworthy insights on model behavior.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ACRONYMS

Notation	Definition
General	
S	A set.
$-S, \overline{S}$	The complement of set S.
$\mathbb{R}$	The set of real numbers.
$\mathbb{R}^d$	The set of d-dimensional vectors.
$\mathbb{N}$	The set of natural numbers.
$[d]$	The set $\{1, 2, \ldots, d\}$ for $d \in \mathbb{N}$.
$2^{[d]}$	The power-set of $[d]$.
$\boldsymbol{x}, \boldsymbol{z}$	Vectors in $\mathbb{R}^d$.
x_j, z_j	jth component of the vector.
$\boldsymbol{A}, \boldsymbol{H}$	Matrices.
Machine Learning	
$\mathcal{X} \subseteq \mathbb{R}^d$	Input domain (a set).
$\mathcal{Y}$	Output domain (a set) .
$\mathcal{H}$	Hypothesis class of functions $h : \mathcal{X} \to \mathcal{Y}'$.
$\ell : \mathcal{Y}' \times \mathcal{Y} \to \mathbb{R}_+$	Loss function.
$\mathcal{D}, \mathcal{B}, \mathcal{F}$	Probability distributions over $\mathcal{X}$ or $\mathcal{X} \times \mathcal{Y}$.
$\mathcal{D}(A), \mathcal{B}(A), \mathcal{F}(A)$	Probability of a measurable set A.
$(\boldsymbol{x}, y) \sim \mathcal{D}, \boldsymbol{z} \sim \mathcal{B}$	Sampling according to the distribution.
$\mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[f(\boldsymbol{z})]$	Expectation of $f(\boldsymbol{z})$ for $\boldsymbol{z} \sim \mathcal{B}$.
$\mathbb{V}_{\boldsymbol{z}\sim\mathcal{B}}[f(\boldsymbol{z})]$	Variance of $f(\boldsymbol{z})$ for $\boldsymbol{z} \sim \mathcal{B}$.
$S = \{(\boldsymbol{x}^{(i)}, y^{(i)})\}_{i=1}^N$	Dataset.
$\mathcal{L}_{\mathcal{D}}(h)$	Population loss $\mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}}[\,\ell(h(\boldsymbol{x}), y)]$.
$\widehat{\mathcal{L}}_S(h)$	Empirical loss $1/N \sum_{i=1}^N \ell(h(\boldsymbol{x}^{(i)}), y^{(i)})$.
$h^\star \in \operatorname{argmin}_{h\in\mathcal{H}} \mathcal{L}_{\mathcal{D}}(h)$	Best in-class model.
$h_S \in \operatorname{argmin}_{h\in\mathcal{H}} \widehat{\mathcal{L}}_S(h) + \lambda \times \mathrm{Reg}(h)$	Empirical loss minimizer.
$\mathcal{H} = \{h_{\boldsymbol{\theta}} : \boldsymbol{\theta} \in \Theta\}$	Parametric Hypothesis class.
$h^{\mathrm{lin}}_{\boldsymbol{\omega}}(\boldsymbol{x}) = \omega_0 + \sum_{j=1}^d \omega_j x_j$	Linear Models.

Notation	Definition
Additive Explanations	
$\phi : \mathcal{H} \times \mathcal{X} \times \mathcal{P}(\mathcal{X}) \to \mathbb{R}^d$	Local Feature Attribution (LFA).
$\Phi : \mathcal{H} \times \mathcal{P}(\mathcal{X}) \to \mathbb{R}^d_+$	Global Feature Importance (GFI).
$h^{\text{add}}(\boldsymbol{x}) := \omega_0 + \sum_{j=1}^d h_j(x_j)$	An additive model.
$G(h, \boldsymbol{x}, \mathcal{B}) := h(\boldsymbol{x}) - \mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{z})]$	Prediction Gap at $\boldsymbol{x}$ relative to $\mathcal{B}$.
$\pi : [d] \to [d]$	Permutation of $[d]$.
$\pi_{:j}$	Features appearing before j in π.
$\nu : 2^{[d]} \to \mathbb{R}$	Coalitional game.
$\nu^{\text{int}}_{h,\boldsymbol{x},\mathcal{B}}(S) := \mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_S, \boldsymbol{z}_{-S})]$	Interventional Game.
$\nu^{\text{obs}}_{h,\boldsymbol{x},\mathcal{B}}(S) := \mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{z})\|\boldsymbol{z}_S = \boldsymbol{x}_S]$	Observational Game.
Functional Decomposition	
$\mathcal{X} = \prod_{j=1}^d \mathcal{X}_j$	Generalized input space.
$\boldsymbol{x}_u = (x_j)_{j\in u}$	$\boldsymbol{x}$ components in the set $u \subseteq [d]$.
$\boldsymbol{r}^{\boldsymbol{z}}_u : \mathcal{X} \to \mathcal{X}$	Replace-function.
$h(\boldsymbol{r}^{\boldsymbol{z}}_u(\boldsymbol{x})) \equiv h(\boldsymbol{x}_u, \boldsymbol{z}_{-u})$	Model called on $\boldsymbol{x}_u$ and $\boldsymbol{z}_{-u}$.
$\mathcal{B}_{\text{ind}} := \prod_{j=1}^d \mathcal{B}_j$	Distribution with independent features.
$h(\boldsymbol{x}) = \sum_{u\subseteq[d]} h_u(\boldsymbol{x})$	Functional decomposition.
$h_{u,\boldsymbol{z}}$	$\boldsymbol{z}$-Anchored Decompositions.
$h_{u,\mathcal{B}}$	$\mathcal{B}$-Interventional Decompositions.
$h_{u,\mathcal{B}_{\text{ind}}}$	ANOVA Decompositions.
$\sigma^2_u := \mathbb{E}_{\boldsymbol{x}\sim\mathcal{B}_{\text{ind}}}[h_{u,\mathcal{B}_{\text{ind}}}(\boldsymbol{x})^2]$	Variance of ANOVA component.
$\Omega = \prod_{j=1}^d \Omega_j$ where $\Omega_j \subseteq \mathcal{X}_j$	Region of $\mathcal{X}$.
$\mathcal{B}_\Omega$	Restriction of $\mathcal{B}$ to the region Ω.
Estimates	
$\boldsymbol{H}^u$	$N \times M$ matrix $H^u_{ij} := h_{u,\boldsymbol{z}^{(j)}}(\boldsymbol{x}^{(i)})$.
$\boldsymbol{\Phi}^k$	$N \times M$ matrix $\Phi^k_{ij} := \phi^{\text{SHAP-int}}_k(h, \boldsymbol{x}^{(i)}, \boldsymbol{z}^{(j)})$.
$\boldsymbol{\xi} : \prod_{j=1}^d \mathcal{X}_j \to \mathbb{R}^{d'}$	Feature Embedding
$h = h^{\text{ML}} \circ \boldsymbol{\xi}$	Composing an embedding with a ML model.
$\mathcal{I}_{\boldsymbol{\xi}} : [d'] \to [d]$	Mapping $j \in [d']$ to its $\boldsymbol{x}$ component.
$\mathcal{I}^{-1}_{\boldsymbol{\xi}} : 2^{[d]} \to 2^{[d']}$	The preimage of $\mathcal{I}_{\boldsymbol{\xi}}$.

Notation	Definition
Interaction Disagreement	
$L_h(\mathcal{B}) \in \mathbb{R}_+$	Lack of Additivity (LoA) of h w.r.t $\mathcal{B}$.
$D(\boldsymbol{\phi}, \boldsymbol{\phi}')$	LFA Distance $\mathbb{E}_{\boldsymbol{x}\sim\mathcal{B}}[\|\boldsymbol{\phi}(h, \boldsymbol{x}, \mathcal{B}) - \boldsymbol{\phi}'(h, \boldsymbol{x}, \mathcal{B})\|_2^2]$.
$D(\boldsymbol{\Phi}, \boldsymbol{\Phi}')$	GFI Distance $\|\boldsymbol{\Phi}(h, \mathcal{B}) - \boldsymbol{\Phi}'(h, \mathcal{B})\|_2^2$.
Subsampling Disagreement	
$x_s \in \{0, 1\}$	Sensitive Feature (*e.g.* Religion, Race).
$D_j \ = \{\boldsymbol{x}^{(i)} : x_s^{(i)} = j\}$	Demographic subgroups.
$S_j \subset D_j$	Subsample of Demographic subgroups.
$S'_j \subset D_j$	Cherry-picked subsamples.
$h(D_j), h(S_j)$	Image of h *i.e.* $h(S) := \{h(\boldsymbol{x}^{(i)}) : \boldsymbol{x}^{(i)} \in S\}$.
$\boldsymbol{\Phi}^{\text{Fair}}(h, \mathcal{F}, \mathcal{B})$	Fair-SHAP feature attributions.
$\widehat{\boldsymbol{\Phi}}^{\text{Fair}}(h, S_0, S_1)$	Fair-SHAP estimate.
$\mathcal{B}_{\boldsymbol{\omega}}$	Weighted background.
$\mathcal{W}(h(\mathcal{B}), h(\mathcal{B}_{\boldsymbol{\omega}}))$	Wassertein distance.
Underspecifiation Disagreement	
$\mathcal{R}(\mathcal{H}, \epsilon)$	Rashomon Set.
$i \preceq_{\epsilon,\boldsymbol{x},\mathcal{B}} j$	Consensus Order Relation on LFA.
$i \preceq_{\epsilon,\mathcal{B}} j$	Consensus Order Relation on GFI.
$\mid \preceq_{\epsilon,\boldsymbol{x},\mathcal{B}} \mid$	Cardinality of the Local Partial Order.

LIST OF APPENDICES

CHAPTER 1 INTRODUCTION

1.1 Context

In July 1969, the National Aeronautics and Space Administration (NASA) supervised the Apollo 11 mission, whose objective was to land the first astronauts on the moon. The historical success of this mission has been the result of technological innovation and collaborations between various scientific domains : Mechanical Engineering, Electrical Engineering, and what became to be known as Software Engineering. The Software branch of the project, handled by scientists at the Massachusetts Institute of Technology (MIT), created an interface between the astronauts and the spacecraft, allowing astronauts to send explicit control commands. The source code (seen on the right Figure[1]) was written in the assembly language, meaning that each individual action performed by the computer was *explicitely* written by the developers. In which register to store the variable encoding acceleration? What quantity of gas to insert in the engine igniter? When to ignite the engine? These are all behaviors that had to be written into assembly instructions.

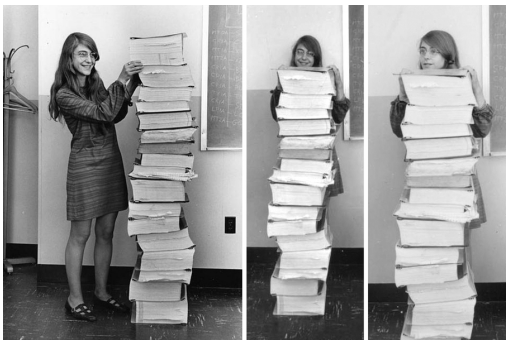


Figure 1.1 Margaret Hamilton, leader of the Software Engineering Division of the MIT, with the written software developed for the Apollo mission.

The development of these instructions was a systematic process. The first step of this process was to define clear *requirements* of the intended spacecraft behavior, see Figure 1.2 (Left). In the case of the landing procedure, the requirements were a predefined landing trajectory, divided into three phases (P63, P64, and P66) with clear checkpoints at given altitudes and times. Given a target trajectory, sequences of high-level instructions (called *algorithms*) were derived to ensure that the actual spacecraft trajectory would match the target, see Figure 1.2 (Middle). As an example, the ignition algorithm in phase P63 of the landing was derived using Linear Algebra to handle changes of coordinates, and Dynamics to predict how engine ignition would affect speed, acceleration, tork, etc. The final step of the development process is to *implement* the algorithms into assembly instructions that can be understood and executed by the computer (Figure 1.2 (Right)).

[1] `https://news.mit.edu/2016/scene-at-mit-margaret-hamilton-apollo-code-0817`

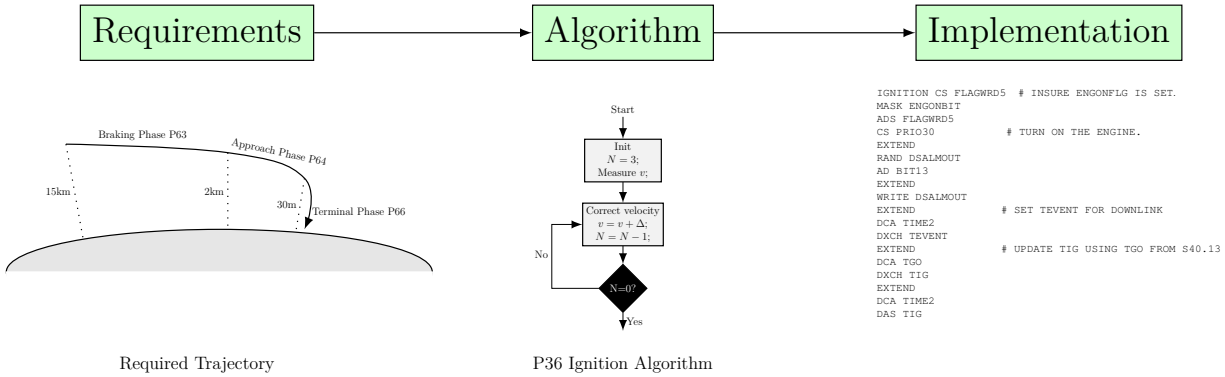


Figure 1.2 Traditional Programming Paradigm. First, clear requirements are stated. Second, an algorithm is developed to meet the requirements. Such algorithms are usually derived based on mathematics and physics. Third, the derived algorithm is implemented using a programming language. These schematics are inspired by the NASA report [Klumpp, 1971] and are by no means exact recreations.

The methodology of going from clear requirements to algorithms to machine instructions is called the *traditional programming paradigm* and it results in a *traditional program*. Today, 55 years after the Apollo 11 mission, traditional programs are successfully applied in various fields where algorithms can be derived from requirements using mathematics and physics. For instance, Ray Tracing algorithms derived from linear algebra have led to increased visual fidelity in modern video games, and formalizing the task of Web Page Ranking as an eigenvalue problem has enabled the most powerful search engine : Google. Additionally, post-Apollo 11, the Aerospace field has seen many software innovations including the plane autopilot and material stress simulations. Both innovations have increased flight safety and were only possible because the physics underlying plane flight are well understood.

Traditional programs struggle, however, whenever a task's requirements are not easily translated into mathematical equations. For example, facial recognition systems involve abstract *concepts* (a person's nose, eyes, hair) that are hard to formalize theoretically but feel very intuitive to humans. Other cases are email spam detectors, which programmers cannot be expected to develop by enumerating all possible dubious characters typically present in spam. In an attempt to solve such tasks, a change of paradigm has been observed in recent years: Machine Learning (ML). The central idea behind the ML paradigm is to do away with explicit requirements, and instead collect observational data that represents various operating scenarios $\boldsymbol{x}$ and the expected system response y, see Figure 1.3 (Left). For face recognition systems, the data could regroup thousands of pictures from individuals tagged with their name. Spam detectors would require collections of email texts along with a label (Spam or Clean).

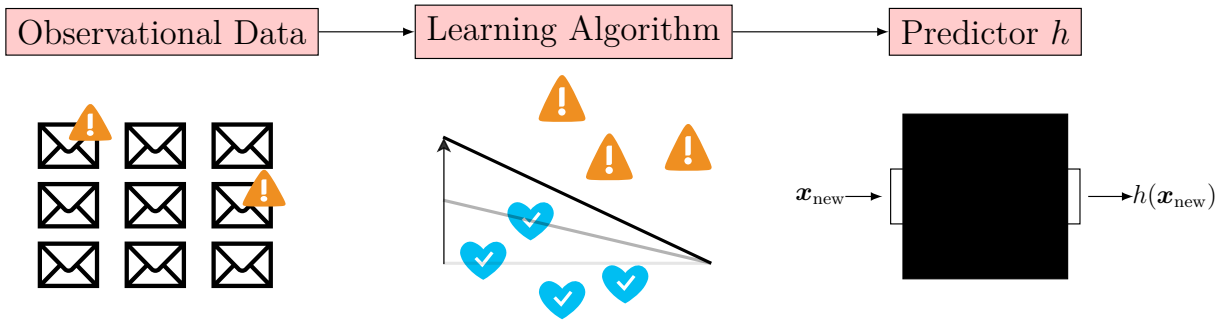


Figure 1.3 Machine Learning Paradigm. First, rather than specifying a list of clear requirements for the task, a dataset storing operating scenarios $\boldsymbol{x}$ (emails) and expected behavior y (spam/clean) is collected. Second, in place of deriving each individual step of the program, a learning algorithm selects the program (predictor) h whose predictions $h(\boldsymbol{x})$ are closest to y, on average. Third, instead of resulting in a series of simple computer instructions, one ends up with a predictor h that can operate in new conditions $\boldsymbol{x}_{\text{new}}$, but whose inner mechanisms are opaque (*i.e.* h is a black-box).

Now, given a dataset that represents the task, ML relies on a *learning algorithm* to decide which program h is best by minimizing the error between the expected y and actual $h(\boldsymbol{x})$ system outputs, see Figure 1.3 (Middle). This stands in contrast to the traditional paradigm, where each step of the program is derived formally from the requirements. At the end of the ML procedure, instead of having a list of instructions implementing the algorithm, one ends up with a predictive model h that returns predictions $h(\boldsymbol{x}_{\text{new}})$ on new operating conditions $\boldsymbol{x}_{\text{new}}$, see Figure 1.3 (Right). We intentionally illustrate h using a *black-box* because the most performant Machine Learning models available today (*e.g.* Tree Ensembles and Deep Neural Networks) are too complex to be interpreted by directly inspecting their code. Such models can only be understood in terms of their input-output relationships.

The ML paradigm has risen in popularity over the past decade, seeing as it has allowed programmers to solve complex tasks that were previously intractable. Facial Recognition systems, which are now present in your cellphone, were made possible by training Deep Neural Networks on immense image datasets. More recently, Large Language Models (LLMs) compress the integrality of the text available on the internet into an abstract knowledge-base that can be queried with natural language. Public interfaces of these models, such as ChatGPT, have allowed the public to reap the benefits of this technology.

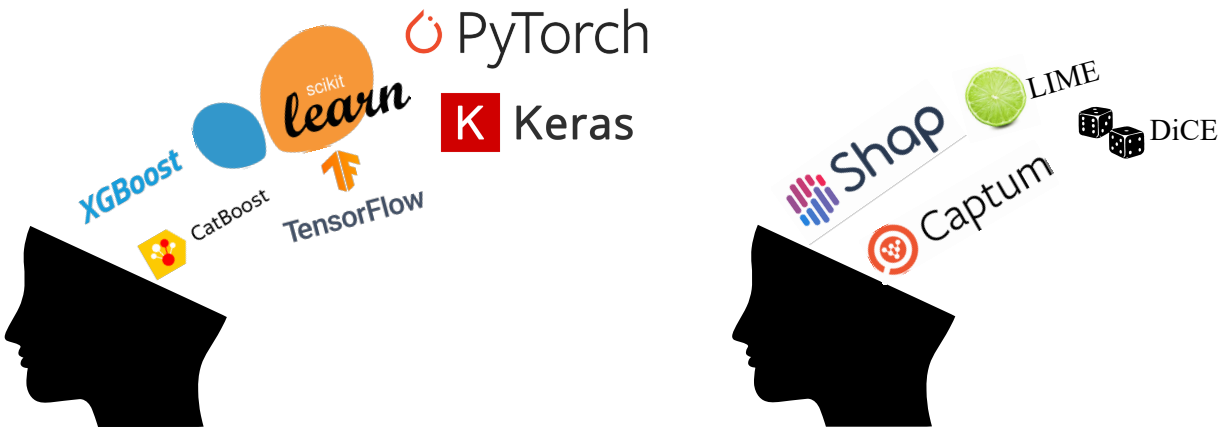
The motto of ML could be stated as follows: : *when we do not know how to specify the program to solve task <T>, we let data and learning algorithms specify the program.* The underlying assumption is that the data alone can specify the program intended by the developer. That is, by collecting sufficiently many samples (*e.g.* emails with/without spam or human face

pictures with tagged names), the learned predictor h should eventually exhibit the intended behavior (accurately tag spam emails or identify faces in deployment). However, there have been famous instances where the data and learning algorithms led to a model that did not have the intended behavior.

Notably, ML models have been shown to accurately predict the death of patients entering hospitals with pneumonia. Still, these models were deemed too risky because they attributed lower risks to asthmatic patients, while asthma is a known risk factor of pneumonia complications [Caruana et al., 2015]. The root cause of this disconnect is that, in the collected data, asthmatic patients were treated more aggressively so their chances of dying were lower than average. In this setting, learning from observational data, no matter how plentiful, will systematically lead to the incorrect predictor. Caruana et al. [2015] could identify the disconnect between h and the desired predictor because h was transparent *i.e.* its predictions could be explained. Nevertheless, the most modern and performant ML models are *black-boxes* and can only be understood through their input-output relationship (Figure 1.3 (Right)). This begs the fundamental question.

How can we verify that the data and learning algorithm returned the intended predictor h when h is a black-box that cannot be interpreted?

Answering this fundamental question is at the heart of the eXplainable Artificial Intelligence (XAI) research field, whose objective is to develop mechanisms to "explain" black-box predictions in order to assert whether they are "right for the right reasons". Efforts to make ML models more explainable date back to before the 21th century, but the terminology XAI has become very popular post-2017, when the Defense Advanced Research Projects Agency (DARPA) initiated its XAI program. Today, XAI is a mature field with many related publications and workshops in top ML conferences. The field has also developed a myriad of techniques that are now easily accessible to the average ML developer.



(a) (b)

Figure 1.4 (a) In ML research, there was an emphasis on developing as many techniques and frameworks as possible and finally comparing on standardized benchmarks. (b) XAI has also seen the development of a variety of techniques and frameworks but, unlike ML, benchmarking efforts have not been fruitful. Since it is hard to define a *ground-truth* for explainability techniques, practitioners are left wondering which method yields the *correct* answer on their use-case.

1.2 Problem Statement

A lot of XAI techniques have been proposed, which is reminiscent of ML research's focus on developing as many tools as possible and then benchmarking them (Figure 1.4). Importantly, the various XAI methods were recently shown to provide different (and even contradictory) conclusions on model behavior [Krishna et al., 2022]. The observation of contradictions between explanations is referred to as the *Disagreement Problem* (DP). While the DP is to be expected since different XAI techniques characterize models differently, practitioners cannot be expected to make informed decisions when presented with contradictory claims on the behavior of their model. What if an explanation technique claims that the model is racially biased, while another claims it is not? Should the model be deployed then? Krishna et al. [2022] surveyed 25 data scientists who use explainability techniques on a day-to-day basis. The data scientists collectively stated that they did not know how to handle disagreements between explanations, and they relied on heuristics (*e.g.* , choosing a favorite method or selecting whichever explanation best matched their intuition). Selecting explanations that way induces risks of *confirmation bias*, where humans think they understand the model because it matches their internal model of the world. However, we argue that trusting or mistrusting an explanation, and by extend a model, should be based on the explanation's *correctness*.

Main Research Question : How can the correctness of conflicting post-hoc additive explanations be determined?

Answering this question is not trivial because, unlike ML whose goal is to predict y given $\boldsymbol{x}$, there is no established *ground-truth* in XAI. Thus, benchmarking (the bread and butter of ML practitioners) is hardly applicable as a mean to decide which explanation is the best. The XAI has tackled the lack of ground-truth from two angles.

First, methods like Shapley Values [Lundberg and Lee, 2017] and the Integrated Gradient [Sundararajan et al., 2017] are motivated as being the *unique* explanations satisfying a set of theoretical properties. As such, they are advertised as a form of ground-truth. Still, in the case of Shapley Values, it was demonstrated that their "Dummy" property can be violated in practice [Sundararajan and Najmi, 2020]. Regarding the Integrated Gradient, its properties were proven to be insufficient at specifying a truly unique explanation [Lerma and Lucas, 2021].

Second, some works are attempting to benchmark explainability methods using *faithfulness metrics* : for example the Insertion/Deletion [Jethani et al., 2021, Petsiuk et al., 2018], μ-Fidelity [Bhatt et al., 2020], and PGU/PGI[Dai et al., 2022]. Unfortunately, faithfulness metrics were shown to be inconsistent : an explanation can be ranked first by a metric and ranked last by another [Tomsett et al., 2020]. Faithfulness rankings were also shown to be sensitive to innocuous choices. For example, different means of "shutting down" a pixel (replacing it with zero, the data mean, white noise) were shown to completely shift the rankings of explanation faithfulness [Tomsett et al., 2020]. In light of these observations, it appears that faithfulness metrics do not measure the same notion of faithfulness, so they cannot be used to select a unique correct explanation whenever there are disagreements.

In this thesis, I propose an orthogonal direction to address the lack of universal ground-truth in XAI. **Rather than following past trends of ML research (*i.e.* developing a variety of techniques and comparing them on benchmarks), I suggest increasing the alignment between competing explainability techniques.**

> **Thesis Statement :** Rather than comparing/benchmarking competing explainability methods, we should increase their alignment/agreement while ensuring that methods collectively converge toward a ground-truth explanation. We propose to use the built-in explanations of an Additive Model as the ground-truth explanation attained in the limit.

Using this Thesis Statement as our starting point, this manuscript will answer the main research question by tackling two sub-questions. The first is

---

Question I : What are the root causes of disagreements between explainability methods?

This question is tackled first because solving a problem requires a fundamental understanding of its source. To answer Question I, we have unified explainability methods through the theory of Functional Decomposition. More specifically, all explainability methods have been expressed in a manner that highlights what they actually measure about the predictive model, and more importantly, *why they disagree*. Notably, so-called *feature interactions* are identified as the culprit that prohibits agreement among the techniques.

Finally, it is demonstrated that, when all explainability methods agree, they coincide with the built-in explanations of an additive model (our ground truth). This suggests that, the goal of increasing unanimity among explanation techniques is aligned with the goal of attaining a single ideal explanation. The second sub-question investigated is:

Question II : How can we increase alignment between contradicting explanations?

To address this question, we define three notions of explanation disagreements 1) disagreements between explainability methods induced by feature interactions (*Interaction Disagreements*), 2) disagreements caused by random subsamples of data (*Subsampling Disagreements*), 3) disagreements induced by the choice of model within an equivalence class of predictors with good empirical performance (*Underspecification Disagreements*). In addition to reporting the strength of said disagreements, we propose methodologies to minimize them *e.g.* by explaining the model regionally, increasing subsample sizes, and treating correlated

(or interacting) features as a single group.

The results of these research questions have been implemented into the `PyFD` (Python Functional Decomposition) framework allowing practitioners to efficiently compute any post-hoc additive explanation and increase their alignment.

1.3 Thesis Outline

The content of this Thesis is structured into three parts. The first two investigate Questions I and II respectively, while the third one puts everything together into a practical framework. We now describe the content of each Part and of each Chapter within.

Part I (Unification)

This part searches for the root cause of disagreements between post-hoc additive explanation methods.

Chapter 3 unifies several additive explainers via the theory of Functional Decomposition. More precisely, all post-hoc additive explainers are expressed in terms of $\boldsymbol{z}$-Anchored Decompositions [Kuo et al., 2010]. Functional Decomposition reveals how different XAI techniques relate and in which contexts they agree/disagree. For instance, We demonstrate that explainability techniques disagree because of feature interactions. More importantly, when all techniques agree, they coincide with the built-in explanations of an additive model. Given this discovery, we advocate increasing agreement between explainability methods rather than benchmarking them.

Chapters 4 & 5 present algorithms for efficiently computing functional decompositions in a model-agnostic (**Chapters 4**) or model-specific (**Chapters 5**) context. These algorithms are important to operationalize the explainability framework that will be used in subsequent parts.

Part II (Alignment)

In light of our unification, we advocate increasing the alignment between contradicting explanations under the promise that they will all eventually converge to an ideal explanation. Each chapter in this part identifies a type of disagreement, and illustrates how to minimize it.

Chapter 6 reports the disagreements between XAI techniques under the label *Interaction Disagreement*. It is demonstrated empirically that restricting the model explanations to well-chosen *regions* with reduced interactions can increase alignment between the techniques. These regions are defined as the leaves of a special Decision Tree called a *FD-Tree* (Functional Decomposition Tree).

Chapter 7 identifies the random choice of data subsample as a type of disagreement between explanations. We refer to this lack of alignment as the *Subsampling Disagreement*. In most scenarios where data is subsampled uniformly at random, these disagreements can be characterized using statistical Confidence Intervals. Nevertheless, this Chapter explores a novel *Audit Scenario* where a Company has a dataset and biased model and an Auditor with limited access to data (due to privacy concerns) must explain the model to identify the bias. We demonstrate that the Company can cherry-pick the samples send to the Auditor in order to provide *arbitrary* model explanations. The Auditor, due to their limited access to the data, cannot detect the malicious manipulation.

Chapter 8 quantifies the contradictions between explanations of models with an equivalent empirical performance (*i.e.* a *Rashomon Set*). This chapter refers to the average rate of model contradictions across a dataset as the *Underspecification Disagreement*. This score is meant to inform practitioners of the amount of statements supported by their model which remain true when considering any competing model. It is finally demonstrated that Underspecification Disagreements can be reduced by treating correlated features as a single group within the explanation.

Part III (Practical Guidelines)

Given that we understand the root-cause of disagreements and that we have methodologies to report/reduce said disagreements, it remains to put everything together into a practical framework that can be used by developers.

Chapter 9 regroups the contributions of parts I and II into a package called `PyFD` (Python Functional Decomposition). More precisely, the model-agnostic and model-specific algorithms presented in part I are implemented, allowing users to efficiently compute $\boldsymbol{z}$-Anchored Decompositions. These decompositions are then aggregated to yield any additive explanation. In addition, the package allows reporting and reducing the Interaction and Subsampling Disagreements.

Chapter 10 illustrates how the `PyFD` package can be applied on two real-world data : the BikeSharing and Marketing UCI repository datasets. Notably, we demonstrate how the package lets users make conscious decisions (fitting a FDTree or treating certain features as a group) that lead to post-hoc additive explanations with increased alignment.

Chapter 11 concludes the thesis and discusses future works towards explainable ML systems.

1.4 Publications

Main Papers The content of this Thesis is based on the following papers

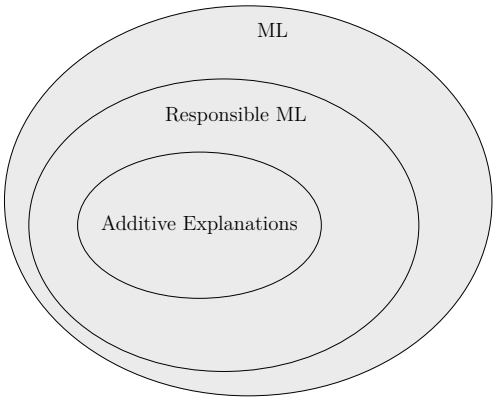
- **Laberge, G.,** Aïvodji, U., Hara, S., Marchand, M., & Khomh, F. (2023, May). *Fooling SHAP with Stealthily Biased Sampling.* In The Eleventh International Conference on Learning Representations.
- **Laberge, G.**, Pequignot, Y., Mathieu, A., Khomh, F., & Marchand, M. (2023). *Partial Order in Chaos: Consensus on Feature Attributions in the Rashomon Set.* Journal of Machine Learning Research, 24(364), 1-50.
- **Laberge, G.**, Pequignot, Y. B., Marchand, M., & Khomh, F. (2024, May). *Tackling the XAI Disagreement Problem with Regional Explanations.* In International Conference on Artificial Intelligence and Statistics (pp. 2017-2025). PMLR.
- **Laberge, G.**, & Pequignot, Y. (2022). Understanding interventional treeshap: How and why it works. **arXiv preprint arXiv:2209.15123**.

Secondary Papers Additional papers have been written during my PhD. While they all investigate the role of explainability in building responsible Machine Learning systems, they did not fit within the specific narrative of the Thesis and so were excluded.

- Tambon, F., **Laberge, G.**, An, L., Nikanjam, A., Mindom, P. S. N., Pequignot, Y., ... & Laviolette, F. (2022). *How to certify machine learning based safety-critical systems? A systematic literature review.* Automated Software Engineering, 29(2), 38.
- Ferry, J., **Laberge, G.**, & Aïvodji, U. (2024). *Learning Hybrid Interpretable Models: Theory, Taxonomy, and Methods.* Transactions on Machine Learning Research, 2835-8856.
- Openja, M., **Laberge, G.**, & Khomh, F. (2024). *Detection and evaluation of bias-inducing features in machine learning.* Empirical Software Engineering, 29(1), 22.
- Roy, S., **Laberge, G.**, Roy, B., Khomh, F., Nikanjam, A., & Mondal, S. (2022, October). *Why don't XAI techniques agree? Characterizing the disagreements between post-hoc explanations of defect predictions.* In 2022 IEEE International Conference on Software Maintenance and Evolution (ICSME) (pp. 444-448). IEEE.
- Oueslati, K., **Laberge, G.**, Lamothe, M., & Khomh, F. (2024). *Mining Action Rules for Defect Reduction* Planning. Proceedings of the ACM on Software Engineering, 1(FSE), 2309-2331.

CHAPTER 2 BACKGROUND

2.1 Supervised Machine Learning



Machine Learning (ML) is a programming paradigm where instead of hard-coding logic and rules, one lets models adapt their internal logic based on observational data. This methodology is useful for solving tasks that involve abstract *concepts* that are hard to characterize using formal mathematics or traditional programming. As an example, predicting whether an email is spam or not can be done by detecting the presence of certain characters in the text. Intuitively, characters such as `!!!`, `please`, and `congratulations` should be highly correlated with the concept of `spam`, but a practitioner cannot be expected to enumerate all these red flags characters. It makes more sense to collect data representing spam and non-spam emails, and let an algorithm inspect the frequency of the various characters and decide which ones are most useful for prediction. As another example, it has been notoriously difficult to perform computer vision and speech recognition. These applications involve high-level concepts such as a cat's face or a consonant in speech, which are complicated to detect automatically. In the past decade, these problems have been successfully tackled by training Deep Neural Networks under the promise that these models learn the relevant concepts on their own [Goodfellow et al., 2016].

This literature review of ML will focus solely on Supervised Learning where one has access to labeled data representing the expected system behavior on a variety of operating conditions. Note that Supervised Learning is only a subset of all possible learning settings: Unsupervised Learning, Semi-Supervised Learning, Reinforcement Learning, Generative AI, etc.

As a simplified scenario to illustrate the utility of Supervised Learning programs, let's imagine you are working for a bike rental company and you want to predict the number of bike rentals at a certain hour given information about the weather and the day of the week. A classical program you could come up with might look like

if `temperature=cold` or `time=late` or `time=early` **then**
    return `few-bike-rentals`
else if `temperature=hot_but_not_too_hot` **then**
    return `many-bike-rentals`
else
    return `medium-bike-rentals`.

This program has several issues. First, it is not clear what `hot` and `cold` temperature actually mean numerically. What thresholds should be used? Should they be chosen by hand or by conducting a line search? Same thing with the statements `time=late` and `time=early`. The solution proposed with Machine Learning is to *learn* those thresholds (and the program structure itself) automatically based on historical data of bike rentals given various hours and temperatures.

To fix your program, you decide to apply the Supervised Learning methodology. You first define an input space $\mathcal{X} \subseteq \mathbb{R}^d$ of d features (d different time/weather measurements), an output space $\mathcal{Y}$ representing all possible numbers of bike rentals, a hypothesis space $\mathcal{H} \subset \mathcal{Y}'^{\mathcal{X}}$ storing potential predictive models $h : \mathcal{X} \to \mathcal{Y}'$, and a loss function $\ell : \mathcal{Y}' \times \mathcal{Y} \to \mathbb{R}_+$ that quantifies the error between predicted and actual bike rentals. After setting up the problem, you collect data that represents the expected system behavior *i.e.* tuples $S := \{(\boldsymbol{x}^{(i)}, y^{(i)})\}_{i=1}^N$ of inputs and ideal outputs. For your specific use-case, these tuples represent time/weather measurements $\boldsymbol{x}^{(i)}$ and their associated number of bike rentals $y^{(i)}$. The data $S \sim \mathcal{D}^N$ is customarily assumed to be iid from an unknown probability distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$. The end goal of your ML pipeline is to find a predictive model in $\mathcal{H}$ with minimal population loss

$$h^\star \in \operatorname*{argmin}_{h \in \mathcal{H}} \mathcal{L}_{\mathcal{D}}(h), \tag{2.1}$$

where

$$\mathcal{L}_{\mathcal{D}}(h) := \mathop{\mathbb{E}}_{(\boldsymbol{x},y)\sim\mathcal{D}}[\ell(h(\boldsymbol{x}), y)]. \tag{2.2}$$

Said otherwise, assuming the distribution $\mathcal{D}$ represents the conditions into which the model will be deployed, you search for the model whose response $h(\boldsymbol{x})$ is on average closest to the actual one y. Yet, since the data-generating distribution $\mathcal{D}$ is unknown, you cannot compute the population loss $\mathcal{L}_{\mathcal{D}}(h)$ directly and must resort to studying the empirical loss on the collected dataset S

$$\widehat{\mathcal{L}}_S(h) := \frac{1}{N}\sum_{i=1}^{N} \ell(h(\boldsymbol{x}^{(i)}), y^{(i)}). \tag{2.3}$$

The empirical loss can be minimized over $\mathcal{H}$ to get an estimate h_S of $h^\star$

$$h_S \in \underset{h \in \mathcal{H}}{\operatorname{argmin}} \ \widehat{\mathcal{L}}_S(h) + \lambda \times \text{Regularization}(h). \tag{2.4}$$

The role of the Regularization term is to specify an a priori preference over functions in $\mathcal{H}$. Undesirable functions are attributed a higher value of Regularization(h) and so they are less likely to be picked as the estimate h_S. Without incorporating a preference toward certain types of functions (either through regularization or by heavily restricting $\mathcal{H}$), the task of learning from data becomes theoretically impossible [Shalev-Shwartz and Ben-David, 2014, Chapter 5]. Regularization is also a vital tool to avoid *overfitting* scenarios where the model h_S makes accurate predictions on S but does not work well on fresh data sampled from $\mathcal{D}$.

When the target is continuous ($\mathcal{Y} \subseteq \mathbb{R}$), the ML task is called regression and the most common loss function employed is the Squared Loss

$$\ell(y', y) = (y' - y)^2. \tag{2.5}$$

The corresponding empirical loss $\widehat{\mathcal{L}}_S(h)$ is called the Mean Squared Error (MSE). Taking its square root $\sqrt{\widehat{\mathcal{L}}_S(h)}$ leads to the Root Mean Squared Error (RMSE), a performance metric that has the advantage of having the same units as the target y. Returning to your bike rental task, directly predicting the number of rentals is a regression problem.

Alternatively, when the target is discrete ($\mathcal{Y} = \{0, 1\}$), the task is called classification. If the model's output space is also discrete ($\mathcal{Y}' = \{0, 1\}$), then the 0-1 loss can be utilized to assess performance

$$\ell(y', y) = \mathbb{1}[y' \neq y]. \tag{2.6}$$

The associated empirical loss $\widehat{\mathcal{L}}_S(h)$ is referred to as the misclassification rate and $1 - \widehat{\mathcal{L}}_S(h)$ is colloquially called the accuracy. Because the 0-1 loss is not differentiable w.r.t the estimate y', it can be hard to minimize Equation 2.4. In fact, this minimization problem is NP-Hard for various hypothesis classes [Shalev-Shwartz and Ben-David, 2014]. A common solution is to first let the model output be continuous ($\mathcal{Y}' = \mathbb{R}$) and then use its sign $y' > 0$ or $y' < 0$ as the discrete prediction. The 0-1 loss becomes

$$\ell(y', y) = \mathbb{1}[\,(2y - 1)\, y' \leq 0\,]. \tag{2.7}$$

Afterward, one can define a convex upper-bound of the 0-1 loss and minimize it to learn the classifier [Mohri et al., 2018, Chapter 4]. A notable example of convex upper-bound is the

logistic-loss

$$\ell(y', y) = \log_2 \Big[1 + \exp(\,-(2y-1)\,y'\,)\Big]. \tag{2.8}$$

When $y = 1$, the logistic-loss heavily penalizes negative values of y' and the reverse occurs when $y = 0$. The task of bike predictions could be phrased as a classification problem if your goal is to predict whether the amounts of rentals is larger than average or not.

We now present several hypothesis spaces $\mathcal{H}$ with varying degrees of expressivity.

2.1.1 Parametric Models

Parametric hypothesis spaces $\mathcal{H} = \{h_{\boldsymbol{\theta}} : \boldsymbol{\theta} \in \Theta\}$ regroup functions $h_{\boldsymbol{\theta}} : \mathcal{X} \to \mathcal{Y}'$ that are indexed using a vector of p parameters $\boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^p$. A parametric hypothesis space is called *identifiable* if each function $h_{\boldsymbol{\theta}} \in \mathcal{H}$ is associated with a unique $\boldsymbol{\theta}$. When discussing such spaces, the notion of "model" can refer to either the function $h_{\boldsymbol{\theta}}$ or the parameters $\boldsymbol{\theta}$ without any ambiguity. If the space is not identifiable, however, many parameters can be associated with the same function, and so the notion of "model" refers exclusively to the function.

Linear Models The simplest parametric hypothesis spaces is the set of Linear Models

$$h^{\text{lin}}_{\boldsymbol{\omega}}(\boldsymbol{x}) = \omega_0 + \sum_{j=1}^{d} \omega_j x_j, \tag{2.9}$$

where the parameter ω_0 is called the *intercept* and ω_j for $j = 1, \ldots, d$ are called the *weights* [Hastie et al., 2009, Chapter 3]. In regression, linear models are typically trained using the squared loss and regularized via the weights norm : $\text{Regularization}(h^{\text{lin}}_{\boldsymbol{\omega}}) = \sum_{j=1}^{d} \omega_j^2$. This penalization enforces a preference over functions that do not vary too much with respect to perturbations of $\boldsymbol{x}$. Letting $\boldsymbol{X}$ be the $N \times (d+1)$ matrix whose ith row is $[1, \boldsymbol{x}^{(i)\,T}]$, and $\boldsymbol{y}$ is the vector of N targets $y^{(i)}$, then Equation 2.4 has a closed form solution

$$\boldsymbol{\omega}_S = (\boldsymbol{X}^T\boldsymbol{X} + \lambda N\, \boldsymbol{I}_{-0})^{-1}\boldsymbol{X}^T\boldsymbol{y}, \tag{2.10}$$

with

$$\boldsymbol{I}_{-0} := \begin{bmatrix} 0 & \mathbf{0}^T \\ \mathbf{0} & \boldsymbol{I}_d \end{bmatrix} \tag{2.11}$$

Linear models also apply to classification. To see how, note that sign of the model output $h^{\text{lin}}_{\boldsymbol{\omega}}(\boldsymbol{x}) > 0$ or $h^{\text{lin}}_{\boldsymbol{\omega}}(\boldsymbol{x}) < 0$ can be used to make discrete predictions. Minimizing the corresponding logistic loss (cf. Equation 2.8), which is a convex upper-bound of the 0-1 loss, leads

to the following empirical loss minimization

$$\min_{\boldsymbol{\omega}\in\mathbb{R}^p} \frac{1}{N}\sum_{i=1}^{N} \log_2\Big[1+\exp\Big(-(2y^{(i)}-1)h^{\text{lin}}_{\boldsymbol{\omega}}(\boldsymbol{x}^{(i)})\Big)\Big] \;+\; \lambda\times\sum_{j=1}^{d}\omega_j^2. \tag{2.12}$$

This minimization problem does not have a closed-form solution. However, it is a convex optimization problem over $\boldsymbol{\omega}$ and so it can be solved with an iterative algorithm *e.g.* Newtons method. A linear model trained by solving Equation 2.12 is commonly referred to as a Logistic Regression, which can be confusing because it actually performs classification.

Because they make very strong assumptions on the relationship between $\boldsymbol{x}$ and y, linear models tend to not perform well in practice. Hence, we need to make them more expressive but we do not want to sacrifice the property that makes their optimization so simple: their output is linear w.r.t the parameters $\boldsymbol{\theta}$. To increase expressivity, it is common to define a feature embedding $\boldsymbol{\xi}:\mathcal{X}\to\mathbb{R}^D$ that maps $\boldsymbol{x}$ to some higher dimensional space $\mathbb{R}^D$. A linear model can then be applied to the embedding instead of the original feature [Hastie et al., 2009, Chapter 5]

$$h^{\text{lin-embed}}_{\boldsymbol{\omega}}(\boldsymbol{x}) = \omega_0 + \sum_{j=1}^{D}\omega_j\xi_j(\boldsymbol{x}). \tag{2.13}$$

Optimizing this model is done by solving Equations 2.10 or 2.12 using the embedding $\boldsymbol{\xi}(\boldsymbol{x})$ in place of the raw-input $\boldsymbol{x}$.

Kernel Methods When embedding $\boldsymbol{x}$, the dimension D can quickly become unmanageable. For example a polynomial expansion of degree p applied on d features leads to an embedding of size $\binom{d+p}{p}$. If $d=p=10$, this space has a dimension of almost 200K. *Kernel methods* tackle this computational challenge when $N\ll D$. They define the kernel

$$k(\boldsymbol{x},\boldsymbol{x}') := \langle\boldsymbol{\xi}(\boldsymbol{x}),\boldsymbol{\xi}(\boldsymbol{x}')\rangle_{\mathbb{R}^D} \tag{2.14}$$

representing scalar products in the embedding $\mathbb{R}^D$, and define the function

$$h^{\text{kernel}}_{\boldsymbol{\alpha}}(\boldsymbol{x}) := \sum_{i=1}^{N}\alpha_i k(\boldsymbol{x},\boldsymbol{x}^{(i)}) \tag{2.15}$$

with coefficients $\boldsymbol{\alpha}\in\mathbb{R}^N$. The Representer Theorem [Shalev-Shwartz and Ben-David, 2014, Section 16.2] states that

$$\min_{\boldsymbol{\omega}\in\mathbb{R}^D}\widehat{\mathcal{L}}_S(h^{\text{lin-embed}}_{\boldsymbol{\omega}})+\lambda\|\boldsymbol{\omega}\|^2 = \min_{\boldsymbol{\alpha}\in\mathbb{R}^N}\widehat{\mathcal{L}}_S(h^{\text{kernel}}_{\boldsymbol{\alpha}})+\lambda\boldsymbol{\alpha}^T\boldsymbol{K}\boldsymbol{\alpha}, \tag{2.16}$$

where $\boldsymbol{K}$ is a $N \times N$ matrix containing elements $K_{ij} = k(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)})$. Moreover, the minimizers $\boldsymbol{\omega}_S$ and $\boldsymbol{\alpha}_S$ of each side of Equation 2.16 are two different parametrization of the same function

$$h^{\text{lin-embed}}_{\boldsymbol{\omega}_S} = h^{\text{kernel}}_{\boldsymbol{\alpha}_S}. \tag{2.17}$$

When doing regression with the squared loss, the solution using a kernel parametrization still has a closed-form [Mohri et al., 2018, Chapter 11]

$$\boldsymbol{\alpha}_S = (\boldsymbol{K} + \lambda N \boldsymbol{I})^{-1} \boldsymbol{y}. \tag{2.18}$$

Kernel parametrization has other advantages beyond efficient computations when $N \ll D$. Indeed, one can let the embedding map to a (possibly infinite dimensional) Hilbert space $\mathbb{H}$, and study the kernel

$$k(\boldsymbol{x}, \boldsymbol{x}') := \langle \boldsymbol{\xi}(\boldsymbol{x}), \boldsymbol{\xi}(\boldsymbol{x}') \rangle_{\mathbb{H}} \tag{2.19}$$

while ignoring the embedding altogether. Kernels that respect Equation 2.19 for some $\mathbb{H}$ are called Positive Definite Symmetric (PDS) kernels [Mohri et al., 2018, Chapter 6]. Notable examples include

- Polynomial Kernel : $k(\boldsymbol{x}, \boldsymbol{x}') = (\gamma \boldsymbol{x}^T \boldsymbol{x}' + 1)^p$.
- Gaussian Kernel : $k(\boldsymbol{x}, \boldsymbol{x}') = \exp(-\gamma \|\boldsymbol{x} - \boldsymbol{x}'\|_2^2)$.
- Laplace Kernel : $k(\boldsymbol{x}, \boldsymbol{x}') = \exp(-\gamma \|\boldsymbol{x} - \boldsymbol{x}'\|_1)$.

Neural Networks Linear models can be made more expressive by training them on a high-dimensional embedding of the input. This embedding could either be explicitly computed or indirectly through kernels. Nonetheless, practitioner need to define the embedding $\boldsymbol{\xi}$ or the kernel $k(\cdot, \cdot)$ a priori. Defining the right embedding requires experimentation and domain knowledge for each separate use-case. This process is difficult, but not impossible. For instance, pre-2012, state-of-the-art models on the ImageNet Image Classification Benchmarks were using domain-knowledge embeddings (Scale-Invariant Feature Transforms and Fisher Vectors) and had reasonable top-5 accuracies of 74.3% [Sánchez and Perronnin, 2011].

Defining the correct embedding for each use case is a time-consuming task. To tackle this issue, the central idea behind Deep Neural Networks is to let the model *learn* the embedding (or representation) for any given task [Goodfellow et al., 2016]. That is, the embedding $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ will itself be a parametric function fitted on data

$$\xi_{k,\{\boldsymbol{\omega},b\}}(\boldsymbol{x}) = a(\boldsymbol{\omega}^T \boldsymbol{x} + b), \tag{2.20}$$

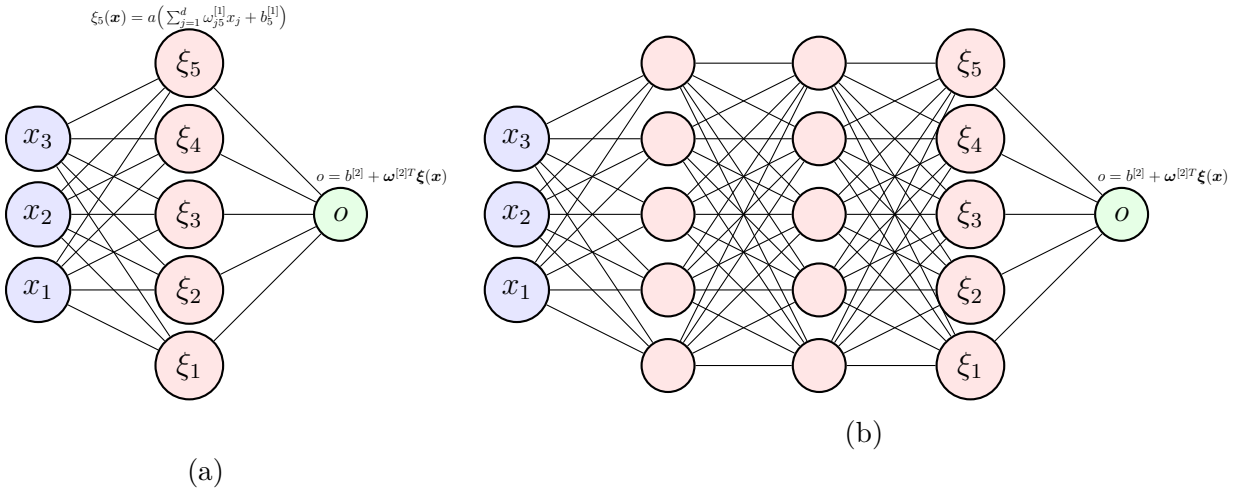


Figure 2.1 Basic Neural Network architectures. (a) A Shallow Neural Network defines the embeddings $\boldsymbol{\xi}(\boldsymbol{x})$ as the composition between an affine function and a non-linear activation a. (b) A Multi-Layered Perceptron (MLP) is a composition of hidden layers, leading up to the final embedding $\boldsymbol{\xi}(\boldsymbol{x})$ used for prediction by a linear model. The premise of Deep Learning is that earlier layers learn simple concepts such as edges and corners in images, while deeper layers learn higher-level concepts such as a nose or eyes.

where $a : \mathbb{R} \to \mathbb{R}$ is a non-linear activation function and $\{\boldsymbol{\omega}, b\}$ are parameters. A Shallow Neural Network takes the form

$$h_{\boldsymbol{\theta}}^{\text{shallow}}(\boldsymbol{x}) := b^{[2]} + \sum_{k=1}^{D} \omega_k^{[2]}\, a\left(\sum_{j=1}^{d} \omega_{jk}^{[1]} x_j + b_k^{[1]}\right), \tag{2.21}$$

see Figure 2.1 (a). In this figure, the middle layer representing the embedding is called a *hidden layer*. These hidden layers can be stacked together, leading to a more expressive model called a Multi-Layered Perceptron (MLP), see Figure 2.1 (b). The compositional structure of MLPs is the basis of Deep Learning, whose premise is that earlier layers learn basic concepts (*e.g.* edges, words) while deeper layers learn higher-level ones (*e.g.* a face, a figure of speech) [Goodfellow et al., 2016].

The idea that DNNs learn the same concepts as humans in their hidden layers is still debated [Freiesleben and König, 2023]. Nevertheless, the empirical success of Deep Learning cannot be denied. Going back to the discussion on the ImageNet Image Classification Benchmark: in 2012, Deep Convolutional Neural Networks famously obtained a 83% top-5 accuracy, improving the latest state-of-the art by about 10% [Krizhevsky et al., 2012]. This historic event has been one of the major catalyst toward the widespread adoption of Deep Learning. Not only did these models conveniently learn the embedding by themselves, they had even better

performance! Throughout subsequent years, the ImageNet top-5 accuracy has been used as the golden standard driving innovations in Deep Neural Network architectures. In 2015, this race has cumulated in ResidualNets that attained a 96.4% top-5 accuracy, surpassing human-level performance [He et al., 2016].

More recently, Deep Learning led to breakthroughs in Natural Language Processing: so-called Large Language Models (LLMs) are trained on all the textual data available on the internet, and their knowledge-base can be queried via natural language. Knowledge extraction from LLMs is demonstrably useful in solving University-level exams and coding tasks [Achiam et al., 2023]. Reminiscent to past competitions on the ImageNet Benchmark, there is an ongoing arms race to develop the penultimate LLM that beats all others on a variety of Benchmarks[1].

While Neural Networks conveniently learn the embedding $\boldsymbol{\xi}$ by themselves, they lose an important property : their output is no longer linear w.r.t the parameters $\boldsymbol{\theta}$. Consequently, even if the loss function ℓ is convex, the minimization problem underlying Equation 2.4 is no longer convex. As a result, Neural Networks trained with iterative algorithms have no guarantee of reaching a global minimum, and it is common to stop the learning procedure whenever the loss computed on held-out data stops decreasing. Another consequence of non-convexity is that models trained from M different parameter initializations lead to different functions $\{h_k^{\text{MLP}}\}_{k=1}^M$. This is, in fact, the basis behind Deep Ensembles that combine independently trained models for uncertainty quantification [Lakshminarayanan et al., 2017].

2.1.2 Non-parametric Models

Non-parametric hypothesis spaces regroup functions that cannot be indexed by a *fixed* number of p parameters. Such functions can still employ parameters, but their amount is adaptively chosen during training.

Decision Trees Decision trees are non-parametric models that partition the input space $\mathcal{X}$ into multiple regions Ω_ℓ (*i.e.* $\mathcal{X} = \bigcup_\ell \Omega_\ell$ s.t. $\Omega_\ell \cap \Omega_{\ell'} = \emptyset$ when $\ell \neq \ell'$). Said partitioning is done in a recursive manner by repeatedly splitting along individual features, see Figure 2.2. The resulting model takes the following form [Hastie et al., 2009, Section 9.2]

$$h^{\text{tree}}(\boldsymbol{x}) = \sum_\ell v_\ell \mathbb{1}(\boldsymbol{x} \in \Omega_\ell), \tag{2.22}$$

[1] `https://www.vellum.ai/llm-leaderboard`

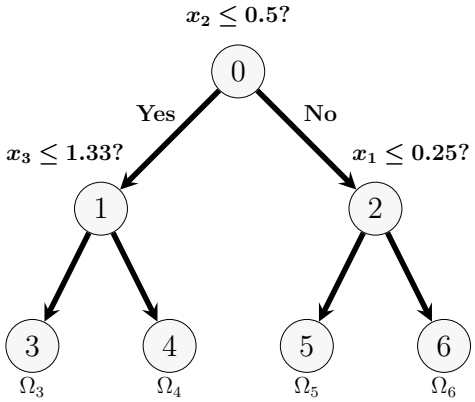


Figure 2.2 Basic example of Decision Tree. Each of its leaf represents an element Ω_ℓ of a partition of $\mathcal{X}$.

where the output is a constant v_ℓ in each region Ω_ℓ. Learning a decision tree requires determining its structure, the splits done at each of its internal node, and the value returned at each of its leaves. When the tree structure and splits are fixed, the value v_ℓ is straightforward to optimize. Indeed, for the squared loss, the optimal prediction is the average target value of data samples in Ω_ℓ. For the 0-1 loss, predicting the majority class of samples in Ω_ℓ is the optimal choice. The tricky part of training is finding the minimal tree structure and its splits (a NP-Hard problem [Shalev-Shwartz and Ben-David, 2014]). Thus, solving to optimality is out of the question and greedy heuristics are used instead: nodes are split by locally minimizing a surrogate of the empirical loss, without any backtracking or consideration for the fitness of subsequent splits. Starting from root, splits are performed greedily until a termination criterion is satisfied [Louppe, 2014]. Examples of termination criteria include: attaining a minimal number of data samples in the node, reaching a maximum depth `max_depth`, or not being able to significantly reduce the objective by splitting further.

Greedy heuristics make training procedure very efficient, but they also introduce instability of the model w.r.t perturbations of data S. Moreover, recursively splitting along features allow decision trees to quickly isolate individual datum *e.g.* 1000 data points can be isolated by a depth 10 tree. These two observations explain why individual decision trees are prone to *overfitting* the training data. To solve this issue, the state-of-the-art practice is to learn an ensemble of M trees $\{h^{\text{tree},[k]}\}_{k=1}^{M}$ and return a linear combination of their output

$$h^{\text{ensemble}}(\boldsymbol{x}) := \omega_0 + \sum_{k=1}^{M} \omega_k \, h^{\text{tree},[k]}(\boldsymbol{x}). \tag{2.23}$$

We now discuss two popular ensemble methods : Random Forests and Gradient Boosting.

Random Forests A Random Forest (RF) is an ensemble of independently trained decision trees whose predictions are averaged. The act of averaging individual decision trees smooths their decision boundary and make them less prone to overfitting. To increase the smoothing effect, trees are diversified by training them on a different bootstrap sample of the original dataset and restricting their internal nodes to split among a random subset of features [Breiman, 2001a]. Letting $r \in \mathbb{N}$ represent the seed encoding all pseudo-random processes in the training of a single tree $h^{\text{tree},[r]}$ and $U([M])$ be the uniform distribution over all M possible random seeds on a computer, the theoretical definition of a Random Forest is

$$h^{\text{rf}}(\boldsymbol{x}) = \mathop{\mathbb{E}}_{r\sim U([M])}[h^{\text{tree},[r]}(\boldsymbol{x})]. \tag{2.24}$$

In practice, the expectation $\mathbb{E}_{r\sim U([M])}$ has to be approximated using Monte-Carlo sampling. We draw m seeds from $R \sim U([M])^m$ and return the sample average

$$h^{\text{rf}}_R(\boldsymbol{x}) = \frac{1}{|R|}\sum_{r\in R} h^{\text{tree},[r]}(\boldsymbol{x}). \tag{2.25}$$

The estimated RF should converge to the true RF (cf. Equation 2.24) as m increases.

Gradient Boosted Trees Instead of learning an ensemble by training trees independently, trees can be learned sequentially so that each additional tree corrects the errors made by its predecessors. More formally, given an intercept ω_0 and sequence of decision trees $(h^{\text{tree},[k]})_{k=1}^m$, the next tree $h^{\text{tree},[m+1]}$ can be learned by minimizing

$$h^{\text{tree},[m+1]} = \mathop{\text{argmin}}_{h^{\text{tree}}} \widehat{\mathcal{L}}_S\Big(\omega_0 + \nu \sum_{k=1}^{m} h^{\text{tree},[k]} + h^{\text{tree}}\Big), \tag{2.26}$$

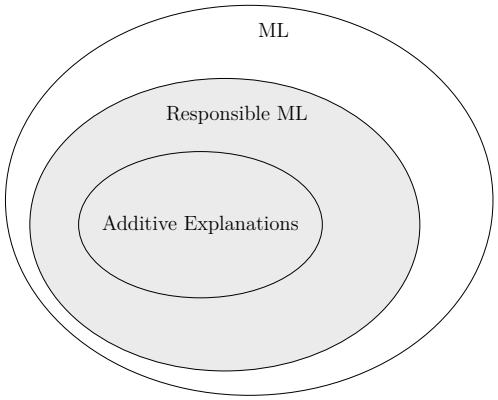
which is a procedure called boosting [Hastie et al., 2009, Chapter 10]. The hyperparameter $\nu \in]0, 1]$ is called the learning rate and it controls the convergence rate of the boosting rounds. After M boosting rounds, the final model is

$$h^{\text{boost}}(\boldsymbol{x}) := \omega_0 + \nu \sum_{k=1}^{M} h^{\text{tree},[k]}(\boldsymbol{x}). \tag{2.27}$$

Solving Equation 2.26 is extremely hard and so boosting algorithms resort to solving it approximately. Assuming the loss function employed is twice differentiable (*e.g.* squared loss and logistic loss), the Gradient Boosted Trees (GBT) algorithm approximately solves Equation 2.26 by leveraging a Taylor Expansion of the loss $\ell(y'+\Delta, y) \approx \ell(y', y)+\ell'(y', y)\Delta+\ell''(y', y)\Delta^2/2$ at the current prediction $y' = \omega_0 + \nu \sum_{k=1}^m h^{\text{tree},[k]}(\boldsymbol{x})$ [Friedman, 2001].

2.2 Responsible Machine Learning

We spent time introducing various ML models used in Supervised Learning and how they are trained. Yet one question remains : *how are they validated?*. The typical approach is to keep some held-out data T and report the empirical error $\widehat{\mathcal{L}}_T(h_S)$, which is an unbiased estimate of the population loss $\mathcal{L}_{\mathcal{D}}(h_S)$. But is reporting this metric enough to guarantee the model will operate properly when deployed? No, for two reasons : shortcut learning and legal regulations.



Shortcut Learning ML models are good at identifying patterns in large datasets. In fact, they are too good: they can find patterns specific to the collected data, allowing them to make accurate predictions without necessarily learning the concepts that were indented by the practitioner. We shall informally refer to "shortcut learning" any disconnect between a ML program's actual and intended behavior caused by the dataset or the learning algorithm. Notorious examples of shortcut learning are animal classifiers trained on image data where certain species only appear in specific environments. For example, cows appear mostly in prairies [Beery et al., 2018] and wolves appear mainly in snowy environments [Ribeiro et al., 2016]. In either case, the models fail whenever they are shown images of an animal in a different biome (*e.g.* cows on beaches), meaning that they are leveraging the image background to make decisions. This is a disconnect between the classifier's intended and actual behavior. Reporting the held-out performance $\widehat{\mathcal{L}}_T(h_S)$ does not highlight this issue if the whole dataset suffers from correlations between animal species and environment.

Why not simply collect more data, then? As more data is collected, wouldn't the model eventually be forced to learn the relevant patterns/concepts seeing as any other pattern will lose its predictive power? This mentality is at the heart of the current trend with Large Language Models that are trained over all data available on the internet. Nevertheless, we argue that more data is not always the answer, as evidenced by the following use-case.

Predicting Mortality from Pneumonia

Caruana et al. [2015] previously investigated the use of Machine Learning to predict the probability of death for patients with pneumonia. The indented use of this model was to help hospitals assign priority access to high-risk subjects. While the various predictive models trained presented encouraging held-out performance, further analysis highlighted a critical flaw in their reasoning. The models were assigning lower risks to certain patients **because** they had asthma. This behavior appears counterintuitive at first, but is actually consistent with the data used to train the models. Indeed, in the collected data, patients with asthma were treated more aggressively and so their chances of dying from pneumonia was actually lower than the general population.

The models learned the shortcut "asthma implies low-risk", allowing them to make accurate predictions, while inducing a disconnect between their actual and intended behaviors. The **intended** behavior is to prioritize hospital access to asthmatic patient because asthma is a known risk factor for pneumonia. The **actual** behavior is the complete opposite. Collecting more data does not solve this issue. Fixing it would require a randomized control trial where patients suffering from pneumonia are randomly assigned degrees of healthcare (staying home, external clinic, hospital, Intensive Care Unit). Conducting this trial would be illegal.

In this example, it was possible to identify the shortcut learning because the model employed was transparent *i.e.* its predictions could be explained. But what if the model is a black-box whose decisions are opaque? Going back to the example of image classifiers that rely on the background to make predictions. How would you affirm that the model is not leveraging information from the background if it is too complex to be introspected? This is one of the important questions tackled in explainability.

Legal regulations Beyond its ineffectiveness at detecting shortcut learning, held-out performance $\widehat{\mathcal{L}}_T(h_S)$ is no longer a sufficient credential because of legal regulations. For example, the European AI-Act and Canada's future C-27 law impose strict regulations on ML models before they can be deployed in society. These legal constraints will be described shortly, along with the research subfields that aim at tackling them: *Fairness* and *Explainability*.

2.2.1 Fairness

ML models just identify and reproduce complex patterns in data. As such, they have no notion of morality or malicious intents. This is good news a priori because ML models make decisions based on objective facts, unlike humans, who are subject to a variety of cognitive biases. Yet, the strong reliance on historical data also means that models can be biased and reproduce/perpetuate past discrimination. To have a clear definition of *discrimination*, we shall study Canada's future C-27 law[2].

> *Under the Artificial Intelligence and Data Act, biased output occurs when there is unjustified and adverse differential impact based on any of prohibited grounds for discrimination in the Canadian Human Rights Act. This includes differentiation that occurs directly or indirectly, such as through variables that act as a proxy for prohibited grounds. [...]*

There are many expressions to unpack from this text. First is the terminology : *differential impact based on any of prohibited grounds for discrimination in the Canadian Human Rights Act*. The phrasing *prohibited grounds of discrimination* can refer to belonging to a certain demographic (*e.g.* sex, race, religion). In the Fairness literature, these demographic subgroups are represented via a sensitive feature $x_s \in \{0, 1\}$ that can differentiate man/woman, Caucasian/African-American etc. The terminology *differential impact* can potentially refer to the many metrics proposed in the Fairness literature. Under the assumption that $h(\boldsymbol{x}) \in \{0, 1\}$, these metrics often take the form

$$\mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{F}}[h(\boldsymbol{x})] - \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}[h(\boldsymbol{x})], \tag{2.28}$$

where $\mathcal{F}$ and $\mathcal{B}$ are probability distributions over different demographic subgroups. Table 2.1 shows the relation between the distributions and the corresponding metric. For instance, the Demographic Parity (DP) [Dwork et al., 2012] enforces equal outcomes between subgroups (*e.g.* equal acceptance rates among men and women). The Predictive Equality (PE) [Corbett-Davies et al., 2017] studies the difference in False Positive Rates (FPR) among subgroups. This metric was famously used in the the context of ML-driven recidivism prediction, where ProPublica demonstrated that African-Americans who did not re-offend ($y = 0$) were more likely to be incorrectly assigned high risk $h(\boldsymbol{x}) = 1$ compared to Caucasians [Larson et al., 2016]. Equal Opportunity (EO) [Hardt et al., 2016] enforces equal True Positive

[2] https://ised-isde.canada.ca/site/innovation-better-canada/en/artificial-intelligence-and-data-act-aida-companion-document

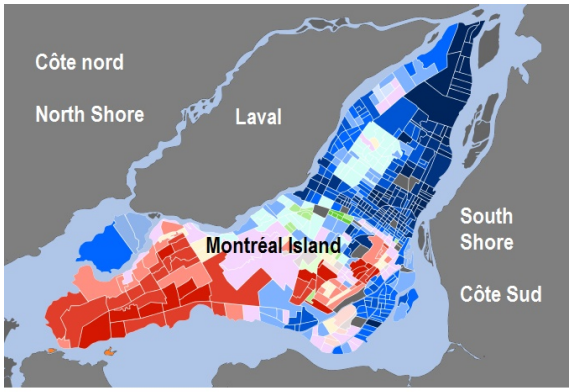
Metric	Description	$\mathcal{F}$	$\mathcal{B}$
DP	Equal Outcomes	$\mathcal{D}\|\{\boldsymbol{x}_s = 0\}$	$\mathcal{D}\|\{\boldsymbol{x}_s = 1\}$
PE	Equal False Positive Rates	$\mathcal{D}\|\{\boldsymbol{x}_s = 0, y = 0\}$	$\mathcal{D}\|\{\boldsymbol{x}_s = 1, y = 0\}$
EO	Equal True Positive Rates	$\mathcal{D}\|\{\boldsymbol{x}_s = 0, y = 1\}$	$\mathcal{D}\|\{\boldsymbol{x}_s = 1, y = 1\}$

Table 2.1 Probability Distributions over demographic subgroups for each Fairness metric. Plugging these distributions into Equation 2.28 yields the metric.

Rate between subpopulations, which is equivalent to enforcing equal False Negative Rates (FNR). Additional metrics compare the Precision $\mathbb{P}[y = 1|h(\boldsymbol{x}) = 1, x_s = s]$ across subgroups and variants thereof. Yet, a famous Impossibility Theorem has demonstrated that enforcing equal Precisions across subgroups is incompatible with enforcing equal FNR and FPR [Chouldechova, 2017]. For a more thorough list of Fairness metrics and their compatibilities/incompatibilities, we refer to the recent work of Defrance and De Bie [2023].

Now that we understand the possible meanings of *differential impact*, we unpack the subsequent statement : *differentiation that occurs directly or indirectly, such as through variables that act as a proxies for prohibited grounds*. This statement distinguishes the notions of *direct* and *indirect* discrimination. Direct discrimination refers to situations where the sensitive attribute x_s is directly used for prediction. Such discrimination is prohibited in US Employment practices according to Title VII of the Civil Rights Act of 1964[3].

Indirect discrimination refers to situations where a feature that is statistically associated with x_s (but not equivalent to it) is used for predictions. For example, consider $x_s \in \{\text{english}, \text{french}\}$ to be the native language of a loan applicant in Montréal. While the explicit use of x_s to accept/refuse a loan is prohibited by law, geographic information can be leveraged to accurately infer native language, see the Figure on the right[4]. Since Montréal is literally split in half between French and English districts, using geographic information in ML models could induce indirect discrimination.



Montréal and its surrounding suburbs. On Montréal Island itself; the darker the read, the more Anglophone, the darker the blue the more Francophone, the closer to green the more Allophone.

Photo credit creative commons

[3] https://www.eeoc.gov/statutes/title-vii-civil-rights-act-1964

[4] https://quebecculturebloc.com/2014/10/28/quebec-trends-in-bilingualism-70/

Berkley Admission Disparities

In 1973, the admission rates at Berkley University exhibited a disparity between men and women applicants. Letting $\mathcal{F} = \mathcal{D}_{\mathcal{X}}|\{x_s = \text{man}\}$ and $\mathcal{B} = \mathcal{D}_{\mathcal{X}}|\{x_s = \text{woman}\}$, the Demographic Parity (cf. Equation 2.28) was about 9% in favor of men. However, this disparity was no longer apparent when investigating every department separately. Subsequent data analysis concluded that acceptance rates were lower for women because they applied to harder departments, such as humanities and social sciences, which had more applicants and less available places [Pearl and Mackenzie, 2018, Page 309]. This is a historic example of *indirect discrimination*, where the proxy variable `departementChoice` induces a disparity between subgroups.

Having defined Fairness metrics and possible causes for their gaps (direct and indirect discrimination), we continue reading the C-27 bill.

> *[...] Adverse differentiation could be considered justified if it is unavoidable in the context of real-world factors affecting a decision or recommendation. For example, individual income often correlates with the prohibited grounds, such as race and gender, but income is also relevant to decisions or recommendations related to credit. The challenge, in this instance, is to ensure that a system does not use proxies for race or gender as indicators of creditworthiness.*

This text clarifies that indirect discrimination could potentially be legal if it is based on *real-world factors affecting a decision or recommendation*. Clarifying the meaning of *real-world factors* is at the heart of Causal Fairness [Kilbertus et al., 2017], where these factors are required to block any causal paths between x_s and y in a causal graph. The choice of *real-world factors* will be task-dependent and probably subject to extensive legal deliberation. Thus, this manuscript will only discuss simplified examples. Notably, firefighting departments, whose sex imbalance can be justified by their high physical standards and associations between strength and sex, will often be brought up.

Causal Fairness investigates how different features x_j impact the model outcome $h(\boldsymbol{x})$. As a result, assessing the fairness of a model is intimately linked to *understanding* its behavior [Rudin et al., 2018]. Shedding light on the reasoning behind model prediction is at the heart of the Explainability subfield, which is discussed next.

2.2.2 Explainability

With the rise in complexity of ML models, there is a simultaneous increase in demand for the *explainability* of their decisions [Arrieta et al., 2020, Belle and Papantonis, 2021]. This is especially true in domains where humans are impacted by the model outcome *e.g.* Medicine, Law, Insurance, and Banking. In fact, article 13 of the European AI-Act[5] reads

> *High-risk AI systems shall be designed and developed in such a way as to ensure that their operation is sufficiently transparent to enable deployers to interpret a system's output and use it appropriately.*

The field of eXplainable Artificial Intelligence (XAI) is currently tackling the challenge of explaining the ML *system's output*. While there is no universal notion of "explanation", multiple definitions have been proposed.

- **Sufficient Explanations (SE)** consider an input $\boldsymbol{x}$ and model h, and provide a region $\Omega \subset \mathcal{X}$ that contains $\boldsymbol{x}$ and such that $h(\boldsymbol{x}) = h(\boldsymbol{x}')$ for all $\boldsymbol{x}' \in \Omega$. This region describes sufficient conditions for making the prediction $h(\boldsymbol{x})$ [Dandl et al., 2023, Ribeiro et al., 2018].

- **Counterfactual Explanations (CE)** consider an input $\boldsymbol{x}$ and model h, and return a region $\Omega \subset \mathcal{X}$ excluding $\boldsymbol{x}$ and such that $h(\boldsymbol{x}) \neq h(\boldsymbol{x}')$ for all $\boldsymbol{x}' \in \Omega$. The region provides a recommendation on how to change $\boldsymbol{x}$ in order to flip the model outcome [Amoukou and Brunel, 2022, Rawal and Lakkaraju, 2020, Wachter et al., 2017].

- **Local Feature Attributions (LFA)** are functionals $\boldsymbol{\phi}^{\text{LFA}} : \mathcal{H} \times \mathcal{X} \times \mathcal{P}(\mathcal{X}) \to \mathbb{R}^d$, where $\mathcal{P}(\mathcal{X})$ refers to the set of all probability distributions over $\mathcal{X}$. The jth component $\phi_j^{\text{LFA}}(h, \boldsymbol{x}, \mathcal{B})$ of the LFA conveys how much each feature j contributes toward the prediction $h(\boldsymbol{x})$ relative to a baseline distribution $\mathcal{B}$ over the features.

- **Global Feature Importance (GFI)** are functionals $\boldsymbol{\Phi}^{\text{GFI}} : \mathcal{H} \times \mathcal{P}(\mathcal{X}) \to \mathbb{R}_+^d$ whose jth component $\Phi_j^{\text{GFI}}(h)$ illustrates how much feature j is used globally by the model over the distribution $\mathcal{B}$. Unlike LFAs, these importance scores are not specific to any input $\boldsymbol{x}$.

[5] https://artificialintelligenceact.eu/article/13/

We emphasize that "explanations" are the output of functionals evaluated on the model h, and they can either be regions or vectors. There are two paradigms in XAI to extract explanations from models : *ante-hoc* and *post-hoc* methods.

Ante-hoc explanations are built-in the model structure [Burkart and Huber, 2021]. Decision Trees are an example of models that provide ante-hoc Sufficient and Counterfactual Explanations, see Figure 2.3. Sufficient Explanations are extracted by taking the conjunction of all boolean statements in the root-leaf path followed by $\boldsymbol{x}$. Letting $\boldsymbol{x}$ be an adult of age 34 making $45K$ dollars per year, a Necessary Explanation for their loan rejection $h(\boldsymbol{x}) = 0$ is $\Omega = (\texttt{age} < 40) \wedge (\texttt{Income} < 50K)$. Counterfactual Explanations are extracted by taking the conjunction of statements going from the root to an alternative leaf with the desired outcome. For the same individual $\boldsymbol{x}$, a counterfactual explanation would be $\Omega = (\texttt{age} < 40) \wedge (\texttt{Income} \geq 50K)$ meaning that increasing their salary within 6 years will grant them the loan.

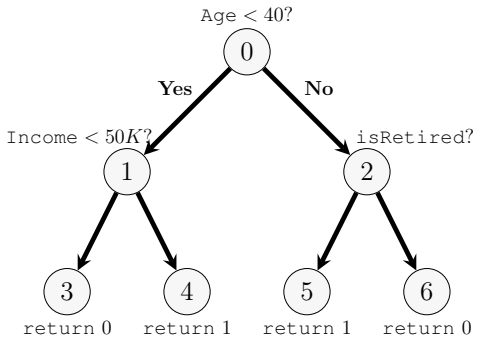


Figure 2.3 Hypothetical Decision Tree predicting whether someone should be given a loan ($y = 1$) or not ($y = 0$). This model provides ante-hoc Sufficient and Counterfactual Explanations.

Post-hoc explanations, on the other hand, are computable on arbitrary functions $h : \mathcal{X} \to \mathcal{Y}'$ and so they are not specific to any model architecture [Burkart and Huber, 2021]. Their definition only requires oracle access to model evaluations $h(\boldsymbol{x})$, or gradients $\nabla h(\boldsymbol{x})$, which makes them applicable to a variety of ML models. For example, while ante-hoc Sufficient Explanations are available from Decision Trees, the Anchor algorithm [Ribeiro et al., 2018] provides post-hoc Sufficient Explanations for arbitrary functions h. Crucially, the distinction between ante-hoc and post-hoc paradigms concerns whether the explanation is built-in the model structure (ante-hoc) or if it is computed from its input-output relationship (post-hoc).

By definition, Post-hoc explanations are *model-agnostic* because their computation only requires querying the model at arbitrary inputs. Yet, *model-specific* implementations can exploit knowledge on the the structure of h to speed up computations. Figure 2.4 clarifies the differences between Interpretability, Explanations, Ante-hoc/Post-hoc paradigms, and model-agnostic/model-specific implementations.

The coming Section reviews a class of ML models that provide ante-hoc LFA and GFI. These ante-hoc explanations are then extended to post-hoc explanations of arbitrary models h.

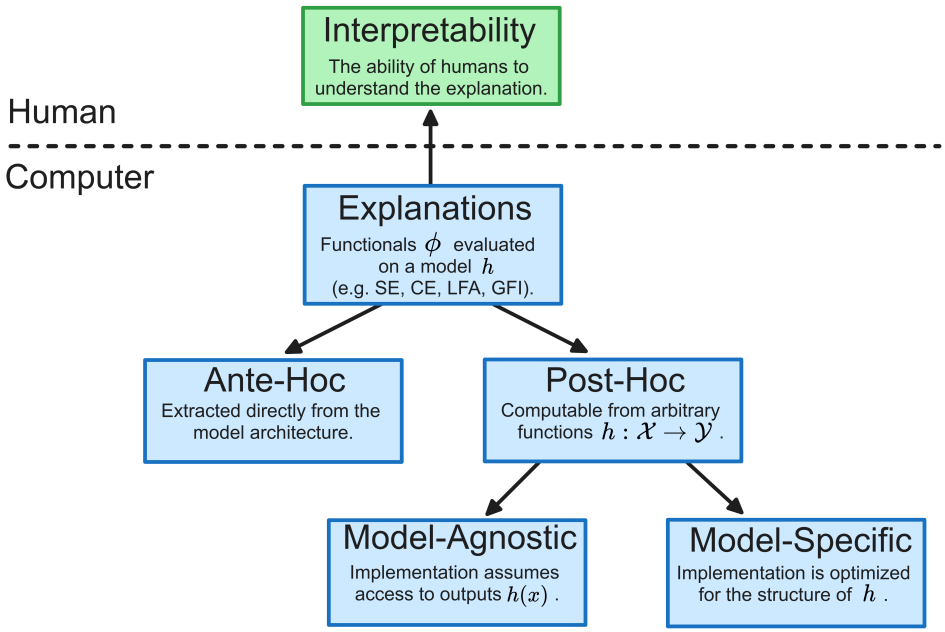
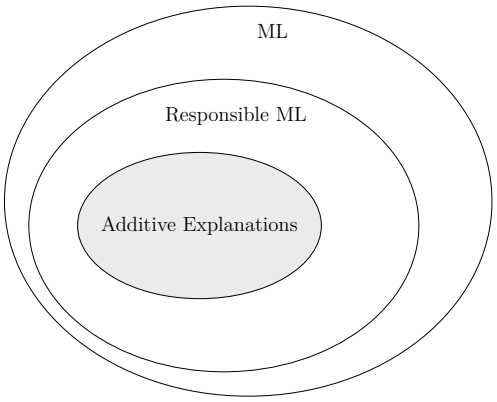


Figure 2.4 XAI Taxonomy.

2.3 Additive Explanations



This section surveys Local Feature Attribution (LFA) and Global Feature Importance (GFI), which are commonly referred to as additive explanations. In light of my Thesis Statement (see Page 7), we will first investigate a class of interpretable ML models called *Additive Models* and demonstrate that they provide ante-hoc additive explanations. These built-in explanations will be considered as a *ground-truth* for the remainder of the manuscript. Because additive models tend to perform worse than their black-box counterparts, there is growing interest in XAI to generalize these feature importance notions to other models types. Thus, the second part of this section reviews the many post-hoc additive explanation proposed to generalize feature importance to non-additive models. Given my Thesis Statement, the following sanity check will be conducted on the various post-hoc additive explanations : *does this technique fall back to ante-hoc explanations whenever the model happens to be additive?* Techniques that fail this sanity check will be discarded, seeing as they cannot be aligned toward this ground-truth.

2.3.1 Ante-hoc Explanations

As a reminder, ante-hoc explanations are extracted from certain ML models by directly looking at their structure. We shall see that, for LFA/GFI explanations, the natural structure is that of *Additive Models*

$$h^{\text{add}}(\boldsymbol{x}) = \omega_0 + \sum_{j=1}^{d} h_j(x_j), \tag{2.29}$$

where each *shape function* h_j only depends on a single feature x_j. Note that linear models are a special case of additive models with $h_j(x_j) = \omega_j x_j$. By design, the contribution of each individual feature toward the output $h(\boldsymbol{x})$ is readily-available, which is why additive models are advertised as being intelligible [Lou et al., 2012]. Moreover, the impact of varying feature x_j on the model response is independent of the values of the remaining features. This allows one to mentally decouple the effects of each feature and investigate the functions h_j individually. To fit an additive model, one must find a way to represent the univariate functions h_j. Both parametric and non-parametric approaches are now discussed.

The shape functions can be modelled non-parametrically as a sum of univariate decision trees. Those trees can be learned by specializing the Gradient Boosted Trees procedure (cf. Equation 2.26). One adaptation of boosting is to simply set the hyperparameter `max_depth=1`. This forces each boosted tree to be a decision stump : $h^{\text{tree},[k]}(\boldsymbol{x}) = v_1 \mathbb{1}[x_j < \gamma] + v_2 \mathbb{1}[x_j \geq \gamma]$. The function h_j is then calculated by summing all decision stumps involving feature j [Chang et al., 2021]. Alternatively, the individual decision trees could have `max_depth>1` but be explicitly constrained to use a single feature. This is the scheme behind the Explainable Boosting Machines (EMBs) of the InterpretML Python library [Nori et al., 2019]. EBMs further modify the boosting procedure by cyclically going through the list of features used by the trees.

The shape functions can be modelled parametrically by defining a basis $\{h_{jk}\}_{k=1}^{M_j}$ along each dimension j and representing h_j as a linear combination of these basis functions [Hastie et al., 2009, Chapter 5]

$$h_{\boldsymbol{\omega}}^{\text{add}}(\boldsymbol{x}) = \omega_0 + \sum_{j=1}^{d} \sum_{k=1}^{M_j} \omega_{jk} h_{jk}(x_j). \tag{2.30}$$

Several choices are available for choosing the bases $\{h_{jk}\}_{k=1}^{M_j}$ of feature j.

1. Linear input-output dependence $h_{j1}(x_j) = x_j$.
2. Piece-wise polynomials, for instance B-Splines [Hastie et al., 2009, Chapter 5].
3. One-hot-encoding of categorical variables $h_{jc}(x_j) = \mathbb{1}[x_j = c]$.

Now, letting

$$\boldsymbol{\omega} := [\omega_0, \underbrace{\omega_{11}, \omega_{12}, \ldots, \omega_{1M_1}}_{\boldsymbol{\omega}_1 \text{ feature 1}}, \underbrace{\omega_{21}, \omega_{22}, \ldots, \omega_{2M_2}}_{\boldsymbol{\omega}_2 \text{ feature 2}}, \ldots, \underbrace{\omega_{d1}, \omega_{d2}, \ldots, \omega_{dM_d}}_{\boldsymbol{\omega}_d \text{ feature } d}]. \tag{2.31}$$

be the parameter vector of the model and $\boldsymbol{H}$ be the $N \times (1 + \sum_{j=1}^{d} M_j)$ matrix whose ith row is

$$[1, \underbrace{h_{11}(x_1^{(i)}), h_{12}(x_1^{(i)}), \ldots, h_{1M_1}(x_1^{(i)})}_{\text{feature 1}}, \ldots, \underbrace{h_{d1}(x_d^{(i)}), h_{d2}(x_d^{(i)}), \ldots, h_{dM_d}(x_d^{(i)})}_{\text{feature d}}], \tag{2.32}$$

Solving Equation 2.4 simply involves fitting a linear model using $\boldsymbol{h}$ in place of $\boldsymbol{x}$ as the data matrix. Regularization can also be adapted to the choice of basis functions. For instance, when using B-splines, it is possible to penalize their first or second degree derivatives using a regularization of the form $\boldsymbol{\omega}^T \boldsymbol{A} \boldsymbol{\omega}$ [Wood, 2017]. These regularizations penalize the slopes of the h_j functions or their wiggliness.

Local Feature Attribution Because the output of an additive model is the sum of all shape functions $h_j(x_j)$ (modulo the intercept) it is very tempting to define the ante-hoc Local Feature Attribution

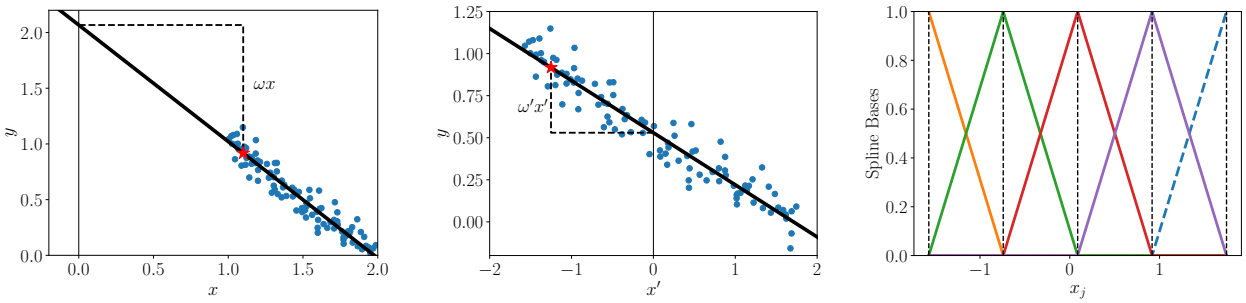
$$\phi_j^{\text{Naive}}(h^{\text{add}}, \boldsymbol{x}) := h_j(x_j), \quad \forall j = 1, 2, \ldots, d. \tag{2.33}$$

However, there are various issues with this formulation. For instance, when the shape functions are modeled non-parametrically, it is always possible to add a constant C to h_j and remove said constant from the intercept ω_0 without changing the function output $h^{\text{add}}(\boldsymbol{x})$. Consequently, the functional $\boldsymbol{\phi}^{\text{Naive}}$ is ill-posed for non-parametric methods. This can be circumvented by imposing constraints on h_j *e.g.* $\mathbb{E}_{\boldsymbol{x} \sim \mathcal{D}_{\mathcal{X}}}[h_j(\boldsymbol{x})] = 0$ [Lou et al., 2013]. Nonetheless, these constraints are still arbitrary and lead to very different values of $\phi_j^{\text{Naive}}(h^{\text{add}}, \boldsymbol{x})$.

The functional $\boldsymbol{\phi}^{\text{Naive}}$ also has issues for parametric additive models. As an example, a linear model has attributions $\phi_j^{\text{Naive}}(h_{\boldsymbol{\omega}}^{\text{lin}}, \boldsymbol{x}) = \omega_j x_j$ that are not invariant to affine mappings of the features $x_j' = a x_j + b$ with $a, b \in \mathbb{R}$. See Figure 2.5 (a) & (b) for an extreme example. Simply put, the attribution of a feature x_j measuring temperature would depend on whether its units are Celsius or Fahrenheit. This is obviously not desirable.

Moreover, $\boldsymbol{\phi}^{\text{Naive}}$ is problematic when a basis of function $\{h_{jk}\}_{k=1}^{M_j}$ is used to model h_j. The reason is that many bases (Splines and Indicator functions) sum up to one

$$\sum_{k=1}^{M_j} h_{jk}(x_j) = 1 \ \ \forall x_j \in \mathcal{X}_j. \tag{2.34}$$



(a) (b) (c)

Figure 2.5 Illustrating the failures of $\phi_j^{\text{Naive}}(h^{\text{add}}, \boldsymbol{x}) := h_j(x_j)$. (a) The local feature attribution of the linear model evaluated at the input x indicated by a red star is $\omega x = -1.15$. (b) After an affine transformation $x' = ax + b$ is performed to standardize the input feature, the feature attribution of the corresponding model on the same instance is now $\omega' x' = 0.39$. (c) The linear spline basis $\{h_{jk}\}_{k=1}^{5}$ plotted here can be used to model h_j as a piece-wise linear function. Nonetheless, to make the model identifiable when the intercept is present, one of the splines must be removed. Discarding the right-most basis (shown as a dashed curve) will result in null attributions $\phi_j^{\text{Naive}}(h^{\text{add}}, \boldsymbol{x}) = 0$ for any input with $x_j \geq 1.75$.

The resulting hypothesis space cannot be *identifiable* if an intercept ω_0 is also included. This is because any vertical shift in predictions can result from varying the intercept, or by adding the same constant to all weights. A common practice to make the parametric additive models identifiable is to remove one of the basis function [Wood, 2017], see Figure 2.5 (c). Yet, the choice of basis to remove is arbitrary and has strong implications on the attribution $\boldsymbol{\phi}^{\text{Naive}}$. For example, in Figure 2.5 (c), removing the right-most basis will yield $\phi_j^{\text{Naive}}(h^{\text{add}}, \boldsymbol{x}) = 0$ whenever $x_j \geq 1.75$.

Since $\boldsymbol{\phi}^{\text{Naive}}$ is sensitive to many arbitrary modeling choices that do not affect the *function* h^{add}, it cannot be used as a LFA. Then what attribution should be used? The solution is to think of Local Feature Attributions as being relative and not absolute. That is, instead of explaining the prediction $h(\boldsymbol{x})$, one should answer a *contrastive question* of the form : why is the model output $h(\boldsymbol{x})$ so high/low compared to a baseline value? The baseline value is commonly chosen to be the average model output $\mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{z})]$ over a distribution $\mathcal{B}$ called the *background*. At the heart of any contrastive question is a quantity called the Gap

$$G(h, \boldsymbol{x}, \mathcal{B}) := h(\boldsymbol{x}) - \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{z})], \tag{2.35}$$

and so asking a contrastive question amounts to measuring a Gap $G(h, \boldsymbol{x}) \neq 0$ and wondering why it is definitely positive or negative. Examples of contrastive questions include:

1. Why is individual $\boldsymbol{x}$ predicted to have higher-than-average risk of heart disease? Here, the Gap is positive and the background $\mathcal{B}$ is the distribution over the whole dataset.

2. Why is house $\boldsymbol{x}$ predicted to have a lower price than house $\boldsymbol{z}$? In that case, the Gap is negative and the background $\mathcal{B}$ is the Dirac measure $\delta_{\boldsymbol{z}}$.

Letting $\mathcal{B}_j$ be the marginal of $\mathcal{B}$ along feature j, the natural way to answer a contrastive question with an additive model h^{add} is the following functional

$$\phi_j^{\text{LFA}}(h^{\text{add}}, \boldsymbol{x}, \mathcal{B}) := h_j(x_j) - \mathop{\mathbb{E}}_{z_j \sim \mathcal{B}_j}[h_j(z_j)]. \tag{2.36}$$

When the model is linear, the functional simplifies to

$$\phi_j^{\text{LFA}}(h^{\text{lin}}_{\boldsymbol{\omega}}, \boldsymbol{x}, \mathcal{B}) := \omega_j \left(x_j - \mathop{\mathbb{E}}_{z_j \sim \mathcal{B}_j}[z_j]\right). \tag{2.37}$$

The crucial property of these LFAs is that they sum up to the gap

$$\sum_{j=1}^{d} \phi_j^{\text{LFA}}(h^{\text{add}}, \boldsymbol{x}, \mathcal{B}) = G(h^{\text{add}}, \boldsymbol{x}, \mathcal{B}). \tag{2.38}$$

Thus, a large positive attribution for feature j can be interpreted as: *the input component x_j increased the model output relative to the baseline*. The reverse interpretation is applicable to negative attributions.

The functional described in Equation 2.36 does not suffer from the drawbacks of $\boldsymbol{\phi}^{\text{Naive}}$ which were previously discussed. Indeed, $\boldsymbol{\phi}^{\text{LFA}}$ is invariant to the addition of arbitrary constants to the shape function h_j and the intercept. When the model is linear and Equation 2.37 holds, the $\boldsymbol{\phi}^{\text{LFA}}$ functional is invariant to affine transformation of the features x_j [Staniak and Biecek, 2018]. Therefore, the attribution for `temperature=10C` is the same as for `temperature=50F`. Finally, the $\boldsymbol{\phi}^{\text{LFA}}$ functional is invariant to the choice of which spline basis to remove when making the hypothesis space identifiable. All these invariance properties highlight the advantage of viewing local feature attributions as relative concepts instead of absolute ones.

Global Feature Importance Given that one is able to compute ante-hoc LFA explanations of additive models $\boldsymbol{\phi}^{\text{LFA}}(h^{\text{add}}, \boldsymbol{x}, \mathcal{B})$, Global Feature Importance can be obtained by aggregating them

$$\Phi_j^{\text{GFI},[p]}(h^{\text{add}}, \mathcal{B}) := \Big(\mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mathcal{B}} \Big[\, |h_j(x_j) - \mathop{\mathbb{E}}_{z_j \sim \mathcal{B}_j}[h_j(z_j)]|^p \, \Big] \Big)^{1/p} \tag{2.39}$$

using $p = 1, 2, \ldots$. Intuitively, these GFI metrics highlight how much feature x_j is used by the model on average when sampling data from $\mathcal{B}$. It is clear that some degree of information is lost from the averaging procedure and so GFI is not a substitute for LFAs. Global importance simply summarizes the information contained in a collection of local attributions.

Taking $p = 1$ is the default approach for EBMs [Nori et al., 2019]. Taking $p = 2$ is also a natural choice since the resulting functional measures the variance of the shape function h_j. Another argument in favor of $p = 2$ used in [Lou et al., 2013] is that, when a linear model is used, the corresponding global importance

$$\Phi_j^{\mathrm{GFI},[2]}(h_{\boldsymbol{\omega}}^{\mathrm{lin}}, \mathcal{B}) = |\omega_j| \ \mathrm{STD}_{x_j \sim \mathcal{B}_j}[x_j], \tag{2.40}$$

are the standardized weights of the linear model. Taking the limit $p \to \infty$

$$\Phi_j^{\mathrm{GFI},[\infty]}(h^{\mathrm{add}}, \mathcal{B}) := \sup_{\boldsymbol{x} \in \mathrm{supp}(\mathcal{B})} |\phi_j^{\mathrm{LFA}}(h^{\mathrm{add}}, \boldsymbol{x}, \mathcal{B})| \tag{2.41}$$

measures the largest possible attribution for this feature. This global importance is pertinent for fairness use-cases where a sensitive feature may have a large attribution on a small minority and no attribution on the majority. This could describe a model that discriminates heavily, but rarely. In such scenarios, the importance $\Phi_j^{\mathrm{GFI},[1]}$ and $\Phi_j^{\mathrm{GFI},[2]}$ might be small because of the imbalance in data but $\Phi_j^{\mathrm{GFI},[\infty]}$ would be large, indicating that there exists extreme cases where a sensitive feature is used to predict.

Explaining the Unfairness The use of feature attributions for fairness auditing is desirable in cases where the interest is on the direct/indirect impact of sensitive attributes on the output of the model. One such situation occurs in the context of Causal Fairness [Chikahara et al., 2021], where we need to ensure that disparities between subgroups are based on *real-world factors affecting a decision or recommendation*.

Importantly, when an additive model is employed, it is possible to attribute a blame to each input feature toward a given disparity measure

$$\Phi_j^{\mathrm{Fair}}(h^{\mathrm{add}}, \mathcal{F}, \mathcal{B}) := \mathop{\mathbb{E}}_{x_j \sim \mathcal{F}_j}[h_j(x_j)] - \mathop{\mathbb{E}}_{z_j \sim \mathcal{B}_j}[h_j(z_j)]. \tag{2.42}$$

These attributions sum up to the Fairness metric induced by the choice of distributions $\mathcal{F}$ and $\mathcal{B}$ (see Table 2.1). Therefore, passing Canada's future C-27 law might require demonstrating that the features causing a large Demography Parity (DP) represent an acceptable basis for decision. For instance, it would likely be illegal for a Bank to use an additive model that

attributes large importance $\Phi_j^{\text{Fair}}(h^{\text{add}}, \mathcal{F}, \mathcal{B})$ to features such as `nativeLanguage` (direct discrimination) and `geolocation` (indirect discrimination), instead of more acceptable risk factors such as `income` and `creditHistory`.

Ante-Hoc Additive Explanations

Additive models take the form

$$h^{\text{add}}(\boldsymbol{x}) = \omega_0 + \sum_{j=1}^{d} h_j(x_j), \tag{2.43}$$

where h_j only depends on x_j. These models provide ante-hoc additive explanations.

- The Local Feature Attribution

$$\phi_j^{\text{LFA}}(h^{\text{add}}, \boldsymbol{x}, \mathcal{B}) := h_j(x_j) - \mathop{\mathbb{E}}_{z_j \sim \mathcal{B}_j}[h_j(z_j)] \tag{2.44}$$

  explains the gap $G(h, \boldsymbol{x}, \mathcal{B}) = h(\boldsymbol{x}) - \mathbb{E}_{\boldsymbol{z} \sim \mathcal{B}}[h(\boldsymbol{z})]$.

- The Global Feature Importance

$$\Phi_j^{\text{GFI},[p]}(h^{\text{add}}, \mathcal{B}) := \mathop{\mathbb{E}}_{x_j \sim \mathcal{B}_j}\Big[\,|h_j(x_j) - \mathop{\mathbb{E}}_{z_j \sim \mathcal{B}_j}[h_j(z_j)]|^p\,\Big]^{1/p}. \tag{2.45}$$

  for $p = 1, 2, \ldots$ describes the variability in h^{add} attributed to feature j. Features that induce low variability are deemed unimportant to h.

- Features causing disparate outcomes betwen subgroups can be blamed via

$$\Phi_j^{\text{Fair}}(h^{\text{add}}, \mathcal{F}, \mathcal{B}) := \mathop{\mathbb{E}}_{x_j \sim \mathcal{F}_j}[h_j(x_j)] - \mathop{\mathbb{E}}_{z_j \sim \mathcal{B}_j}[h_j(z_j)] \tag{2.46}$$

  where, for each fairness metrics, $\mathcal{F}$ and $\mathcal{B}$ are provided in Table 2.1.

Following my Thesis Statement, I stress that above definitions for ante-hoc LFA, GFI, and Fairness Attribution will be considered *ground-truths* through-out the rest of this manuscript. Consequently, whenever the definitions above are generalized to models that are not additive, we will often make the sanity check : *does this new quantity fall back to ante-hoc explanations whenever the model happens to be additive?* Methods that fail this sanity check will be discarded, seeing as they cannot be aligned toward this ground-truth.

2.3.2 Post-hoc Explanations

Additive models are great: they can explain their predictions, explain their lack of fairness, and provide principled feature importance. So why isn't their use more widespread? The reason is that intelligibility is both a strength and a weakness. Remember that additive models are intelligible because the impact of varying feature x_j on the output $h(\boldsymbol{x})$ is the same regardless of the fixed value of other features $\boldsymbol{x}_{-j}$. This is a very strong restriction and, as a result, additive models are limited in the kinds of patterns they can fit.

As an illustrative example, imagine you wish to predict a student's grade at a difficult exam based on their `sleep_time` and `study_time`. Intuitively, the grade should increase monotonically with both features

$$\texttt{grade} = \omega_0 + \omega_1 \, \texttt{sleep\_time} + \omega_2 \, \texttt{study\_time}. \tag{2.47}$$

However, this additive model fails to account for the fact that the productivity of study hours depends on sleep quality. Ten hours of study for a well-rested student cannot be equivalent to ten hours of study for a student who stayed awake two days in a row. When the effect of varying a feature (study time) depends on the value of the other (sleep time), the two variables are said to interact.

Definition 2.3.1 (Feature Interactions)**.** *Feature j interacts with feature k if the effect of varying x_j on the response $h(\boldsymbol{x})$ depends on the fixed value of x_k, and vice versa.*

To make accurate grade predictions, an interaction should be included in the model

$$\texttt{grade} = \omega_0 + \omega_1 \, \texttt{sleep\_time} \times \texttt{study\_time}. \tag{2.48}$$

Here, the grade increase per additional study hour is proportional to the sleep amount. Simply put, students with zero hours of sleep cannot improve their grade by studying more. In opposition, students with enough sleep will see their grade improve with each additional hour of study. The more sleep, the greater the improvement.

The interaction had to be manually inserted in Equation 2.48 based on background knowledge about the task. This is not a scalable process. Therefore, many ML models are designed to automatically specify interactions terms based on the data. For instance, decision trees of depth-D are expressive enough to represent interactions between D features $\mathbb{1}[x_1 \leq \gamma_1] \times \ldots \times \mathbb{1}[x_D \leq \gamma_D]$. Models that train an ensemble of such as trees (*e.g.* Random Forests and Gradient Boosted Trees) can thus learn a large of set of complex feature interactions. In fact, these models are so expressive that many such interactions could be spurious (*i.e.* due to

noise). Competing approaches such as EBMs only include an interaction if it has statistical significance [Lou et al., 2013].

Models that contain feature interactions are no longer additive, so the ante-hoc explanations from Page 35 are not applicable. Nonetheless, because ante-hoc additive explanations are so intuitive, considerable efforts have been undertaken to provide post-hoc additive explanations of arbitrary functions h.

Post-Hoc Additive Explanations

The literature on post-hoc additive explanations can be categorized as

- **Partial Dependence** : Low-dimensional visualizations of how varying $\boldsymbol{x}$ impacts the model response $h(\boldsymbol{x})$.
- **Local Linearity** : Break interactions by explaining the model on small localities.
- **Importance by "removing" a feature** : Heuristically remove a feature and look at the impact on model output/performance.
- **Cooperative Game Theory** : Redistribute the effects of interactions to individual features via game theory.

In the sequel, we will overload notation $\boldsymbol{\phi}(h, \boldsymbol{x}, \boldsymbol{z}) \equiv \boldsymbol{\phi}(h, \boldsymbol{x}, \delta_{\boldsymbol{z}})$ when a Dirac measure is used.

Partial Dependence This line of work aims at providing low-dimensional graphical visualizations of how varying $\boldsymbol{x}$ impacts the model response $h(\boldsymbol{x})$. When h is additive, one only needs to visualize the shape function h_j to understand how perturbing x_j affects the output. However, in the presence of feature interactions, the impact of changing x_j may depend on the fixed values of other features. The solution proposed by Goldstein et al. [2015] is to illustrate how x_j impacts the response while fixing $\boldsymbol{x}_{-j}$ to some value $\boldsymbol{z}_{-j}$ chosen randomly from the dataset. This leads to the so-called Individual Conditional Expectations (ICE) curves

$$\phi_j^{\text{ICE}}(h, \boldsymbol{x}, \boldsymbol{z}) := h(x_j, \boldsymbol{z}_{-j}). \tag{2.49}$$

A distinct ICE curve exists for each data point $\boldsymbol{z}$ and Goldstein et al. suggest plotting them all simultaneously as a function of x_j. The motivation behind this mathematical formulation is that, when h is additive in feature j (*i.e.* $h(\boldsymbol{x}) = h_j(x_j) + h_{-j}(\boldsymbol{x}_{-j})$), all ICE curves ϕ_j^{ICE} are parallel to h_j. Thus, we are able to recover h_j up to a constant. Asserting parallelism of

curves requires comparing the curve's shapes (rather than their value), so the ICEs can be centered w.r.t x_j [Goldstein et al., 2015]

$$\phi_j^{\text{ICE-c}}(h, \boldsymbol{x}, \boldsymbol{z}) := h(x_j, \boldsymbol{z}_{-j}) - \mathop{\mathbb{E}}_{x_j \sim \mathcal{B}_j}\Big[h(x_j, \boldsymbol{z}_{-j})\Big] \tag{2.50}$$

before comparison. Differences between the centered ICE curves are indicative of feature interactions involving feature j. The drawback of ICEs is that they become unreadable when too many curves are plotted. Information overload can be reduced by averaging the ICE curves w.r.t the background sample $\boldsymbol{z}$ leading to single a curve called a Partial Dependence Plot [Friedman, 2001]

$$\phi_j^{\text{PDP}}(h, \boldsymbol{x}, \mathcal{B}) := \mathop{\mathbb{E}}_{\boldsymbol{z} \sim \mathcal{B}}[h(\boldsymbol{x}_j, \boldsymbol{z}_{-j})]. \tag{2.51}$$

By plotting $\phi_j^{\text{PDP}}(h, \boldsymbol{x}, \mathcal{B})$ as a function of x_j, one can understand how increasing/decreasing x_j influences the output signal *on average*. Because PDPs are the average of ICE curves, they share the ability to recover the shape functions when h is additive. Also, since PDPs alleviate the burden of plotting multiple curves, they can be calculated for feature subsets $S \subset [d]$ ($|S| \leq 3$)

$$\phi_S^{\text{PDP}}(h, \boldsymbol{x}, \mathcal{B}) := \mathop{\mathbb{E}}_{\boldsymbol{z} \sim \mathcal{B}}[h(\boldsymbol{x}_S, \boldsymbol{z}_{-S})] \tag{2.52}$$

and visualized to understand how groups of features impact the response. PDPs were later extended to Global Feature Importance by taking their variance [Greenwell et al., 2018]

$$\Phi_j^{\text{PDP}}(h, \mathcal{B}) := \mathbb{V}_{\boldsymbol{x} \sim \mathcal{B}}[\phi_j^{\text{PDP}}(h, \boldsymbol{x}, \mathcal{B})], \tag{2.53}$$

and to feature interactions quantification via the so-called H^2 statistics [Friedman and Popescu, 2008]. One such statistic measures interactions between features j and k

$$\Phi_{jk}^{\text{Inter}}(h, \mathcal{B}) := \mathbb{V}_{\boldsymbol{x} \sim \mathcal{B}}\Big[\, \phi_{\{j,k\}}^{\text{PDP}}(h, \boldsymbol{x}, \mathcal{B}) - \phi_j^{\text{PDP}}(h, \boldsymbol{x}, \mathcal{B}) - \phi_k^{\text{PDP}}(h, \boldsymbol{x}, \mathcal{B}) \,\Big], \tag{2.54}$$

while the other statistic quantifies interactions between feature j and all other features

$$\Phi_{j:}^{\text{Inter}}(h, \mathcal{B}) := \mathbb{V}_{\boldsymbol{x} \sim \mathcal{B}}\Big[\, h(\boldsymbol{x}) - \phi_j^{\text{PDP}}(h, \boldsymbol{x}, \mathcal{B}) - \phi_{-j}^{\text{PDP}}(h, \boldsymbol{x}, \mathcal{B}) \,\Big]. \tag{2.55}$$

By definition, PDPs evaluates the model at a synthetic point which is the concatenation of $\boldsymbol{z}_{-j}$ sampled from the marginal $\mathcal{B}_{-j}$ and x_j fixed. When feature j is correlated with other features, this synthetic point may lie out of the data distribution. For example, if one feature measures `age` and the other `salary`, fixing `age=15` and sampling `salary` from its marginal will create unrealistic instances of very rich teenagers. Since h was trained on the

data distribution, its behavior outside said the data support can be unpredictable. To avoid extrapolation it is possible to replace the $\mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_j, \boldsymbol{z}_{-j})]$ with a conditional expectation $\mathbb{E}_{\boldsymbol{x}\sim\mathcal{B}}[h(\boldsymbol{x})|x_j]$, but the resulting plot is no-longer guaranteed to recover the h_j functions when h is additive [Friedman, 2001]. This is because the impact of feature j "leaks into" other features with which it is correlated. The search for a compromise between recovering additive structure and not breaking feature correlations has led to the Accumulated Local Effect [Apley and Zhu, 2020]. This technique takes conditional expectations of the model **gradient** (instead of model values) and accumulates them up to the point x_j of interest

$$\phi_j^{\text{ALE}}(h, \boldsymbol{x}, \mathcal{B}) = \int_{x_{j,\min}}^{x_j} \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}} \left[\frac{\partial h}{\partial z_j}(\boldsymbol{z}) \middle| z_j \right] dz_j. \tag{2.56}$$

Like ICE/PDP, this functional is able to recover additive structure but is also drastically reduces extrapolation. Its empirical estimation is more complicated however : conditional expectations are estimated by binning along x_j and partial derivatives are estimated with finite differences [Apley and Zhu, 2020].

Local Linearity For a sufficiently smooth model h, the following Taylor decomposition holds

$$h(\boldsymbol{x}) = h(\boldsymbol{z}) + \nabla h(\boldsymbol{z})^T(\boldsymbol{x} - \boldsymbol{z}) + o(\|\boldsymbol{x} - \boldsymbol{z}\|). \tag{2.57}$$

So, in a small locality around $\boldsymbol{z}$, the model $h(\boldsymbol{x})$ can be approximated by a linear model that contains *no interactions*. This fundamental property has encouraged a large body of literature focusing on local linear approximations of h as a mean of breaking feature interactions. The first method was to report output gradient [Simonyan et al., 2013]

$$\boldsymbol{\phi}^{\text{Grad}}(h, \boldsymbol{x}) := \nabla h(\boldsymbol{x}). \tag{2.58}$$

Although the original work of Simonyan et al. [2013] only investigated the gradients of Deep Neural Networks, gradients are available for other hypothesis classes *e.g.* kernel methods with smooth kernels. In image classification tasks, the saliency maps obtained by taking the gradient of a DNN output with respect to the input pixels are often very noisy. A solution to this problem is SmoothGrad (SG), which computes the average gradient according to a distribution $\mathcal{B}$ in the neighborhood of the instance $\boldsymbol{x}$ [Smilkov et al., 2017]

$$\boldsymbol{\phi}^{\text{SG}}(h, \boldsymbol{x}, \mathcal{B}) := \mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[\nabla h(\boldsymbol{z})]. \tag{2.59}$$

Local linear approximations of model outputs have also been proposed for non-smooth models. The well-established LIME (Local Interpretable Model-agnostic Explanations) [Ribeiro et al., 2016] is one such method. LIME samples synthetic inputs $Z = \{\boldsymbol{z}^{(i)}\}_{i=1}^{M}$ from the distribution $\mathcal{B}$ over the neighborhood of $\boldsymbol{x}$, evaluates the black box at those points $h(Z)$, and finally fits a linear model on the regression task $(Z, h(Z))$

$$(\omega_0, \underbrace{\omega_1, \omega_2, \ldots, \omega_d}_{\boldsymbol{\phi}^{\text{LIME}}(h,\boldsymbol{x},\mathcal{B})}) = \underset{\boldsymbol{\omega}\in\mathbb{R}^{d+1}}{\operatorname{argmin}} \mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}\left[\left(h(\boldsymbol{z}) - \sum_{j=1}^{d}\omega_j z_j - \omega_0\right)^2\right]. \tag{2.60}$$

This surrogate model $h_{\boldsymbol{\omega}}^{\text{lin}}$ is the best linear approximation of h in the neighborhood of the point of interest $\boldsymbol{x}$. Note that the intercept ω_0 is rejected from the local attribution since it does not encode interesting information about the behavior of h near $\boldsymbol{x}$. SG and LIME have been proven to be equivalent when features are continuous, $\mathcal{B}$ is an isotropic Gaussian centered at $\boldsymbol{x}$, and h is smooth [Agarwal et al., 2021]. Thus, the following discussions apply to both methods, although we will only explicitly refer to LIME.

To determine if LIME is a generalization of the ante-hoc additive models explanations introduced in Section 2.3.1, one needs to make two verifications. 1) Does LIME answer a contrastive question? 2) If so, does LIME return Equation 2.44 when the underlying model is additive? We shall see that the answers to both questions depend on the domain $\mathcal{X}$.

Let $\mathcal{X} = \{0,1\}^d$ be a binary domain, which is used when LIME is applied to text and image data and values $x_j \in \{0,1\}$ represent the absence and presence of a word/super-pixel in the original text/image. In this representation, the original text/image to explain is the point $\boldsymbol{x} = \mathbf{1}$, where all words/super-pixels are present. Now, let $\mathcal{B}$ be a distribution over $\mathcal{X}$ such that $\mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[z_j] = p$ for all $j = 1, 2 \ldots, d$ and which involves randomly shutting down words/super-pixels with probability $(1-p)$. Then, the feature attributions of the local linear model $h_{\boldsymbol{\omega}}^{\text{lin}}$ fitted in Equation 2.60 are

$$\phi_j^{\text{LFA}}(h_{\omega}^{\text{lin}}, \mathbf{1}, \mathcal{B}) = \omega_j(1 - \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[z_j]) = \omega_j\,(1-p). \tag{2.61}$$

The weights ω_j returned by LIME are *proportional* to a local feature attribution answering a contrastive question on $h_{\boldsymbol{\omega}}^{\text{lin}}$. Consequently, up to a multiplicative constant, LIME does answer a contrastive question. Additionally, note that an additive model on $\mathcal{X} = \{0,1\}^d$ can be perfectly approximated by a linear one. So, the attribution of the surrogate linear model (cf. Equation 2.61) coincides with the attributions of the underlying additive model (cf. Equation 2.44).

Let $\mathcal{X} \subseteq \mathbb{R}^d$ be a real-valued domain, the default background choice in LIME is an isotropic Gaussian centered at $\boldsymbol{x}$: $\mathcal{B} = \mathcal{N}(\boldsymbol{x}, \sigma^2 \boldsymbol{I})$. In this setting, the coefficient ω_j returned by LIME does not by itself answer a contrastive question. The coefficient instead tells one the effect of increasing x_j on the local linear surrogate. Still, is it possible to answer a contrastive question by leveraging the linear surrogate $h^{\text{lin}}_{\boldsymbol{\omega}} \approx h$? The resulting Gap is

$$\begin{aligned} G(h^{\text{lin}}_{\boldsymbol{\omega}}, \boldsymbol{x}, \mathcal{B}) &= h^{\text{lin}}_{\boldsymbol{\omega}}(\boldsymbol{x}) - \mathop{\mathbb{E}}_{\boldsymbol{z} \sim \mathcal{B}}[h^{\text{lin}}_{\boldsymbol{\omega}}(\boldsymbol{z})] \\ &= \sum_{j=1}^{d} \omega_j x_j + \omega_0 - \sum_{j=1}^{d} \omega_j \mathop{\mathbb{E}}_{\boldsymbol{z} \sim \mathcal{B}}[z_i] - \omega_0 \\ &= \sum_{j=1}^{d} \omega_j x_j - \sum_{j=1}^{d} \omega_j x_j = 0. \end{aligned} \tag{2.62}$$

Crucially, local linear models with backgrounds centered as $\boldsymbol{x}$ do not have an interesting gap to explain at $\boldsymbol{x}$. So, the default configurations of LIME cannot answer a useful contrastive question. Solutions to answer contrastive questions with LIME would require either allowing for non-symmetric neighborhood distributions, or fitting an additive local approximation of h in place of a linear one. Both extensions will be discussed in Chapter 3.

We just saw that, when $\mathcal{X} \subseteq \mathbb{R}^d$, local linear approximations of h may or may not answer a useful contrastive question depending on the choice of background distribution. Yet, there exists an alternative way of leveraging local linearity while providing useful attributions for any background. To see how, one must take a closer look at Equation 2.57. If $\boldsymbol{x}$ is close to $\boldsymbol{z}$ then the attribution of feature j towards the gap $h(\boldsymbol{x}) - h(\boldsymbol{z})$ is approximately $(x_j - z_j) \frac{\partial h}{\partial x_j}(\boldsymbol{z})$. When $\boldsymbol{x}$ is far from $\boldsymbol{z}$, a linear path could be drawn between the two points and the local feature attributions of infinitesimal steps along this path could be accumulated. This is the intuition behind the Integrated/Expected Gradient [Erion et al., 2021, Sundararajan et al., 2017]. The general definition of EG is

$$\phi^{\text{EG}}_j(h, \boldsymbol{x}, \mathcal{B}) = \mathop{\mathbb{E}}_{\substack{\boldsymbol{z} \sim \mathcal{B} \\ t \sim U(0,1)}} \left[(x_j - z_j) \frac{\partial h}{\partial x_j}((1-t)\boldsymbol{z} + t\boldsymbol{x}) \right]. \tag{2.63}$$

By averaging gradients along linear paths between background samples and $\boldsymbol{x}$, one obtains attributions that sum up to the gap and that fall back to Equation 2.44 when h is additive. When the background distribution degenerates to a single atom at input $\boldsymbol{z}$ ($\mathcal{B} = \delta_{\boldsymbol{z}}$), the Expected Gradient is also called the Integrated Gradient. For the derivation of Equation 2.63, we refer to Appendix A.1.

Global Importance by "removing" a feature Permutation Feature Importance (PFI) [Breiman, 2001a] was introduced as a Global Feature Importance technique for Random Forest although its definition is model-agnostic. The general idea is to *replace feature k with noise and report the impact on model performance.* The theoretical definition is [Gregorutti et al., 2017]

$$\Phi_k^{\text{PFI-O}}(h, \mathcal{D}) := \mathop{\mathbb{E}}_{\substack{(\boldsymbol{x},y)\sim\mathcal{D} \\ (\boldsymbol{z},y')\sim\mathcal{D}}} \Big[\Big(h(\boldsymbol{x}_{-k}, \boldsymbol{z}_k) - y \Big)^2 \Big] - \mathop{\mathbb{E}}_{(\boldsymbol{x},y)\sim\mathcal{D}} \Big[\Big(h(\boldsymbol{x}) - y \Big)^2 \Big], \tag{2.64}$$

which compares the model performance on the original data and on synthetic data where feature k is replaced by a sample from its marginal. Intuitively, by replacing a feature with noise, its relationship with the target is broken and hence a large drop in performance is interpreted as evidence that the model relies strongly on this feature to generalize. The nomenclature *Permutation* Feature Importance is not immediately clear by looking at Equation 2.64. Although the functional itself does not involve any permutation, we will see that its empirical estimates often do.

Speaking of empirical estimates, the first term of the PFI can be estimated using a V-statistic, which are consistent estimators [Fisher et al., 2019]

$$\frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} \Big(h(\boldsymbol{x}_{-k}^{(i)}, \boldsymbol{x}_k^{(j)}) - y^{(i)} \Big)^2 \xrightarrow{p} \mathop{\mathbb{E}}_{\substack{(\boldsymbol{x},y)\sim\mathcal{D} \\ (\boldsymbol{z},y')\sim\mathcal{D}}} \Big[\Big(h(\boldsymbol{x}_{-k}, \boldsymbol{z}_k) - y \Big)^2 \Big]. \tag{2.65}$$

Since this V-statistic requires evaluating the model at N^2 points, it was proposed to estimate it using random permutations. Letting π be a permutation of $[d]$, the idea is to switch the k^{th} feature of the i^{th} instance with the $\pi[i]^{\text{th}}$ instance. This yields [Fisher et al., 2019]

$$\frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} \Big(h(\boldsymbol{x}_{-k}^{(i)}, \boldsymbol{x}_k^{(j)}) - y^{(i)} \Big)^2 = \frac{1}{d!} \sum_{\text{perms } \pi} \frac{1}{N} \sum_{i=1}^{N} \Big(h(\boldsymbol{x}_{-k}^{(i)}, \boldsymbol{x}_k^{(\pi[i])}) - y^{(i)} \Big)^2 \tag{2.66}$$

Approximating Equation 2.66 by sampling M random permutations provides an consistent estimate of the V-statistic, which is itself a consistent estimate of the PFI (cf. Equation 2.65). This is the current Monte-Carlo scheme behind the `permutation_importance`[6] function of the Scikit-Learn library. Note that two convergences are involved when estimating PFI, 1) According to Equation 2.66, as more permutations are sampled, one can better approximate the V-statistic. 2) According to Equation 2.65, as $N \to \infty$, the V-statistic converges to the

[6] `https://scikit-learn.org/stable/modules/generated/sklearn.inspection.permutation_importance.html#sklearn.inspection.permutation_importance`

population-level functional.

The original definition of PFI involves the label y, yet it is possible to express it in a form that only involves the model h. Under the following assumption : $h(\boldsymbol{x}) - y = \epsilon$ where ϵ is a random variable that is independent of $\boldsymbol{x}$ and such that $\mathbb{E}[\epsilon] = 0$ and $\mathbb{E}[\epsilon^2] = \sigma^2$, we get [Gregorutti et al., 2017]

$$\Phi_k^{\text{PFI}}(h, \mathcal{B}) = \mathop{\mathbb{E}}_{\substack{\boldsymbol{x}\sim\mathcal{B} \\ \boldsymbol{z}\sim\mathcal{B}}} \Big[\left(h(\boldsymbol{x}) - h(\boldsymbol{x}_{-k}, \boldsymbol{z}_k)\right)^2 \Big]. \tag{2.67}$$

This form also simplifies under the assumption that h is additive

$$\Phi_k^{\text{PFI}}(h^{\text{add}}, \mathcal{B}) = 2\, \mathbb{V}_{\boldsymbol{x}\sim\mathcal{B}}[h_k(\boldsymbol{x})\Big]. \tag{2.68}$$

It is reassuring to observe that the PFI falls back to a well-known measure of importance when the model is additive (cf. Equation 2.45). However, when the model contains interactions and features are correlated, issues can arise with the PFI. Hooker et al. [2021] present a synthetic regression experiment where features have unit variance, the target is generated by a linear model $y = x_1 + x_2 + x_3 + \epsilon$, and fitted with a Random Forest h^{rf}. Since the ground-truth is symmetric w.r.t all features, an ideal feature importance $\boldsymbol{\Phi}(h^{\text{rf}}, \mathcal{B})$ should return the same value for any x_k. Yet, Hooker et al. show that the PFI of correlated features inflate compared to uncorrelated ones. Their explanation for this phenomenon is that, like the PDP, PFI "removes" feature x_k by replacing its value with a sample z_k from the marginal. When input features are correlated, this replacement will evaluate the model on a point $h(\boldsymbol{x}_{-k}, \boldsymbol{z}_k)$ that lies outside the support of the data.

Many PFI corrections have been proposed to address the extrapolation occurring when features are correlated. First, [Bénard et al., 2021] define the Marginal Sobol Index

$$\Phi_k^{\text{Marginal-Sobol}}(h, \mathcal{B}) := \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}} \Big[\mathop{\mathbb{V}}_{z\sim\mathcal{B}}[h(\boldsymbol{x}_{-k}, \boldsymbol{z}_k)] \Big], \tag{2.69}$$

an importance measure that is invariant to the correlation between feature k and remaining features. Appendix 1 of [Bénard et al., 2021] presents a toy example with feature correlations ρ to illustrate that the Marginal Sobol Index in unaffected by increases in ρ while the Permutation Feature Importance inflates with correlations.

Second, various work advocates *retraining* the model on the synthetic data where feature k was broken. Breaking feature k can be done by permuting it [Hooker et al., 2021] or completely removing the k^{th} column of the data matrix $\boldsymbol{X}$ [Williamson et al., 2021]. In either case, given that the number of data $N \to \infty$, that $\mathcal{H}$ has enough capacity to represent

any conditional expectation $\mathbb{E}[y|\boldsymbol{x}_S]$, and that the squared loss is used, it was proven that all these methods converge to the so-called Total Sobol Index [Hooker et al., 2021, Theorem 2]

$$\Phi_k^{\text{Total-Sobol}}(h, \mathcal{B}) = \mathop{\mathbb{E}}_{\boldsymbol{x}_{-k}} \Big[\mathop{\mathbb{V}}_{x_k}[h(\boldsymbol{x})|\boldsymbol{x}_{-k}] \Big]. \tag{2.70}$$

This functional measures the variance of h w.r.t x_k given that we are conditioning on $\boldsymbol{x}_{-k}$. The conditional variance is hard to estimate for general data distributions and model, but efficient and consistent estimates have been proposed when h^{rf} is a Random Forest [Bénard et al., 2021].

Third, a family of PFI corrections learn a partition $(\Omega_{-k}^{[1]}, \dots, \Omega_{-k}^{[M]})$ of $\mathcal{X}_{-k}$ and let $\mathcal{B}_{\Omega^{[m]}}$ be the restriction of $\mathcal{B}$ to the region $\Omega^{[m]} := \Omega_{-k}^{[m]} \times]-\infty, \infty[$. The resulting functional

$$\Phi_k^{\text{CPFI}}(h, \mathcal{B}) := \sum_{m=1}^{M} \mathcal{B}(\Omega^{[m]}) \times \Phi_k^{\text{PFI}}(h, \mathcal{B}_{\Omega^{[m]}}), \tag{2.71}$$

is called *Conditional* PFI since feature x_k is now broken by replacing it with noise conditioned on $\boldsymbol{x}_{-k} \in \Omega_{-k}^{[m]}$. When h is a tree ensemble, Strobl et al. [2008] suggest leveraging the node splits involving features in $[d] \setminus \{k\}$ to define the partition. Molnar et al. [2023] instead advocate training a Regression tree to predict x_k given $\boldsymbol{x}_{-k}$ and defining the partition of $\mathcal{X}_{-k}$ as the leaves of said tree.

Cooperative Game Theory Coalitional Game Theory studies situations where d players collaborate toward a common outcome. Formally, letting $[d] := \{1, 2, \dots, d\}$ be the set of all d players, this theory is concerned with games $\nu : 2^{[d]} \to \mathbb{R}$, which describe the collective payoff that any set of players can gain by forming a coalition. In this context, the challenge is to assign a credit (score) $\phi_j(\nu) \in \mathbb{R}$ to each player $j \in [d]$ based on their contribution toward the total value $\nu([d]) - \nu(\emptyset)$ (the collective gain when all players join, taking away the gain when no one joins). Namely, such scores should satisfy:

$$\sum_{j=1}^{d} \phi_j(\nu) = \nu([d]) - \nu(\emptyset). \tag{2.72}$$

The intuition behind this Equation is to think of the outcomes $\nu([d])$ and $\nu(\emptyset)$ as the profit of a company involving all employees and no employee. In that case, the score $\phi_j(\nu)$ can be seen as the salary of employee j based on their productivity. Because players could interact (the effect of player i might depend on the presence/absence of player j), the scores ϕ are non-unique. Cooperative Game Theory tackles this non-unicity by imposing additional constraints on ϕ, for instance, the Dummy, Symmetry, and Linearity properties.

In cooperative games, a player j is called a *dummy* if $\forall S \subseteq [d] \setminus \{j\}\ \ \nu(S \cup \{j\}) = \nu(S)$. Dummy players basically never contribute to the game. A desirable property of a score is that dummy players should not be given any credit, *i.e.* employees that do not work do not earn a salary,

$$\Big[\forall S \subseteq [d] \setminus \{j\}\ \ \nu(S \cup \{j\}) = \nu(S)\Big] \Rightarrow \phi_j(\nu) = 0. \tag{2.73}$$

Another property is symmetry which states that players with equivalent roles in the game should have the same score *i.e.* employees with the same productivity should have the same salary

$$\Big[\forall S \subseteq [d] \setminus \{j,k\} \quad \nu(S \cup \{j\}) = \nu(S \cup \{k\})\Big] \Rightarrow \phi_j(\nu) = \phi_k(\nu). \tag{2.74}$$

The last desirable property is that scores are linear w.r.t games, *i.e.* letting $\boldsymbol{\phi}(\nu) := [\phi_1(\nu), \phi_2(\nu), \ldots, \phi_d(\nu)]^T$, then for all games $\nu, \mu : 2^{[d]} \to \mathbb{R}$ and all $\alpha \in \mathbb{R}$, we want

$$\boldsymbol{\phi}(\nu + \mu) = \boldsymbol{\phi}(\nu) + \boldsymbol{\phi}(\mu). \tag{2.75}$$

$$\boldsymbol{\phi}(\alpha\mu) = \alpha\boldsymbol{\phi}(\mu). \tag{2.76}$$

The reasoning behind this property is a bit more involved than the previous two. For Equation 2.75, imagine that the two games ν and μ represent two different companies and that employee j works at both. In that case, the salary of employee j from company ν should not be affected by their performance in the company μ and vice versa. Importantly, the total salary of employee j ideally should be the sum of the salaries at both companies. For Equation 2.76, imagine that the company is subject to a lawsuit in the end of a quarter which results in a sudden reduction in profits by a factor of α. Then, given that each employee had fixed productivity during this quarter, it is only fair that all their salaries diminish by the same factor α.

In his seminal work, Lloyd Shapley has proven the existence of a **unique** score function that respects Equations 2.72-2.76: the so-called Shapley values.

Definition 2.3.2 (Shapley Values [Shapley, 1953])**.** *Given a set $[d] := \{1, 2, \ldots, d\}$ of players and a cooperative game $\nu : 2^{[d]} \to \mathbb{R}$, the Shapley values are defined as*

$$\phi_j^{SHAP}(\nu) = \sum_{S \subseteq [d] \setminus \{j\}} W(|S|, d)\Big(\nu(S \cup \{j\}) - \nu(S)\Big) \qquad j \in [d], \tag{2.77}$$

where

$$W(|S|, d) := \frac{|S|!(d - |S| - 1)!}{d!} \tag{2.78}$$

is the proportion of all $d!$ orderings of $[d]$ such that $\pi_{:j} = S$.

Intuitively, the credit $\phi_j(\nu)$ is the weighted average contribution of adding them to coalitions S that excludes them. The beauty in Equation 2.77 is that it provides a principled way of attributing credit to individual players despite the presence of interactions in the game ν. For this reason, Shapley values have become extremely popular in the XAI literature and much effort as been made to establish the correct coalitional game ν to explain a model locally and globally.

For explaining a model locally, any game $\nu_{h,\boldsymbol{x},\mathcal{B}}$ such that $\nu_{h,\boldsymbol{x},\mathcal{B}}([d]) = h(\boldsymbol{x})$ and $\nu_{h,\boldsymbol{x},\mathcal{B}}(\emptyset) = \mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{z})]$ results in Shapley values that sum up to the Gap $h(\boldsymbol{x}) - \mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{z})]$ and could be seen as explaining the prediction [Merrick and Taly, 2020, Sundararajan and Najmi, 2020]. The multiplicity of such possible games has lead to a variety of Shapley values attributions in the literature.

Definition 2.3.3 (The Interventional Game [Janzing et al., 2020])**.** *Comparing the model predictions at $\boldsymbol{x}$ and over $\mathcal{B}$ can be done by computing the Shapley values of the following game:*

$$\nu^{int}_{h,\boldsymbol{x},\mathcal{B}}(S) := \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_S, \boldsymbol{z}_{-S})]. \tag{2.79}$$

In this game, the score of a coalition S of features consist in the expected prediction when features in S are taken from the point to explain $\boldsymbol{x}$ while others are sampled from the marginal $\mathcal{B}_{-S}$. The Shapley values of the resulting game are called the *Interventional Shapley Values*

$$\boldsymbol{\phi}^{\text{SHAP-int}}(h, \boldsymbol{x}, \mathcal{B}) \equiv \boldsymbol{\phi}^{\text{SHAP}}(\nu^{\text{int}}_{h,\boldsymbol{x},\mathcal{B}}). \tag{2.80}$$

An alternative perspective on Interventional Shapley Values is to define the baseline game $\nu^{\text{baseline}}_{h,\boldsymbol{x},\boldsymbol{z}}(S) := h(\boldsymbol{x}_S, \boldsymbol{z}_{-S})$ where features are removed by fixing their value to $\boldsymbol{z}_{-S}$. Then, by linearity, the Interventional Shapley Values are

$$\boldsymbol{\phi}^{\text{SHAP-int}}(h, \boldsymbol{x}, \mathcal{B}) = \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[\boldsymbol{\phi}^{\text{SHAP}}(\nu^{\text{baseline}}_{h,\boldsymbol{x},\boldsymbol{z}})]. \tag{2.81}$$

This Equation will prove useful when estimating Interventional Shapley Values. Like other feature attribution methods that remove features by sampling from the marginal irrespective of the value of $\boldsymbol{x}_S$ (*e.g.* PDP and PFI), the Interventional Shapley Values might extrapolate the data. It was discussed previously that ALE and Conditional PFI were later introduced as modifications of PDP/PFI to avoid extrapolation. Similarly, an alternative cooperative game has been defined with the aim of staying faithful to the data distribution.

Definition 2.3.4 (The Observational Game [Frye et al., 2020])**.** *Comparing the model predictions at $\boldsymbol{x}$ and over $\mathcal{B}$ can be done by computing the Shapley values of the following game:*

$$\nu^{obs}_{h,\boldsymbol{x},\mathcal{B}}(S) := \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{z})|\boldsymbol{z}_S = \boldsymbol{x}_S]. \tag{2.82}$$

The corresponding attributions are called the *Observational Shapley Values*

$$\boldsymbol{\phi}^{\text{SHAP-obs}}(h, \boldsymbol{x}, \mathcal{B}) \equiv \boldsymbol{\phi}^{\text{SHAP}}(\nu^{\text{obs}}_{h,\boldsymbol{x},\mathcal{B}}). \tag{2.83}$$

Unlike the Interventional Shapley Values, the observational ones do not extrapolate the data since their computation relies on *conditional* expectation given that $\boldsymbol{x}_S$ is fixed. Still, beyond the difficulty of computing conditional expectations on real-world data, the main complication with Observational Shapley Values is that they do not fall back to Equation 2.44 when h is additive. Even worst, they can even give a non-null attribution to a feature that is not referenced by the model h, as highlighted by the following proposition.

Proposition 2.3.1. *Let $d = 2$, $h^{lin}_{\boldsymbol{\omega}}$ be a linear model, and $\mathcal{B} = \mathcal{N}(\mathbf{0}, (1-\rho)\boldsymbol{I} + \rho\mathbf{1})$ be a distribution over two correlated Gaussian variables. Computing Shapley values of the Interventional and Observational games yields the feature attributions*

$$\begin{aligned}
\boldsymbol{\phi}^{SHAP\text{-}int}(h^{lin}_{\boldsymbol{\omega}}, \boldsymbol{x}, \mathcal{B}) &= [\omega_1 x_1, \omega_2 x_2]^T \\
\boldsymbol{\phi}^{SHAP\text{-}obs}(h^{lin}_{\boldsymbol{\omega}}, \boldsymbol{x}, \mathcal{B}) &= [\omega_1 x_1 + \frac{\rho}{2}(\omega_2 x_1 - \omega_1 x_2), \omega_2 x_2 + \frac{\rho}{2}(\omega_1 x_2 - \omega_2 x_1)]^T.
\end{aligned} \tag{2.84}$$

The proof is presented in Appendix A.2.

As we can see, when x_1 and x_2 are correlated, a non-null attribution can be given to x_j even though $\omega_j = 0$. This failure of Observational Shapley Values is well-known in the XAI and Sensitivity Analysis communities: the former calls it the "Failure of the Dummy" [Sundararajan and Najmi, 2020] while the latter refers to it as the "Shapley's joke" [Herin et al., 2022]. Seeing as Interventional and Observational games have their respective strengths and weaknesses, subsequent work has argued that none is inherently better than the other. Supposedly Interventional games explain the model, Observational ones explain the data, and practitioners should pick which ever better suits their use-case [Chen et al., 2020]. There have also been attempts to get the best of both paradigms by defining new coalitional games ν which reweight the interventional game based on a density estimate of the data [Yeh et al., 2022] or a 0/1 extrapolation detector [Taufiq et al., 2023]. These alternative game definitions

will not be investigated further in this manuscript because they considerably complicate the interpretation of the game.

For the remainder of the Thesis, we shall employ Interventional Shapley Values because, in light of Equation 2.84, they fall back to ante-hoc explanations when h is additive. The issues on how $\mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_S, \boldsymbol{z}_{-S})]$ causes data extrapolations will be tackled with *Regional Backgrounds* in Chapter 6.

Once the right coalitional game has been chosen, it remains to compute Shapley values using Equation 2.77. Since it involves a summation over all subsets $S \subseteq [d] \setminus \{j\}$ and that there are exponentially many such sets, the exact computation of Shapley values is NP-Hard in the general case. Yet, if the model architecture is directly accessible (*i.e.* the implementation is no longer model-agnostic), it is possible to compute Shapley Values more efficiently. Notably, polynomial algorithms have been proposed when h is an ensemble of decision trees [Lundberg et al., 2018, 2020].

For model-agnostic implementations with numerous features d, the Shapley values must be estimated. The most common estimation scheme is a Monte-Carlo algorithm that relies on a reformulation of Equation 2.77. Let π be a permutation of $[d]$, $\pi[j]$ be the position of the feature j in π, and $\pi_{:j} = \{k \in [d] : \pi[k] < \pi[j]\}$, the Shapley values can be reformulated as

$$\phi_j^{\text{SHAP-int}}(h, \boldsymbol{x}, \mathcal{B}) := \mathop{\mathbb{E}}_{\substack{\pi\sim\Omega \\ \boldsymbol{z}\sim\mathcal{B}}} \left[\nu^{\text{baseline}}_{h,\boldsymbol{x},\boldsymbol{z}}(\pi_{:j} \cup \{j\}) - \nu^{\text{baseline}}_{h,\boldsymbol{x},\boldsymbol{z}}(\pi_{:j})\right], \tag{2.85}$$

where Ω is the uniform distribution over all $d!$ permutations of $[d]$. To see that Equations 2.85 and 2.77 are equivalent, note that, for any set $S \subseteq [d]$ there are $|S|!(d - |S| - 1)$ permutations for which $\pi_{:j} = S$. These redundant permutations can be aggregated leading to a weighted summation over subsets $S \subseteq [d] \setminus \{j\}$ rather than a summation over $d!$ permutations. Equation 2.85 suggests a straight-forward Monte-Carlo estimate for Shapley values: 1) sample a random permutations $\pi \sim \Omega$, 2) sample a background point $\boldsymbol{z} \sim \mathcal{B}$, 3) compute $\nu^{\text{baseline}}_{h,\boldsymbol{x},\boldsymbol{z}}(\pi_{:j} \cup \{j\}) - \nu^{\text{baseline}}_{h,\boldsymbol{x},\boldsymbol{z}}(\pi_{:j})$, 4) repeat M times and average [Štrumbelj and Kononenko, 2014].

The permutation formulation of the Shapley values has also inspired an alternative post-hoc attribution method called Breakdown [Staniak and Biecek, 2018]

$$\phi_j^{\text{BD-}\pi}(h, \boldsymbol{x}, \mathcal{B}) := \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}} \left[\nu^{\text{baseline}}_{h,\boldsymbol{x},\boldsymbol{z}}(\pi_{:j} \cup \{j\}) - \nu^{\text{baseline}}_{h,\boldsymbol{x},\boldsymbol{z}}(\pi_{:j})\right]. \tag{2.86}$$

The difference between Breakdown and SHAP is that Breakdown leverages a single permu-

tation π while SHAP averages over all. The permutation π used by Breakdown can either be chosen by the user, or determined heuristically. The current heuristics are greedy and differ between the original Breakdown implementation [Staniak and Biecek, 2018] and the `Dalex` Python package [Baniecki et al., 2021]. For this reason, we will not describe these heuristics in detail and we prefer to let π be a hyperparameter of the post-hoc explainer. An important property of Breakdown is that it can detect feature interactions. Indeed, if there are two permutations π, π' such that $\phi_j^{\text{BD-}\pi}(h, \boldsymbol{x}, \mathcal{B}) \neq \phi_j^{\text{BD-}\pi'}(h, \boldsymbol{x}, \mathcal{B})$, then the feature j must interact with some other feature [Gosiewska and Biecek, 2019].

We have just described how coalitional game theory can provide Local Feature Attributions. But what about Global Feature Importance? A first possibility is to aggregate local Shapley Values into global scores

$$\Phi_j^{\text{SHAP},[\text{p}]}(h, \mathcal{B}) := \Big[\mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mathcal{B}} \big[\, |\phi_j^{\text{SHAP-int}}(h, \boldsymbol{x}, \mathcal{B})|^p \, \big] \Big]^{1/p} \tag{2.87}$$

Taking $p = 1$ is the default approach of the `SHAP` library [Lundberg and Lee, 2017], although other values of p could be reasonable. Note that Equation 2.87 falls back to ante-hoc GFIs (cf. Equation 2.45) whenever h^{add} is additive.

Another possibility is to define a cooperative game $\nu(S)$ that returns the test performance of the fake model $\mathbb{E}_{\boldsymbol{z} \sim \mathcal{B}}[h(\boldsymbol{x}_S, \boldsymbol{z}_{-S})]$ that only uses the features $S \subset [d]$. The resulting Shapley Values sum up to the difference $\widehat{\mathcal{L}}_T(h) - \widehat{\mathcal{L}}_T(\overline{h})$ and so they can be interpreted as a form of Global Feature Importance [Covert et al., 2020]. Global Shapley Values based on the test performance will not be investigated further in this manuscript because they do not fall back to Equation 2.45 when the model is additive.

The Shapley Values attribute a score to each player $j \in [d]$ participating in a coalitional game ν. Subsequent work generalized Shapley Values by introducing the Shapley-Taylor Indices that attribute a score, not only to each player $j \in [d]$, but to each pair of players $(j, k) \in [d]^2$.

Definition 2.3.5 (Shapley Taylor Indices [Sundararajan et al., 2020])**.** *Given a set* $[d] := \{1, 2, \ldots, d\}$ *of players and a cooperative game* $\nu : 2^{[d]} \to \mathbb{R}$*, the Shapley-Taylor indices are defined as*

$$\phi_{jk}^{SHAPT}(\nu) = \begin{cases} \nu(\{j\}) - \nu(\emptyset) & \text{if } j = k, \\ \sum_{S \subseteq [d] \setminus \{j,k\}} W(|S|, d) \nabla_{jk}(S) & \text{if } j \neq k. \end{cases} \tag{2.88}$$

with

$$\nabla_{jk}(S) = \nu(S \cup \{j, k\}) - \nu(S \cup \{k\}) - \Big[\nu(S \cup \{j\}) - \nu(S) \Big]. \tag{2.89}$$

The motivation behind the Shapley-Taylor Indices was to provide interaction strength scores that respect the same additive property as the original Shapley Values (cf. Equation 2.72)

$$\sum_{j=1}^{d}\sum_{k=1}^{d}\phi_{jk}^{\text{SHAPT}}(\nu) = \nu([d]) - \nu(\emptyset). \tag{2.90}$$

Operationalizing these indices still requires the definition of a coalitional game. In the context of explaining a model prediction $h(\boldsymbol{x})$, the Interventional game shall be employed leading to the functional

$$\phi_{jk}^{\text{SHAPT}}(h, \boldsymbol{x}, \mathcal{B}) \equiv \phi_{jk}^{\text{SHAPT}}(\nu_{h,\boldsymbol{x},\mathcal{B}}^{\text{int}}). \tag{2.91}$$

We end this subsection by mirroring that of Subsection 2.3.1, which identified the input features responsible for the unfairness of h^{add}. Equation 2.46 illustrates how ante-hoc additive explanations can attribute a blame for each feature in fairness auditing. Generalizing these feature attributions to arbitrary functions h is possible by leveraging any Local Feature Attribution $\boldsymbol{\phi}$ that sums to the Gap [Begley et al., 2020].

Definition 2.3.6. *Given a post-hoc Local Feature Attribution $\boldsymbol{\phi}(h, \boldsymbol{x}, \boldsymbol{z})$ whose components sum to the Gap $G(h, \boldsymbol{x}, \boldsymbol{z}) := h(\boldsymbol{x}) - h(\boldsymbol{z})$ (e.g. SHAP, IG, Breakdown), the corresponding post-hoc Fairness Attribution is*

$$\Phi_j^{Fair}(h, \mathcal{F}, \mathcal{B}) := \mathop{\mathbb{E}}_{\substack{\boldsymbol{x}\sim\mathcal{F}\\ \boldsymbol{z}\sim\mathcal{B}}}\big[\phi_j(h, \boldsymbol{x}, \boldsymbol{z})\big], \quad j = 1, 2, \ldots, d. \tag{2.92}$$

This represents the average attribution of feature j towards the Gap $h(\boldsymbol{x}) - h(\boldsymbol{z})$ where $\boldsymbol{x} \sim \mathcal{F}$ and $\boldsymbol{z} \sim \mathcal{B}$.

Crucially, given distributions $\mathcal{F}$ and $\mathcal{B}$ over two demographic subgroups, the post-hoc Fairness Attributions sum to the corresponding Fairness metric

$$\begin{aligned}
\sum_{j=1}^{d}\Phi_j^{\text{Fair}}(h, \mathcal{F}, \mathcal{B}) &= \sum_{j=1}^{d}\mathop{\mathbb{E}}_{\substack{\boldsymbol{x}\sim\mathcal{F}\\ \boldsymbol{z}\sim\mathcal{B}}}[\,\phi_j(h, \boldsymbol{x}, \boldsymbol{z})\,] \\
&= \mathop{\mathbb{E}}_{\substack{\boldsymbol{x}\sim\mathcal{F}\\ \boldsymbol{z}\sim\mathcal{B}}}\left[\sum_{j=1}^{d}\phi_j(h, \boldsymbol{x}, \boldsymbol{z})\right] \\
&= \mathop{\mathbb{E}}_{\substack{\boldsymbol{x}\sim\mathcal{F}\\ \boldsymbol{z}\sim\mathcal{B}}}[\,h(\boldsymbol{x}) - h(\boldsymbol{z})\,] \\
&= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{F}}[h(\boldsymbol{x})] - \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{z})].
\end{aligned} \tag{2.93}$$

2.4 Thesis Main Research Question

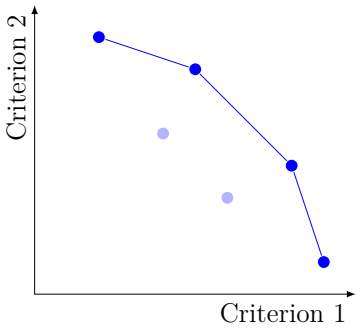
As you can see after reading Section 2.3.2, a lot of post-hoc additive explanation techniques have been proposed. Importantly, the various XAI methods were recently shown to provide different (and even contradictory) conclusions on model behavior [Krishna et al., 2022]. The observation of contradictions between explanations is referred to as the *Disagreement Problem* (DP). While the DP is to be expected since different XAI techniques characterize models differently, practitioners cannot be expected to make informed decisions when presented with contradictory claims on the behavior of their model. What if an explanation technique claims that the model is racially biased, while another claims it is not? Should the model be deployed then? Krishna et al. [2022] surveyed 25 data scientists who use explainability techniques on a day-to-day basis. The data scientists collectively stated that they did not know how to handle disagreements between explanations, and they relied on heuristics (*e.g.* choosing a favorite method or selecting whichever explanation best matched their intuition). Selecting explanations that way induces risks of *confirmation bias*, where humans think they understand the model because it matches their internal model of the world. However, we argue that trusting or mistrusting an explanation, and by extend a model, should be based on the explanation's *correctness*.

Main Research Question : How can the correctness of conflicting post-hoc additive explanations be determined?

2.4.1 Benchmarking Efforts

Current efforts to answer our main research question go in three distinct directions.

First, methods like Shapley Values [Lundberg and Lee, 2017] and the Integrated Gradient [Sundararajan et al., 2017] are motivated as ideal because they are the *unique* post-hoc additive explanations satisfying a set of theoretical properties. Although promising, these theoretical properties are still not sufficient to specify a truly unique post-hoc additive explanation. In the case of IG, subsequent work has shown that the parametric path between baseline $\boldsymbol{z}$ and input $\boldsymbol{x}$ can differ from a straight line while still respecting the theoretical properties [Lerma and Lucas, 2021]. Consequently, the original claims about uniqueness were false. Concerning Shapley Values, their properties highlighted in Equations 2.72-2.76 involve an abstract cooperative game ν and not the model h being explained. Different games have



(a) Example of Pareto Front containing all explanations which are not worse than any other w.r.t two criteria.

Faithfulness

Method	FA ↑	RA	SA	SRA	RC	PRA	PGI	PGU
Vanilla Gradient	0.923	0.921	0.138	0.136	1.000	1.000	0.297	0.391
SmoothGrad	0.923	0.923	0.741	0.741	1.000	1.000	0.485	0.082
Integrated Gradient	0.923	0.923	0.138	0.138	1.000	1.000	0.297	0.392
LIME	0.869	0.697	0.858	0.689	0.921	0.913	0.420	0.269
SHAP	0.601	0.105	0.133	0.009	0.379	0.655	0.391	0.205
Gradient x Input	0.567	0.075	0.070	0.003	0.281	0.580	0.395	0.193

(b) OpenXAI Leaderboard `https://open-xai.github.io/leaderboard`. The most faithful method for each metric are shown in red. In this benchmark, the Pareto Front regroups four methods : Gradient, SmoothGrad, IG, and LIME.

Figure 2.6 Issues when benchmarking using a suite of faithfulness metric. No single "best" explanation can be identified and so practitioners are left with a Pareto Front of plausible explanations. In this specific example, the Pareto Front contains four methods.

been proposed to explain model predictions $h(\boldsymbol{x})$ (Interventional and Observational) leading to contradicting feature attributions even for simple linear models (cf. Proposition 2.3.1). In short, lists of theoretical properties can only get you so far.

Second, some papers define a criterion that good additive explanations should satisfy and then return the explanation that optimizes this criterion Kwon and Zou [2022], Yeh et al. [2019]. For example, Kwon and Zou [2022] introduce a faithfulness metric for additive explanations called *Insertion*, and then modify the weights SHAP assign to different coalition sizes to optimize this metric. Their approach is validated by comparing the resulting Insertion score to that of vanilla SHAP. Yeh et al. [2019] defined an alternative measure of faithfulness, and proved that the additive explanation maximizing it has a closed-form. They finally compared the unfaithfulness of this explanation to competing method and, unsurprisingly, confirmed that their method is better. The issue with this methodology is that comparisons are not objective since competing methods were not invented with the specific quality criterion in mind [Freiesleben and König, 2023]. The analogous fallacy in ML research would be to invent a variant of Deep Neural Networks that better handles class imbalance and then compare it with a vanilla DNN on imbalanced data. A more meaningful comparison should involve classifiers that also handle class imbalance.

Third, to avoid the biases of reporting a single quality criterion, many explanations benchmarks report a whole suite of quality metrics. Boisnard et al. [2023] present the `Xplique` Python Package that supports for a variety of quality metrics: Insertion/Deletion [Jethani

et al., 2021, Petsiuk et al., 2018] and μ-Fidelity [Bhatt et al., 2020] faithfulness scores, in addition to the average explanations stability w.r.t random perturbations of the input $\boldsymbol{x}$ [Bhatt et al., 2020]. In a similar effort, OpenXAI has been proposed as a benchmark supporting 22 quality metrics and online leaderboards [Agarwal et al., 2022b]. The quality metrics range from the various faithfulness metrics introduced in [Dai et al., 2022] to stability metrics of Agarwal et al. [2022a]. Because multiple quality metrics are being compared simultaneously, practitioners are actually left with a Pareto Front rather than a single optimal explanation. A Pareto Front contains all methods which are not worse than any other w.r.t all criteria, see Figure 2.6 (a). Looking at a specific OpenXAI leaderboard (Figure 2.6 (b)), the choice of the most faithful explanation depends on the metric. Here, the Pareto Front contains the Gradient, SmoothGrad, IG and LIME explainers.

2.4.2 Faithfulness Metrics

Faithfulness metrics have other issues beyond their ineffectiveness at identifying a unique optimal additive explanations. We shall see that the Insertion/Deletion metrics, introduced by Petsiuk et al. [2018], and currently implemented in the `Xplique` explainability library [Boisnard et al., 2023] fail a basic sanity check.

Definition 2.4.1 (Insertion & Deletion)**.** *Given a predictor h, a local feature attribution functional $\boldsymbol{\phi}$, a reference distribution $\mathcal{B}$, and a point of interest $\boldsymbol{x}$, define the function $I : [d] \to 2^{[d]}$ that maps k to the set $I(k) \subseteq [d]$ containing the k features with the largest importance $|\phi_j(h, \boldsymbol{x}, \mathcal{B})|$. Then, the Insertion and Deletion unfaithfulness metrics follow*

$$\begin{aligned} \textit{Insertion}(h, \boldsymbol{\phi}, \boldsymbol{x}, \mathcal{B}) &:= \sum_{k=1}^{d-1} |h(\boldsymbol{x}) - \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_{I(k)}, \boldsymbol{z}_{\overline{I(k)}})]| \\ \textit{Deletion}(h, \boldsymbol{\phi}, \boldsymbol{x}, \mathcal{B}) &:= \sum_{k=1}^{d-1} |\mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{z})] - \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_{\overline{I(k)}}, \boldsymbol{z}_{I(k)})]| \end{aligned} \tag{2.94}$$

Both metrics return an Area Under the Curve (AUC), see Figure 2.7. For Insertion, this curve is generated by adding features in order of importance and reporting the error with the prediction $h(\boldsymbol{x})$. Deletion starts from the prediction $h(\boldsymbol{x})$, iteratively removes features in order of importance, and reports the error with the baseline $\mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{z})]$. Intuitively, if the explanation truly highlights important features, then the Insertion curve should quickly converge to $h(\boldsymbol{x})$ and Deletion should quickly converge to the baseline. Since, the Insertion and Deletion scores do not agree when the model contains feature interactions, it is recommended to report their Pareto front [Boisnard et al., 2023]. Regarding the choice of baseline

$\mathcal{B}$, Insertion/Deletion where originally applied to Image Classification so a Dirac $\mathcal{B} = \delta(\mathbf{0})$ over a black image was employed [Petsiuk et al., 2018]. Kwon and Zou [2022] later generalized these definitions to more models and distributions $\mathcal{B}$.

Are Insertion and Deletion good unfaithfulness metrics? Well it depends! Section 2.3.1 previously demonstrated that ante-hoc additive explanations can be extracted from additive models and take the form : $\phi_j^{\text{LFA}}(h^{\text{add}}, \boldsymbol{x}, \mathcal{B}) = h_j(x_j) - \mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[h_j(z_j)]$ for some baseline $\mathcal{B}$. Moreover, as presented in Section 2.3.2, some post-hoc local feature attributions fall back to this definition when the model is additive (*e.g.* SHAP). Consequently, SHAP is faithful to the model h, at least when h is additive. Given this fact, a sensible unfaithfulness metric should be minimized when running SHAP on an additive model h^{add}. We now show that Insertion and Deletion fail this sanity check.

Proposition 2.4.1. *Let $\boldsymbol{\phi}$ be a post-hoc additive explainer that falls back to the ante-hoc explanation $\phi_j(h^{add}, \boldsymbol{x}, \mathcal{B}) = h_j(x_j) - \mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[h_j(z_j)]$ when the model is additive (e.g. SHAP). Also, let $\boldsymbol{\phi}'$ be any alternative local feature attribution that does not fall back to ante-hoc definitions.*

Then, if h^{add} is additive, the Insertion and Deletion metrics are equivalent

$$\begin{aligned} \mathit{Insertion}(h^{add}, \boldsymbol{\phi}, \boldsymbol{x}, \mathcal{B}) &= \mathit{Deletion}(h^{add}, \boldsymbol{\phi}, \boldsymbol{x}, \mathcal{B}) \\ \mathit{Insertion}(h^{add}, \boldsymbol{\phi}', \boldsymbol{x}, \mathcal{B}) &= \mathit{Deletion}(h^{add}, \boldsymbol{\phi}', \boldsymbol{x}, \mathcal{B}). \end{aligned} \tag{2.95}$$

Moreover, when all local feature attributions $\{\phi_j(h^{add}, \boldsymbol{x}, \mathcal{B})\}_{j=1}^d$ have the same sign, we have

$$\mathit{Insertion}(h^{add}, \boldsymbol{\phi}, \boldsymbol{x}, \mathcal{B}) \leq \mathit{Insertion}(h^{add}, \boldsymbol{\phi}', \boldsymbol{x}, \mathcal{B}). \tag{2.96}$$

Insertion/Deletion metrics rightfully claim that $\boldsymbol{\phi}$ is more faithful to h^{add} than $\boldsymbol{\phi}'$. However, if we allow local feature attributions $\{\phi_j(h^{add}, \boldsymbol{x}, \mathcal{B})\}_{j=1}^d$ to have both positive and negative signs, then there exists a functional $\boldsymbol{\phi}'$, model h^{add}, input $\boldsymbol{x}$ and reference $\mathcal{B}$ where

$$\mathit{Insertion}(h^{add}, \boldsymbol{\phi}', \boldsymbol{x}, \mathcal{B}) < \mathit{Insertion}(h^{add}, \boldsymbol{\phi}, \boldsymbol{x}, \mathcal{B}). \tag{2.97}$$

Insertion and Deletion fail as unfaithfulness metrics. The proof is presented in App. A.3.

This proposition demonstrates that Insertion and Deletion are only sensible unfaithfulness metrics if all local feature attributions have the same sign. In this ideal scenario, the Insertion/Deletion curves are monotonic (see Figure 2.7). When the local feature attributions have both positive and negative signs, the Insertion/Deletion curves oscillate and cannot be used as unfaithfulness metrics.

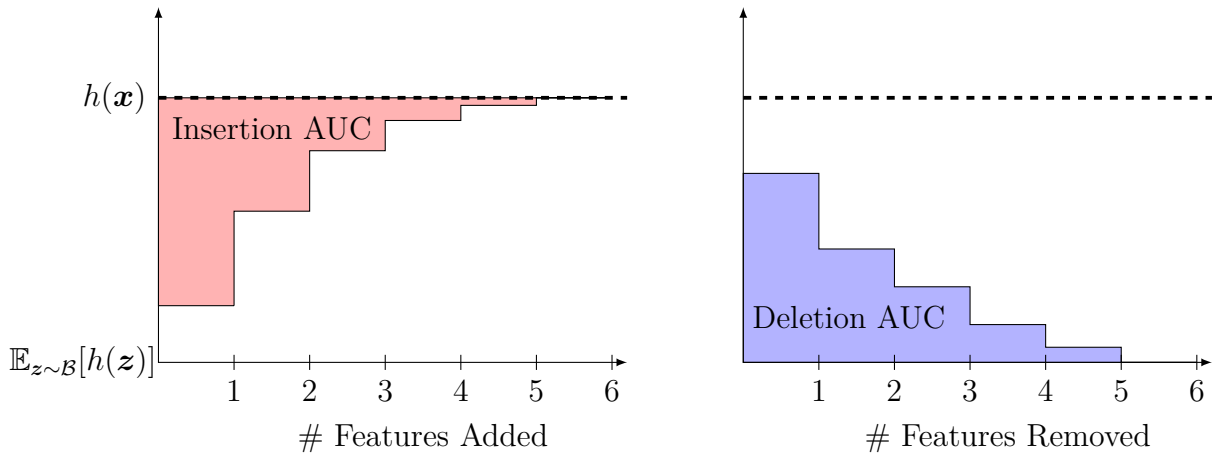


Figure 2.7 Intuition behind the Insertion and Deletion unfaithfulness metrics. (Right) Insertion iteratively adds features into the model in order of importance. A low AUC is desirable since it implies that predictions converge quickly to $h(\boldsymbol{x})$. (Left) Deletion starts from the full set of features and progressively removes them in order of importance. A lower AUC is better since it highlights a rapid convergence to the baseline $\mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{z})]$.

2.4.3 Align instead of Benchmark

Past XAI research has not been able to pinpoint the ideal additive explanation of a non-additive model. As we just discussed, lists of theoretical properties (SHAP, IG), faithfulness metrics, or suites thereof have not yet been successful. Rather than continuing in this direction, we propose a drastic shift. First, we investigate the question:

Question I : What are the root causes of disagreements between post-hoc additive explanation methods?

This research question is tackled in part I of the Thesis. All post-hoc additive explanations presented in Section 2.3.2 are unified through the lens of Functional Decomposition. More specifically, post-hoc additive explanations are expressed in terms of $\boldsymbol{z}$-Anchored Decompositions [Kuo et al., 2010], which are well-established in Sensitivity Analysis, but less so in XAI. These results reveal what each explainer actually measure, and more importantly, in what contexts these methods agree or disagree. For instance, it is shown that many explainers (*e.g.* PDP, SHAP, PFI, ALE, LIME, Breakdown) agree when the model is additive. Thus, *feature interactions* are the culprit that prohibit agreement among the competing approaches. More

importantly, when the aforementioned methods agree with each other, they coincide with the ante-hoc additive explanations of an additive model. So, the goal of increasing agreement among explanation techniques is aligned with the goal of attaining an ideal ante-hoc explanation.

In light of this unification and convergence toward an ideal ante-hoc explanation, our motto becomes : *align instead of benchmark*. We then ask

---

Question II : How can we increase alignment between contradicting explanations?

This question is investigated in part II. We identify three measures of disagreements between explanations and present methodologies to reduce them.

The first disagreement investigated is the *Interaction Disagreement*, which reports the strength of interactions and how they prohibit agreement among the many post-hoc explainers. We then reduce feature interactions by partitioning the input space using a so-called FD-Tree (Functional Decomposition Tree). Since post-hoc explainers have greater agreement when restricted to the leaves of the FD-Tree, the question of which method is "best" becomes less relevant.

The second and third disagreement are called *Subsampling Disagreement* and *Underspecification Disagreement*. They both relate to the sensitivity of post-hoc explanations w.r.t innocuous choices such as the data subsample used to approximate explanations and the choice of model h within an equivalence class of models with good performance (*i.e.* a Rashomon Set). Characterizing these two disagreements is important if practitioners are to extract robust insights additive epxlanations.

CHAPTER 3 FUNCTIONAL DECOMPOSITION

3.1 Functional Decompositions

To understand the root cause of disagreements between post-hoc additive explanations, we express the various techniques in terms of a novel functional decomposition, which we call the *Interventional Decomposition*. This expression of the post-hoc methods reveals the source of their disagreement: feature interactions.

3.1.1 Replace-Function

Up to this point, we have assumed that the model of interest h takes as input a d-dimensional vector $\boldsymbol{x} \in \mathbb{R}^d$ and returns a real value $h(\boldsymbol{x}) \in \mathbb{R}$. Each component $x_j \in \mathbb{R}$ of the input was called a *feature* and LFA/GFI methods aimed to convey their individual importance toward the model response. Henceforth, we generalize the meaning of *feature* by letting the input domain

$$\mathcal{X} = \prod_{j=1}^{d} \mathcal{X}_j \tag{3.1}$$

be a Cartesian product of **arbitrary** spaces $\mathcal{X}_j$. Indeed, $\mathcal{X}_j$ need not be $\mathbb{R}$ so one can represent a categorical feature $\mathcal{X}_j = [\text{dog}, \text{cat}, \text{hamster}]$, or even a feature $\mathcal{X}_j = \mathbb{R}^M$ that is itself a vector/matrix/tensor. For example, a feature vector could be

$$\boldsymbol{x} = [2.5, \text{Cat}, \text{Man}, (2.56, 75.23)]^T,$$

where two features are categorical and another is a tuple. In the context of digit classification, each feature could be a 4x4 patch of the image $\mathcal{X}_j = [256]^{16}$

Letting $u \subseteq [d]$ be a subset of features indices and $\boldsymbol{x}_u := (x_j)_{j \in u}$ be their values, understanding the sensitivity of h w.r.t $\boldsymbol{x}_u$ can be done by freezing $\boldsymbol{x}_{-u}$ at some baseline $\boldsymbol{z}_{-u}$. This is the idea behind the *replace-function* $\boldsymbol{r}_u^{\boldsymbol{z}} : \mathcal{X} \to \mathcal{X}$ defined as

$$r_u^{\boldsymbol{z}}(\boldsymbol{x})_j = \begin{cases} x_j & \text{if } j \in u \\ z_j & \text{otherwise.} \end{cases} \tag{3.2}$$

The replace-function is able to map $\mathcal{X}$ back to itself because the input space is a Cartesian

product. The function $h : \mathcal{X} \to \mathbb{R}$ can then be evaluated on $\boldsymbol{r}_u^{\boldsymbol{z}}(\boldsymbol{x})$

$$h(\boldsymbol{r}_u^{\boldsymbol{z}}(\boldsymbol{x})) \equiv h(\boldsymbol{x}_u, \boldsymbol{z}_{-u}). \tag{3.3}$$

Both of these notations are equivalent, but they come with their own strengths and weaknesses. On the one hand, $h(\boldsymbol{x}_u, \boldsymbol{z}_{-u})$ makes it clear what features are set to $\boldsymbol{x}$ and which are set to $\boldsymbol{z}$. Yet, the set u must be written twice, which may clutter notation when u is a large expression. On the other hand, $h(\boldsymbol{r}_u^{\boldsymbol{z}}(\boldsymbol{x}))$ only requires writing u once, but it is not immediately evident what features are $\boldsymbol{x}$ and which are $\boldsymbol{z}$. One has to remember the convention set by Equation 3.2. In the sequel, whatever notation is most convenient will be used.

3.1.2 Anchored Decomposition

A Decomposition of function h expresses it as a sum of 2^d sub-functions

$$h(\boldsymbol{x}) = \sum_{u \subseteq [d]} h_u(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \mathcal{X}, \tag{3.4}$$

where h_u does not depend on $\boldsymbol{x}_{-u}$.

Definition 3.1.1 (Functional Dependence)**.** *Let $\mathcal{X} = \prod_{j=1}^d \mathcal{X}_j$ be the input space. The function $g : \mathcal{X} \to \mathbb{R}$, does not depend on u if for all $\boldsymbol{x} \in \mathcal{X}$ and $\boldsymbol{z}_u \in \mathcal{X}_u$ the following holds*

$$g(\boldsymbol{x}) = g(\boldsymbol{x}_{-u}, \boldsymbol{z}_u). \tag{3.5}$$

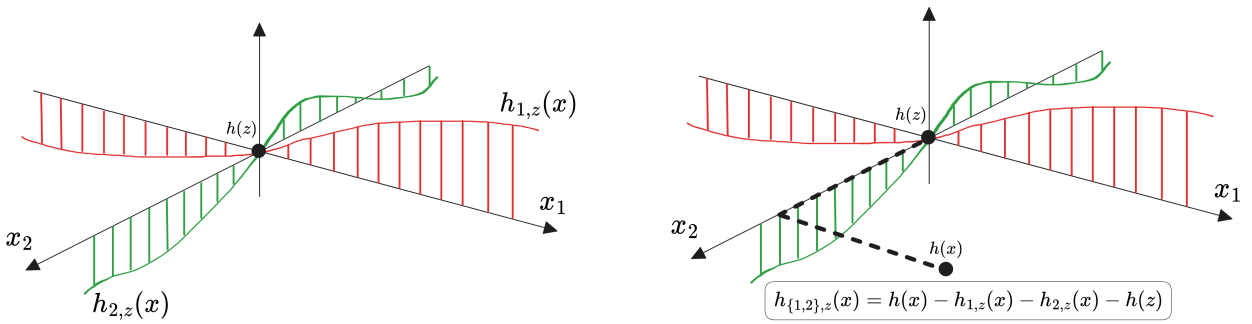
That is, the function is not affected by replacing the features u with another value $\boldsymbol{z}_u$ from their domain $\mathcal{X}_u$.

The term $h_\emptyset$ of the functional decomposition is a constant called the "intercept". The terms h_j are called the "main effects" and each depend on a single feature. Finally, the terms h_u with $|u| \geq 2$ are referred to as "$|u|$-way interactions" and depend on multiple features simultaneously. The function h is said to be additive if there exists a decomposition where $h_u = 0$ whenever $|u| \geq 2$ *i.e.* there are no interactions.

Definition 3.1.2 (Additive Function)**.** *Let $\mathcal{X} = \prod_{j=1}^d \mathcal{X}_j$ be the input space. The function $h : \mathcal{X} \to \mathbb{R}$ is called additive if it can be decomposed as*

$$h(\boldsymbol{x}) = \sum_{u \subseteq [d] : |u| \leq 1} h_u(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \mathcal{X}. \tag{3.6}$$

That is, there exists a decomposition of h that does not contain any interactions.



(a) Additive components

(b) Interaction component

Figure 3.1 Illustration of the $\boldsymbol{z}$-Anchored Decomposition with $\mathcal{X} = \mathbb{R}^2$.

Additive functions are advertised as being inherently intelligible because the impact of varying a feature on the output is independent of other features [Lou et al., 2012]. In oposition, interactions terms h_u ($|u| \geq 2$) are harder to understand because the impact of varying feature $j \in u$ on the output depends on other features in u.

The assumption that the domain of h is a carterian product is important to avoid edges cases where models "appear" additive because of the geometry of $\mathcal{X}$. For instance, imagine a function $h(\boldsymbol{x}) = x_1 x_2$ with the domain $\mathcal{X} = \{\boldsymbol{x} \in \mathbb{R}^2 : x_1 = x_2\}$. The function admits an additive decomposition $h(\boldsymbol{x}) = x_1 x_2 = x_1^2 \ \forall \boldsymbol{x} \in \mathcal{X}$ but is not additive in the intuitive sense.

Functional decompositions are not unique. For example, let $h(\boldsymbol{x}) = 2x_1$. This function could be expressed with a single main effect $h_1(\boldsymbol{x}) = 2x_1$, but it could also be rewritten

$$h(\boldsymbol{x}) = \underbrace{x_1}_{h_1(\boldsymbol{x})} + \underbrace{x_2}_{h_2(\boldsymbol{x})} + \underbrace{(x_1 - x_2)}_{h_{\{1,2\}}(\boldsymbol{x})}. \tag{3.7}$$

Although valid, this decomposition is undesirable because it introduces more terms than necessary. Mathematically speaking, this decomposition is not *Minimal* [Kuo et al., 2010]. Minimality is one of the arguments in favor of the so-called *Anchored Decomposition*.

Definition 3.1.3 ($\boldsymbol{z}$-Anchored Decomposition [Kuo et al., 2010])**.** *Let $\mathcal{X} = \prod_{j=1}^{d} \mathcal{X}_j$ be the input space. Given a function $h : \mathcal{X} \to \mathbb{R}$ and a fixed reference $\boldsymbol{z} \in \mathcal{X}$, the $\boldsymbol{z}$-Anchored Decomposition is*

$$\begin{aligned} h_{\emptyset,\boldsymbol{z}} &:= h(\boldsymbol{z}) \\ h_{j,\boldsymbol{z}}(\boldsymbol{x}) &:= h(\boldsymbol{x}_j, \boldsymbol{z}_{-j}) - h_{\emptyset,\boldsymbol{z}} \\ &\dots \\ h_{u,\boldsymbol{z}}(\boldsymbol{x}) &:= h(\boldsymbol{x}_u, \boldsymbol{z}_{-u}) - \sum_{v \subset u} h_{v,\boldsymbol{z}}(\boldsymbol{x}). \end{aligned} \tag{3.8}$$

Recipe for z-Anchored Decompositions [Kuo et al., 2010]

Let $\boldsymbol{z} \in \mathcal{X}$ be a reference input that is *fixed*.

- The intercept $h_{\emptyset,\boldsymbol{z}}$ is the prediction at the reference

$$h_{\emptyset,\boldsymbol{z}} := h(\boldsymbol{z}). \tag{3.9}$$

- The main effects $h_{j,\boldsymbol{z}}(\boldsymbol{x})$ are built by varying the jth feature away from z_j and reporting the increase/decrease relative to the reference $h(\boldsymbol{z})$:

$$\begin{aligned} h_{1,\boldsymbol{z}}(\boldsymbol{x}) &:= h(\boldsymbol{x}_1, \boldsymbol{z}_{-1}) - h(\boldsymbol{z}) \\ &\cdots \\ h_{d,\boldsymbol{z}}(\boldsymbol{x}) &:= h(\boldsymbol{x}_d, \boldsymbol{z}_{-d}) - h(\boldsymbol{z}). \end{aligned} \tag{3.10}$$

  Figure 3.1(a) conveys the intuition behind main effects.

- The pairwise interactions $h_{\{j,k\},\boldsymbol{z}}$ are constructed by evaluating the model at $h(\boldsymbol{x}_{\{j,k\}}, \boldsymbol{z}_{-\{j,k\}})$ and reporting the difference with the sum of the intercept and the two main effects:

$$h_{\{j,k\},\boldsymbol{z}}(\boldsymbol{x}) := h(\boldsymbol{x}_{\{j,k\}}, \boldsymbol{z}_{-\{j,k\}}) - h_{j,\boldsymbol{z}}(\boldsymbol{x}) - h_{k,\boldsymbol{z}}(\boldsymbol{x}) - h(\boldsymbol{z}) \tag{3.11}$$

  See Figure 3.1(b) for the intuition.

- Finally, general u-way interactions are computed via

$$h_{u,\boldsymbol{z}}(\boldsymbol{x}) := h(\boldsymbol{x}_u, \boldsymbol{z}_{-u}) - \sum_{v \subset u} h_{v,\boldsymbol{z}}(\boldsymbol{x}). \tag{3.12}$$

The $\boldsymbol{z}$-Anchored Decomposition is minimal and will not introduce more terms than necessary *i.e.* Equation 3.7 will never occur.

Theorem 3.1.1 (Minimality Theorem from [Kuo et al., 2010]). *Let $\mathcal{X} = \prod_{j=1}^{d} \mathcal{X}_j$ be the input space, $h : \mathcal{X} \to \mathbb{R}$ be a function and $\sum_{u \subseteq [d]} g_u$ be any functional decomposition of h. Also, assume that a subset $v \subset [d]$ exists such that*

$$v \subseteq u \Rightarrow \forall \boldsymbol{x} \in \mathcal{X} \quad g_u(\boldsymbol{x}) = 0.$$

Then, the $\boldsymbol{z}$-Anchored Decomposition respects

$$v \subseteq u \Rightarrow \forall \boldsymbol{x}, \boldsymbol{z} \in \mathcal{X} \quad h_{u,\boldsymbol{z}}(\boldsymbol{x}) = 0.$$

The assumption that the domain of h is a Cartesian product of features $\prod_{j=1}^{d} \mathcal{X}_j$ is crucial for the Theorem to hold. Indeed, without this assumption, Equation 3.3 would not make sense and the $\boldsymbol{z}$-Anchored Decomposition would be ill-defined.

3.1.3 Interventional Decompositions

Rather than using a fixed reference point $\boldsymbol{z}$, we propose to use references sampled from a probability distribution $\mathcal{B}$.

Definition 3.1.4 (Interventional Decomposition)**.** *Let $\mathcal{X} = \prod_{j=1}^{d} \mathcal{X}_j$ be the input space. Given a function $h : \mathcal{X} \to \mathbb{R}$ and a distribution $\mathcal{B}$, we define the Interventional Decomposition as:*

$$\begin{aligned} h_{\emptyset,\mathcal{B}} &:= \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{z})] \\ h_{j,\mathcal{B}}(\boldsymbol{x}) &:= \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_j, \boldsymbol{z}_{-j})] - h_{\emptyset,\mathcal{B}} \\ &\dots \\ h_{u,\mathcal{B}}(\boldsymbol{x}) &:= \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_u, \boldsymbol{z}_{-u})] - \sum_{v\subset u} h_{v,\mathcal{B}}(\boldsymbol{x}). \end{aligned} \tag{3.13}$$

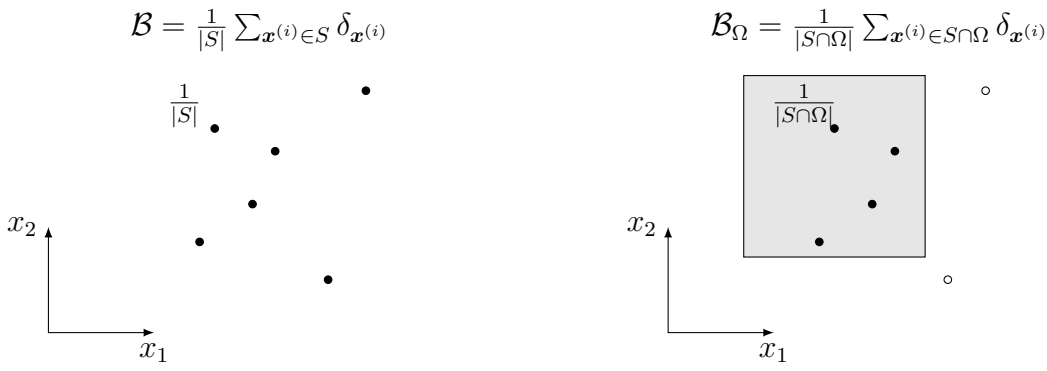
We refer to this decomposition as the *Interventional Decomposition* by taking inspiration from Interventional Shapley Values [Janzing et al., 2020] which break feature correlations in $\mathcal{B}$ by replacing replacing $\boldsymbol{x}_{-u}$ components with $\boldsymbol{z}_{-u}$ sampled from the marginal $\mathcal{B}_{-u}$. Note that, by linearity of the expectation, Anchored and Interventional Decompositions are related via:

$$h_{u,\mathcal{B}}(\boldsymbol{x}) = \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h_{u,\boldsymbol{z}}(\boldsymbol{x})]. \tag{3.14}$$

Computing an Interventional Decomposition amounts to computing multiple $\boldsymbol{z}$-Anchored Decompositions and averaging them. Viewing Interventional Decompositions that way (instead of through Equation 3.13) has two advantages:

1. If the background is restricted to a region, the Interventional Decomposition is simply the aggregation of all $\boldsymbol{z}$-Anchored Decompositions from this locality.

2. By linearity of the expectation, Interventional Decompositions inherit many of the theoretical properties of $\boldsymbol{z}$-Anchored Decompositions demonstrated by Kuo et al. [2010].



(a) Non-Regional (b) Regional

Figure 3.2 Empirical distribution over a dataset S and its regional counterpart on Ω.

Regional Distributions First, the background distribution $\mathcal{B}$ could be restricted to a region $\Omega = \prod_{j=1}^{d} \Omega_j$ where $\Omega_j \subseteq \mathcal{X}_j$. If $\mathcal{B}(\Omega) > 0$, the *regional* background $\mathcal{B}_\Omega$ is the measure:

$$\mathcal{B}_\Omega(A) := \frac{\mathcal{B}(A \cap \Omega)}{\mathcal{B}(\Omega)}, \tag{3.15}$$

for any measurable subset $A \subset \mathcal{X}$. For measures endowed with a probability density ρ over $\mathcal{X}$, their regional counterpart has density

$$\rho_\Omega(\boldsymbol{x}) := \frac{\rho(\boldsymbol{x}) \mathbb{1}(\boldsymbol{x} \in \Omega)}{\int_\Omega \rho(\boldsymbol{x}) d\boldsymbol{x}}. \tag{3.16}$$

For discrete measures, such as the empirical distribution over the dataset $\mathcal{B} = \frac{1}{|S|} \sum_{\boldsymbol{x}^{(i)} \in S} \delta_{\boldsymbol{x}^{(i)}}$, the corresponding regional measure is

$$\mathcal{B}_\Omega = \frac{1}{|S \cap \Omega|} \sum_{\boldsymbol{x}^{(i)} \in S \cap \Omega} \delta_{\boldsymbol{x}^{(i)}}. \tag{3.17}$$

This formula is illustrated in Figure 3.2. Simply put, a regional discrete distribution removes all data that land outside of Ω, and reweights the remaining instances. From Equation 3.14, it becomes apparent that computing an Interventional Decomposition $h_{u,\mathcal{B}_\Omega}$ amounts to selecting a subset of data based on Ω, computing their corresponding $\boldsymbol{z}$-Anchored Decomposition, and averaging them.

Theoretical Properties Since Interventional Decompositions are the average of $\boldsymbol{z}$-Anchored Decompositions, they inherit many of their theoretical properties. First, the Möbius Inverse

formula respected by Anchored Decompositions (Example 2.3 (b) [Kuo et al., 2010]) translates to

$$\mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_u, \boldsymbol{z}_{-u})] = \sum_{v\subseteq u} h_{v,\mathcal{B}}(\boldsymbol{x}) \iff h_{u,\mathcal{B}}(\boldsymbol{x}) = \sum_{v\subseteq u}(-1)^{|u|-|v|} \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_v, \boldsymbol{z}_{-v})]. \tag{3.18}$$

This Equation is especially useful when trying to make interactions terms appear out of algebraic expressions involving $\mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_u, \boldsymbol{z}_{-u})]$. The Minimality property is also inherited.

Corollary 3.1.1. *Let $\Omega = \prod_{j=1}^{d} \Omega_j$ be a subset of $\mathcal{X}$, $h : \mathcal{X} \to \mathbb{R}$ be a function and*

$$h(\boldsymbol{x}) = \sum_{u\subseteq[d]} g_u(\boldsymbol{x}) \quad \textit{for all } \boldsymbol{x} \in \Omega,$$

be a functional decomposition of h over the region Ω. Also, assume that a subset $v \subset [d]$ exists such that

$$v \subseteq u \Rightarrow \forall \boldsymbol{x} \in \Omega \;\; g_u(\boldsymbol{x}) = 0.$$

Then, for any probability distribution $\mathcal{B}$ such that $\mathcal{B}(\Omega) > 0$ the $\mathcal{B}_\Omega$-Interventional-Decomposition respects

$$v \subseteq u \Rightarrow \forall \boldsymbol{x} \in \Omega \;\; h_{u,\mathcal{B}_\Omega}(\boldsymbol{x}) = 0.$$

Proof. Since $\mathcal{B}(\Omega) > 0$, the regional background $\mathcal{B}_\Omega$ is well defined and respects $\text{supp}(\mathcal{B}_\Omega) \subseteq \Omega$. Therefore, any sample $\boldsymbol{z} \sim \mathcal{B}_\Omega$ from the regional background must land inside Ω. Moreover, since the region Ω is a cartesian product, Theorem 3.1.1 is applicable to the restricted function $h : \Omega \to \mathbb{R}$.

Now, let u be a super-set of v ($v \subseteq u$). By Theorem 3.1.1, for any $\boldsymbol{x}, \boldsymbol{z} \in \Omega$ we have $h_{u,\boldsymbol{z}}(\boldsymbol{x}) = 0$. So, for any $\boldsymbol{x} \in \Omega$

$$h_{u,\mathcal{B}_\Omega}(\boldsymbol{x}) = \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}_\Omega}[h_{u,\boldsymbol{z}}(\boldsymbol{x})] = \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}_\Omega}[0] = 0.$$

□

Minimality is a very general property that has important consequences depending on the subset v used in Corollary 3.1.1. For instance, if the model h does not depend on feature j, then there exists a decomposition such as $g_u(\boldsymbol{x}) = 0$ whenever $\{j\} \subseteq u$. As a result of minimality, the Interventional Decomposition cannot have terms involving j. This logic translates to arbitrarily many dummy features.

Dummy Property

When h does not depend on w over the region $\Omega = \prod_{j=1}^{d} \Omega_j$, the following holds

$$h_{u,\mathcal{B}_\Omega}(\boldsymbol{x}) = 0 \quad \text{whenever } u \cap w \neq \emptyset. \tag{3.19}$$

No attribution can be given to a feature not used by the model in the region.

Moreover, let h be additive w.r.t j, that is, $h(\boldsymbol{x}) = g_j(x_j) + g_{-j}(\boldsymbol{x}_{-j})$. For any other feature k, $g_u(\boldsymbol{x}) = 0$ whenever $\{j, k\} \subseteq u$ and so the Interventional Decomposition cannot have interactions involving $\{j, k\}$. Thus if h is additive w.r.t to j, then so is the Interventional Decomposition.

Additive Recovery

Let $\Omega = \prod_{j=1}^{d} \Omega_j$ be a region. When $h(\boldsymbol{x}) = g_j(x_j) + g_{-j}(\boldsymbol{x}_{-j})\ \forall \boldsymbol{x} \in \Omega$, the Interventional Decomposition yields $h(\boldsymbol{x}) = h_{j,\mathcal{B}_\Omega}(\boldsymbol{x}) + \sum_{u \subseteq [d] \setminus \{j\}} h_{u,\mathcal{B}_\Omega}(\boldsymbol{x})$ with main effect for feature j

$$h_{j,\mathcal{B}_\Omega}(\boldsymbol{x}) = g_j(x_j) - \mathop{\mathbb{E}}_{z \sim \mathcal{B}_\Omega}[g_j(z_j)]. \tag{3.20}$$

When h is additive w.r.t feature j, the true dependence on feature j is recovered up to a constant.

3.1.4 ANOVA Decomposition

When $\mathcal{B} = \mathcal{B}_{\text{ind}} := \prod_{j=1}^{d} \mathcal{B}_j$ (*i.e.* input features are independent), the Interventional Decomposition falls back to the so-called *ANOVA Decomposition* [Hooker, 2004]. In such cases, the components $h_{u,\mathcal{B}_{\text{ind}}}$ and $h_{v,\mathcal{B}_{\text{ind}}}$ are zero-mean, orthogonal, and the total variance decomposes as the sum of the variance for each individual component

$$\mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mathcal{B}_{\text{ind}}}[\,(h(\boldsymbol{x}) - h_{\emptyset,\mathcal{B}_{\text{ind}}})^2] = \sum_{\substack{u \subseteq [d] \\ |u| \geq 1}} \sigma_u^2 \tag{3.21}$$

with

$$\sigma_u^2 := \mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mathcal{B}_{\text{ind}}}[h_{u,\mathcal{B}_{\text{ind}}}(\boldsymbol{x})^2]. \tag{3.22}$$

For general $\mathcal{B}$, these properties may not hold however. The ANOVA Decomposition will rarely be used in this manuscript because the assumption of total feature independence is too strong for any realistic Machine Learning use-case. ANOVA will be mostly of theoretical interest when analyzing LIME in Section 3.2.1, when introducing the VIN algorithm in Section 4.2.1,

and when presenting the definition of *Interaction Disagreements* in Chapter 6.

3.2 Unification of Post-hoc Additive Explanations

This section leverages the Anchored/Interventional/ANOVA Decompositions introduced previously to unify various post-hoc additive explanations. Each separate method will be expressed in terms of the components $\{h_{u,\mathcal{B}}\}_{u\subseteq[d]}$, thus highlighting what model characteristics are measured by each explainer and, most importantly, why do explainers disagree.

3.2.1 Local Feature Attributions

Partial Dependence Plots Trivially, the PDP is the first component of the Interventional Decomposition, up to a constant

$$\phi_j^{\text{PDP}}(h, \boldsymbol{x}, \mathcal{B}) := h_{j,\mathcal{B}}(\boldsymbol{x}) + \text{Constant}. \tag{3.23}$$

Since only the profile of the PDP curve is useful (not its value), the *Constant* can be fixed to zero without impacting its interpretation.

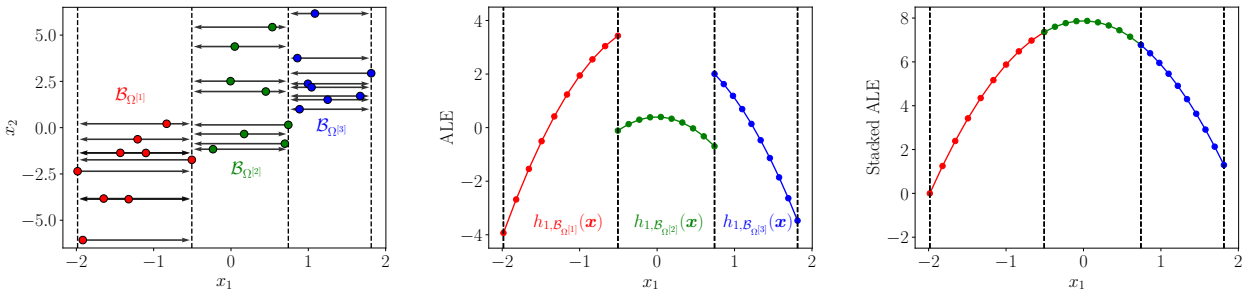
ALE As a reminder, the theoretical definition of Accumulated Local Effects (ALE) is

$$\phi_j^{\text{ALE-Theory}}(h, \boldsymbol{x}, \mathcal{B}) := \int_{x_{j,\min}}^{x_j} \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}\left[\frac{\partial h}{\partial z_j}(\boldsymbol{z})\middle| z_j\right] dz_j. \tag{3.24}$$

Although elegant, this formulation is not very informative as to what this explainer actually measures. Studying the empirical estimate of ALE leads to more insights. In practice, the expectations conditioned on z_j are estimated by binning feature j into M intervals $]\gamma_{t-1}, \gamma_t]$ with $\gamma_0 < \gamma_1 < \ldots < \gamma_M$. These intervals are then extended to regions $\Omega^{[t]} := \mathcal{X}_{-j}\times]\gamma_{t-1}, \gamma_t]$. Now, conditioning on $z_j \in]\gamma_{t-1}, \gamma]$ is approximated by employing the regional background $\mathcal{B}_{\Omega^{[t]}}$, see the colored points in Figure 3.3 (a). Also, the derivative $\frac{\partial h}{\partial z_j}$ is approximated with finite-difference $h(\gamma_t, \boldsymbol{z}_{-j}) - h(\gamma_{t-1}, \boldsymbol{z}_{-j})$, see the dark arrows in Figure 3.3 (a). Returning the average finite difference across $\boldsymbol{z} \sim \mathcal{B}_{\Omega^{[t]}}$ amounts to computing the functional decompositions $h_{j,\mathcal{B}_{\Omega^{[t]}}}$

$$\phi_j^{\text{ALE-Practice}}(h, \boldsymbol{x}, \mathcal{B}) := h_{j,\mathcal{B}_{\Omega^{[t]}}}(\boldsymbol{x}) \quad \text{where} \quad \boldsymbol{x} \in \Omega^{[t]} := \mathcal{X}_{-j}\times]\gamma_{t-1}, \gamma_t], \tag{3.25}$$

see Figure 3.3 (b). Finally, these regional profiles can be "glued together" from left to right to generate the ALE profile, see Figure 3.3 (c). Yet, we argue that this stacking is not necessary and that Equation 3.25 is the most insightful way to conceptualize ALE. From this point of



(a) Input space is partitioned by binning along x_1 leading to regional distributions $\mathcal{B}_{\Omega[t]}$.

(b) Regional Interventional Decompositions are computed on each separate regions.

(c) Optionally, the profiles can be glued together from left-to-right.

Figure 3.3 Intuition of how ALE relates to Regional Interventional Decompositions $h_{j,\mathcal{B}_{\Omega[t]}}(\boldsymbol{x})$.

view, ALE is actually a collection regional functional decompositions, each applied over a separate bin of feature j.

LIME The central idea behind Local Interpretable Model-agnostic Explanations (LIME) is to provide local feature attributions at $\boldsymbol{x}$ by locally approximating h with a linear model in the neighborhood of $\boldsymbol{x}$. Letting $\mathcal{B}$ be a probability distribution centered at $\boldsymbol{x}$, we saw previously that LIME returns

$$(\omega_0, \underbrace{\omega_1, \omega_2, \ldots, \omega_d}_{\boldsymbol{\phi}^{\text{LIME}}(h,\boldsymbol{x},\mathcal{B})}) = \underset{\boldsymbol{\omega}\in\mathbb{R}^{d+1}}{\operatorname{argmin}} \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}} \left[\left(h(\boldsymbol{z}) - \omega_0 - \sum_{j=1}^{d} \omega_j z_j \right)^2 \right]. \tag{3.26}$$

The following results clarifies the link between approximating h with a local interpretable model and computing a ANOVA decomposition.

Lemma 3.2.1 (**Lemma A.5** [Owen, 2013])**.** *Let $\mathcal{B}_{ind} = \prod_{j=1}^{d} \mathcal{B}_j$ be a background of independent features and let $L_0^2(\mathcal{X}_j)$ be the subset of $L^2(\mathcal{X}_j)$ containing functions f s.t. $\mathbb{E}_{z_j\sim\mathcal{B}_j}[f(z_j)] = 0$, then the following holds*

$$(h_{\emptyset,\mathcal{B}_{ind}}, h_{1,\mathcal{B}_{ind}}, \ldots, h_{d,\mathcal{B}_{ind}}) \in \underset{(\omega_0,g_1,\ldots,g_d)\in\mathbb{R}\times\prod_{j=1}^{d} L_0^2(\mathcal{X}_j)}{\operatorname{argmin}} \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}_{ind}} \left[\Big(h(\boldsymbol{x}) - \omega_0 + \sum_{j=1}^{d} g_j(z_j) \Big)^2 \right]. \tag{3.27}$$

The striking resemblance between Equations 3.26 and 3.27 suggests that it may be possible to frame LIME as a functional decomposition. Cases where $\mathcal{X} = \{0,1\}^d$ (Text/Image LIME) and where $\mathcal{X} \subseteq \mathbb{R}^d$ (Tabular LIME) are discussed separately.

Let $\mathcal{X} = \{0,1\}^d$, as stated in the previous chapter, this scenario occurs when LIME is used on Text/Images and the binary features encode the presence or removal of a specific part of the input (a specific word or a patch of pixels.) The background distribution $\mathcal{B}$ is chosen such that $\mathbb{E}_{\boldsymbol{x}\sim\mathcal{B}}[x_j] = p$, *i.e.* it randomly shuts down components of the input with probability $(1-p)$. If features are independent, the LIME methodology of fitting a linear surrogate of h and reporting the weights coincides with a functional decomposition.

Proposition 3.2.1. *Let $\mathcal{B}_{ind} = \prod_{j=1}^{d} \mathcal{B}_j$ be a background distribution of independent features over $\mathcal{X} = \{0,1\}^d$ such that $\mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[z_j] = p$. The classical LIME approach advocates for fitting a linear model*

$$(\omega_0, \omega_1, \ldots, \omega_d) = \underset{\boldsymbol{\omega}\in\mathbb{R}^{d+1}}{\operatorname{argmin}} \mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}_{ind}}\left[\Big(h(\boldsymbol{z}) - \sum_{j=1}^{d} \omega_j z_j - \omega_0\Big)^2\right] \tag{3.28}$$

and reporting the coefficients $(w_1, w_2, \ldots, w_d)$ as the local feature importance for input $\boldsymbol{x} = \mathbf{1}$ (i.e. the full text/image). These weights are proportional to the functional decomposition

$$h_{j,\mathcal{B}_{ind}}(\mathbf{1}) = \omega_j \times (1-p). \tag{3.29}$$

The proof is presented in Appendix B.1.

This proposition reveals that the weights returned by Text/Image LIME are quite sensible measures of feature attribution. Indeed, they actually perform ANOVA functional decomposition although this was not the initial intent of the approach.

When $\mathcal{X} \subseteq \mathbb{R}^d$, a local linear surrogate may not approximate the model h, even if it is additive. Another issue with these surrogates is that the default setting $\mathcal{B}_{\text{ind}} = \mathcal{N}(\boldsymbol{x}, \sigma^2\boldsymbol{I})$ of LIME results in a null gap (cf. Equation 2.62). In light of Lemma 3.2.1, one could modify LIME by fitting a local *additive* surrogate (rather than a local linear one). This local additive surrogate could be obtained by either minimizing the squared error (cf. Equation 3.27) or by directly computing the functional component

$$\phi_j^{\text{LIME}}(h, \boldsymbol{x}, \mathcal{B}_{\text{ind}}) := h_{j,\mathcal{B}_{\text{ind}}}(\boldsymbol{x}). \tag{3.30}$$

Fitting a surrogate additive model has two advantages: 1) when h is additive, LIME will recover its ante-hoc feature attributions, 2) The additive surrogate can have a non-zero gap at $\boldsymbol{x}$.

SHAP The Shapley values arise from Cooperative Game Theory. Letting $\nu^{\text{int}}_{h,\boldsymbol{x},\mathcal{B}} : 2^{[d]} \to \mathbb{R}$ be the interventional game that takes $u \subseteq [d]$ and returns $\nu^{\text{int}}_{h,\boldsymbol{x},\mathcal{B}}(u) = \mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_u, \boldsymbol{z}_{-u})]$, the

Interventional Shapley Values are

$$\phi_j^{\text{SHAP-int}}(h, \boldsymbol{x}, \mathcal{B}) := \sum_{u \subseteq [d] \setminus \{j\}} W(|u|, d) \Big[\nu^{\text{int}}_{h,\boldsymbol{x},\mathcal{B}}(u \cup \{j\}) - \nu^{\text{int}}_{h,\boldsymbol{x},\mathcal{B}}(u) \Big] \tag{3.31}$$

where $W(|u|, d) := |u|!(d-|u|-1)!/d!$. The Shapley value of feature j is the weighted average contribution of adding the feature to any coalition u that excludes it. Equation 3.31 represents how Shapley values are typically computed in practice although its interpretation regarding the model h is not straightforward. This is because the Shapley Values are expressed as a complicated function of a coalitional game $\nu^{\text{int}}_{h,\boldsymbol{x},\mathcal{B}}$. Yet, our goal is to understand the original model h and not some abstract game applied to it. An alternative expression of the Shapley Values recently discussed by Herren and Hahn [2022]

$$\phi_j^{\text{SHAP-int}}(h, \boldsymbol{x}, \mathcal{B}) = \sum_{u \subseteq [d] : j \in u} \frac{h_{u,\mathcal{B}}(\boldsymbol{x})}{|u|} \tag{3.32}$$

reveals that SHAP values assign a local attribution to every feature by evenly sharing the u-way interactions of the Interventional Decomposition among all features involved. This is an important result because the functional decomposition $h_{u,\mathcal{B}}(\boldsymbol{x})$ is known to have desirable properties (*e.g.* minimality), which SHAP inherits by linearity. Equation 3.32 is well established in the game theory community, where the terms $h_{u,\mathcal{B}}(\boldsymbol{x})$ are known as the *Harsenyi Dividends* [Harsanyi, 1963]. Nonetheless, this expression of the Shapley values was not known by the XAI community until very recently [Bordt and von Luxburg, 2023, Herbinger et al., 2023, Hiabu et al., 2023]. For the proof that Equation 3.32 is equivalent to Equation 3.31, we refer to [Strumbelj and Kononenko, 2010, Theorem 2].

Breakdown As a reminder, the Breakdown feature attribution lets π be a permutation of $[d]$, and lets $\pi_{j:} := \{k \in [d] : \pi[k] < \pi[j]\}$ be all features that appear before j in the permutation. The attribution is [Staniak and Biecek, 2018]

$$\phi_j^{\text{BD-}\pi}(h, \boldsymbol{x}, \mathcal{B}) := \mathop{\mathbb{E}}_{\boldsymbol{z} \sim \mathcal{B}}[h(\boldsymbol{r}^{\boldsymbol{z}}_{\pi_{:j} \cup \{j\}}(\boldsymbol{x}))] - \mathop{\mathbb{E}}_{\boldsymbol{z} \sim \mathcal{B}}[h(\boldsymbol{r}^{\boldsymbol{z}}_{\pi_{:j}}(\boldsymbol{x}))]. \tag{3.33}$$

In light of the Möbius inverse (cf. Equation 3.18), Breakdown feature attributions can be expressed in terms of the Interventional Decomposition

$$\phi_j^{\text{BD-}\pi}(h, \boldsymbol{x}, \mathcal{B}) = \sum_{u \subseteq \pi_{:j}} h_{u \cup \{j\},\mathcal{B}}(\boldsymbol{x}). \tag{3.34}$$

It is apparent that Breakdown provides attributions to feature j by summing its main effect $h_{j,\mathcal{B}}(\boldsymbol{x})$ with all interactions between j and features in $\pi_{:j}$ (features that appear before j in the permutation). As previously noted in Chapter 2, if there are two permutations π, π' such that $\phi_j^{\text{BD-}\pi}(h, \boldsymbol{x}, \mathcal{B}) \neq \phi_j^{\text{BD-}\pi'}(h, \boldsymbol{x}, \mathcal{B})$, then the feature j must interact with some other feature. Equation 3.34 allows making this claim more formal : $\phi_j^{\text{BD-}\pi}(h, \boldsymbol{x}, \mathcal{B}) \neq \phi_j^{\text{BD-}\pi'}(h, \boldsymbol{x}, \mathcal{B})$ implies the existence of an interaction between j and a feature in the symmetric difference of the sets $\pi_{:j}$ and $\pi'_{:j}$.

Expected Gradient Assuming that $\mathcal{X} \subseteq \mathbb{R}^d$ and that $h \in \mathbb{C}^1(\mathcal{X})$ has continuous partial derivatives, the Expected Gradient attribution is

$$\phi_j^{\text{EG}}(h, \boldsymbol{x}, \mathcal{B}) = \mathop{\mathbb{E}}_{\substack{\boldsymbol{z} \sim \mathcal{B} \\ t \sim U(0,1)}} \left[(x_j - z_j) \frac{\partial h}{\partial x_j}((1-t)\boldsymbol{z} + t\boldsymbol{x}) \right]. \tag{3.35}$$

Expressing the EG in terms of the Interventional Decomposition

$$\phi_j^{\text{EG}}(h, \boldsymbol{x}, \mathcal{B}) = h_{j,\mathcal{B}_j \times \mathcal{B}_{-j}}(\boldsymbol{x}) + e(h), \tag{3.36}$$

is less elegant than previous LFAs due to its reliance on a parametric path between the baselines $\boldsymbol{z}$ and the evaluation point $\boldsymbol{x}$. It is only important to note that the correction

$$e(h) := \sum_{u \subseteq [d] : j \in u, |u| \geq 2} \mathop{\mathbb{E}}_{\substack{\boldsymbol{z} \sim \mathcal{B} \\ t \sim U(0,1)}} \left[(x_j - z_j) \frac{\partial h_{u,\mathcal{B}}}{\partial x_j}((1-t)\boldsymbol{z} + t\boldsymbol{x}) \right] \tag{3.37}$$

is null when the model is additive w.r.t feature j.

Unification of Post-hoc Local Feature Attributions

Various post-hoc LFAs introduced in the literature can be expressed in terms of Interventional Decompositions, which are an aggregate of $\boldsymbol{z}$-Anchored Decompositions.

- $\phi_j^{\text{PDP}}(h, \boldsymbol{x}, \mathcal{B}) = h_{j,\mathcal{B}}(\boldsymbol{x})$
- $\phi_j^{\text{LIME}}(h, \boldsymbol{x}, \mathcal{B}_{\text{ind}}) = h_{j,\mathcal{B}_{\text{ind}}}(\boldsymbol{x})$
- $\phi_j^{\text{ALE}}(h, \boldsymbol{x}, \mathcal{B}) = h_{j,\mathcal{B}_{\Omega^{[t]}}}(\boldsymbol{x})$ where $\boldsymbol{x} \in \Omega^{[t]} := \mathcal{X}_{-j} \times]\gamma_{t-1}, \gamma_t]$
- $\phi_j^{\text{SHAP-int}}(h, \boldsymbol{x}, \mathcal{B}) = \sum_{u \subseteq [d]: j \in u} h_{u,\mathcal{B}}(\boldsymbol{x}) \,/\, |u|$
- $\phi_j^{\text{BD-}\pi}(h, \boldsymbol{x}, \mathcal{B}) = \sum_{u \subseteq \pi_{:j}} h_{u \cup \{j\},\mathcal{B}}(\boldsymbol{x})$
- $\phi_j^{\text{EG}}(h, \boldsymbol{x}, \mathcal{B}) = h_{j,\mathcal{B}_j \times \mathcal{B}_{-j}}(\boldsymbol{x}) + e(h)$ where $e(g_j + g_{-j}) = 0$.

If the model is additive w.r.t feature j ($h(\boldsymbol{x}) = g_j(x_j) + g_{-j}(\boldsymbol{x}_{-j})\ \forall \boldsymbol{x} \in \mathcal{X}$), by minimality of the Interventional Decomposition, the PDP/LIME/SHAP/Breakdown/EG explainers all yield the attribution

$$\phi_j^{\text{LFA}}(h, \boldsymbol{x}, \mathcal{B}) = h_{j,\mathcal{B}}(\boldsymbol{x}) = g_j(x_j) - \mathop{\mathbb{E}}_{z_j \sim \mathcal{B}_j}[g_j(z_j)]. \tag{3.38}$$

This coincides with the ante-hoc additive explanations extracted from additive models (cf. Equation 2.44). Moreover, although ALE does not provide the exact same feature attribution as other methods, the curve $\phi_j^{\text{ALE}}(h, \boldsymbol{x}, \mathcal{B})$ plotted as a function of $x_j \in]\gamma_{t-1}, \gamma_t]$ is parallel to g_j.

3.2.2 Global Feature Importance

Partial Dependence Plots The variance of a PDP was proposed as an importance metric by Greenwell et al. [2018]

$$\Phi_j^{\text{PDP-Variance}}(h, \mathcal{B}) := \mathbb{V}_{\boldsymbol{x}\sim\mathcal{B}}[h_{j,\mathcal{B}}(\boldsymbol{x})]. \tag{3.39}$$

In terms of the Interventional Decomposition, this yields

$$\Phi_j^{\text{PDP-Variance}}(h, \mathcal{B}) = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{B}_j\times\mathcal{B}_{-j}}[h_{j,\mathcal{B}_j\times\mathcal{B}_{-j}}(\boldsymbol{x})^2]. \tag{3.40}$$

To avoid relying on the background $\mathcal{B}_j \times \mathcal{B}_{-j}$, which breaks correlation between j and other features, we propose to redefine the PDP importance as

$$\Phi_j^{\text{PDP},[2]}(h, \mathcal{B}) := \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}[\, h_{j,\mathcal{B}}(\boldsymbol{x})^2\,]. \tag{3.41}$$

That is, instead of computing the variance of the component $h_{j,\mathcal{B}}(\boldsymbol{x})$, we compute its squared amplitude.

SHAP It is common practice to aggregate local Shapley Values to yield global feature importance

$$\Phi_j^{\text{SHAP},[\text{p}]}(h, \mathcal{B}) := \left[\mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\Big[\,\Big|\sum_{u\subseteq[d]:j\in u} h_{u,\mathcal{B}}(\boldsymbol{x})\,/\,|u|\,\Big|^p\Big]\right]^{1/p}. \tag{3.42}$$

The default choice in the SHAP Python library [Lundberg and Lee, 2017] is $p = 1$ although other values of p could be reasonable.

Permutation Feature Importance The original definition of PFI is [Gregorutti et al., 2017]

$$\Phi_j^{\text{PFI-O}}(h, \mathcal{D}) := \mathop{\mathbb{E}}_{\substack{(\boldsymbol{x},y)\sim\mathcal{D}\\(\boldsymbol{z},y')\sim\mathcal{D}}}\Big[\,\Big(h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j) - y\Big)^2\,\Big] - \mathop{\mathbb{E}}_{(\boldsymbol{x},y)\sim\mathcal{D}}\Big[\,\Big(h(\boldsymbol{x}) - y\Big)^2\,\Big], \tag{3.43}$$

which compares the model performance on the original data and on synthetic data where feature j is replaced by a sample from the marginal. This replacement is typically estimated by permuting the jth column of the data matrix $\boldsymbol{X}$, justifying the terminology Permutation Feature Importance. This feature importance can be expressed in terms of Interventional Decompositions under certain assumptions on the target y.

Proposition 3.2.2. *Assume that $y = h(\boldsymbol{x}) + \epsilon$ where ϵ is a random variable that is independent of $\boldsymbol{x}$, $\mathbb{E}[\epsilon] = 0$, and $\mathbb{E}[\epsilon^2] = \sigma^2$. Then,*

$$\begin{aligned}\Phi_j^{\text{PFI-O}}(h, \mathcal{B}) &= \mathop{\mathbb{E}}_{\substack{\boldsymbol{x}\sim\mathcal{B}\\ \boldsymbol{z}\sim\mathcal{B}}}\Big[\Big(h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j) - h(\boldsymbol{x})\Big)^2\Big] \\ &= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}_j\times\mathcal{B}_{-j}}\Big[\Big(\sum_{u\subseteq[d]:j\in u} h_{u,\mathcal{B}_j\times\mathcal{B}_{-j}}(\boldsymbol{x})\Big)^2\Big] + \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\Big[\Big(\sum_{u\subseteq[d]:j\in u} h_{u,\mathcal{B}}(\boldsymbol{x})\Big)^2\Big].\end{aligned} \tag{3.44}$$

The proof is presented in Appendix B.1.

This expression of the PFI involves **two** Interventional Decompositions using backgrounds $\mathcal{B}_j \times \mathcal{B}_{-j}$ and $\mathcal{B}$ respectively. This explains why PFI falls back to **twice** the variance of h_j when the model is additive (cf. Equation 2.68). These two decompositions have their respective strengths and weaknesses. The first term

$$\Phi_j^{\text{Marginal-Sobol}}(h, \mathcal{B}) = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}_j\times\mathcal{B}_{-j}}\Big[\Big(\sum_{u\subseteq[d]:j\in u} h_{u,\mathcal{B}_j\times\mathcal{B}_{-j}}(\boldsymbol{x})\Big)^2\Big] \tag{3.45}$$

is the Marginal Sobol Index (cf. Equation 2.69) and is invariant to the correlation structure between j and $-j$. Therefore, this importance score cannot be altered/manipulated by varying feature correlation. Yet the expectation $\boldsymbol{x} \sim \mathcal{B}_j \times \mathcal{B}_{-j}$ is likely to be outside the data distribution and so extrapolation may become an issue. The second term $\mathbb{E}_{\boldsymbol{x}\sim\mathcal{B}}\Big[\Big(\sum_{u\subseteq[d]:j\in u} h_{u,\mathcal{B}}(\boldsymbol{x})\Big)^2\Big]$ has never yet been proposed in the XAI literature. In opposition to the Marginal Sobol Index, this functional could have reduced extrapolation due to the average $\boldsymbol{x} \sim \mathcal{B}$. In an effort to reduce data extrapolation, we propose reformulate the PFI so that it does not involve the background $\mathcal{B}_j \times \mathcal{B}_{-j}$.

$$\Phi_j^{\text{PFI}}(h, \mathcal{D}) := \mathop{\mathbb{E}}_{(\boldsymbol{x},y)\sim\mathcal{D}}\Big[\Big(\mathop{\mathbb{E}}_{(\boldsymbol{z},y')\sim\mathcal{D}}\Big[h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j)\Big] - y\Big)^2\Big] - \mathop{\mathbb{E}}_{(\boldsymbol{x},y)\sim\mathcal{D}}\Big[\Big(h(\boldsymbol{x}) - y\Big)^2\Big]. \tag{3.46}$$

Crucially, we moved the expectation w.r.t the noise sample $\boldsymbol{z}$ inside the square $(\cdot)^2$. This definition is still in line with the high-level idea of *replacing a feature with noise and reporting the impact on performance*. However, we now average the model predictions on noisy samples before comparing them to the labels. Under the same noise assumptions behind Proposition 3.2.2, one can show that

$$\Phi_j^{\text{PFI}}(h, \mathcal{B}) = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\left[\Big(h(\boldsymbol{x}) - \mathop{\mathbb{E}}_{z\sim\mathcal{B}}[h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j)]\Big)^2\right] = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\left[\left(\sum_{u\subseteq[d]:j\in u} h_{u,\mathcal{B}}(\boldsymbol{x})\right)^2\right]. \tag{3.47}$$

This alternative formulation of the PFI involves a single Interventional Decomposition and

attributes importance to j by summing all interactions terms that involve said feature.

Sobol & CPFI The Total Sobol Index is [Hooker et al., 2021]

$$\Phi_j^{\text{Total-Sobol}}(h, \mathcal{B}) = \mathop{\mathbb{E}}_{\boldsymbol{x}_{-j}} \Big[\mathop{\mathbb{V}}_{x_j}[h(\boldsymbol{x})|\boldsymbol{x}_{-j}] \Big]. \tag{3.48}$$

To be able to express this quantity in terms of functional decompositions, we shall rely on the Conditional-PFI approaches developed in [Molnar et al., 2023, Strobl et al., 2008]. In this work, a partition $(\Omega_{-j}^{[1]}, \ldots, \Omega_{-j}^{[M]})$ of $\mathcal{X}_{-j}$ is learned and the background $\mathcal{B}_{\Omega^{[t]}}$ is defined by restricting $\mathcal{B}$ to the subset $\Omega^{[t]} := \Omega_{-j}^{[t]} \times \mathcal{X}_j$. The proposed functional is

$$\Phi_j^{\text{CPFI}}(h, \mathcal{B}) := \sum_{t=1}^{M} \mathcal{B}(\Omega^{[t]}) \times \Phi_j^{\text{PFI-O}}(h, \mathcal{B}_{\Omega^{[t]}}), \tag{3.49}$$

According to Proposition 3.2.2, this functional involves the Marginal-Sobol and PFI importance

$$\Phi_j^{\text{CPFI}}(h, \mathcal{B}) = \sum_{t=1}^{M} \mathcal{B}(\Omega^{[t]}) \times \Big[\Phi_j^{\text{Marginal-Sobol}}(h, \mathcal{B}_{\Omega^{[t]}}) + \Phi_j^{\text{PFI}}(h, \mathcal{B}_{\Omega^{[t]}}) \Big], \tag{3.50}$$

both of which are easily expressed in terms of functional decompositions (cf. Equation 3.45 & 3.47). But how can we relate this quantity to the Total Sobol Index? Establishing the connection requires making an independence assumption between x_j and $\boldsymbol{x}_{-j}$ conditioned on $\boldsymbol{x}_{-j} \in \Omega_{-j}^{[t]}$.

Proposition 3.2.3. *Let $(\Omega_{-j}^{[1]}, \ldots, \Omega_{-j}^{[M]})$ be a partition of $\mathcal{X}_{-j}$ and define the corresponding regions $\Omega^{[t]} := \Omega_{-j}^{[t]} \times \mathcal{X}_j$. If $x_j \perp \boldsymbol{x}_{-j} | \{\boldsymbol{x}_{-j} \in \Omega_{-j}^{[t]}\}$ for all $t = 1, 2, \ldots M$, it holds that*

$$\Phi_j^{\textit{Total-Sobol}}(h, \mathcal{B}) = \frac{1}{2}\Phi_j^{\textit{CPFI}}(h, \mathcal{B}) = \sum_{t=1}^{M} \mathcal{B}(\Omega^{[t]})\, \Phi_j^{\textit{Marginal-Sobol}}(h, \mathcal{B}_{\Omega^{[t]}}) = \sum_{t=1}^{M} \mathcal{B}(\Omega^{[t]})\, \Phi_j^{\textit{PFI}}(h, \mathcal{B}_{\Omega^{[t]}}). \tag{3.51}$$

The proof is presented in Appendix B.1.

This proposition suggests that computing regional Interventional Decompositions using a fine-grained partition of $\mathcal{X}_{-j}$ can estimate the Total Sobol Index.

Unification of Post-hoc Global Feature Importance

The PDP, SHAP, PFI, Marginal-Sobol, and PFI-O feature importance can be expressed in terms of the Interventional Decomposition

- $\Phi_j^{\text{PDP-Variance}}(h, \mathcal{B}) = \mathbb{E}_{\boldsymbol{x} \sim \mathcal{B}_j \times \mathcal{B}_{-j}}[h_{j,\mathcal{B}_j \times \mathcal{B}_{-j}}(\boldsymbol{x})^2]$
- $\Phi_j^{\text{PDP},[2]}(h, \mathcal{B}) = \mathbb{E}_{\boldsymbol{x} \sim \mathcal{B}}[\, h_{j,\mathcal{B}}(\boldsymbol{x})^2 \,]$
- $\Phi_j^{\text{SHAP},[\text{p}]}(h, \mathcal{B}) = \Big[\, \mathbb{E}_{\boldsymbol{x} \sim \mathcal{B}} \Big[\, \Big| \sum_{u \subseteq [d]: j \in u} h_{u,\mathcal{B}}(\boldsymbol{x}) \,/\, |u| \, \Big|^p \,\Big] \,\Big]^{1/p}$
- $\Phi_j^{\text{PFI}}(h, \mathcal{B}) = \mathbb{E}_{\boldsymbol{x} \sim \mathcal{B}} \Big[\Big(\sum_{u \subseteq [d]: j \in u} h_{u,\mathcal{B}}(\boldsymbol{x}) \Big)^2 \Big]$
- $\Phi_j^{\text{Marginal-Sobol}}(h, \mathcal{B}) = \mathbb{E}_{\boldsymbol{x} \sim \mathcal{B}_j \times \mathcal{B}_{-j}} \Big[\Big(\sum_{u \subseteq [d]: j \in u} h_{u,\mathcal{B}_j \times \mathcal{B}_{-j}}(\boldsymbol{x}) \Big)^2 \Big]$
- $\Phi_j^{\text{PFI-O}}(h, \mathcal{B}) = \Phi_j^{\text{Marginal-Sobol}}(h, \mathcal{B}) + \Phi_j^{\text{PFI}}(h, \mathcal{B})$.

This unification has several implications.

1) If the model is additive w.r.t feature j ($h(\boldsymbol{x}) = g_j(x_j) + g_{-j}(\boldsymbol{x}_{-j})\ \forall \boldsymbol{x} \in \mathcal{X}$), by minimality of the Interventional Decomposition, the PDP-Variance, PDP-[2], SHAP-[2], PFI, Marginal-Sobol, and PFI-O/2 on the global importance of feature j and yield

$$\Phi_j^{\text{GFI}}(h, \mathcal{B}) = \mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mathcal{B}}[h_{j,\mathcal{B}}(\boldsymbol{x})^2] = \mathop{\mathbb{E}}_{x_j \sim \mathcal{B}_j}[\,(\, g_j(x_j) - \mathop{\mathbb{E}}_{z_j \sim \mathcal{B}_j}[g_j(z_j)]\,)^2\,]. \tag{3.52}$$

This coincides with the ante-hoc global feature importance of an additive model (cf. Equation 2.45).

2) PDP-Variance agrees with PDP-[2] and Marginal-Sobol agrees with PFI when feature x_j is independent of $\boldsymbol{x}_{-j}$ in the distribution $\mathcal{B}$.

3) Given a partition $(\Omega_{-j}^{[1]}, \ldots, \Omega_{-j}^{[M]})$ of $\mathcal{X}_{-j}$ and corresponding regions $\Omega^{[t]} := \Omega_{-j}^{[t]} \times \mathcal{X}_j$, if $x_j \perp \boldsymbol{x}_{-j} | \{\boldsymbol{x}_{-j} \in \Omega_{-j}^{[t]}\}$ for all $t = 1, 2, \ldots M$, then the Total-Sobol and CPFI are related to regional Marginal-Sobol and PFI feature importance

$$\begin{aligned} \Phi_j^{\text{Total-Sobol}}(h, \mathcal{B}) &= \frac{1}{2} \Phi_j^{\text{CPFI}}(h, \mathcal{B}) \\ &= \sum_{t=1}^{M} \mathcal{B}(\Omega^{[t]})\, \Phi_j^{\text{Marginal-Sobol}}(h, \mathcal{B}_{\Omega^{[t]}}) = \sum_{t=1}^{M} \mathcal{B}(\Omega^{[t]})\, \Phi_j^{\text{PFI}}(h, \mathcal{B}_{\Omega^{[t]}}). \end{aligned} \tag{3.53}$$

3.2.3 Interaction Quantification

Disagreement among additive explanation techniques can arise from interactions and therefore it is interesting to quantify the presence of interactions.

H^2 statistics The pairwise H^2 statistic measures the interaction strength between features j and k [Friedman and Popescu, 2008]

$$\Phi_{jk}^{\text{Inter}}(h, \mathcal{B}) := \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\Big[\Big(\mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_{\{j,k\}}, \boldsymbol{z}_{-\{j,k\}}) - h(\boldsymbol{x}_j, \boldsymbol{z}_{-j}) - h(\boldsymbol{x}_k, \boldsymbol{z}_{-k}) + h(\boldsymbol{z})]\Big)^2\Big]. \tag{3.54}$$

The Möbius Inverse (cf. Equation 3.18) reveals that this is simply the squared amplitude of the $h_{\{j,k\},\mathcal{B}}$ component

$$\Phi_{jk}^{\text{Inter}}(h, \mathcal{B}) = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}[\, h_{\{j,k\},\mathcal{B}}(\boldsymbol{x})^2\,]. \tag{3.55}$$

The one-vs-all H^2 statistic reports the strength of interactions between j and all other features [Friedman and Popescu, 2008]

$$\Phi_{j:}^{\text{Inter}}(h, \mathcal{B}) := \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\left[\Big(\mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}) - h(\boldsymbol{x}_j, \boldsymbol{z}_{-j}) - h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j) + h(\boldsymbol{z})]\Big)^2\right]. \tag{3.56}$$

Straight-forward manipulations involving the Möbius Inverse yield

$$\Phi_{j:}^{\text{Inter}}(h, \mathcal{B}) = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\left[\Big(\sum_{u\subseteq[d]:|u|\geq 2, j\in u} h_{u,\mathcal{B}}(\boldsymbol{x})\Big)^2\right]. \tag{3.57}$$

Shapley Taylor The Shapley-Taylor index can actually be expressed as [Sundararajan et al., 2020]

$$\phi_{jk}^{\text{SHAPT}}(h, \boldsymbol{x}, \mathcal{B}) = \begin{cases} h_{j,\mathcal{B}}(\boldsymbol{x}) & \text{if } j = k \\ \sum_{\substack{u\subseteq[d] \\ \{j,k\}\subseteq u}} h_{u,\mathcal{B}}(\boldsymbol{x})\binom{|u|}{2}^{-1} & \text{if } j \neq k. \end{cases} \tag{3.58}$$

That is, the u-way interactions are shared evenly between all *pairs* of features involved.

Contributions

To conclude this Chapter, we expressed all post-hoc explainers from the XAI literature as functions involving the *Interventional Decomposition* $\{h_{u,\mathcal{B}}\}_{u\subseteq[d]}$ relative to some background distribution $\mathcal{B}$. This functional decomposition is novel and it was shown to respect an important property : minimality. Expressing post-hoc exlainers in terms of said the Interventional Decomposition reveals that they disagree because they have different schemes of sharing interactions to the individual features involved. If the model is additive, interactions are null by minimality and all post-hoc explainers agree with the ante-hoc explanation.

This unification also implies that any software that can rapidly calculate $\boldsymbol{z}$-Anchored Decompositions $\{h_{u,\boldsymbol{z}}\}_{u\subseteq[d]}$ can compute any additive explanations. The two coming Chapters demonstrate how to efficiently compute $\boldsymbol{z}$-Anchored Decompositions in *model-agnostic* and *model-specific* settings, respectively.

CHAPTER 4 MODEL-AGNOSTIC ESTIMATES

4.1 Computing a Functional Component

This chapter tackles the challenge of computing Anchored/Interventional Decompositions of functions while remaining agnostic to their internal structure. That is, we are only allowed to generate an input $\boldsymbol{x}$ and evaluate $h(\boldsymbol{x})$. All presented algorithms, except for VIN and PermutationSHAP, are novel contributions.

4.1.1 Anchored

A single component from the $\boldsymbol{z}$-Anchored Decomposition is calculated as follows

$$h_{u,\boldsymbol{z}}(\boldsymbol{x}) = h(\boldsymbol{x}_u, \boldsymbol{z}_{-u}) - \sum_{v \subset u} h_{v,\boldsymbol{z}}(\boldsymbol{x}). \tag{4.1}$$

Assuming the components $\{h_{v,\boldsymbol{z}}(\boldsymbol{x})\}_{v \subset u}$ have already been obtained, the u component requires a single function evaluation $h(\boldsymbol{x}_u, \boldsymbol{z}_{-u})$ and $2^{|u|} - 1$ subtractions. Since we are agnostic to the structure of h, the only way to rapidly compute Equation 4.1 is to *not compute it* when the result is known in advance to be zero. A simple way to detect null components a priori is to leverage so-called "annihilation property"

Definition 4.1.1 (Annihilation Property [Kuo et al., 2010])**.** *The $\boldsymbol{z}$-Anchored Decomposition respects*

$$v \cap u \neq \emptyset \;\Rightarrow\; h_{u,\boldsymbol{z}}(\boldsymbol{x}_{-v}, \boldsymbol{z}_v) = 0. \tag{4.2}$$

Said otherwise, $h_{u,\boldsymbol{z}}(\boldsymbol{x})$ evaluates to zero whenever any component of its input is equal to z_j for some j in u.

When some features have low cardinality ($\mathcal{X}_j = [\text{Monday}, \text{Tuesday}, \ldots, \text{Sunday}]$) or low variability (edge pixels of a MNIST image) the "annihilation property" is likely to occur, and function calls $h(\boldsymbol{x}_u, \boldsymbol{z}_{-u})$ with $j \in u$ can be avoided. Algorithm 1 illustrates this optimization.

In full generality, we consider a list of anchors $\{\boldsymbol{z}^{(j)}\}_{j=1}^{M}$ and a list of evaluation points $\{\boldsymbol{x}^{(i)}\}_{i=1}^{N}$ for the decompositions. The results for the h_u component can be aggregated in a $N \times M$ matrix $\boldsymbol{H}^u$

$$H^u_{ij} := h_{u,\boldsymbol{z}^{(j)}}(\boldsymbol{x}^{(i)}). \tag{4.3}$$

The first index of this matrix is the evaluation point index while the second index is the anchor index. Computing such a matrix takes time $\mathcal{O}(N \times M \times 2^{|u|})$.

Algorithm 1 Compute a $\boldsymbol{z}$-Anchored Component $h_{u,\boldsymbol{z}}(\boldsymbol{x})$

```
procedure GetComponent(h, u, x, z, {h_{v,z}(x)}_{v⊂u})
    if ∃k ∈ u such that x_k = z_k then
        % Annihilation Property
        return 0;
    else
        h_{u,z}(x) = h(x_u, z_{-u});
        for v ⊂ u do
            h_{u,z}(x) -= h_{v,z}(x);
        return h_{u,z}(x);
```

4.1.2 Interventional

The $\boldsymbol{z}$-Anchored Decomposition involves two inputs 1) the anchor $\boldsymbol{z}$ around which the decomposition is done 2) the evaluation point $\boldsymbol{x}$ at which the decomposition is evaluated. The notation $h_{u,\boldsymbol{z}}(\boldsymbol{x})$ highlights the different roles of each input. A similar distinction must also be applied when these inputs are sampled from a random distribution. The anchor is sampled from a background $\boldsymbol{z} \sim \mathcal{B}$ while the evaluation point is sampled from a foreground $\boldsymbol{x} \sim \mathcal{F}$. Most applications (but not all) will have $\mathcal{B} = \mathcal{F}$. Examples of situations where $\mathcal{B} \neq \mathcal{F}$ include fairness where two distinct distributions might separate men from women, and scenarios where a detailed plot $h_{u,\mathcal{B}}(\boldsymbol{x})$ is required and so $\boldsymbol{x}$ is chosen from a grid.

If the anchors are sampled iid from the background $\{\boldsymbol{z}^{(j)}\}_{j=1}^{M} \sim \mathcal{B}^M$, then averaging $\boldsymbol{H}^u$ along columns is a consistent estimate of the Interventional Decomposition.

$$\frac{1}{M}\sum_{j=1}^{M} H_{ij}^{u} \xrightarrow{p} h_{u,\mathcal{B}}(\boldsymbol{x}^{(i)}). \tag{4.4}$$

We have seen previously that the PDP, LIME, and ALE explainers can be formulated as main effects $h_{j,\mathcal{B}}$ using different backgrounds $\mathcal{B}$, $\mathcal{B}_{\text{ind}}$, and $\mathcal{B}_{\Omega^{[t]}}$. Consequently, computing the matrices $\{\boldsymbol{H}^k\}_{k=1}^{d}$ is sufficient to estimate PDP, LIME, and ALE local feature attributions.

But how good are these estimates given that only M anchors are sampled? Assuming the function output is bounded, $h_{u,\boldsymbol{z}}(\boldsymbol{x})$ must have finite moments w.r.t the random variable $\boldsymbol{z}$. As a result, the Central Limit Theorem [Owen, 2013, Theorem 2.1] is applicable and allows one to compute asymptotic confidence intervals

Theorem 4.1.1 (Decomposition Confidence Interval)**.** *Let* $\{\boldsymbol{z}^{(j)}\}_{j=1}^{M} \sim \mathcal{B}^M$ *be a sequence of* M *iid background observations, moreover let* $\mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[\, h_{u,\boldsymbol{z}}(\boldsymbol{x})\,]$ *and* $\mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[\, h_{u,\boldsymbol{z}}(\boldsymbol{x})^2\,]$ *be finite for any* $\boldsymbol{x} \in \mathcal{X}$*, then the following holds for any* $i \in [N]$ *and* $\delta \in]0,1[$

$$\lim_{M\to\infty} \mathbb{P}\left(\left| 1/M \sum_{j=1} H_{ij}^{u} - h_{u,\mathcal{B}}(\boldsymbol{x}^{(i)}) \right| \geq F^{-1}_{\mathcal{N}(0,1)}(1-\delta/2)) \frac{s_M^{(i)}}{\sqrt{M}} \right) \leq \delta,$$

where $F^{-1}_{\mathcal{N}(0,1)}$ *is the inverse Cumulative Distribution Function (CDF) of the standard normal distribution, and* $s_M^{(i)} = [1/M \sum_{j=1}^{M} (H_{ij}^{u} - \frac{1}{M}\sum_{\ell=1}^{M} H_{i\ell}^{u})^2]^{1/2}$ *is the sample variance.*

Averaging the matrix $\boldsymbol{H}^u$ along its second axis leads to consistent estimates of the Interventional Decomposition component $h_{u,\mathcal{B}}(\boldsymbol{x}^{(i)})$. By curiosity, what happens when averaging along the first axis?

Proposition 4.1.1 (Duality)**.** *Let* $\{\boldsymbol{x}^{(i)}\}_{i=1}^{N} \sim \mathcal{F}^N$ *be a sequence of* N *iid foreground observations, then the following holds*

$$\frac{(-1)^{|u|}}{N} \sum_{i=1}^{N} H_{ij}^{u} \xrightarrow{p} \sum_{v\subseteq[d]:u\subseteq v} h_{v,\mathcal{F}}(\boldsymbol{z}^{(j)}). \tag{4.5}$$

The proof in presented in Appendix B.2.

By averaging along the other axis, the roles of anchor and evaluation point have switched and the results $\sum_{v\subseteq[d]:u\subseteq v} h_{u,\mathcal{F}}(\boldsymbol{z}^{(j)})$ quantifies **all** high-order interactions that involve u. This Duality has several practical implications.

When $\mathcal{B} = \mathcal{F}$ and $\boldsymbol{z}^{(i)} = \boldsymbol{x}^{(i)}$ for $i = 1, 2, \ldots, N$, the $\boldsymbol{H}^k$ matrix has size $N \times N$ and stores the elements $H_{ij}^{k} = h_{k,\boldsymbol{x}^{(j)}}(\boldsymbol{x}^{(i)})$. Averaging the matrix along axis 2 yields a consistent estimate of the PDP $h_{k,\mathcal{B}}(\boldsymbol{x}^{(i)})$ while averaging it along axis 1 yields a consistent estimate of $\sum_{u\in[d]:k\in u} h_{k,\mathcal{B}}(\boldsymbol{x}^{(i)})$ which is linked to the PFI (cf. Equation 3.47). Hence, by computing an additive decomposition $\{\boldsymbol{H}^k\}_{k=1}^{d}$ one can already extract estimates of the PDP and PFI importance. In fact, given the relationship between PFI, PFI-O and Marginal-Sobol highlighted

in Proposition 3.2.2, one can estimate most importance measures using $\{\boldsymbol{H}^k\}_{k=1}^d$

$$\begin{aligned}
\frac{1}{N}\sum_{i=1}^{N}\Big(\frac{1}{N}\sum_{j=1}^{N}\Big(H_{ij}^k-\frac{1}{N}\sum_{\ell=1}^{N}H_{\ell j}^k\Big)\Big)^2 &\xrightarrow{p}\Phi_k^{\text{PDP-Variance}}(h,\mathcal{B})\\
\frac{1}{N}\sum_{i=1}^{N}\Big(\frac{1}{N}\sum_{j=1}^{N}H_{ij}^k\Big)^2 &\xrightarrow{p}\Phi_k^{\text{PDP,[2]}}(h,\mathcal{B})\\
\frac{1}{N}\sum_{j=1}^{N}\Big(\frac{1}{N}\sum_{i=1}^{N}H_{ij}^k\Big)^2 &\xrightarrow{p}\Phi_k^{\text{PFI}}(h,\mathcal{B})\\
\frac{1}{N^2}\sum_{j=1}^{N}\sum_{i=1}^{N}\Big(H_{ij}^k-\frac{1}{N}\sum_{\ell=1}^{N}H_{\ell j}^k\Big)^2 &\xrightarrow{p}\Phi_k^{\text{Marginal-Sobol}}(h,\mathcal{B})\\
\frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}(H_{ij}^k)^2 &\xrightarrow{p}\Phi_k^{\text{PFI-O}}(h,\mathcal{B}).
\end{aligned}\tag{4.6}$$

Under the assumptions of Proposition 3.2.3, the Total-Sobol and CPFI feature importance can be calculated by computing regional Marginal-Sobol or PFI. To conclude, the matrices $\{\boldsymbol{H}^k\}_{k=1}^d$ are all you need to compute feature importance.

4.1.3 Experiments

The following toy experiment demonstrates how the $\{\boldsymbol{H}^k\}_{k=1}^d$ matrices can estimate the various GFI methods unified in Sections 3.2.2. The example is taken directly from [Bénard et al., 2021, Appendix 1] because closed-forms for the GFI are already provided. The input $\boldsymbol{x}\in\mathbb{R}^5$ follows a multivariate Gaussian $\mathcal{B}:=\mathcal{N}(\mathbf{0},\boldsymbol{\Sigma})$ with a covariance matrix that is 1 on the diagonal, and 0 everywhere except for $\Sigma_{1,2}=\Sigma_{2,1}=\rho_{1,2}$ and $\Sigma_{4,5}=\Sigma_{5,4}=\rho_{4,5}$. The model under study is

$$h(\boldsymbol{x})=\alpha x_1x_2\,\mathbb{1}[x_3\geq 0]+\beta x_4x_5\,\mathbb{1}[x_3<0].\tag{4.7}$$

This simple model is already hard to explain because it involves strong 3-way interactions. Bénard et al. [2021] have demonstrated that

$$\begin{aligned}
\Phi^{\text{PDP-Variance}}(h,\mathcal{B})&=1/2[0,0,\,(\alpha\rho_{1,2}-\beta\rho_{4,5})^2/2,\,0,0]^T\\
\Phi^{\text{PDP-[2]}}(h,\mathcal{B})&=1/2[\alpha^2\rho_{1,2}^2,\alpha^2\rho_{1,2}^2,\,(\alpha\rho_{1,2}-\beta\rho_{4,5})^2/2,\,\beta^2\rho_{4,5}^2,\beta^2\rho_{4,5}^2]^T
\end{aligned}\tag{4.8}$$

$$
\begin{aligned}
\Phi^{\text{PFI}}(h, \mathcal{B}) &= 1/2[\alpha^2(1+2\rho_{1,2}^2), \alpha^2(1+2\rho_{1,2}^2), \\
&\qquad \alpha^2/2(1+\rho_{1,2}^2) + \beta^2/2(1+\rho_{4,5}^2) + (\alpha\rho_{1,2} - \beta\rho_{4,5})^2/2, \\
&\qquad \beta^2(1+2\rho_{4,5}^2), \beta^2(1+2\rho_{4,5}^2)]^T \\
\Phi^{\text{Marginal-Sobol}}(h, \mathcal{B}) &= 1/2[\alpha^2, \alpha^2, \\
&\qquad \alpha^2/2(1+\rho_{1,2}^2) + \beta^2/2(1+\rho_{4,5}^2) + (\alpha\rho_{1,2} - \beta\rho_{4,5})^2/2, \\
&\qquad \beta^2, \beta^2]^T \\
\Phi^{\text{Total-Sobol}}(h, \mathcal{B}) &= 1/2[\alpha^2(1-\rho_{1,2}^2), \alpha^2(1-\rho_{1,2}^2), \\
&\qquad \alpha^2/2(1+\rho_{1,2}^2) + \beta^2/2(1+\rho_{4,5}^2) + (\alpha\rho_{1,2} - \beta\rho_{4,5})^2/2, \\
&\qquad \beta^2(1-\rho_{4,5}^2), \beta^2(1-\rho_{4,5}^2)]^T.
\end{aligned}
\tag{4.9}
$$

None of the feature importance agree because of feature correlations and interactions. In fact, PDP-Variance and PDP-[2] attribute the same importance to feature x_3 (because it is uncorrelated) but disagree on the importance of the remaining correlated features. The PDP-[2] inflates their importance with correlation while the PDP-Variance assigns them none.

Also, PFI, Marginal-Sobol, and Total-Sobol agree on the importance of x_3 (because it is uncorrelated) but disagree on the importance of the remaining correlated features. PFI inflates the importance α^2 of x_1 with correlation, Total-Sobol deflates the importance with correlation, and Marginal-Sobol leaves it intact.

To estimate these various feature importance, we fixed $\alpha = 1$ and $\beta = 2$, and then varied the correlations $\rho_{1,2}, \rho_{4,5} = 0.2, 0.5, 0.75$. For 20 random seeds and for each of the nine correlation scenarios, we sampled N synthetic data points, computed their $N \times N$ functional decomposition matrices $\{\boldsymbol{H}^k\}_{k=1}^d$, and applied Equation 4.6. The error between the ground truth importance $\boldsymbol{\Phi}$ and its estimate $\widehat{\boldsymbol{\Phi}}$ was measured using the Euclidean distance

$$
D(\boldsymbol{\Phi}, \widehat{\boldsymbol{\Phi}}) := \|\boldsymbol{\Phi}(h, \mathcal{B}) - \widehat{\boldsymbol{\Phi}}(h, \mathcal{B})\|_2^2. \tag{4.10}
$$

These errors are reported as a function of $N = 100, \ldots, 5000$ for PDP-Variance/PDP-[2] in Figure 4.1 and for Marginal-Sobol/PFI in Figure 4.2. The errors decrease as N increases, which confirms the validity of Equation 4.6.

Traditional explainability software would require a separate API call for computing each feature importance. This might introduce unnecessary abstractions and could cause practitioners to treat the explainer as "another black-box". In constrast, our unification of global feature importance sheds light on the nature of each techniques: they are simply different means of aggregating components stored in the $\{\boldsymbol{H}^k\}_{k=1}^d$ matrices.

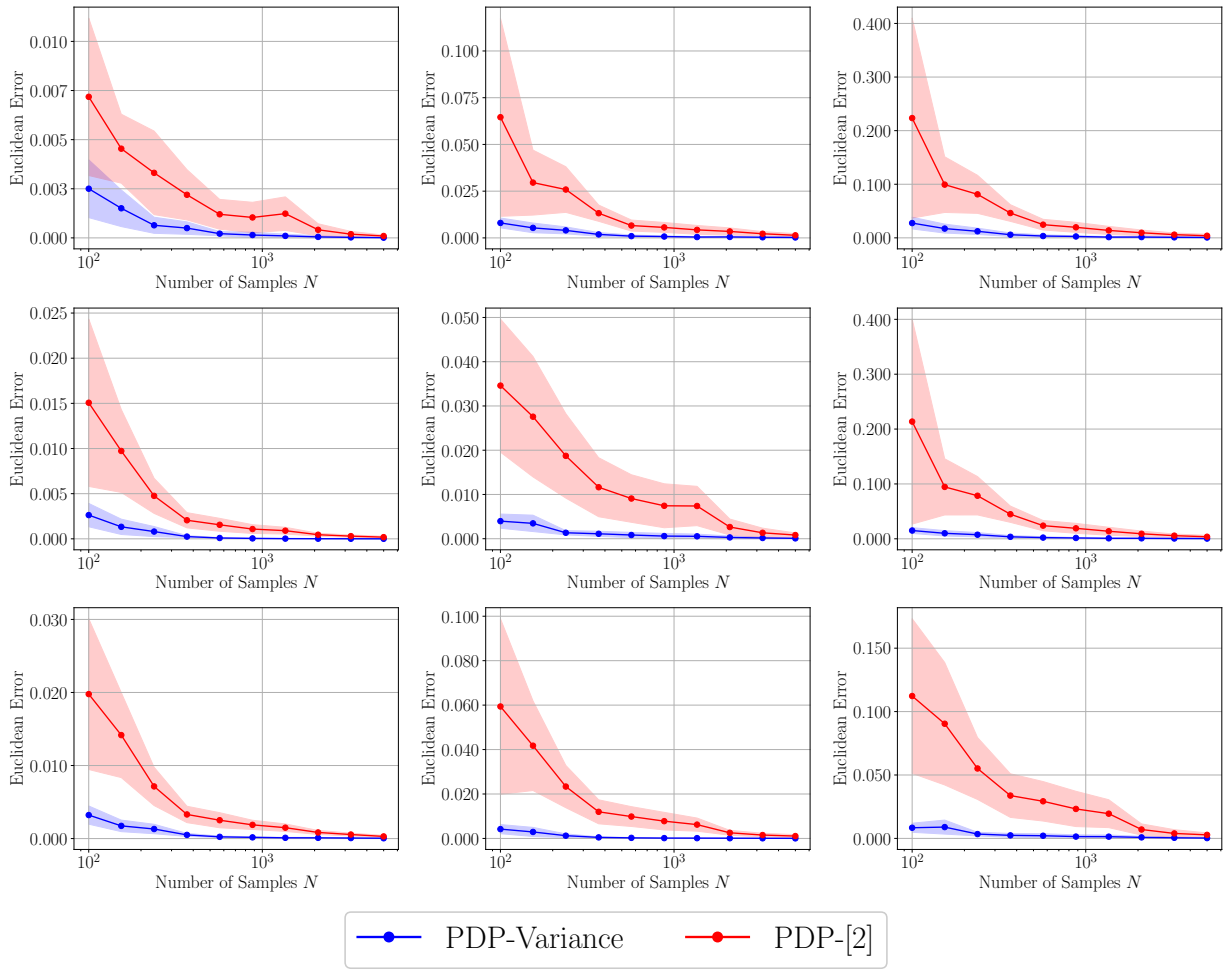


Figure 4.1 Convergence of PDP-Variance and PDP-[2]. The three columns indicate correlation $\rho_{45} = 0.2, 0.5, 0.75$ while the three rows indicate correlation $\rho_{12} = 0.2, 0.5, 0.75$.

We are left with estimating the Total-Sobol importance. As discussed previously in Page 44, Theorem 2 of [Hooker et al., 2021] states that Total-Sobol can be estimated by removing feature x_j from the dataset, and refitting a model on the target y. Currently, the target is $h(\boldsymbol{x})$ and so the Remove-and-Retrain (RaR) estimate is

$$\widehat{\Phi}_j^{\text{RaR}}(h, \mathcal{B}) = \mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mathcal{B}}[(h(\boldsymbol{x}) - \widehat{h}_{-j}(\boldsymbol{x}_{-j}))^2], \tag{4.11}$$

where $\widehat{h}_{-j}(\boldsymbol{x}_{-j})$ is a regressor trained with squared loss to predict $h(\boldsymbol{x})$ given $\boldsymbol{x}_{-j}$. RaR estimates are highly sensitive to the performance of the model $\widehat{h}_{-j}(\boldsymbol{x}_{-j})$: it must be flexible enough to represent conditional expectations, but it must not overfit the data. Moreover, practitioners cannot be expected to optimize the hyperparameters for d additional models so the default Scikit-Learn hyperparameters are employed. Figure 4.3 presents the slow

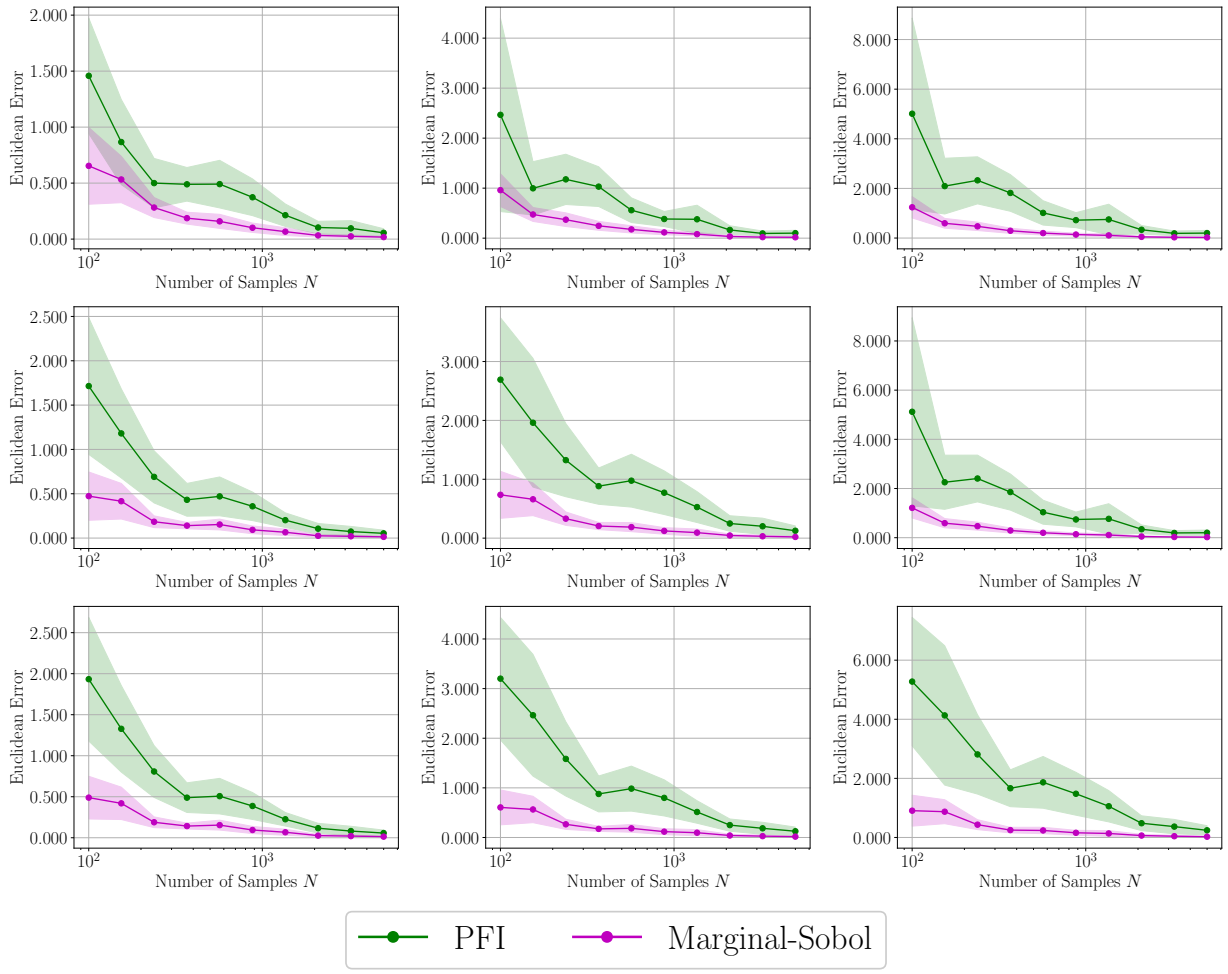


Figure 4.2 Convergence of Marginal-Sobol and PFI. The three columns indicate correlation $\rho_{45} = 0.2, 0.5, 0.75$ while the three rows indicate correlation $\rho_{12} = 0.2, 0.5, 0.75$.

convergence of RaR estimates using GBTs and MLPs to learn $\widehat{h}_{-j}(\boldsymbol{x}_{-j})$.

Proposition 3.2.3 suggests an alternative estimate of the Total-Sobol importance : the Conditional-PFI. Indeed, if the partition of $\mathcal{X}_{-j}$ is fine enough so that feature x_j is independent of $\boldsymbol{x}_{-j}$ given $\boldsymbol{x}_{-j} \in \Omega^{[t]}_{-j}$, then the CPFI will accurately estimate the Total-Sobol importance. Molnar et al. [2023] advocate defining the regions as the leaves of a decision tree trained to predict x_j from $\boldsymbol{x}_{-j}$. By setting the tree hyperparameter `min_samples_leaf=50`, we can regularize the tree-growth while also allowing the number of regions to increase with N.

Once the partition of $\mathcal{X}_{-j}$ is learned, the CPFI estimate is

$$\widehat{\Phi}^{\text{CPFI}}_j(h, \mathcal{B}) = \sum_{t=1}^{M} \mathcal{B}(\Omega^{[t]})\, \Phi^{\text{Marginal-Sobol}}_j(h, \mathcal{B}_{\Omega^{[t]}}), \tag{4.12}$$

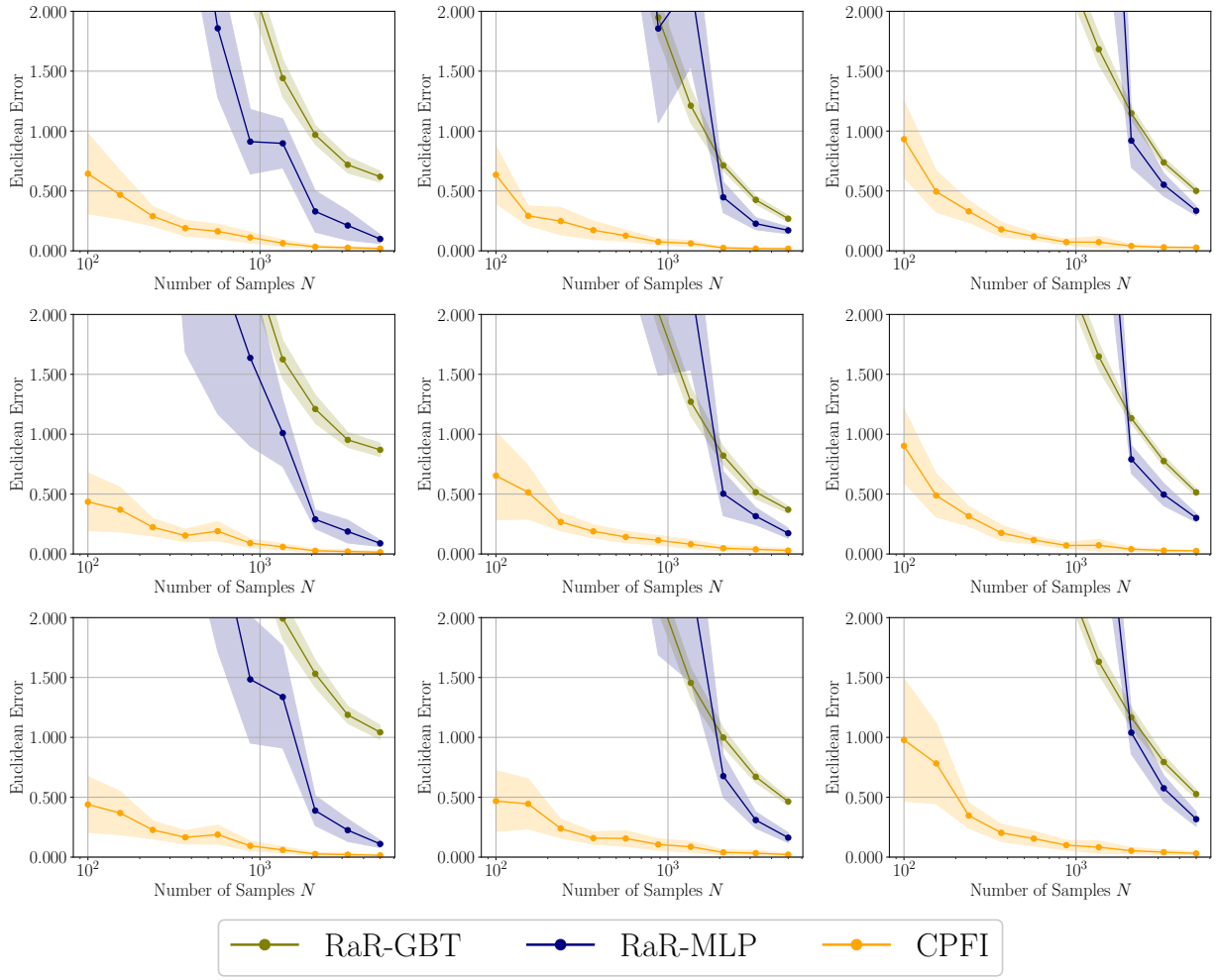


Figure 4.3 Convergence of the RaR-GBT, RaR-MLP, and CPFI to the Total-Sobol Index. The three columns indicate correlation $\rho_{45} = 0.2, 0.5, 0.75$ while the three rows indicate correlation $\rho_{12} = 0.2, 0.5, 0.75$.

where the regional Marginal-Sobol importance $\Phi_j^{\text{Marginal-Sobol}}(h, \mathcal{B}_{\Omega^{[t]}})$ is estimated using the matrix $\boldsymbol{H}^j$. According to Figure 4.3, the CPFI estimate converges faster to Total-Sobol in all nine correlation settings. This demonstrates that computing CPFI from $\boldsymbol{z}$-Anchored Decompositions is a viable mean to extract Total-Sobol feature importance.

Like RaR-GBT and RaR-MLP, the CPFI method requires fitting d additional models. However, unlike the GBTs and MLPs used in the RaR approach, the decisions trees employed in CPFI are interpretable by-design so we are not using a black-box to explain a black-box.

4.2 Exploring the Lattice Space

4.2.1 The VIN Algorithm

Computing a single component $h_{u,\boldsymbol{z}}(\boldsymbol{x})$ involves a function evaluation and $2^{|u|}-1$ subtractions. Such calculations are tractable given the high efficiency of subtraction operations on a CPU. Still, tractability becomes a major issue when one aims to compute a component $h_{u,\boldsymbol{z}}(\boldsymbol{x})$ for every possible subset of features $u \subseteq [d]$. Machine Learning problems are often high-dimensional ($d = 10, 50, 100$) and so computing all 2^d components is infeasible. Avoiding this exponential complexity is non-trivial in a model-agnostic setting because no prior assumptions can be made on the feature interactions present in h.

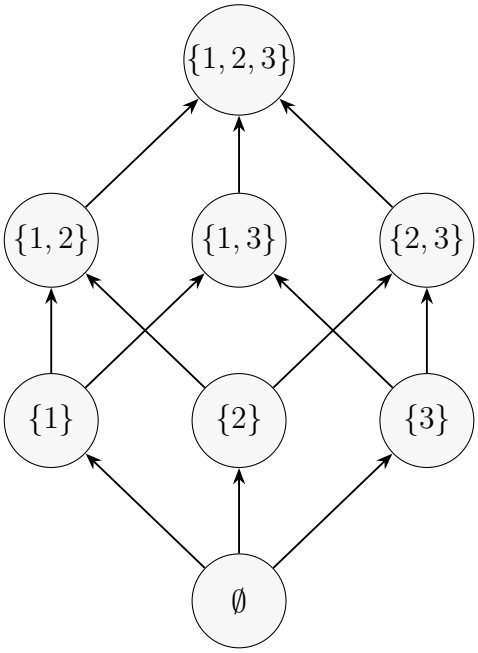


Figure 4.4 Lattice Space of the Functional Decomposition. The partial order of set inclusion is indicated by pointed arrows. This space has size 2^d and so exploring it in ML settings is challenging.

The solution : iteratively probe the model and determine what interactions h_u are non-significant. If insignificant interactions can be identified in advance, then one can avoid computing them altogether. This is the main idea behind the Variable Interaction Network (VIN) algorithm proposed by Hooker [2004]. The author first defined the *lattice space* as the representation of all possible components of the FD arranged hierarchically, see Figure 4.4. The lattice space is a graph where each node is a subset of features $u \subseteq [d]$ and each edge points from a set u to one of its supersets $v \supset u$ such that $|v| = |u| + 1$. The VIN algorithm explores the graph starting from the root $\emptyset$ and iteratively looking at children of the current explored nodes. Although this search space has an exponential size, not all of its nodes u may represent interactions h_u that have significant variability. Assuming feature independence, the VIN algorithm [Hooker, 2004] quantifies the *significance* of interaction $u \subseteq [d]$ as follows

$$\bar{\sigma}_u^2 := \sum_{v \subseteq [d]: u \subseteq v} \sigma_v^2 \geq \epsilon, \tag{4.13}$$

where

$$\sigma_v^2 := \mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mathcal{B}_{\text{ind}}} [h_{v,\mathcal{B}_{\text{ind}}}(\boldsymbol{x})^2] \tag{4.14}$$

is the variance of a v-way interaction for the Interventional Decomposition with background $\mathcal{B}_{\text{ind}}$. VIN considers an interaction u significant if the total variance of all of its supersets exceeds a certain threshold. This implies that a non-negligible amount of the variance in h is induced by interactions involving u. The monotonic property [Hooker, 2004]

$$u \subset v \Rightarrow \bar{\sigma}_u^2 \geq \bar{\sigma}_v^2 \tag{4.15}$$

guarantees that an insignificant interaction u cannot have significant children $v \supset u$. Consequently, the search of the Lattice space can be aggressively pruned.

Equations 4.13 and 4.15, which are at the basis of the VIN algorithm, rely on the decorrelated background $\mathcal{B}_{\text{ind}}$. We propose a generalization of VIN that does not make any assumptions about the background distribution $\mathcal{B}$.

4.2.2 VIN without Independence

In this section, we again assume that $\mathcal{B} = \mathcal{F}$, $\boldsymbol{z}^{(i)} = \boldsymbol{x}^{(i)}$ for $i = 1, 2, \ldots, N$, and that the $\boldsymbol{H}^u$ matrix has size $N \times N$ and stores $H^u_{ij} = h_{u,\boldsymbol{x}^{(j)}}(\boldsymbol{x}^{(i)})$. Now, Duality (cf. Proposition 4.1.1) implies that differences between averaging $\boldsymbol{H}^u$ along axes 1 and 2 are induced by higher order interactions involving u. Formally, the quantity

$$\Psi(u) := \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\Big[\Big(\sum_{v\in[d]:u\subset v} h_{v,\mathcal{B}}(\boldsymbol{x})\Big)^2\Big] \tag{4.16}$$

that measures interaction strength of strict supersets of u can be estimated via

$$\widehat{\Psi}(\boldsymbol{H}^u) := \frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{N}\sum_{j=1}^{N} H^u_{ij} - (-1)^{|u|}H^u_{ji}\right)^2 \xrightarrow{p} \Psi(u). \tag{4.17}$$

Note that $\Psi(u)$ is equivalent to $\bar{\sigma}_u^2 - \sigma_u^2$ when features are independent. Thus, our methodology based on $\Psi(u)$ falls back to VIN when a product measure is used as background.

The generalized VIN algorithm goes as follows : initialize the set $U = \emptyset \cup \{\{j\}\}_{j=1}^d$ of already explored nodes with components $\{\boldsymbol{H}^u\}_{|u|\leq 1}$. Also initialize a set of nodes $E = \{\{j\}\}_{j=1}^d$ that can be potentially extended. Then, the node $u^\star = \text{argmin}_{u\in E}\,\widehat{\Psi}(\boldsymbol{H}^u)$ with the largest potential is considered for extension. That is, define a candidate $u_{\text{candidate}} := u^\star \cup \{k\}$ by extending $u^\star$ with any missing element $k \in [d] \setminus u^\star$. Candidate generation continues until a candidate has been proposed by all of its parents. At this point one is able to compute the component $\boldsymbol{H}^{u_{\text{candidate}}}$ and add $u_{\text{candidate}}$ to the set of visited nodes U. The search is conducted

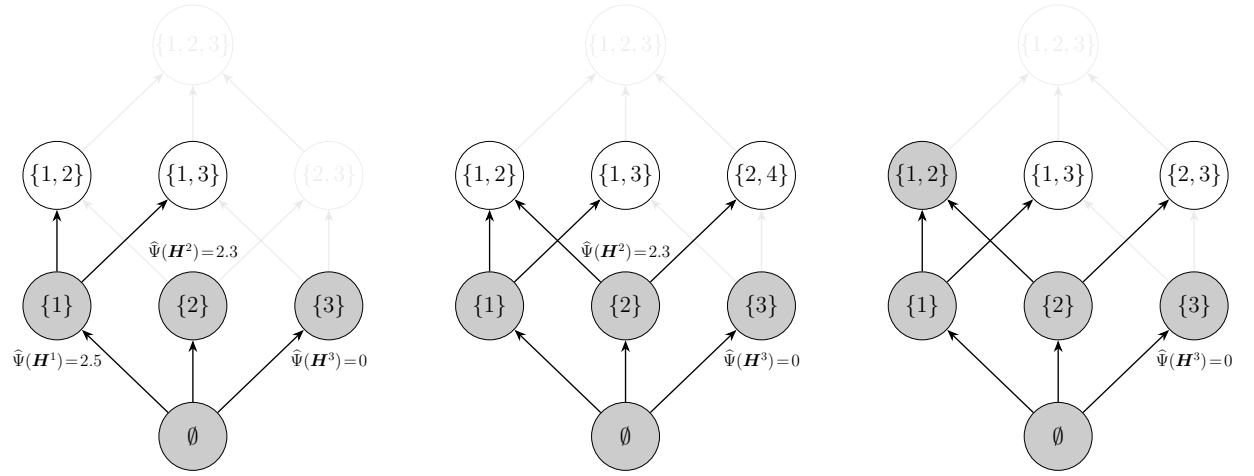
until the reconstruction loss

$$\ell = \frac{1}{N}\sum_{i=1}^{N}\Big(h(\boldsymbol{x}^{(i)}) - \frac{1}{N}\sum_{j=1}^{N}\sum_{u\in U} H_{ij}^{u}\Big)^2 \tag{4.18}$$

falls bellow some threshold $\tau := \epsilon \times \mathbb{V}_{\boldsymbol{x}\sim\mathcal{B}}[h(\boldsymbol{x})]$. Algorithm 2 presents the procedure and Figure 4.5 illustrates it on a toy model.

Algorithm 2 Exploring the Lattice Space

procedure ExploreLattice(h, $\{\boldsymbol{H}^u\}_{|u|\leq 1}$, ϵ)
  $C = \texttt{dict}()$; % Empty Dictionary of Candidate Nodes
  $U = \emptyset \cup \{\{j\}\}_{i=1}^{d}$; % Visited nodes
  $E = \{\{j\}\}_{i=1}^{d}$; % Nodes to extend
  $\tau = \epsilon \times \mathbb{V}_{\boldsymbol{x}\sim\mathcal{B}}[h(\boldsymbol{x})]$; % Stopping Criteron
  $\ell = \frac{1}{N}\sum_{i=1}^{N}(h(\boldsymbol{x}^{(i)}) - \frac{1}{N}\sum_{j=1}^{N}\sum_{u\in U} H_{ij}^{u})^2$;
  while $\ell > \tau$ **do**
    % Choose next node to extend;
    $u^\star = \text{argmax}_{u\in E}\,\widehat{\Psi}(\boldsymbol{H}^u)$;
    $E = E \setminus u^\star$;
    % Extend with any missing feature;
    for $k \in [d] \setminus u^\star$ **do**
      $u_{\text{candidate}} = u^\star \cup \{k\}$;
      if $u_{\text{candidate}} \notin C.\text{keys}$ **then**
        $C[u_{\text{candidate}}] = 1$;
      else
        $C[u_{\text{candidate}}] += 1$;
        % Check if all children of the node have been visited
        if $C[u_{\text{candidate}}] = |u_{\text{candidate}}|$ **then**
          Compute $\boldsymbol{H}^{u_{\text{candidate}}}$;
          $U = U \cup u_{\text{candidate}}$
          $\ell = \frac{1}{N}\sum_{i=1}^{N}(h(\boldsymbol{x}^{(i)}) - \frac{1}{N}\sum_{j=1}^{N}\sum_{u\in U} H_{ij}^{u})^2$;
  return $\{\boldsymbol{H}^u\}_{u\in U}$;



(a) Step 1.
$U = \{\emptyset, \{1\}, \{2\}, \{3\}\}$
$E = \{\{1\}, \{2\}, \{3\}\}$.
The node $u^\star = \{1\}$ has the highest potential $\widehat{\Psi}(\boldsymbol{H}^1)$ and so it is chosen for extension. The two candidates are $\{1, 2\}$ and $\{1, 3\}$. $u^\star$ is then removed from the set E.

(b) Step 2.
$U = \{\emptyset, \{1\}, \{2\}, \{3\}\}$
$E = \{\{2\}, \{3\}\}$
The node $u^\star = \{2\}$ has the highest potential $\widehat{\Psi}(\boldsymbol{H}^2)$ and so it chosen for extension. The two candidates are $\{1, 2\}$ and $\{2, 3\}$. $u^\star$ is then removed from the set E.

(c) Step 3.
$U = \{\emptyset, \{1\}, \{2\}, \{3\}, \{1, 2\}\}$
$E = \{\{3\}\}$
$u_{\text{candidate}} = \{1, 2\}$ was proposed by all its direct parent so its component $\boldsymbol{H}^{\{1,2\}}$ is computed and added to U. The reconstruction loss is null and the algorithm stops.

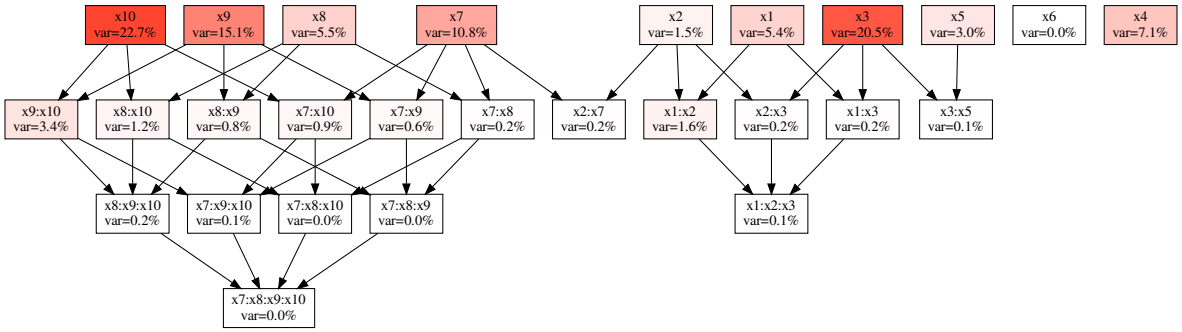
Figure 4.5 Toy Example of lattice space exploration algorithm using the toy function $h(\boldsymbol{x}) = x_1x_2 - x_3$. Visited nodes U are shown in gray while candidate nodes are colored white.

4.2.3 Experiments

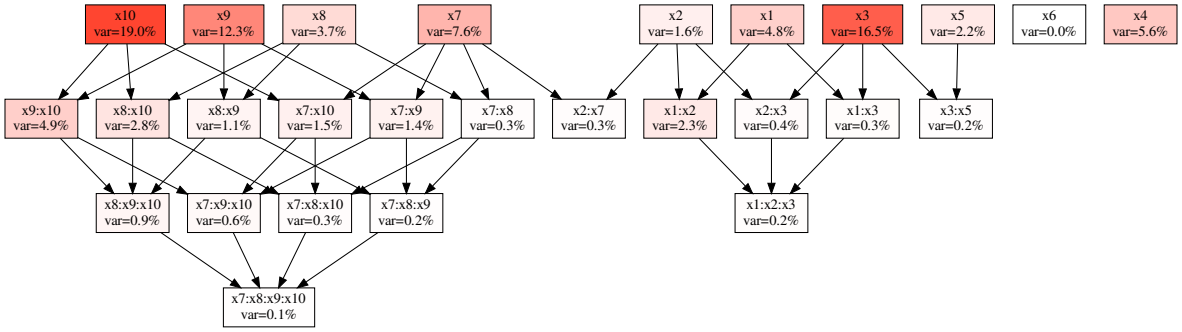
The viability of Algorithm 2 as an alternative to VIN is demonstrated on the toy example used originally to illustrate the VIN algorithm [Hooker, 2004]

$$h(\boldsymbol{x}) = \pi^{x_1x_2}\sqrt{2x_3} - \sin^{-1}(x_4) + \log(x_3 + x_5) - \frac{x_9}{x_{10}}\sqrt{\frac{x_7}{x_8}} - x_2x_7. \tag{4.19}$$

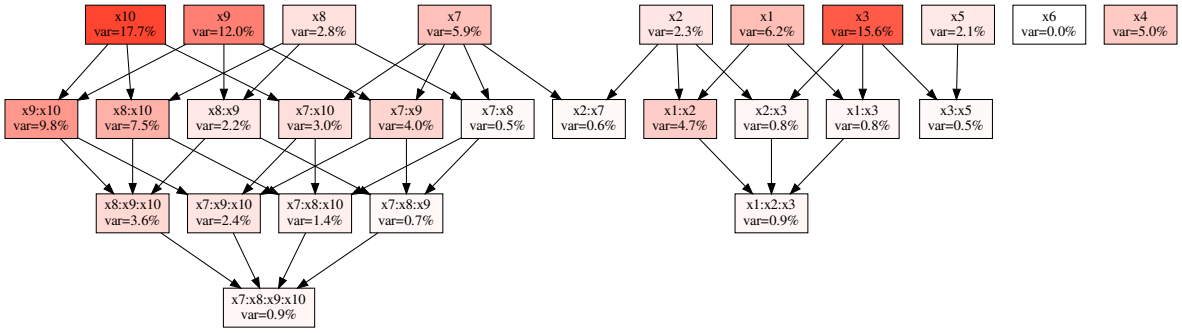
The features are sampled uniformly between 0.2 and 1 and their dependence structure is a Gaussian Copula with correlation ρ. The result of Algorithm 2 using $\epsilon = 1 \times 10^{-5}$ for $\rho = 0, 0.5, 0.95$ are presented in Figure 4.6. Algorithm 2 is able to recover the ground truth lattice space regardless of the degree of feature correlation. This is a consequence of the minimality of the Interventional Decompositions $\{h_{u,\mathcal{B}}(\boldsymbol{x})\}_{u\subseteq[d]}$, which holds for any background $\mathcal{B}$ (with or without correlations). Nonetheless, note that correlations do affect the amplitudes of the functional components. On this toy example, higher correlation reduce the amplitude of main effects and increase the magnitude of interactions.



(a) $\rho = 0$



(b) $\rho = 0.5$



(c) $\rho = 0.95$

Figure 4.6 Lattice Space obtained with Algorithm 2 on the toy model from [Hooker, 2004].

4.3 Shapley Values

If one were able to compute all the matrices $\{\boldsymbol{H}^u\}_{u\subseteq[d]}$, then computing Interventional Shapley Values would be as simple as averaging said matrices along their second axis and equally sharing the result among the $|u|$ features involved. Yet, computing the whole decomposition is not realistic in a model-agnostic setting and so Shapley Values must be estimated.

4.3.1 Permutations Estimate

Common Shapley Value estimators rely on random permutations. Let π be a permutation of $[d]$, $\pi[j]$ be the position of the feature j in π, and $\pi_{:j} = \{k \in [d] \,:\, \pi[k] < \pi[j]\}$. The Interventional Shapley Values can be expressed as:

$$\phi_j^{\text{SHAP-int}}(h, \boldsymbol{x}, \mathcal{B}) = \mathop{\mathbb{E}}_{\substack{\pi\sim\Omega \\ \boldsymbol{z}\sim\mathcal{B}}} \left[h(\, \boldsymbol{r}^{\boldsymbol{z}}_{\pi_{:j}\cup\{j\}}(\boldsymbol{x})\,) - h(\, \boldsymbol{r}^{\boldsymbol{z}}_{\pi_{:j}}(\boldsymbol{x})\,) \right], \tag{4.20}$$

where Ω is the uniform distribution over all $d!$ permutations of the features. This equation suggests a simple Monte Carlo estimate 1) sample a random permutations $\pi \sim \Omega$, 2) sample a background point $\boldsymbol{z} \sim \mathcal{B}$, 3) compute $h(\, \boldsymbol{r}^{\boldsymbol{z}}_{\pi_{:j}\cup\{j\}}(\boldsymbol{x})\,) - h(\, \boldsymbol{r}^{\boldsymbol{z}}_{\pi_{:j}}(\boldsymbol{x})\,)$, 4) repeat M times and average [Štrumbelj and Kononenko, 2014]. This basic algorithm must be applied independently over each features j, which leads to sub-optimal computations. Indeed, the function call $h(\, \boldsymbol{r}^{\boldsymbol{z}}_{\pi_{:j}}(\boldsymbol{x})\,)$ used when estimating the Shapley Value of feature j cannot be reused by other features $k \neq j$ unless a the same baseline $\boldsymbol{z}$ is resampled and a new permutation π' is sampled such that $\pi_{:j} = \pi'_{:k}$. In a model-agnostic setting, function calls can be costly and so it is important to make the most out of them. To reuse model calls, it is better to sample P permutations ($\{\pi^{(\ell)}\}_{\ell=1}^{P} \sim \Omega^P$), M background samples ($\{\boldsymbol{z}^{(i)}\}_{i=1}^{M} \sim \mathcal{B}^M$), and use each combination of those. Letting

$$\Delta_j(h, \boldsymbol{x}, \boldsymbol{z}, \pi) := h(\, \boldsymbol{r}^{\boldsymbol{z}}_{\pi_{:j}\cup\{j\}}(\boldsymbol{x})\,) - h(\, \boldsymbol{r}^{\boldsymbol{z}}_{\pi_{:j}}(\boldsymbol{x})\,), \tag{4.21}$$

be the marginal contribution of feature j given the background sample $\boldsymbol{z}$ and permutation π, the permutationSHAP estimator is

$$\widehat{\phi}_j^{\text{SHAP-int}}(h, \boldsymbol{x}, \{\boldsymbol{z}^{(i)}\}_{i=1}^{M}) := \frac{1}{PM} \sum_{\ell=1}^{P} \sum_{i=1}^{M} \Delta_j(h, \boldsymbol{x}, \boldsymbol{z}^{(i)}, \pi^{(\ell)}). \tag{4.22}$$

This is the current scheme behind the `PermutationExplainer`[1] of the SHAP library. With this estimate, each function call is guaranteed to be used for the Shapley values of at least two features. This statistic, known as a two-sample U-statistic, has desirable properties such as consistency and asymptotic normality [Lee, 2019, Section 3.7.1].

Theorem 4.3.1 (PermutationSHAP Confidence Interval)**.** *Let $\phi_j^{SHAP\text{-}int}$ be the ground-truth Shapley values from Equation 4.20 and let $\widehat{\phi}_j^{SHAP\text{-}int}$ be the two-sample U-statistic defined in Equation 4.22. Moreover, assume the variance $\mathbb{V}_{\pi\sim\Omega,z\sim\mathcal{B}}[\Delta_j(h,\boldsymbol{x},\boldsymbol{z},\pi)]$ is finite. Then, the following holds for any $\delta \in]0,1[$*

$$\lim_{\substack{P+M\to\infty \\ s.t.\ P/(P+M)\to p\in(0,1)}} \mathbb{P}\left(|\widehat{\phi}_j^{SHAP\text{-}int} - \phi_j^{SHAP\text{-}int}| \geq \frac{F^{-1}_{\mathcal{N}(0,1)}(1-\delta/2)}{\sqrt{P+M}} \left[\frac{\sigma_{10}^2}{p} + \frac{\sigma_{01}^2}{1-p} \right] \right) = \delta,$$

where $\sigma_{10}^2 = \mathbb{V}_{\pi\sim\Omega}[\,\mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[\Delta_j(h,\boldsymbol{x},\boldsymbol{z},\pi)]\,]$ and $\sigma_{01}^2 = \mathbb{V}_{\boldsymbol{z}\sim\mathcal{B}}[\,\mathbb{E}_{\pi\sim\Omega}[\Delta_j(h,\boldsymbol{x},\boldsymbol{z},\pi)]\,]$.

This Theorem allows one to compute confidence intervals that capture the ground-truth Shapley Values with probability $1-\delta$.

4.3.2 Lattice-based Estimate

PermutationSHAP estimator can be interpreted as a uniform exploration of the lattice space. Although this generic approach is guaranteed to converge to the true Shapley values given enough samples, it may require more function calls than necessary. In fact, if the lattice space is sparse, we show that $h(\boldsymbol{r}_u^{\boldsymbol{z}}(\boldsymbol{x}))$ should only be called C times with $C \ll 2^d$. To see this, remember that the Shapley values can be expressed as an egalitarian redistribution of Interventional Decompositions

$$\phi_j^{\text{SHAP-int}}(h,\boldsymbol{x},\mathcal{B}) = \sum_{u\subseteq[d]:j\in u} \frac{h_{u,\mathcal{B}}(\boldsymbol{x})}{|u|}. \tag{4.23}$$

[1] `https://shap.readthedocs.io/en/latest/generated/shap.PermutationExplainer.html#shap.PermutationExplainer`

However, it is possible that only a small number of subsets $U \subset 2^{[d]}$ (returned by Algorithm 2) is required to accurately reconstruct h. We have

$$\begin{aligned}\widehat{\phi}_j^{\text{SHAP-int}}(h, \boldsymbol{x}, \mathcal{B}) &= \sum_{u \in U : j \in u} \frac{h_{u,\mathcal{B}}(\boldsymbol{x})}{|u|} \\ &= \sum_{u \in U : j \in u} |u|^{-1} \sum_{v \subseteq u} (-1)^{|u|-|v|} \mathop{\mathbb{E}}_{\boldsymbol{z} \sim \mathcal{B}}[h(\boldsymbol{r}_v^{\boldsymbol{z}}(\boldsymbol{x}))] \\ &= \sum_{v \in U} w_j(v) \mathop{\mathbb{E}}_{\boldsymbol{z} \sim \mathcal{B}}[h(\boldsymbol{r}_v^{\boldsymbol{z}}(\boldsymbol{x}))]\end{aligned} \tag{4.24}$$

for some weights $w_j(v)$ computed via Algorithm 3. We call this estimator LatticeSHAP. When $U \ll 2^d$, LatticeSHAP can potentially be an accurate estimate while requiering a small number of function calls.

Algorithm 3 Compute Shapley Weights

```
procedure ShapleyWeights(U)
    w_j(v) = 0 for all j = 1, 2, ..., d and v ∈ U;
    for u ∈ U do
        for v ⊆ u do
            for j ∈ u do
                w_j(v) += (−1)^{|u|−|v|}/|u|;
    return {w(v)}_{v∈U};
```

4.3.3 Experiments

This experiment is the continuation of the toy example from Section 4.2.3 that demonstrated the viability of Algorithm 2 to explore the lattice space. We now present preliminary evidence that the lattice space can be leveraged to efficiently calculate Interventional Shapley Values. For the three correlation values $\rho = 0, 0.5, 0.95$ investigated previously, we computed Interventional Shapley Values using Equation 4.24 and the `ExactExplainer` from the SHAP library, and compared the computation times. LatticeSHAP took about 10 seconds to run while the `ExactExplainer` took 6 minutes. Both algorithms return the exact same quantity, but LatticeSHAP is much more efficient because the lattice space is sparse $|U| = 27 < 2^d = 1024$.

As future work, it remains to demonstrate the applicability of Algorithm 2 to identify feature interactions in real ML models. Assuming the algorithm is able to accurately recover the subset $U \subset 2^{[d]}$ of meaningful interactions, we then wish to highlight the advantages of computing LatticeSHAP estimates (cf. Equation 4.24) rather than the more common PermutationSHAP estimates (cf. Equation 4.22). Proving the superiority of LatticeSHAP over

PermutationSHAP would require comparing the estimation error to the number of model calls.

Given that these empirical comparisons have not yet been conducted at scale, we currently advocate sticking to PermutationSHAP in model-agnostic settings.

Contributions

To conclude this Chapter, we computed $\boldsymbol{z}$-Anchored components with minimal function calls using the "annihilation property". We stored these functional components in matrices $\boldsymbol{H}$. By exploiting a property we called *duality*, we showed how these matrices can be used to accurately estimate the various post-hoc additive explanation from the literature. Duality also allowed use to generalize the VIN algorithm to accept background which correlated features. This generalization of the VIN algorithm was finally used to derive a new model-agnostic estimate of Shapley Values, LatticeSHAP, which holds the promise of reduced computational cost whenever the black-box only contains few features interactions.

CHAPTER 5 MODEL-SPECIFIC ESTIMATES

This chapter demonstrates how to develop model-specific implementations of $\boldsymbol{z}$-Anchored Decomposition. As shown previously, these functional decompositions are the fundational blocks used to compute the various post-hoc additive explanations. We first formalize how general features, categorical, or vectorial, are handled by a ML model. Then, we derive efficient algorithms for Parametric Additive Models, Kernel Methods, and Tree Ensembles. The presented algorithms are all new contributions.

5.1 Feature Embeddings

Functional Decompositions is applicable to any function h that maps a feature space $\prod_{j=1}^{d} \mathcal{X}_j$ to $\mathbb{R}$. Each feature domain $\mathcal{X}_j$ is not necessarily $\mathbb{R}$ so features can be categorical and even vectorial. Nonetheless, most ML models (*e.g.* kernel methods, tree ensembles, neural networks) require that each input component is a scalar. Hence, the feature vector $\boldsymbol{x}$ must be passed to an embedder $\boldsymbol{\xi} : \prod_{j=1}^{d} \mathcal{X}_j \to \mathbb{R}^{d'}$ before being fed to the ML model h^{ML}. Consequently, the function h being explained is the composition

$$h = h^{\text{ML}} \circ \boldsymbol{\xi}. \tag{5.1}$$

Practitioners familiar with the Scikit-Learn API should find Equation 5.1 reminiscent of `Pipelines`[1], which separate the feature embedding from the ML model itself. The $\boldsymbol{\xi}$ map works as follows: each component x_j is mapped through a *component-embedding* $\xi_j : \mathcal{X}_j \to \mathbb{R}^{d_j}$. Letting $d' = \sum_{j=1}^{d} d_j$, the embedding function $\boldsymbol{\xi} : \prod_{j=1}^{d} \mathcal{X}_j \to \mathbb{R}^{d'}$ maps any vector $\boldsymbol{x}$ to the concatenation $\boldsymbol{\xi}(\boldsymbol{x}) = [\xi_1(x_1), \ldots, \xi_d(x_d)]^T$. In practice, these concatenations of component-embeddings are conducted with the `ColumnTransformer` of the Scikit-Learn Python library.

[1] `https://scikit-learn.org/stable/modules/compose`

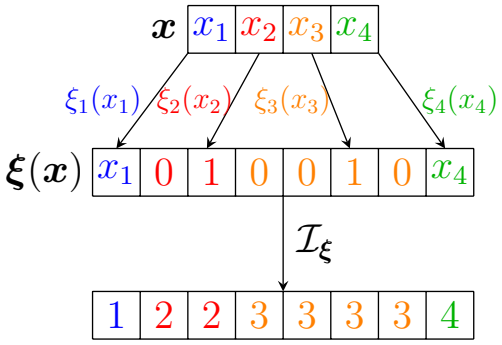


Figure 5.1 Example of embedding $\boldsymbol{\xi}(\boldsymbol{x}) \in \mathbb{R}^8$. Here, x_1 and x_4 are kept intact while x_2 and x_3 are one-hot-encoded. The bottom of the figure presents the function $\mathcal{I}_{\boldsymbol{\xi}}$ that maps the index of an embedded coordinate to the index of its associated $\boldsymbol{x}$ component.

Component Embeddings

We distinguish the following types of component-embedding used in practice.

- Heavy-tailed numerical features x_j can be monotonically transformed to have a distribution closer to normal : $\xi_j(x_j) = \log(1 + x_j) \in \mathbb{R}^1$.
- Numerical features x_j are standardized before being fed to a linear model : $\xi_j(x_j) = \big(x_j - \bar{x}_j\big)/\sqrt{1/N \sum_{i=1}^{N} (x_j^{(i)} - \bar{x}_j)^2} \in \mathbb{R}^1$ with $\bar{x}_j := \frac{1}{N} \sum_{i=1}^{N} x_j^{(i)}$.
- Polynomial powers of x_j can be extracted as part of feature engineering to increase the model expressivity : $\xi_j(x_j) = [x_j, x_j^2, x_j^3]^T \in \mathbb{R}^3$.

More Component Embeddings

- Categorical Features with C categories ($\mathcal{X}_j = [C]$) can be embedded into
  - $\mathbb{R}^C$ via One-Hot-Encoding (OHE) : $\xi_j(x_j) = \boldsymbol{\delta}_{x_j}$, where the components of the vector $\boldsymbol{\delta}_j$ are all 0 except for the jth component which is 1.
  - $\mathbb{R}^{d_j}$ via Entity Embedding (EE) : $\xi_j(x_j) = \mathbf{E}[:, j] \in \mathbb{R}^{d_j}$, where the embedding matrix $\mathbf{E} \in \mathbb{R}^{d_j \times C}$ is learned a priori via a Neural Network [Guo and Berkhahn, 2016].
- A vectorial feature in $\mathcal{X}_j = \mathbb{R}^m$ (*e.g.* pixels patches in a image or (latitude, longitude) tuples) is mapped to m different columns in the embedding $\xi_j(x_j) = [x_{j1}, x_{j2}, \ldots, x_{jm}]^T$.

Notice that the embedding $\boldsymbol{\xi}$ induces a surjective function $\mathcal{I}_{\boldsymbol{\xi}} : [d'] \to [d]$ given by

$$\mathcal{I}_{\boldsymbol{\xi}}(i) = \min \left\{ j \in [d] : i \leq \sum_{k=1}^{j} d_k \right\} \tag{5.2}$$

that maps each embedded coordinate $i \in [d']$ to the index of its associated $\boldsymbol{x}$ component. Figure 5.1 shows an example of embedding with One-Hot-Encodings and the resulting map $\mathcal{I}_{\boldsymbol{\xi}}$. The pre-image $\mathcal{I}_{\boldsymbol{\xi}}^{-1} : 2^{[d]} \to 2^{[d']}$ returns the set of all embedded coordinates that map to a specific $\boldsymbol{x}$ components.

When the embedding results from a composition of embeddings $\boldsymbol{\xi}_3 \circ \boldsymbol{\xi}_2 \circ \boldsymbol{\xi}_1$, the pre-image is easy to compute

$$\mathcal{I}^{-1}_{\boldsymbol{\xi}_3 \circ \boldsymbol{\xi}_2 \circ \boldsymbol{\xi}_1} = \mathcal{I}^{-1}_{\boldsymbol{\xi}_3} \circ \mathcal{I}^{-1}_{\boldsymbol{\xi}_2} \circ \mathcal{I}^{-1}_{\boldsymbol{\xi}_1}. \tag{5.3}$$

This formula describes which components of composition of embeddings correspond to specific $\boldsymbol{x}$ components. Finally, the replace-function (cf. Equation 3.3) can also be expressed in terms of the ML model and the embedding

$$h(\boldsymbol{r}^{\boldsymbol{z}}_u(\boldsymbol{x})) = h^{\mathrm{ML}}\Big(\boldsymbol{r}^{\boldsymbol{\xi}(\boldsymbol{z})}_{\mathcal{I}^{-1}(u)}(\boldsymbol{\xi}(\boldsymbol{x})) \Big), \tag{5.4}$$

which expresses the replacement after embedding that corresponds to a given replacement before embedding, see Figure 5.2. Equation 5.4 highlights the importance of the pre-image $\mathcal{I}_{\boldsymbol{\xi}}^{-1}$ for calculating functional decompositions of functions h with compositional structures. We now present how to efficiently compute the Anchored/Interventional decomposition given different architectures for h^{ML}.

5.2 Additive Models

As a reminder, additive models take the form

$$h^{\mathrm{add}}(\boldsymbol{x}) = \omega_0 + \sum_{j=1}^{d} h_j(x_j). \tag{5.5}$$

By minimality of the Interventional Decomposition, the components $h_{u,\mathcal{B}}(\boldsymbol{x})$ are null for every $|u| \geq 2$ so one only needs to estimate the main effects $h_{j,\mathcal{B}}(\boldsymbol{x})$ for $j = 1, \ldots, d$. When M anchors are sampled $\{\boldsymbol{z}^{(k)}\}_{k=1}^{M} \sim \mathcal{B}$, the estimates are

$$h_j(x_j) - \frac{1}{M} \sum_{k=1}^{M} h_j(z_j^{(k)}) \xrightarrow{p} h_{j,\mathcal{B}}(\boldsymbol{x}). \tag{5.6}$$

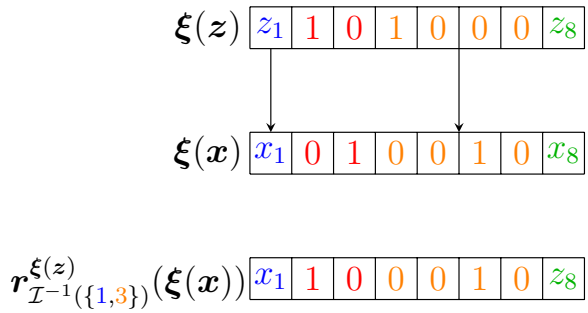


Figure 5.2 The replace function applied to an embedding of 4 features to $\mathbb{R}^8$. Importantly, all embedded components associated with a given feature x_j are replaced simultaneously.

When h_j are modeled non-parametrically (for example via boosted trees), and are fitted directly on the feature x_j, Equation 5.6 is directly calculable from the model h^{add}.

Yet, it is also possible to model h_j parametrically so that the additive model is equivalent to a linear model evaluated on an embedding (*i.e.* a basis expansion). Letting $h^{\text{lin}}_{\boldsymbol{\omega}}$ be the linear model described by Equation 2.9, a parametric additive model takes the composite form

$$h^{\text{add}}_{\boldsymbol{\omega}} = h^{\text{lin}}_{\boldsymbol{\omega}} \circ \boldsymbol{\xi}. \tag{5.7}$$

Common Basis Expansions

Here are some common basis expansions for parametric additive models

- For $x_j \in \mathbb{R}$, one can keep a linear dependence with the output $\xi_j(x_j) = x_j$.
- For $x_j \in \mathbb{R}$, a basis of B-Splines $\{h_{jk}\}_{k=1}^{d_j}$ can be used

$$\xi_j(x_j) = [h_{j1}(x_j), h_{j2}(x_j), \ldots, h_{jd_j}(x_j)]^T \in \mathbb{R}^{d_j}$$

- A factor variable $x_j \in [C]$ can be one-hot encoded $\xi_j(x_j) = \boldsymbol{\delta}_{\boldsymbol{x}_j} \in \mathbb{R}^C$ before being fed to the linear model.
- For $x_j \in \mathbb{R}$, given a set of bins edges $\{b_k\}_{k=0}^{d_j}$, the embedding

$$\xi_j(x_j) = \Big[\mathbb{1}[b_0 \leq x_j < b_1], \mathbb{1}[b_1 \leq x_j < b_2], \ldots, \mathbb{1}[b_{d_j-1} \leq x_j \leq b_{d_j}]\Big]^T \in \mathbb{R}^{d_j}$$

  describes a step-function basis expansion.

When explaining a parametric additive model, the jth component of Interventional Decom-

position is computed

$$h_j(x_j) - \frac{1}{M}\sum_{k=1}^{M} h_j(z_j^{(k)}) = \sum_{k \in \mathcal{I}_{\boldsymbol{\xi}}^{-1}(\{j\})} \omega_k \Big(\xi_k(\boldsymbol{x}) - \frac{1}{M}\sum_{i=1}^{M} \xi_k(\boldsymbol{z}^{(i)})\Big) := \boldsymbol{\omega}_j^T \, \overline{\boldsymbol{\xi}}_j(\boldsymbol{x}), \tag{5.8}$$

which is a linear function of the weights $\boldsymbol{\omega}$. Notice that we introduced the vector $\boldsymbol{\omega}_j := (\omega_k)_{k \in \mathcal{I}_{\boldsymbol{\xi}}^{-1}(\{j\})}$ that regroups all weights modeling the dependency w.r.t feature j. The embedding vector $\overline{\boldsymbol{\xi}}_j(\boldsymbol{x})$ corresponding to feature j is defined similarly.

Now investigating Global Feature Importance, the functional $\Phi_j^{\mathrm{GFI},[2]}(h, \mathcal{B}) := \mathbb{E}_{\boldsymbol{x}\sim\mathcal{B}}[h_{j,\mathcal{B}}(\boldsymbol{x})^2]$ can be estimated by

$$\frac{1}{N}\sum_{i=1}^{N} (\, \boldsymbol{\omega}_j^T \, \overline{\boldsymbol{\xi}}_j(\boldsymbol{x}^{(i)}) \,)^2 \xrightarrow{p} \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}[h_{j,\mathcal{B}}(\boldsymbol{x})^2], \tag{5.9}$$

which is a quadratic function of the weights

$$\begin{aligned} \frac{1}{N}\sum_{i=1}^{N} (\, \boldsymbol{\omega}_j^T \, \overline{\boldsymbol{\xi}}_j(\boldsymbol{x}^{(i)}) \,)^2 &= \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{\omega}_j^T \, \overline{\boldsymbol{\xi}}_j(\boldsymbol{x}^{(i)}) \, \overline{\boldsymbol{\xi}}_j(\boldsymbol{x}^{(i)})^T \boldsymbol{\omega}_j \\ &= \boldsymbol{\omega}_j^T \left(\frac{1}{N}\sum_{i=1}^{N} \overline{\boldsymbol{\xi}}_j(\boldsymbol{x}^{(i)}) \, \overline{\boldsymbol{\xi}}_j(\boldsymbol{x}^{(i)})^T \right) \boldsymbol{\omega}_j = \boldsymbol{\omega}_j^T \boldsymbol{B}_j \boldsymbol{\omega}_j. \end{aligned} \tag{5.10}$$

Takeaways for Parametric Additive Models

By correctly handling feature embeddings, we can express additive explanations of parametric additive models directly in terms of the coefficients $\boldsymbol{\omega}$.

- Local Feature Attributions $h_{j,\mathcal{B}}(\boldsymbol{x})$ are linear functions of $\boldsymbol{\omega}$.
- Global Feature Importance $\mathbb{E}_{\boldsymbol{x}\sim\mathcal{B}}[h_{j,\mathcal{B}}(\boldsymbol{x})^2]$ are quadratic functions of $\boldsymbol{\omega}$.

5.3 Kernel Methods

For simplicity, we will not consider any embedding for Kernel Methods. So, we will assume that the input of the ML model is the unprocessed feature vector $\boldsymbol{x} \in \mathbb{R}^d$. As a reminder, kernel models take the form

$$h_{\boldsymbol{\alpha}}^{\mathrm{kernel}}(\boldsymbol{x}) = \sum_{\ell=1}^{R} \alpha_\ell k(\boldsymbol{x}, \boldsymbol{r}^{(\ell)}), \tag{5.11}$$

where the function $k(\cdot, \cdot)$ is a Positive Definite Symmetric (PDS) kernel. Many choices of kernels induce high-order interactions in the model : p-polynomial kernels introduce order

p interactions and Gaussian/Laplace kernels introduce order d interactions. For this reason, model-specific optimizations are limited to specific functionals and do not address the challenge of computing the whole functional decomposition.

One such optimization concerns linear functionals $\phi(\cdot, \boldsymbol{x}, \mathcal{B})$ that can be expressed

$$\phi(h^{\text{kernel}}_{\boldsymbol{\alpha}}, \boldsymbol{x}, \mathcal{B}) = \sum_{\ell=1}^{R} \alpha_\ell \, \phi(k(\cdot, \boldsymbol{r}^{(\ell)}), \boldsymbol{x}, \mathcal{B}) \tag{5.12}$$

Because linear functionals are applied to kernels $\phi(k(\cdot, \boldsymbol{r}^{(\ell)}), \boldsymbol{x}, \mathcal{B})$, it is not necessary to call the model $h^{\text{kernel}}_{\boldsymbol{\alpha}}$ multiple times as in model-agnostic algorithms. If the kernel has certain properties, the computation of $\phi(k(\cdot, \boldsymbol{r}^{(\ell)}), \boldsymbol{x}, \mathcal{B})$ can be further optimized. We present two notable examples.

First, let $\phi(\cdot, \boldsymbol{x}, \mathcal{B})$ be the linear functional that maps h to its component $h_{u,\mathcal{B}}(\boldsymbol{x})$ evaluated at $\boldsymbol{x}$. If $k(\boldsymbol{r}, \boldsymbol{r}') = \prod_{j=1}^{d} k_j(r_j, r'_j)$ (*e.g.* Gaussian and Laplace kernels) and the background $\mathcal{B}$ is the empirical distribution over the dictionary $\{\boldsymbol{r}^{(i)}\}_{i=1}^{R}$, then

$$\phi(k(\cdot, \boldsymbol{r}^{(\ell)}), \boldsymbol{x}, \mathcal{B}) = \frac{1}{R} \sum_{i=1}^{R} \left[\prod_{j \notin u} k_j(r_j^{(i)}, r_j^{(\ell)}) \prod_{j \in u} \Big(k_j(x_j, r_j^{(\ell)}) - k_j(r_j^{(i)}, r_j^{(\ell)}) \Big) \right]. \tag{5.13}$$

The terms $k_j(r_j^{(i)}, r_j^{(\ell)})$ for $i, \ell = 1, \ldots, R$ should be available after fitting $h^{\text{kernel}}_{\boldsymbol{\alpha}}$. Hence, after evaluating $k_j(x_j, r_j^{(\ell)})$ for $j = 1, \ldots, d$ and $\ell = 1, \ldots, R$, one can leverage Equation 5.13 to compute any functional component $h_{u,\mathcal{B}}(\boldsymbol{x})$ using $\mathcal{O}(Rd)$ multiplications.

Second, assuming the kernel has continuous partial derivatives ($k \in \mathbb{C}^1(\mathcal{X} \times \mathcal{X})$), the Integrated Gradient $\phi_j^{\text{IG}}(\cdot, \boldsymbol{x}, \boldsymbol{z})$ can be applied on individual kernels

$$\phi_j^{\text{IG}}(k(\cdot, \boldsymbol{r}^{(\ell)}), \boldsymbol{x}, \boldsymbol{z}) = (x_j - z_j) \int_0^1 \frac{\partial k(\cdot, \boldsymbol{r}^{(\ell)})}{\partial x_j}((1-t)\boldsymbol{z} + t\boldsymbol{x})dt. \tag{5.14}$$

The integral can be estimated with the Trapezoid rule

$$\int_0^1 f(t)dt = \frac{1}{2N} \sum_{i=1}^{N} \Big(f(i/N) + f((i-1)/N) \Big) + \mathcal{O}(1/N^2). \tag{5.15}$$

The $\mathcal{O}(1/N^2)$ error implies that using 10 times more points in the discretization result in a 100 factor improvement on the error. But, how can we report this error without access to the ground truth $\phi_j^{\text{IG}}(k(\cdot, \boldsymbol{r}^{(\ell)}), \boldsymbol{x}, \boldsymbol{z})$, but only its Quadrature approximation $\widehat{\phi}_j^{\text{IG}}(k(\cdot, \boldsymbol{r}^{(\ell)}), \boldsymbol{x}, \boldsymbol{z})$? Since the ground-truth attributions sum to the Gap $G(h^{\text{kernel}}_{\boldsymbol{\alpha}}, \boldsymbol{x}, \boldsymbol{z}) := h^{\text{kernel}}_{\boldsymbol{\alpha}}(\boldsymbol{x}) - h^{\text{kernel}}_{\boldsymbol{\alpha}}(\boldsymbol{z})$

the following

$$\left| G(h^{\text{kernel}}_{\boldsymbol{\alpha}}, \boldsymbol{x}, \boldsymbol{z}) - \sum_{j=1}^{d} \widehat{\phi}^{\text{IG}}_j(h^{\text{kernel}}_{\boldsymbol{\alpha}}, \boldsymbol{x}, \boldsymbol{z}) \right| \tag{5.16}$$

can be used as a measure for the quality of the quadratures.

We have just seen that functional components $h_{u,\mathcal{B}}(\boldsymbol{x})$ are linear functions of the coefficients $\boldsymbol{\alpha}$. In light of Equation 4.6, the many measures of Global Feature Importance (*e.g.* PDP, PFI, Marginal-Sobol, Total-Sobol) must be quadratic functions of $\boldsymbol{\alpha}$.

Takeaways for Kernel Methods

By applying linear functionals ϕ directly on the kernels, we obtain efficient implementation that depend explicitely on the coefficients $\boldsymbol{\alpha}$.

- Local Feature Attributions are linear functions of $\boldsymbol{\alpha}$.
- Global Feature Importance are quadratic functions of $\boldsymbol{\alpha}$.

5.4 Tree Ensembles

In this section, we generalize the well-known Interventional TreeSHAP algorithm [Lundberg et al., 2020] to handle feature embeddings (*e.g.* one-hot-encoded categorical features) and to compute the $\boldsymbol{z}$-Anchored Decomposition.

5.4.1 Decision Tree

A directed graph $G = (N, E)$ is a set of nodes $n \in N$ and edges $e \in E \subset N \times N$. The node at the tail of e is $e_1 \in N$ while the node at the head of the edge is $e_2 \in N$. The node e_2 is called a child of e_1 and node e_1 is called the parent of e_2. A full binary tree is a rooted tree in which every node has exactly two children or none, and only one parent. An *internal node* is a node n that has a left child l and a right child r. We say that (n, r) is a right edge and (n, l) is a left edge. A leaf is a node with no children and all leaves are stored in the set $L \subseteq N$.

Definition 5.4.1 (Binary Decision Tree)**.** *A Binary Decision Tree on $\mathbb{R}^d$ is full binary tree $T = (N, E)$ in which every internal node n is labeled by a pair $(i_n, \gamma_n) \in [d] \times \mathbb{R}$ and every leaf $l \in L$ is labeled with a value $v_l \in \mathbb{R}$. For an internal node n, the label (i_n, γ_n) encodes the boolean function $R_n : \mathbb{R}^d \to \{0, 1\}$ given by $R_n(\boldsymbol{x}) := \mathbb{1}(x_{i_n} \leq \gamma_n)$.*

See Figure 5.3 for a simple example of Binary Decision Tree. Any Binary Decision Tree

induces a function $h^{\text{tree}} : \mathbb{R}^d \to \mathbb{R}$ defined as follows. For every $\boldsymbol{x} \in \mathbb{R}^d$, we start at the top of the tree (the root) and check the condition $R_1(\boldsymbol{x}) := \mathbb{1}(x_{i_1} \leq \gamma_1)$: if it is true we go down to the left child of the root, and if it is false we go down to the right child. This procedure is repeated until we reach a leaf node l and the model outputs $h^{\text{tree}}(\boldsymbol{x}) = v_l$. For instance, in Figure 5.3, we see that for the input $\boldsymbol{x} = (3.4, 0.2, 2)^T$, we go from the root to the node 2 and end up at the leaf 5, so $h^{\text{tree}}(\boldsymbol{x}) = v_5$. We note that the input "flowed through" the sequence of edges $(\,(1,2),(2,5)\,)$ which goes from root to leaf. We shall refer to such a sequence as a maximal path.

Definition 5.4.2 (Maximal Path)**.** *A directed path P in a full binary tree $T = (N, E)$ is sequence $P = (e^{[k]})_{k=1}^{\ell}$ of edges in E such that the $e_2^{[k]} = e_1^{[k+1]}$ for all $k = 1, 2, \ldots, \ell - 1$. A maximal path is a directed path $P = (e^{[k]})_{k=1}^{\ell}$ that cannot be extended.*

We now aim at describing the model h^{tree} in a more formal manner. Since the intuition about h is that $\boldsymbol{x}$ flows downward on edges selected by the Boolean functions, notation should only involve edges. Hence, for an edge $e = (e_1, e_2)$, define i_e as the feature index of the internal node e_1 at its tail. Also, for each edge $e = (e_1, e_2)$ define the Boolean function $R_e(\boldsymbol{x})$ by $R_{e_1}(\boldsymbol{x})$ if e is a left edge and $1 - R_{e_1}(\boldsymbol{x})$ otherwise. When $R_e(\boldsymbol{x}) = 1$ for some edge e, we say that $\boldsymbol{x}$ "flows though" the edge e. Now, for each maximal path P ending at some leaf $l \in L$ define $h^P : \mathbb{R}^d \to \mathbb{R}$ by

$$h^P(\boldsymbol{x}) = v_l \prod_{e \in P} R_e(\boldsymbol{x}). \tag{5.17}$$

This step function outputs zero unless the input flows through all the edges in the maximal path P. For example, in Figure 5.3, the function h^P associated with $P = (\,(1,2),(2,5)\,)$ is $h^P(\boldsymbol{x}) = v_5 \mathbb{1}(x_2 \leq 0.5)\mathbb{1}(x_3 > 1.33)$. Finally, given an input $\boldsymbol{x}$, there exists only one maximal path P such that $h^P(\boldsymbol{x}) \neq 0$. This is because any given $\boldsymbol{x}$ can only flow through one path P from root to leaf. Hence,

$$h^{\text{tree}}(\boldsymbol{x}) = \sum_P h^P(\boldsymbol{x}), \tag{5.18}$$

where the sum is taken over all maximal paths in the Decision Tree. Note that the decision tree h^{tree} of Figure 5.3 can be written as

$$\begin{aligned} h^{\text{tree}}(\boldsymbol{x}) = \quad & v_4 \mathbb{1}(x_2 \leq 0.5)\mathbb{1}(x_3 \leq 1.33) + v_5 \mathbb{1}(x_2 \leq 0.5)\mathbb{1}(x_3 > 1.33) \\ & + v_6 \mathbb{1}(x_2 > 0.5)\mathbb{1}(x_1 \leq 0.25) + v_7 \mathbb{1}(x_2 > 0.5)\mathbb{1}(x_1 > 0.25). \end{aligned}$$

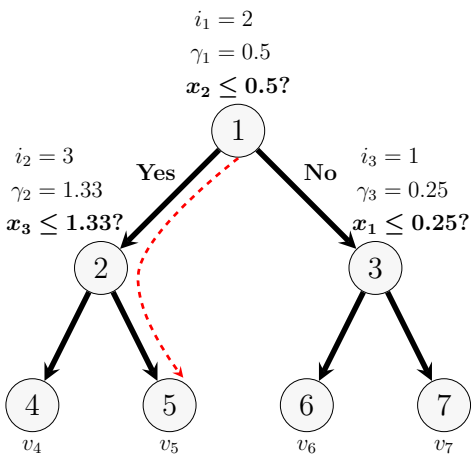


Figure 5.3 Basic example of Binary Decision Tree. In red we highlight the maximal path followed by the input $\boldsymbol{x} = (3.4, 0.2, 2)^T$.

5.4.2 Anchored Decompositions

Anchored decompositions assume that input space is the Cartesian product of arbitrary feature spaces $\mathcal{X} = \prod_{j=1}^{d} \mathcal{X}_j$. Yet, as just discussed, the boolean function $R_n(\boldsymbol{x}) := \mathbb{1}(x_{i_n} \leq \gamma_n)$ encoded at each internal node n requires a scalar input $x_{i_n} \in \mathbb{R}$. To remain general w.r.t the domain $\mathcal{X}$, we assume that an embedding $\boldsymbol{\xi} : \mathcal{X} \to \mathbb{R}^{d'}$ is performed before feeding the input to the decision tree model h^{tree}

$$h = h^{\text{tree}} \circ \boldsymbol{\xi}. \tag{5.19}$$

Since the input of the decision tree is $\boldsymbol{\xi}(\boldsymbol{x}) \in \mathbb{R}^{d'}$, each internal node is actually labeled by a pair $(i_n, \gamma_n) \in [d'] \times \mathbb{R}$ that encodes the boolean function $R_n : \mathbb{R}^{d'} \to \{0, 1\}$ given by $R_n(\boldsymbol{\xi}(\boldsymbol{x})) := \mathbb{1}(\xi(\boldsymbol{x})_{i_n} \leq \gamma_n)$. Crucially, the binary splits are performed along the embeddings and not necessarily along the original features of $\boldsymbol{x}$. The replace-function is also applied on the embedding

$$h(\boldsymbol{r}_u^{\boldsymbol{z}}(\boldsymbol{x})) = h^{\text{tree}}\Big(\boldsymbol{r}_{\mathcal{I}^{-1}(u)}^{\boldsymbol{\xi}(\boldsymbol{z})}(\boldsymbol{\xi}(\boldsymbol{x}))\Big), \tag{5.20}$$

where we have redefined the pre-image $\mathcal{I}^{-1} \equiv \mathcal{I}_{\boldsymbol{\xi}}^{-1}$ for simplicity. Given a *fixed* anchor $\boldsymbol{z}$ and evaluation point $\boldsymbol{x}$, we now present how to exploit the tree structure to efficiently compute the $\boldsymbol{z}$-Anchored components $h_{u,\boldsymbol{z}}(\boldsymbol{x})$ for any $u \subseteq [d]$. Since the anchored decomposition is linear w.r.t the model output,

$$h_{u,\boldsymbol{z}}(\boldsymbol{x}) = (h^{\text{tree}} \circ \boldsymbol{\xi})_{u,\boldsymbol{z}}(\boldsymbol{x}) = (\sum_P h^P \circ \boldsymbol{\xi})_{u,\boldsymbol{z}}(\boldsymbol{x}) = \sum_P (h^P \circ \boldsymbol{\xi})_{u,\boldsymbol{z}}(\boldsymbol{x}). \tag{5.21}$$

We can now fix a maximal path P in the decision tree and focus on its decomposition $(h^P \circ \boldsymbol{\xi})_{u,\boldsymbol{z}}(\boldsymbol{x})$. We first identify four types of edges X, Z, F, B which can occur in P.

Definition 5.4.3 (Edge Type)**.** *We say that an edge $e \in E$ is of*

$$\begin{aligned}
&\textit{Type } X \quad \textit{if} \quad R_e(\,\boldsymbol{\xi}(\boldsymbol{x})\,) = 1 \textit{ and } R_e(\,\boldsymbol{\xi}(\boldsymbol{z})\,) = 0\\
&\textit{Type } Z \quad \textit{if} \quad R_e(\,\boldsymbol{\xi}(\boldsymbol{x})\,) = 0 \textit{ and } R_e(\,\boldsymbol{\xi}(\boldsymbol{z})\,) = 1\\
&\textit{Type } F \quad \textit{if} \quad R_e(\,\boldsymbol{\xi}(\boldsymbol{x})\,) = 1 \textit{ and } R_e(\,\boldsymbol{\xi}(\boldsymbol{z})\,) = 1\\
&\textit{Type } B \quad \textit{if} \quad R_e(\,\boldsymbol{\xi}(\boldsymbol{x})\,) = 0 \textit{ and } R_e(\,\boldsymbol{\xi}(\boldsymbol{z})\,) = 0.
\end{aligned} \tag{5.22}$$

Here X stands for $\boldsymbol{x}$ flows, Z stands for $\boldsymbol{z}$ flows, F stands for both Flow, and B stands for both are Blocked.

Lemma 5.4.1. *If P contains an edge of type B, then*

$$\forall\, u \subseteq [d] \quad (h^P \circ \boldsymbol{\xi})_{u,\boldsymbol{z}}(\boldsymbol{x}) = 0.$$

Proof. Let e be an edge of type B in P. Then for every subset of features $u \subseteq [d]$, we have $R_e\Big(\boldsymbol{r}^{\boldsymbol{\xi}(\boldsymbol{z})}_{\mathcal{I}^{-1}(u)}(\boldsymbol{\xi}(\boldsymbol{x}))\Big) = 0$, which implies that $(h^P \circ \boldsymbol{\xi})(\,\boldsymbol{r}^{\boldsymbol{z}}_u(\boldsymbol{x})\,) = h^P\Big(\,\boldsymbol{r}^{\boldsymbol{\xi}(\boldsymbol{z})}_{\mathcal{I}^{-1}(u)}(\boldsymbol{\xi}(\boldsymbol{x}))\,\Big) \propto R_e\Big(\boldsymbol{r}^{\boldsymbol{\xi}(\boldsymbol{z})}_{\mathcal{I}^{-1}(u)}(\boldsymbol{\xi}(\boldsymbol{x}))\Big) = 0$. As a consequence, the u component of the anchored decomposition is $(h^P \circ \boldsymbol{\xi})_{u,\boldsymbol{z}}(\boldsymbol{x}) = \sum_{v\subseteq u}(-1)^{|u|-|v|}(h^P \circ \boldsymbol{\xi})(\,\boldsymbol{r}^{\boldsymbol{z}}_v(\boldsymbol{x})\,) = \sum_{v\subseteq u}(-1)^{|u|-|v|} \times 0 = 0$. □

Now, assuming P does not contain any type B edges, the following holds

$$\begin{aligned}
(h^P \circ \boldsymbol{\xi})(\boldsymbol{r}^{\boldsymbol{z}}_u(\boldsymbol{x})) &= v_l \prod_{e\in P} R_e\Big(\boldsymbol{r}^{\boldsymbol{\xi}(\boldsymbol{z})}_{\mathcal{I}^{-1}(u)}(\boldsymbol{\xi}(\boldsymbol{x}))\Big)\\
&= v_l \prod_{\substack{e\in P\\ e \text{ type X}}} R_e\Big(\boldsymbol{r}^{\boldsymbol{\xi}(\boldsymbol{z})}_{\mathcal{I}^{-1}(u)}(\boldsymbol{\xi}(\boldsymbol{x}))\Big) \prod_{\substack{e\in P\\ e \text{ type Z}}} R_e\Big(\boldsymbol{r}^{\boldsymbol{\xi}(\boldsymbol{z})}_{\mathcal{I}^{-1}(u)}(\boldsymbol{\xi}(\boldsymbol{x}))\Big) \underbrace{\prod_{\substack{e\in P\\ e \text{ type F}}} R_e\Big(\boldsymbol{r}^{\boldsymbol{\xi}(\boldsymbol{z})}_{\mathcal{I}^{-1}(u)}(\boldsymbol{\xi}(\boldsymbol{x}))\Big)}_{=1\ \forall\, u}\\
&= v_l \prod_{\substack{e\in P\\ e \text{ type X}}} \mathbb{1}(i_e \in \mathcal{I}^{-1}(u)) \prod_{\substack{e\in P\\ e \text{ type Z}}} \mathbb{1}(i_e \notin \mathcal{I}^{-1}(u)).\\
&= v_l \prod_{\substack{e\in P\\ e \text{ type X}}} \mathbb{1}(\mathcal{I}(i_e) \in u) \prod_{\substack{e\in P\\ e \text{ type Z}}} \mathbb{1}(\mathcal{I}(i_e) \notin u).
\end{aligned} \tag{5.23}$$

The second to last line follows from the observation that, for an edge e of type X , the only times $R_e\Big(\boldsymbol{r}^{\boldsymbol{\xi}(\boldsymbol{z})}_{\mathcal{I}^{-1}(u)}(\boldsymbol{\xi}(\boldsymbol{x}))\Big) = 1$ are when the i_eth component of $\boldsymbol{r}^{\boldsymbol{\xi}(\boldsymbol{z})}_{\mathcal{I}^{-1}(u)}(\boldsymbol{\xi}(\boldsymbol{x}))$ is set to $\boldsymbol{\xi}(\boldsymbol{x})$ and not $\boldsymbol{\xi}(\boldsymbol{z})$ ($i_e \in \mathcal{I}^{-1}(u)$). Alternatively, for an edge e of type Z , $R_e\Big(\boldsymbol{r}^{\boldsymbol{\xi}(\boldsymbol{z})}_{\mathcal{I}^{-1}(u)}(\boldsymbol{\xi}(\boldsymbol{x}))\Big) = 1$ if and

only if the i_eth component of $\boldsymbol{r}^{\boldsymbol{\xi}(\boldsymbol{z})}_{\mathcal{I}^{-1}(u)}(\boldsymbol{\xi}(\boldsymbol{x}))$ is set to $\boldsymbol{\xi}(\boldsymbol{z})$ and not $\boldsymbol{\xi}(\boldsymbol{x})$ ($i_e \notin \mathcal{I}^{-1}(u)$). We define the three following sets

$$\begin{aligned} S_X &:= \{i_e : e \in P, \text{ and } e \text{ is of type } X\}, \\ S_Z &:= \{i_e : e \in P, \text{ and } e \text{ is of type } Z\}. \\ S_{XZ} &:= S_X \cup S_Z. \end{aligned} \tag{5.24}$$

It is important to realize that the sets S_X, S_Z, and S_{XZ} depend on the current maximal path P. Still, we assume that P is fixed and hence we do not make the dependence explicit in the notation.

Lemma 5.4.2. *If $\mathcal{I}(S_X) \cap \mathcal{I}(S_Z) \neq \emptyset$, then*

$$\forall\, u \subseteq [d] \quad (h^P \circ \boldsymbol{\xi})_{u,\boldsymbol{z}}(\boldsymbol{x}) = 0. \tag{5.25}$$

Proof. Since $\mathcal{I}(S_X) \cap \mathcal{I}(S_Z) \neq \emptyset$, we can find an index $j \in [d]$ that belongs to both $\mathcal{I}(S_X)$ and $\mathcal{I}(S_Z)$. By definition of S_X and S_Z, this means that we can find in P an edge e of X and an edge e' of type Z such that $\mathcal{I}(i_e) = j = \mathcal{I}(i_{e'})$. Now using Equation 5.23, for all $u \subseteq [d]$

$$(h^P \circ \boldsymbol{\xi})(\, \boldsymbol{r}^{\boldsymbol{z}}_u(\boldsymbol{x})\,) \propto \mathbb{1}(\mathcal{I}(i_e) \in u)\mathbb{1}(\mathcal{I}(i_{e'}) \notin u) = \mathbb{1}(j \in u)\mathbb{1}(j \notin u) = 0.$$

The rest of the proof follows the same logic as the one for Lemma 5.4.1. □

Lemmas 5.4.1 & 5.4.2 provide sufficient conditions for avoiding the maximal path P during a tree traversal. The coming result highlights what features are not affected by the replace-function.

Lemma 5.4.3. *If a feature j does not belong to $\mathcal{I}(S_{XZ})$, then*

$$\forall\, u \subseteq [d] \setminus \{j\} \quad (h^P \circ \boldsymbol{\xi})(\, \boldsymbol{r}^{\boldsymbol{z}}_{u\cup\{j\}}(\boldsymbol{x})\,) = (h^P \circ \boldsymbol{\xi})(\, \boldsymbol{r}^{\boldsymbol{z}}_u(\boldsymbol{x})\,). \tag{5.26}$$

Said otherwise, such features are dummies and can be safely ignored when computing functional decompositions of $h^P \circ \boldsymbol{\xi}$.

Proof. Let $j \notin \mathcal{I}(S_{XZ})$ be a feature and $u \subseteq [d] \setminus \{j\}$ be an arbitrary subset of features that excludes it. Since $j \notin \mathcal{I}(S_{XZ})$, there are no edges e of type X or Z such that $\mathcal{I}(i_e) = j$. Moreover, since $j \neq \mathcal{I}(i_e)$ implies the equivalence $\mathcal{I}(i_e) \in u \iff \mathcal{I}(i_e) \in u \cup \{j\}$, Equation

5.23 yields

$$\begin{aligned}(h_P \circ \boldsymbol{\xi})(\boldsymbol{r}_u^{\boldsymbol{z}}(\boldsymbol{x})) &= v_l \prod_{\substack{e \in P \\ e \text{ type X}}} \mathbb{1}(\mathcal{I}(i_e) \in u) \prod_{\substack{e \in P \\ e \text{ type Z}}} \mathbb{1}(\mathcal{I}(i_e) \notin u) \\ &= v_l \prod_{\substack{e \in P \\ e \text{ type X}}} \mathbb{1}(\mathcal{I}(i_e) \in u \cup \{j\}) \prod_{\substack{e \in P \\ e \text{ type Z}}} \mathbb{1}(\mathcal{I}(i_e) \notin u \cup \{j\}) \\ &= (h_P \circ \boldsymbol{\xi})(\boldsymbol{r}_{u \cup \{j\}}^{\boldsymbol{z}}(\boldsymbol{x})).\end{aligned}$$

□

In the following key result, we assume w.l.o.g that the current maximal path contains no type B edges and that $\mathcal{I}(S_X)$ and $\mathcal{I}(S_Z)$ are disjoint. Indeed, failing to reach these requirements will simply cause the functional decomposition to be zero.

Lemma 5.4.4. *If P contains no type B edges and the sets $\mathcal{I}(S_X)$ and $\mathcal{I}(S_Z)$ are disjoint, then for any $u \subseteq \mathcal{I}(S_{XZ})$ we have*

$$(h^P \circ \boldsymbol{\xi})(\, \boldsymbol{r}_u^{\boldsymbol{z}}(\boldsymbol{x})\,) = \begin{cases} v_l & \text{if } u = \mathcal{I}(S_X) \\ 0 & \text{otherwise,} \end{cases} \tag{5.27}$$

where $l \in L$ is the leaf node at the end of P. The proof is presented in Appendix B.3

The following Theorem demonstrates how to efficiently compute any component of the $\boldsymbol{z}$-Anchored Decomposition of the decision stump $h^P \circ \boldsymbol{\xi}$. This Theorem is the culmination of all previous results : Lemmas 5.4.1, 5.4.2, 5.4.3 & 5.4.4.

Theorem 5.4.1. *If P contains no type B edges and the sets $\mathcal{I}(S_X)$ and $\mathcal{I}(S_Z)$ are disjoint, then*

$$(h^P \circ \boldsymbol{\xi})_{u,\boldsymbol{z}}(\boldsymbol{x}) = \begin{cases} (-1)^{|u|-|\mathcal{I}(S_X)|}\, v_l & \text{if } \mathcal{I}(S_X) \subseteq u \subseteq \mathcal{I}(S_{XZ}) \\ 0 & \text{otherwise.} \end{cases} \tag{5.28}$$

The proof is presented in Appendix B.3

In virtue of Theorem 5.4.1, one only needs to store the sets $\mathcal{I}(S_X)$ and $\mathcal{I}(S_Z)$ in memory when traversing the tree. Once a leaf is reached (and P becomes a maximal path), these two sets are used to compute functional components.

It remains to optimize the tree traversal itself. Lemmas 5.4.1 & 5.4.2 provide sufficient conditions for when the maximal path cannot contribute to the $\boldsymbol{z}$-Anchored Decomposition. Hence, the tree traversal must avoid such maximal paths.

First, from Lemma 5.4.1, one should avoid going down type B edges during the traversal. This is because any maximal path P that follows this edge will contain a type B edge and hence have a null functional component. Therefore, if during the tree traversal one encounters a split where both $\boldsymbol{x}$ and $\boldsymbol{z}$ flow through the same edge, only follow the type F edge, see Figure 5.4(a).

Second, Lemma 5.4.2 states that the sets $\mathcal{I}(S_X)$ and $\mathcal{I}(S_Z)$ must remain disjoint during the tree traversal. Otherwise, any maximal path will have null functional decompositions resulting in wasted computations. Hence, if a node n is encountered such that $\boldsymbol{x}$ and $\boldsymbol{z}$ go different ways, and the associated feature $\mathcal{I}(i_n)$ is already in $\mathcal{I}(S_{XZ})$, then

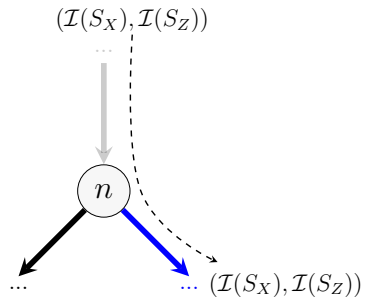
1. if $\mathcal{I}(i_n) \in \mathcal{I}(S_X)$, go down the same direction as $\boldsymbol{x}$
2. if $\mathcal{I}(i_n) \in \mathcal{I}(S_Z)$, go down the same direction as $\boldsymbol{z}$.

This is necessary to keep the sets $\mathcal{I}(S_X)$ and $\mathcal{I}(S_Z)$ disjoint. These two cases are illustrated in Figure 5.4(b) and (c).

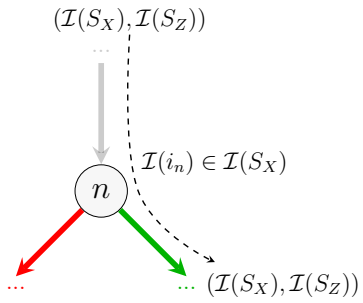
The final scenario that can occur is reaching a node where $\boldsymbol{x}$ and $\boldsymbol{z}$ do not go the same way and the feature i_n is not already in $\mathcal{I}(S_{XZ})$. In that case. one must visit both branches and update the sets $\mathcal{I}(S_X), \mathcal{I}(S_Z)$ accordingly, see Figure 5.4 (d). We have now identified every scenario that can occur during the tree traversal, as well as the two data structures $\mathcal{I}(S_X), \mathcal{I}(S_Z)$ that must be stored dynamically.

Algorithm 4 presents the pseudocode for efficiently computing the $\{\boldsymbol{H}^u\}_{u \in U}$. Since the derivation relies on a fixed anchor $\boldsymbol{z}$ and evaluation point $\boldsymbol{x}$, we must repeat the tree-traversal NM. However, in the scenario where background data and foreground data are identical $\boldsymbol{z}^{(i)} = \boldsymbol{x}^{(i)}$ and $N = M$, a symmetry argument allows to only do $N(N-1)/2$ tree traversals instead of N^2. Since the $\boldsymbol{z}$-Anchored Decomposition is linear w.r.t the model output, Algorithm 4 can be generalized to tree ensembles (*e.g.* Random Forests and Gradient Boosted trees) by looping over each individual tree and summing the results.

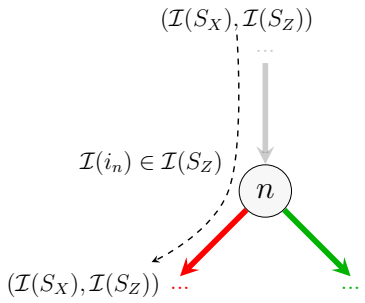
We finally analyze the complexity of this algorithm when explaining an ensemble of T trees of depth D on N instances using M background instances. For simplicity, we only calculate main effects $\{\boldsymbol{H}^k\}_{k=1}^d$. In a model-agnostic setting, there are $\mathcal{O}(NMd)$ functional calls, and each call must visit T trees and at most D nodes per tree. So, the model-agnostic time complexity is $\mathcal{O}(N\,M\,T\,D\,d)$. For Algorithm 4, in the worst case one would visit $\mathcal{O}(\min\{D, d\})$ leaves per tree. One of these leaves would have $\mathcal{I}(S_X) = \emptyset$ and so line 11 would iterate over at most $\min\{D, d\}$ features. The remaining visited leaves would all have $|\mathcal{I}(S_X)| = 1$ and line 11 would iterate over a single feature. The worst-case time complexity is thus



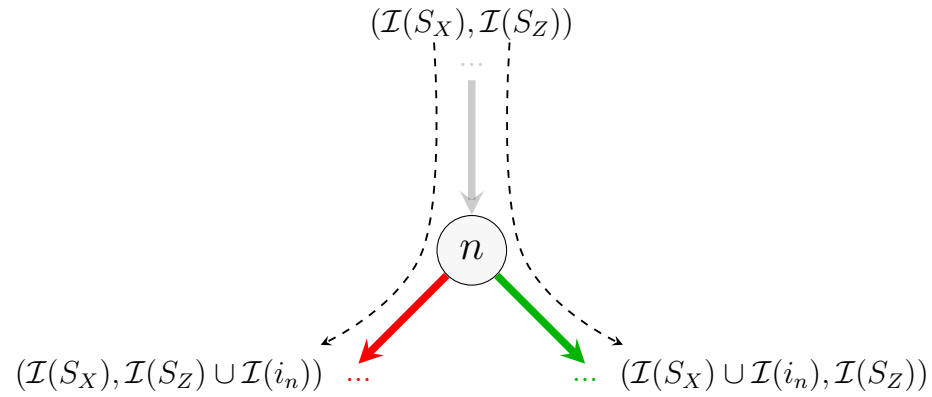
(a) Scenario 1: Ignore the type B edge and follow the type F edge without storing $\mathcal{I}(i_n)$.



(b) Scenario 2: Feature seen before in $\mathcal{I}(S_X)$ so go down the direction of $\boldsymbol{x}$.



(c) Scenario 2: Feature seen before in $\mathcal{I}(S_Z)$ so go down the direction of $\boldsymbol{z}$.



(d) Scenario 3: Feature $\mathcal{I}(i_n)$ never seen before in $\mathcal{I}(S_X)$ and $\mathcal{I}(S_Z)$ so go down both directions and update $\mathcal{I}(S_X)$ and $\mathcal{I}(S_Z)$ accordingly.

Figure 5.4

$\mathcal{O}(N\,M\,T\,\min\{D, d\})$. When $D < d$, the runtime of Algorithm 4 becomes independent of the number of features d.

Algorithm 4 Compute $\{\boldsymbol{H}^u\}_{u\in U}$ efficiently for h^{tree}.

$\boldsymbol{H}^u = \texttt{zeros}(N, M) \quad \forall\, u \in U$;
for $i = 1, 2 \ldots, N$ **do**
  for $j = 1, 2 \ldots, M$ **do**
    $\boldsymbol{x} = \boldsymbol{x}^{(i)}$;
    $\boldsymbol{z} = \boldsymbol{z}^{(j)}$;
    stack $= ((1, \emptyset, \emptyset))$;
    while not stack.isempty() **do**
      $(n, \mathcal{I}(S_X), \mathcal{I}(S_Z)) =$ stack.pop();
      % Reach a leaf so apply Theorem 5.4.1
      if $n \in L$ **then**
        for $u \in U$ s.t. $\mathcal{I}(S_X) \subseteq u \subseteq I(S_{XZ})$ **do**
          H^u_{ij} += $(-1)^{|u|-|\mathcal{I}(S_X)|} \times v_l$;
      % Avoid Type B edge
      else if $\boldsymbol{x}_{\text{child}} = \boldsymbol{z}_{\text{child}}$ **then**
        stack.push$((\boldsymbol{x}_{\text{child}}, \mathcal{I}(S_X), \mathcal{I}(S_Z)))$;
      % Keep $\mathcal{I}(S_X)$ and $\mathcal{I}(S_Z)$ disjoint
      else if $\mathcal{I}(i_n) \in \mathcal{I}(S_X) \cup \mathcal{I}(S_Z)$ **then**
        if $\mathcal{I}(i_n) \in \mathcal{I}(S_X)$ **then**
          stack.push$((\boldsymbol{x}_{\text{child}}, \mathcal{I}(S_X), \mathcal{I}(S_Z)))$;
        else
          stack.push$((\boldsymbol{z}_{\text{child}}, \mathcal{I}(S_X), \mathcal{I}(S_Z)))$;
      else
        if $\exists u \in U$ s.t. $\mathcal{I}(S_X) \cup \mathcal{I}(i_n) \subseteq u$ **then**
          stack.push$((\boldsymbol{x}_{\text{child}}, \mathcal{I}(S_X) \cup \mathcal{I}(i_n), S_Z))$;
        stack.push$((\boldsymbol{z}_{\text{child}}, \mathcal{I}(S_X), \mathcal{I}(S_Z) \cup \mathcal{I}(i_n)))$;
return $\{\boldsymbol{H}^u\}_{u\in U}$;

5.4.3 Shapley Values

Shapley Values are an alternative to computing functional components for all $u \subseteq [d]$. Let

$$\phi_k^{\text{SHAP-int}}(h^P \circ \boldsymbol{\xi}, \boldsymbol{x}, \boldsymbol{z}) = \sum_{u \subseteq [d]: k \in u} \frac{(h^P \circ \boldsymbol{\xi})_{u,\boldsymbol{z}}(\boldsymbol{x})}{|u|} \tag{5.29}$$

be the Shapley Value for the $\boldsymbol{z}$-Anchored Decomposition evaluated as $\boldsymbol{x}$. Note that Shapley Values are a compression of the full decomposition $\{h_{u,\boldsymbol{z}}(\boldsymbol{x})\}_{u \subseteq [d]}$ where u-way interactions are shared evenly between all features involved.

It is more common to express Shapley Values using notation inherited from Cooperative Game Theory. Letting

$$\nu^{\text{int}}_{h,\boldsymbol{x},\boldsymbol{z}}(u) := (h^P \circ \boldsymbol{\xi})(\, \boldsymbol{r}^{\boldsymbol{z}}_u(\boldsymbol{x})\,) \tag{5.30}$$

be the Interventional Game, Shapley Values are often written

$$\phi_k^{\text{SHAP-int}}(h^P \circ \boldsymbol{\xi}, \boldsymbol{x}, \boldsymbol{z}) = \sum_{u \subseteq [d] \setminus \{k\}} W(|u|, d) \left(\nu^{\text{int}}_{h,\boldsymbol{x},\boldsymbol{z}}(u \cup \{k\}) - \nu^{\text{int}}_{h,\boldsymbol{x},\boldsymbol{z}}(u) \right), \tag{5.31}$$

with $W(|u|, d) := |u|!\,(d - |u| - 1)!/d!$. This formulation involving a summation over all subsets $u \subseteq [d] \setminus \{k\}$ highlights that computing Shapley Values scales exponentially with d in the general case. Yet, the following theorem demonstrates that this exponential complexity becomes $\mathcal{O}(1)$ for a decision stump $h^P \circ \boldsymbol{\xi}$.

Theorem 5.4.2. *Let P be a maximal path that contains no type B edges and let the sets $\mathcal{I}(S_X)$ and $\mathcal{I}(S_Z)$ be disjoint. Moreover, define the interventional game $\nu^{int}_{h,\boldsymbol{x},\boldsymbol{z}}(u) := (h^P \circ \boldsymbol{\xi})(\, \boldsymbol{r}^{\boldsymbol{z}}_u(\boldsymbol{x})\,)$. Then all features that are not in $\mathcal{I}(S_{XZ})$ are dummies and get a zero Shapley value. Features $k \in \mathcal{I}(S_{XZ})$ get a Shapley value of*

$$\phi_k^{SHAP\text{-}int}(h^P \circ \boldsymbol{\xi}, \boldsymbol{x}, \boldsymbol{z}) = \sum_{u \subseteq \mathcal{I}(S_{XZ}) \setminus \{k\}} W(|u|, |\mathcal{I}(S_{XZ})|) \Big(\nu^{int}_{h,\boldsymbol{x},\boldsymbol{z}}(u \cup \{k\}) - \nu^{int}_{h,\boldsymbol{x},\boldsymbol{z}}(u) \Big) \tag{5.32}$$

The exponential cost $\mathcal{O}(2^{|\mathcal{I}(S_{XZ})|})$ of computing these terms reduces to $\mathcal{O}(1)$ following

$$k \in \mathcal{I}(S_X) \Rightarrow \phi_k^{SHAP\text{-}int}(h^P \circ \boldsymbol{\xi}, \boldsymbol{x}, \boldsymbol{z}) = W(|\mathcal{I}(S_X)| - 1, |\mathcal{I}(S_{XZ})|) v_l \tag{5.33}$$

$$k \in \mathcal{I}(S_Z) \Rightarrow \phi_k^{SHAP\text{-}int}(h^P \circ \boldsymbol{\xi}, \boldsymbol{x}, \boldsymbol{z}) = -W(|\mathcal{I}(S_X)|, |\mathcal{I}(S_{XZ})|) v_l. \tag{5.34}$$

The proof is presented in Appendix B.3.

With this Theorem, it is possible to efficiently compute Shapley Values $\phi_k^{\text{SHAP-int}}(h^{\text{tree}}, \boldsymbol{x}^{(i)}, \boldsymbol{z}^{(j)})$ for collections of foreground $\{\boldsymbol{x}^{(i)}\}_{i=1}^N \sim \mathcal{F}^N$ and background $\{\boldsymbol{z}^{(j)}\}_{j=1}^M \sim \mathcal{B}^M$ samples. The procedure is presented in Algorithm 5 and its results are stored in the form of matrices $\{\boldsymbol{\Phi}^k\}_{k=1}^d$ of size $N \times M$

$$\Phi_{ij}^k := \phi_k^{\text{SHAP-int}}(h^{\text{tree}}, \boldsymbol{x}^{(i)}, \boldsymbol{z}^{(j)}). \tag{5.35}$$

We now discuss the time complexity of Algorithm 5 when explaining T trees of depth D. In the worst case, one would need to visit $2^{\min\{D,d\}}$ leaves and at each iterate over at most $\min\{D, d\}$ features. The total complexity is hence $\mathcal{O}(N\,M\,T\,\min\{D,d\}2^{\min\{D,d\}})$. It is actually possible to attain complexity $\mathcal{O}(N\,M\,T\,2^{\min\{D,d\}})$ by avoiding the for-loops at line 11 and 13 of Algorithm 5. Rather than looping over features, the contributions $W(|\mathcal{I}(S_X)|-1, |\mathcal{I}(S_{XZ})|)\,v_n$ and -$W(|\mathcal{I}(S_X)|, |\mathcal{I}(S_{XZ})|)\,v_n$ can be propagated up the tree via dynamic programming and recursion. This leads to a constant time cost per node visited. Algorithm 5 only avoids the exponential complexity of Shapley Values w.r.t d if $D < d$, meaning that decision trees should be kept shallow.

Unlike the PermutationSHAP algorithm presented in the previous Chapter, Algorithm 5 returns values $\phi_k^{\text{SHAP-int}}(h^{\text{tree}}, \boldsymbol{x}^{(i)}, \boldsymbol{z}^{(j)})$ that are exact. This has several implications.

First, the Interventional Shapley Values $\phi_k^{\text{SHAP-int}}(h^{\text{tree}}, \boldsymbol{x}^{(i)}, \mathcal{B})$ can be estimated by averaging the $\boldsymbol{\Phi}^k$ matrix along its second axis

$$\frac{1}{M}\sum_{j=1}^M \Phi_{ij}^k \xrightarrow{p} \phi_k^{\text{SHAP-int}}(h^{tree}, \boldsymbol{x}^{(i)}, \mathcal{B}). \tag{5.36}$$

The Central Limit Theorem is also applicable to compute asymptotic confidence intervals.

Theorem 5.4.3 (TreeSHAP Confidence Interval)**.** *Let $\{\boldsymbol{z}^{(j)}\}_{j=1}^M \sim \mathcal{B}^M$ be a sequence of M iid background observations, moreover assume $\phi_k^{SHAP\text{-}int}(h^{tree}, \boldsymbol{x}, \boldsymbol{z})$ has finite first and second moments w.r.t $\boldsymbol{z} \sim \mathcal{B}$ for any $\boldsymbol{x} \in \mathcal{X}$. Then, the following holds for any $i \in [N]$, and $\delta \in]0,1[$:*

$$\lim_{M\to\infty} \mathbb{P}\left(\Big|\frac{1}{M}\sum_{j=1}^M \Phi_{ij}^k - \phi_k^{SHAP\text{-}int}(h^{tree}, \boldsymbol{x}^{(i)}, \mathcal{B})\Big| \geq F_{\mathcal{N}(0,1)}^{-1}(1-\delta/2))\frac{s_M^{(i,k)}}{\sqrt{M}}\right) \leq \delta,$$

where $F_{\mathcal{N}(0,1)}^{-1}$ is the inverse Cumulative Distribution Function (CDF) of the standard normal distribution, and $s_M^{(i,k)} = [1/M\sum_{j=1}^M(\Phi_{ij}^k - \frac{1}{M}\sum_{\ell=1}^M \Phi_{i\ell}^k)^2]^{1/2}$ is the sample variance.

In opposition to Theorem 4.3.1 presented in model-agnostic settings, Theorem 5.4.3 need not consider errors caused by sampling random permutations.

Algorithm 5 Compute the SHAP matrices $\{\boldsymbol{\Phi}^k\}_{k=1}^d$ for h^{tree}.

$\boldsymbol{\Phi}^k = \texttt{zeros}(N, M)$ for $k = 1, 2, \ldots, d$;
for $i = 1, 2 \ldots, N$ **do**
  for $j = 1, 2 \ldots, M$ **do**
    $\boldsymbol{x} = \boldsymbol{x}^{(i)}$;
    $\boldsymbol{z} = \boldsymbol{z}^{(j)}$;
    stack $= ((1, \emptyset, \emptyset))$;
    while not stack.isempty() **do**
      $(n, \mathcal{I}(S_X), \mathcal{I}(S_Z)) =$ stack.pop();
      % Reach a leaf so apply Theorem 5.4.2
      if $n \in L$ **then**
        for $k \in \mathcal{I}(S_X)$ **do**
          $\Phi_{ij}^k \mathrel{+}= W(|\mathcal{I}(S_X)| - 1, |\mathcal{I}(S_{XZ})|)\, v_n$;
        for $k \in \mathcal{I}(S_Z)$ **do**
          $\Phi_{ij}^k \mathrel{-}= W(|\mathcal{I}(S_X)|, |\mathcal{I}(S_{XZ})|)\, v_n$;
      % Avoid Type B edge
      else if $\boldsymbol{x}_{\text{child}} = \boldsymbol{z}_{\text{child}}$ **then**
        stack.push$((\boldsymbol{x}_{\text{child}}, \mathcal{I}(S_X), \mathcal{I}(S_Z)))$;
      % Keep $\mathcal{I}(S_X)$ and $\mathcal{I}(S_Z)$ disjoint
      else if $\mathcal{I}(i_n) \in \mathcal{I}(S_X) \cup \mathcal{I}(S_Z)$ **then**
        if $\mathcal{I}(i_n) \in \mathcal{I}(S_X)$ **then**
          stack.push$((\boldsymbol{x}_{\text{child}}, \mathcal{I}(S_X), \mathcal{I}(S_Z)))$;
        else
          stack.push$((\boldsymbol{z}_{\text{child}}, \mathcal{I}(S_X), \mathcal{I}(S_Z)))$;
      else
        stack.push$((\boldsymbol{x}_{\text{child}}, \mathcal{I}(S_X) \cup \mathcal{I}(i_n), S_Z))$;
        stack.push$((\boldsymbol{z}_{\text{child}}, \mathcal{I}(S_X), \mathcal{I}(S_Z) \cup \mathcal{I}(i_n)))$;
return $\{\boldsymbol{\Phi}^k\}_{k=1}^d$;

Second, recall from Definition 2.3.6 that any Fairness metric (cf. Equation 2.28) can be decomposed as the sum of the feature attributions

$$\boldsymbol{\Phi}^{\text{Fair}}(h, \mathcal{F}, \mathcal{B}) := \mathop{\mathbb{E}}_{\substack{\boldsymbol{x}\sim\mathcal{F}\\ \boldsymbol{z}\sim\mathcal{B}}}[\boldsymbol{\phi}^{\text{SHAP-int}}(h, \boldsymbol{x}, \boldsymbol{z})], \tag{5.37}$$

where the correct choices of distributions $\mathcal{F}$ and $\mathcal{B}$ for a given Fairness metric are given in Table 2.1. Since $\phi_k^{\text{SHAP-int}}(h^{\text{tree}}, \boldsymbol{x}^{(i)}, \boldsymbol{z}^{(j)})$ are computed exactly for tree ensembles, they can be used to approximate the Fairness attribution. Letting $\{\boldsymbol{x}^{(i)}\}_{i=1}^N \sim \mathcal{F}^N$ and $\{\boldsymbol{z}^{(j)}\}_{j=1}^M \sim \mathcal{B}^M$ be samples from two subgroups,

$$\frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M}\Phi_{ij}^k \xrightarrow{p} \Phi_k^{\text{Fair}}(h, \mathcal{F}, \mathcal{B}). \tag{5.38}$$

The estimator being a two-samples U-statistics, it enjoys asymptotic normality [Lee, 2019, Section 3.7.1].

Theorem 5.4.4 (FairSHAP Confidence Interval)**.** *Let $\{\boldsymbol{x}^{(i)}\}_{i=1}^N \sim \mathcal{F}^N$ and $\{\boldsymbol{z}^{(j)}\}_{j=1}^M \sim \mathcal{B}^M$ be iid samples from the foreground and background. Moreover, assume that the variance $\mathbb{V}_{\boldsymbol{x}\sim\mathcal{F},\boldsymbol{z}\sim\mathcal{B}}[\phi_k^{SHAP\text{-}int}(h, \boldsymbol{x}, \boldsymbol{z})]$ is finite. Then the following holds for any $\delta \in]0, 1[$*

$$\lim_{\substack{N+M\to\infty\\ s.t.\ N/(N+M)\to p\in(0,1)}} \mathbb{P}\left(\left|\frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M}\Phi_{ij}^k - \Phi_k^{Fair}(h, \mathcal{F}, \mathcal{B})\right| \geq \frac{F^{-1}_{\mathcal{N}(0,1)}(1-\delta/2)}{\sqrt{M+N}}\left[\frac{\sigma_{10}^2}{p} + \frac{\sigma_{01}^2}{1-p}\right]\right) = \delta,$$

where $\sigma_{10}^2 = \mathbb{V}_{\boldsymbol{x}\sim\mathcal{F}}[\,\mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[\phi_k^{SHAP\text{-}int}(h, \boldsymbol{x}, \boldsymbol{z})]\,]$ and $\sigma_{01}^2 = \mathbb{V}_{\boldsymbol{z}\sim\mathcal{B}}[\,\mathbb{E}_{\boldsymbol{x}\sim\mathcal{F}}[\phi_k^{SHAP\text{-}int}(h, \boldsymbol{x}, \boldsymbol{z})]\,]$.

5.4.4 Experiments

As discussed in Section 5.4.2, computing the matrices $\{\boldsymbol{H}\}_{k=1}^d$ takes time $\mathcal{O}(N\,M\,T\,D\,d)$ in model-agnostic settings and $\mathcal{O}(N\,M\,T\,\min\{D, d\})$ using the Algorithm 4 specific to trees. Therefore, the model-specific algorithm is extremely advantageous when $D < d$ seeing as its runtime becomes independent of the number of features d.

We illustrate this optimization on the real-world dataset NOMAO [Candillier and Lemaire, 2012] involving $d = 119$ features and $N = 34K$ instances. This is a classification dataset based on features with cryptic names : `v1`,`v2`, ..., `v119`, that are not well explained in the online documentation. Nonetheless, this dataset was utilized because d is very large so Algorithm 2 should dominate its model-agnostic counterpart. Following the same methodology as [Neuhof and Benjamini, 2024], we sampled the train/test sets with 60:40 ratio and trained Gradient Boosted Trees with test accuracies of 97%.

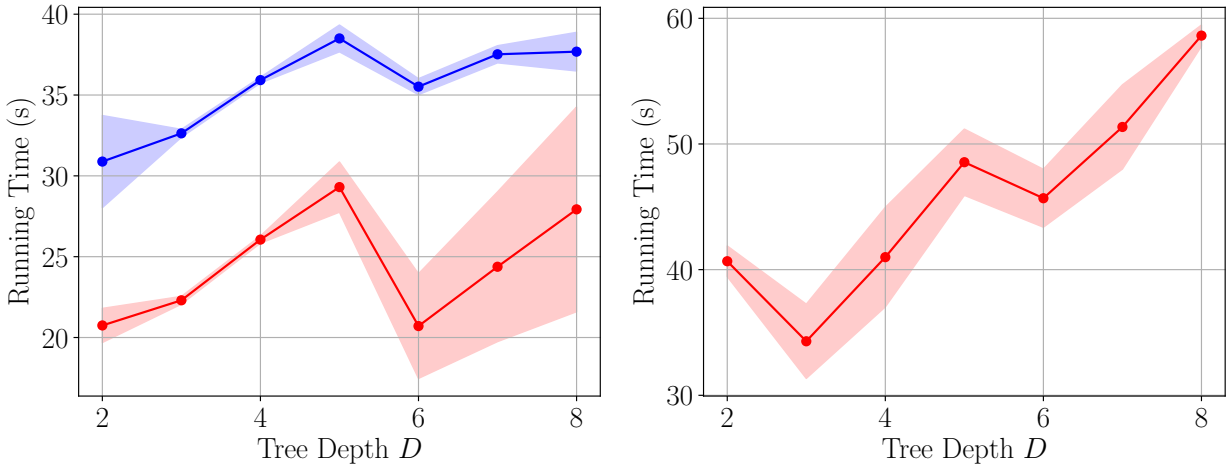


Figure 5.5 Runtime Comparisons. (Left) Computing the $\{\boldsymbol{H}\}_{k=1}^{d}$ matrices with the model-agnostic (blue) and model-specific (red) algorithms. (Right) Computing the Interventional Shapley Values of 5000 test set instances with the model-specific algorithm.

The 1000×1000 matrices $\{\boldsymbol{H}^k\}_{k=1}^{d}$ were computed with model-agnostic and model-specific algorithms. Runtimes for various maximum tree depth D are shown in Figure 5.5 (Left). The two algorithms present similar trends as D increases but the model-specific implementation consistently takes about 10 seconds less. Note that the runtimes are not increasing monotonically with D as suggested by our complexity analysis. This is because the worst-case complexity ignores the training dynamics of Gradient Boosted Trees.

It was demonstrated in Section 4.1.3 that the matrices $\{\boldsymbol{H}\}_{k=1}^{d}$ could estimate multiple Global Feature Importance. Figure 5.7 presents the PDP-[2], PFI, PDP-Variance, and Marginal-Sobol for a GBT of depth 8 trained on NOMAO. The four importance measures disagree since features are correlated and interact with each other. For example, the PDP-[2] and PFI importance of feature `V1` disagree greatly, meaning that this feature must interact with some others. Moreover, the Marginal-Sobol importance of `V30` is a lot larger than the PFI seeing as this feature is highly correlated with `V29`, `V28`, `V27`.

Despite their disagreements, all techniques agree on the ordering of the top-4 features `V90`, `V6`, `V97`, `V1`. Hence, we confirm that these four features are really important to the model and study them at a more granular level. To do so, we compute the Interventional Shapley Values of 5000 test data points with Algorithm 5. The runtimes are presented in Figure 5.5 (Right). No curve is presented for exact model-agnostic algorithms since they are intractable on this dataset. The model-specific algorithm is tractable because its worst-case complexity is $\mathcal{O}(N\,M\,T\,2^D)$ whenever $D < d$, which is the case here. The predicted exponential run-time w.r.t D appears to be conservative in this instance.

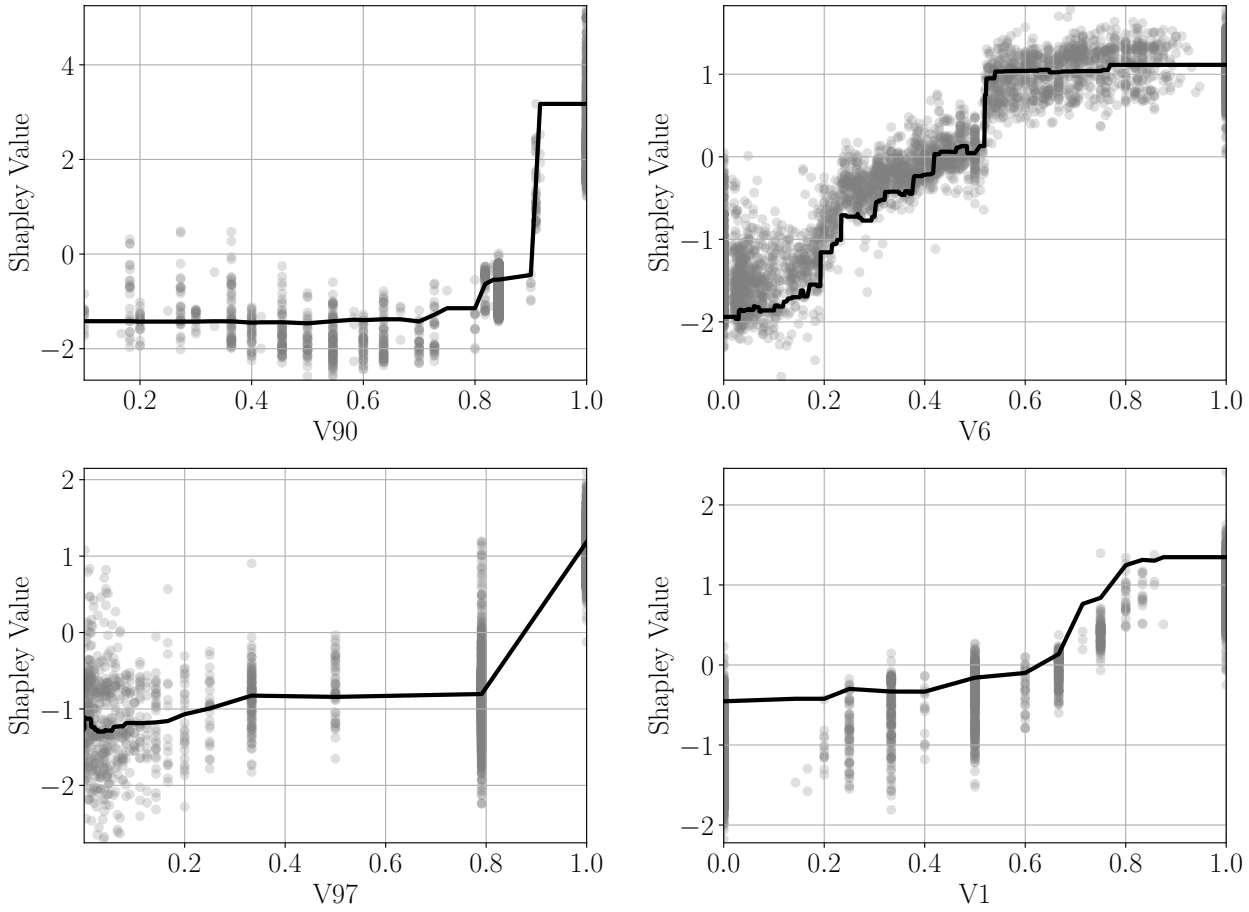


Figure 5.6 Local Feature Attributions of the Top-4 features on the NOMAO dataset. The Interventional Shapley Values are shown as gray dots while the PDP is plotted as a black line.

Figure 5.6 illustrate the Shapley Values of the top four features `V90`, `V6`, `V97`, `V1` as gray dots along with the Partial Dependence Plot (*i.e.* $h_{j,\mathcal{B}}(\boldsymbol{x})$) as a black line. While monotonic trends are apparent in the PDP, there is no definitive trend in Shapley Values. This is because the Shapley Values spread vertically and appear noisy as a result. Some extreme exaples are Shapley Values at `V6=0`, `V97=0`, and `V1=0`, where the vertical spread is so large that Shapley Values cross the origin. This is problematic because the *sign* of a local feature attribution is an important part of their interpretation. A positive (resp. negative) sign implies that a feature has increased (resp. decrease) the model output relative to the baseline. Given the current Shapley Values, we cannot confirm whether low values of `V6`, `V97`, and `V1` increase of decrease the model output relative to the average prediction.

Part II of this manuscript will reduce the disagreement between the PDP (dark line) and the Shapley Values (grey dots) by carefully designing the background $\mathcal{B}$.

Contributions

To conclude this Chapter, we showed how to efficiently compute the $\boldsymbol{z}$-Anchored Decomposition, the Interventional Decomposition, and the Shapley Values for a variety of ML models : Additive models, Kernel Methods, and Tree Ensembles. These novel algorithms will be employed throughout the remainder of the manuscript whenever one of these hypothesis classes is studied.

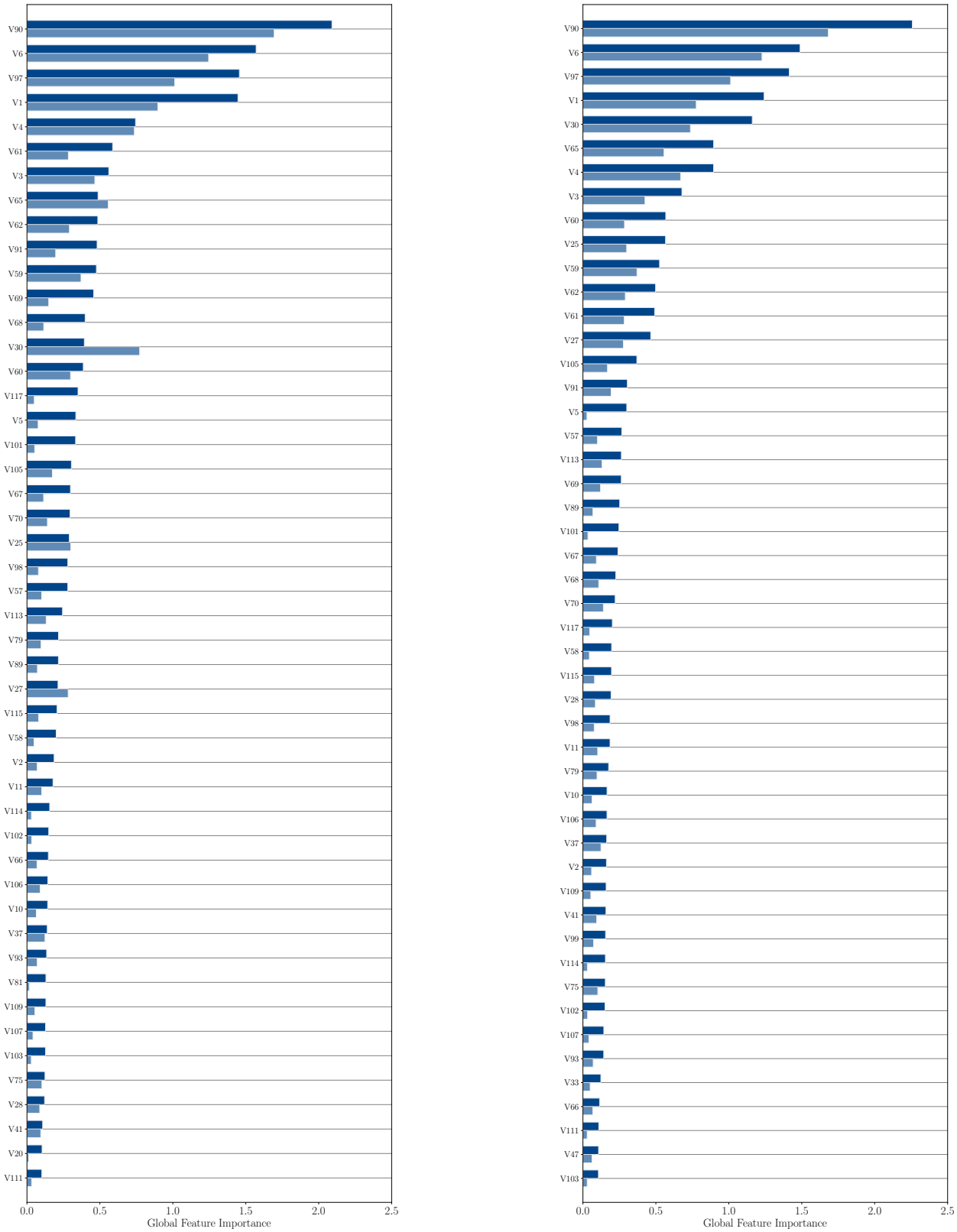


Figure 5.7 Global Feature Importance of a GBT fitted on the NOMAO dataset. (Left) The PFI and PDP-[2] importance are shown as opaque and transparent bars. (Right) The Marginal-Sobol and PDP-Variance importance are shown as opaque and transparent bars.

CHAPTER 6 INTERACTION DISAGREEMENT

6.1 Motivation

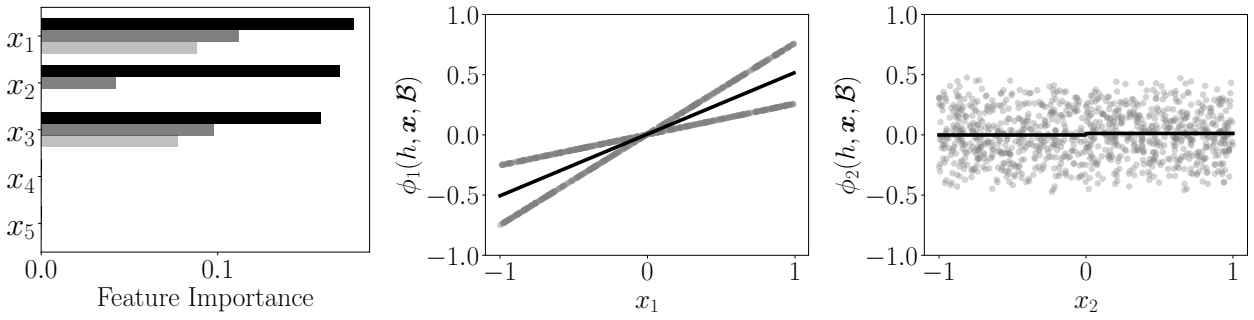
The premise of post-hoc additive explanations is to take any model prediction $h(\boldsymbol{x})$ and decompose it as a summation of d scores $\phi_j(h, \boldsymbol{x}, \mathcal{B})$, one per feature. Interpreting models that way is sensible when the output is an additive function of each individual feature. Nonetheless, models will generally involve feature interactions and so additive explanations cannot be 100% faithful. The reason being that models with interactions can be decomposed as summations $h(\boldsymbol{x}) = \sum_{u \in U} h_{u,\mathcal{B}}(\boldsymbol{x})$ where $|U| > d$, meaning that their inherent dimensionality is larger than d. Taking functions of dimension $D > d$ and summarizing them with only d feature attribution scores is guaranteed to distort their behavior.

(Analogy) *A cube is an inherently 3-dimensional object. Whenever a cube is drawn on a 2D sheet of paper, its geometry is distorted. For instance, not all of its angles can be 90 degrees. This is because the sheet of paper is lacking one dimension.*

The following toy example presents potential issues occurring when explaining a non-additive model using additive explanations. Consider the features $x_j \sim U(\text{-}1, 1)$ with $j = 1, 2, \ldots 5$ and the model

$$h(\boldsymbol{x}) = \begin{cases} x_1 & \text{if } x_2 \geq 0 \\ x_3 & \text{otherwise,} \end{cases} \tag{6.1}$$

which is locally additive but non globally additive. If we compute the PDP/SHAP/PFI global importance using the whole dataset as background $\mathcal{B}$, one observes strong disagreements regarding the importance of x_2, see Figure 6.1 (a). Indeed, PDP gives it no importance while PFI ranks it second. These two interpretations cannot simultaneously be faithful to the model. Additionally, we can compute the local PDP/SHAP feature attributions, see Figure 6.1 (b) & (c). Regarding the local attributions of feature x_1, PDP and SHAP both suggest a linear dependence with the output. These explanations are not entirely faithful to the model because, when x_2 is negative, the model output does not depend on x_1. Neither PDP nor SHAP illustrates this conditional dependence. Concerning the attribution of feature x_2, the PDP is flat suggesting no dependence, while the Shapley Values are all over the place.



(a) (b) (c)

Figure 6.1 Toy Example. (a) Global Feature Importance when using the whole dataset as reference. The PDP (transparent), SHAP (semi-transparent), and PFI (opaque) importance are differentiated via their opacity. (b)&(c) The PDP (line) and SHAP (dots) local feature attributions using the whole data as reference.

6.2 Disagreement Measure

In the above example, post-hoc additive methods were not faithful to h and they disagreed. Yet what is the relationship between explanation agreement and faithfulness? The key insight from our unifying theory of Section 3.2 is that post-hoc additive explanations agree **and** are faithful when the model happens to be additive. This suggests that aligning explainability methods (indirectly rendering them more faithful), requires minimizing the strength of feature interactions in h relative to the background $\mathcal{B}$ employed.

Definition 6.2.1 (Lack of Additivity)**.** *A function $L_h(\mathcal{B}) \in \mathbb{R}^+$ measures Lack of Additivity (LoA) of h w.r.t $\mathcal{B}$ if it respects the following properties.*

1. *If h is additive on a rectangular domain $R \supseteq supp(\mathcal{B})$, then $L_h(\mathcal{B}) = 0$.*
2. *There is a function $w : 2^{[d]} \to \mathbb{R}^+$ such that*

$$L_h(\mathcal{B}_{ind}) = \sum_{u \subseteq [d]: |u| \geq 2} w(u)\sigma_u^2. \tag{6.2}$$

3. *If h is additive in feature j, then*

$$L_h(\mathcal{B}_j \times \mathcal{B}_{-j}) = L_h(\mathcal{B}'_j \times \mathcal{B}_{-j}) \tag{6.3}$$

for any distributions $\mathcal{B}_j$ and $\mathcal{B}'_j$ on feature j and $\mathcal{B}_{-j}$ on features $[d] \setminus j$. Simply put, the LoA is not affected by additive features unless they correlate with interacting features.

Property 1 implies that LoA are sensible unfaithfulness metric for post-hoc additive explanations. Indeed, unlike Insertion and Deletion [Jethani et al., 2021, Petsiuk et al., 2018] (cf Proposition 2.4.1), the LoA metrics are always minimized whenever the model is additive over the support of $\mathcal{B}$. If the LoA is null, practitioners have the guarantee that post-hoc additive explainers will return $\phi_j(h^{\text{add}}, \boldsymbol{x}, \mathcal{B}) = h_j(x_j) - \mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[h_j(z_j)]$, which is a faithful description of h^{add}.

Property 2 is a sanity check that in the ideal scenario of independent features, the LoA penalizes the variances σ_u^2 which are well-established measures of interaction strength [Hooker, 2004, Owen, 2013]. Property 3 reduces the search space when minimizing the LoA w.r.t the background $\mathcal{B}$. Indeed, if only a subset $I \subset [d]$ of features interact, then we only need to minimize L_h w.r.t $\mathcal{B}_I$ and ignore other features.

Theorem 6.2.1. *Any function*

$$L_h(\mathcal{B}) = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\left[\sum_{\substack{u,v\subseteq[d]\\|u|\geq 2,|v|\geq 2}} a(u,v) h_{u,\mathcal{B}}(\boldsymbol{x}) h_{v,\mathcal{B}}(\boldsymbol{x})\right].$$

for some $a : 2^{[d]} \times 2^{[d]} \to \mathbb{R}$ *is a LoA.*

The proof is presented in Appendix C.1. As a result of this Theorem, many possible functions can quantify the LoA. A first possibility would be the L_2 Cost of Exclusion (CoE) [Hooker, 2004], which computes the error between the model and its additive decomposition

$$L_h^{\text{CoE}}(\mathcal{B}) := \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\left[\left(h(\boldsymbol{x}) - \sum_{\substack{u\subset[d]\\|u|\leq 1}} h_{u,\mathcal{B}}(\boldsymbol{x})\right)^2\right]. \tag{6.4}$$

Other possibilities would be any disagreement between local feature attributions. We let

$$\phi_j^{\text{PFI}}(h, \boldsymbol{x}, \mathcal{B}) = \sum_{u\subseteq[d]:j\in u} h_{u,\mathcal{B}}(\boldsymbol{x}) \tag{6.5}$$

be the local counterpart of the PFI global importance (cf. Equation 3.47), and also define the distances

$$D(\boldsymbol{\phi}, \boldsymbol{\phi}') := \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}[\|\boldsymbol{\phi}(h, \boldsymbol{x}, \mathcal{B}) - \boldsymbol{\phi}'(h, \boldsymbol{x}, \mathcal{B})\|_2^2] \tag{6.6}$$

$$D(\boldsymbol{\Phi}, \boldsymbol{\Phi}') := \|\boldsymbol{\Phi}(h, \mathcal{B}) - \boldsymbol{\Phi}'(h, \mathcal{B})\|_2^2 \tag{6.7}$$

between local feature attributions and global feature importance. Then the following holds.

Corollary 6.2.1. *The distances $D(\boldsymbol{\phi}, \boldsymbol{\phi}')$ between local PDP/SHAP/PFI feature attributions, as well as the CoE are all LoA functions.*

The proof is presented in Appendix C.1. Another LoA involves the ICE curves [Goldstein et al., 2015]

$$\phi_k^{\text{ICE}}(h, x_k, \boldsymbol{z}_{-k}) := h(x_k, \boldsymbol{z}_{-k}) \tag{6.8}$$

which can be centered with respect to x_k to obtain the mean-centered ICE curve

$$\phi_k^{\text{ICE-c}}(h, x_k, \boldsymbol{z}_{-k}) := h(x_k, \boldsymbol{z}_{-k}) - \mathop{\mathbb{E}}_{x_k \sim \mathcal{B}_k}\Big[h(x_k, \boldsymbol{z}_{-k})\Big]. \tag{6.9}$$

These curves can be visualized as a function of x_k to understand the effect of feature k when the other features are set to $\boldsymbol{z}_{-k}$. See Figure 6.2 for a toy example of how centered (and uncentered) ICE curves are typically visualized. Subsequently, by averaging the mean-centered ICE curves w.r.t $\boldsymbol{z}_{-k}$, we obtain the mean-centered PDP

$$\phi_k^{\text{PDP-c}}(h, x_k, \mathcal{B}) := \mathop{\mathbb{E}}_{\boldsymbol{z}_{-k} \sim \mathcal{B}_{-k}}\Big[\phi_k^{\text{ICE-c}}(h, x_k, \boldsymbol{z}_{-k})\Big]. \tag{6.10}$$

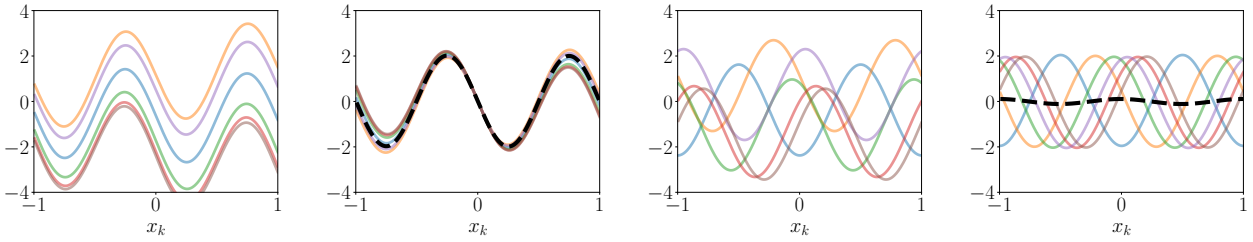
Like ICEs, the centered PDP is visualized as a function of x_k but it now represents the average effect of setting feature k to x_k. If there are no interactions in h involving feature k, then the ICE and PDP curves are parallel when plotted as functions of x_k. Consequently, the *centered* ICEs and PDP should perfectly agree. The following metric called GADGET-PDP [Herbinger et al., 2023] measures the average disagreement

$$\begin{aligned} L_h^{\text{GADGET-PDP}}(\mathcal{B}) &:= \sum_{k=1}^{d} \mathop{\mathbb{E}}_{\substack{x_k \sim \mathcal{B}_k \\ \boldsymbol{z}_{-k} \sim \mathcal{B}_{-k}}} \Big[\Big(\phi_k^{\text{ICE-c}}(h, x_k, \boldsymbol{z}_{-k}) - \phi_k^{\text{PDP-c}}(h, x_k, \mathcal{B})\Big)^2\Big] \\ &= \sum_{k=1}^{d} \mathop{\mathbb{E}}_{x_k \sim \mathcal{B}_k}\Big[\mathbb{V}_{\boldsymbol{z}_{-k} \sim \mathcal{B}_{-k}}\Big[\phi_k^{\text{ICE-c}}(h, x_k, \boldsymbol{z}_{-k})\Big]\Big]. \end{aligned} \tag{6.11}$$

The intuition behind this loss is illustrated in Figure 6.2.

Theorem 6.2.2. *GADGET-PDP is a LoA.*

The proof is presented in Appendix C.1. The fact that the GADGET-PDP loss is a LoA implies that it penalizes interactions and so it can potentially reduce the disagreements between PDP/SHAP/PFI explanations. This is an interesting result since GADGET-PDP was initially designed with PDP and ICE in mind.



(a) ICE curve (weak interactions)

(b) ICE-c curve (weak interactions)

(c) ICE curve (strong interactions)

(d) ICE-c curve (strong interactions)

Figure 6.2 Intuition behind GADGET-PDP. The colored lines are the (centered or uncentered) ICE curves for various values of $\boldsymbol{z}_{-k}$. The dashed dark line is the centered PDP. (a) There are weak interactions involving feature k so the ICE curves for various $\boldsymbol{z}_{-k}$ are nearly parallel. (b) After centering the ICE curves, the centered PDP is computed and is a good estimate of the centered ICEs. Thus the GADGET-PDP loss is very low. (c) There are strong interactions involving feature k and the ICE curves are not parallel. (b) After centering, the PDP is a poor estimate of the ICEs and the GADGET-PDP loss is large.

Interaction Disagreement

Post-hoc additive explanations agree (and are faithful) when the model happens to be additive. If the model contains interactions however, additive explanations may disagree and not faithfully describe its behavior. The impact of interactions on the disagreement between PDP/SHAP/PFI is what we call the *Interaction Disagreement*. Quantifying it requires reporting Lack of Additivity (LoA) metrics. Two such metrics have already been proposed in the literature.

- Hooker [2004] proposed the Cost of Exclusion (cf. Equation 6.4)
- Herbinger et al. [2023] defined the GADGET-PDP loss (cf. Equation 6.11).

In addition to these two pre-existing metrics, we advocate directly reporting the disagreements $D(\boldsymbol{\phi}, \boldsymbol{\phi}'), D(\boldsymbol{\Phi}, \boldsymbol{\Phi}')$ between PDP/SHAP/PFI as a LoA (cf. Corollary 6.2.1).

LoA metrics rely on a probability distribution $\mathcal{B}$ and so in practice they will need to be estimated. We present how to do so while borrowing notation from Chapters 4 and 5.

We are assuming that $\mathcal{B} = \mathcal{F}$, that N samples $\{\boldsymbol{x}^{(i)}\}_{i=1}^{N} \sim \mathcal{B}^N$ are available, and that the corresponding anchored components have been stored in the $N \times N$ matrices $\{\boldsymbol{H}^k\}_{k=1}^{d}$ (cf.

Equation 4.3). Given a subset of indices $S \subseteq [N]$, the empirical CoE is

$$\widehat{L}_h^{\text{CoE}}(S) = \frac{1}{|S|} \sum_{i \in S} \left(H_{ii}^{\text{add}} - \frac{1}{|S|} \sum_{j \in S} H_{ij}^{\text{add}} \right)^2, \tag{6.12}$$

where $H_{ij}^{\text{add}} := \sum_{k=1}^{d} H_{ij}^k + h(\boldsymbol{x}^{(j)})$. The disagreement between local PDP and PFI $D(\boldsymbol{\phi}^{\text{PDP}}, \boldsymbol{\phi}^{\text{PFI}})$ is also simple to compute

$$\widehat{L}_h^{\text{PDP-PFI}}(S) = \frac{1}{|S|} \sum_{k \in [d]} \sum_{i \in S} \left(\frac{1}{|S|} \sum_{j \in S} H_{ij}^k + H_{ji}^k \right)^2. \tag{6.13}$$

To compute the error $D(\boldsymbol{\phi}^{\text{PDP}}, \boldsymbol{\phi}^{\text{SHAP}})$ between the PDP and SHAP, one needs to precompute the $N \times N$ matrices $\{\boldsymbol{\Phi}^k\}_{k=1}^d$ storing Shapley Values (cf. Equation 5.35). Then,

$$\widehat{L}_h^{\text{PDP-SHAP}}(S) = \frac{1}{|S|} \sum_{k \in [d]} \sum_{i \in S} \left(\frac{1}{|S|} \sum_{j \in S} \Phi_{ij}^k - H_{ij}^k \right)^2. \tag{6.14}$$

Finally, to estimate the GADGET-PDP loss, we leverage the $\{\boldsymbol{R}^k\}_{k=1}^d$ matrices of size $N \times N$ with components

$$R_{ij}^k := h(x_k^{(i)}, \boldsymbol{x}_{-k}^{(j)}). \tag{6.15}$$

The corresponding empirical GADGET-PDP loss is

$$\widehat{L}_h^{\text{GADGET-PDP}}(S) = \frac{1}{|S|} \sum_{k=1}^{d} \sum_{i,j \in S} \left(R_{ij}^k - \frac{1}{|S|} \sum_{\ell \in S} R_{\ell j}^k - \frac{1}{|S|} \sum_{m \in S} R_{im}^k + \frac{1}{|S|^2} \sum_{\ell,m \in S} R_{\ell m}^k \right)^2. \tag{6.16}$$

The recurring theme behind these empirical estimates is that they require precomputing large $N \times N$ matrices, which might take some time. Still, such computations only need to be done once, and the results can be stored. Afterward, estimating the LoA for any subset $S \subset [N]$ of data only requires indexing the matrices and computing mean squared errors (cf. Equations 6.12, 6.13, 6.14, & 6.16).

6.3 Disagreement Reduction

Lack of Additivity $L_h(\mathcal{B})$ is a family of metrics that convey the unfaithfulness of post-hoc additive explanations. To compute more faithful explanations, we therefore wish to minimize the LoA. This can be done by fitting a new model h' with heavier constraints on the interactions it can represent *e.g.* training a tree ensemble with shallower trees. Still, to remain model-agnostic, we propose to leave h intact and instead learn regions $\Omega \subset \mathcal{X}$ that cover a

large subset of data ($\mathcal{B}(\Omega)$ is large) while also exhibiting fewer interactions ($L_h(\mathcal{B}_\Omega) \ll L_h(\mathcal{B})$). To understand why this might work, remember the informal definition of feature interaction (cf. Definition 2.3.1). It states that feature i interacts with feature j if and only if the effect of varying x_i on the response $h(\boldsymbol{x})$ depends on the fixed value of x_j. Intuitively, by restricting $\mathcal{B}$ to a region Ω, we condition the distribution on $x_j \in \Omega_j$. The constraint $x_j \in \Omega_j$ on feature x_j reduces its influence on feature x_i.

Rather than learning a single region, we advocate learning a partition of $\mathcal{X}$ into M regions $(\Omega^{[1]}, \Omega^{[2]}, \ldots, \Omega^{[M]})$ that minimizes the total LoA

$$\underset{(\Omega^{[1]},\Omega^{[2]},\ldots,\Omega^{[M]})}{\operatorname{argmin}} \sum_{k=1}^{M} |S \cap \Omega^{[k]}| \times \widehat{L}_h(S \cap \Omega^{[k]}). \tag{6.17}$$

Equation 6.17 is intractable, so solving it requires an approximation. We shall restrict the set of all partitions of $\mathcal{X}$ to the set of leaves of depth $\log_2(M)$ decision trees. These decision trees will be grown in a greedy fashion where at each internal node n, we will search for the feature $i_n \in [d]$ and threshold $\gamma_n \in \mathbb{R}$ such that splitting examples according to $\mathbb{1}(x_{i_n} \leq \gamma_n)$ minimizes the objective. We shall refer to such Decision Trees as *Functional Decomposition Trees* (FD-Trees).

Let's go back to the toy example of Equation 6.1 and see how FD-Trees can increase the faithfulness of additive explanations. Since the model is piece-wise linear, it makes sense to partition the input space into two disjoint regions $\Omega_- = \{\boldsymbol{x} \in \mathbb{R}^5 : x_j \leq \gamma\}$ and $\Omega_+ = \{\boldsymbol{x} \in \mathbb{R}^5 : x_j > \gamma\}$ and explain the model separately on each region. To automatically identify the regions, we select the feature x_j and threshold γ that minimize the CoE objective. According to Figure 6.3 (a), the optimal regions are $x_2 \leq 0.02$ and $x_2 > 0.02$. Then, running PDP/SHAP/PFI with backgrounds $\mathcal{B}_{\Omega_-}$ and $\mathcal{B}_{\Omega_+}$ leads to a strong agreement between the different techniques, see Figure 6.3 (b)&(c). Remember that disagreements between PDP/SHAP/PFI is a LoA metric: perfect agreement between the techniques guarantees explanation faithfulness.

How to reduce Disagreement?

To reduce the Interaction Disagreement, we propose partitioning the input space $\mathcal{X}$ into regions $(\Omega^{[1]}, \Omega^{[2]}, \ldots, \Omega^{[M]})$ with reduced feature interactions. Then, instead of running post-hoc explainers using the whole data distribution as reference $\mathcal{B}$, one can restrict the distribution $\mathcal{B}_\Omega$ to a region Ω of interest. These partitions are found automatically by a FD-Tree.

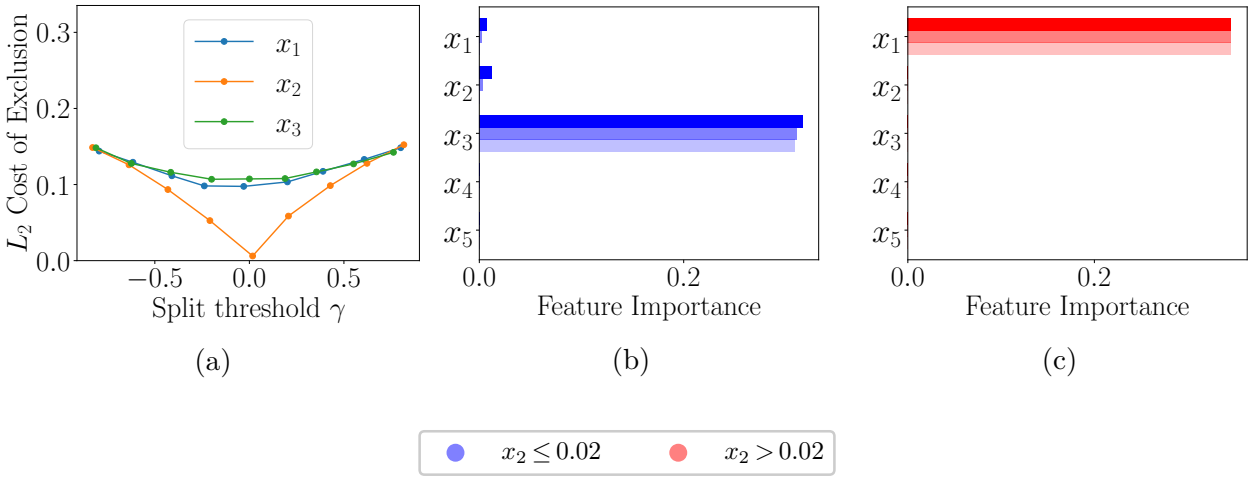


Figure 6.3 Toy Example Revisited. (a) The Cost of Exclusion is minimized by splitting the input space at $x_2 \leq 0.02$. (b) & (c) Global feature importance when the reference data is restricted to each region. The two regions are indicated by red/blue colors.

6.4 Experiments

We experimentally assessed the viability of FD-Trees on various datasets and models[1]. The UCI classification datasets Adults, Default-Credit, and Marketing were employed while, for regression tasks, the UCI dataset BikeSharing, Kin8nm, and the StatLib dataset California were investigated. The two types of models that were considered are Random Forests (RF) and Gradient Boosted Trees (GBT). The Scikit-Learn [Pedregosa et al., 2011] implementations of these models were used. For each dataset and model type, we trained a separate model for five different random seeds. Then, for each of the resulting 60 models, 9 FD-trees were fitted with maximum depth $1, 2, 3$ and objectives GADGET-PDP, CoE, and PDP-PFI. A total of $60 \times 9 = 540$ FD-Trees were obtained.

6.4.1 FDTree Training

Efficiently fitting FDTrees requires two importance choices 1) the features among which to split 2) the size of the $\{\boldsymbol{H}^k\}_{k=1}^d$ matrices.

Feature Split Candidates As the third property of Definition 6.2.1 suggests, a split along an additive feature j can only decrease the LoA if it is correlated with another feature that interacts. Since in those cases, j only acts as a proxy, we argue that splitting should only be

[1]The code to reproduce our experiments is available at `https://github.com/gablabc/UXAI_ANOVA`

done on features that interact. We propose quantifying pair-wise interaction strength using the Shapley-Taylor indices (cf. Equation 3.58) and visualizing them as a matrix heatmap.

Adults Figure 6.4 shows the interactions in the Adults dataset. We note that the strongest interactions involve the features `age`, `capital-gain`, `marital-status`, and `relationship`. Therefore, only these four features were split candidates.

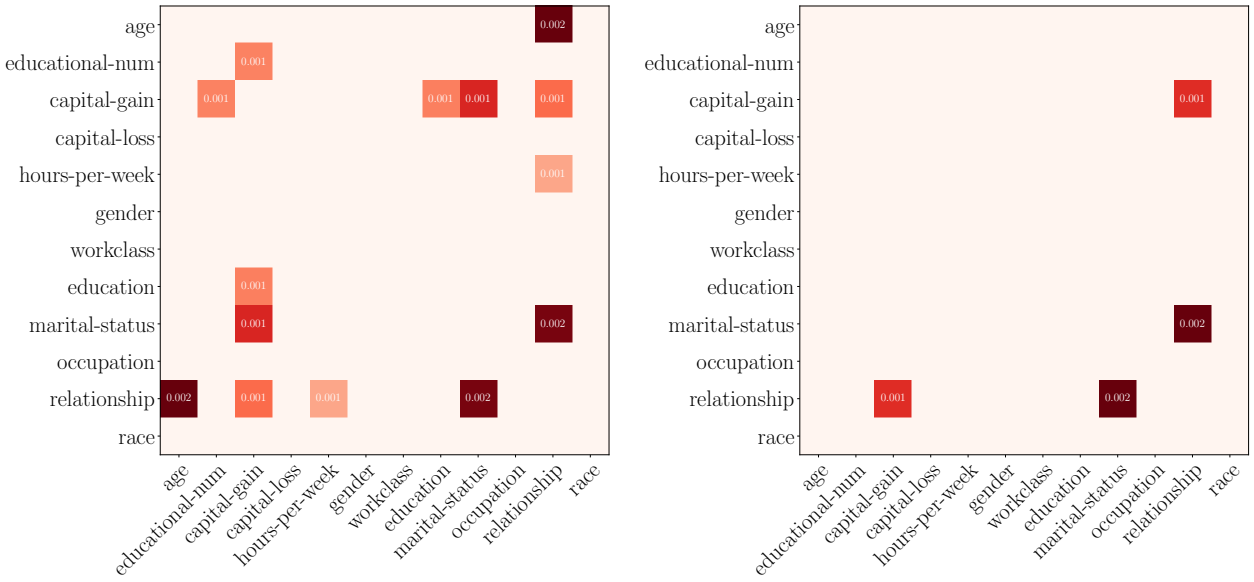
BikeSharing Figure 6.5 presents the interactions in the BikeSharing dataset. The dominating interactions are among the features `hr`, `workingday`, `year`, and `temp`. Hence, these four features were split upon by the FD-Trees.

Marketing Figure 6.6 highlights interactions in Marketing. The main interactions involve `month`, `day`, `contact`, `pdays`, which will be used by the FD-Trees.

Default-Credit Figure 6.7 presents the interactions in the Default-Credit dataset. Notable interactions involve `Delay-Sep`, `Delay-Aug`, `Bill-Sep`, and `Bill-Aug`. These features are the four split candidates in FD-Trees.

Kin8nm Figure 6.8 illustrates interactions in Kin8nm. All features interact with other features so they must all be considered when fitting FD-Trees.

California Figure 6.9 shows the interactions in the California Housing data. Strong interactions between `Lattitude` and `Longitude` are present. We also see weaker interactions between `Occupancy`, `MedInc`, and `HouseAge`. These five features were used by the FD-Trees to define the regions.

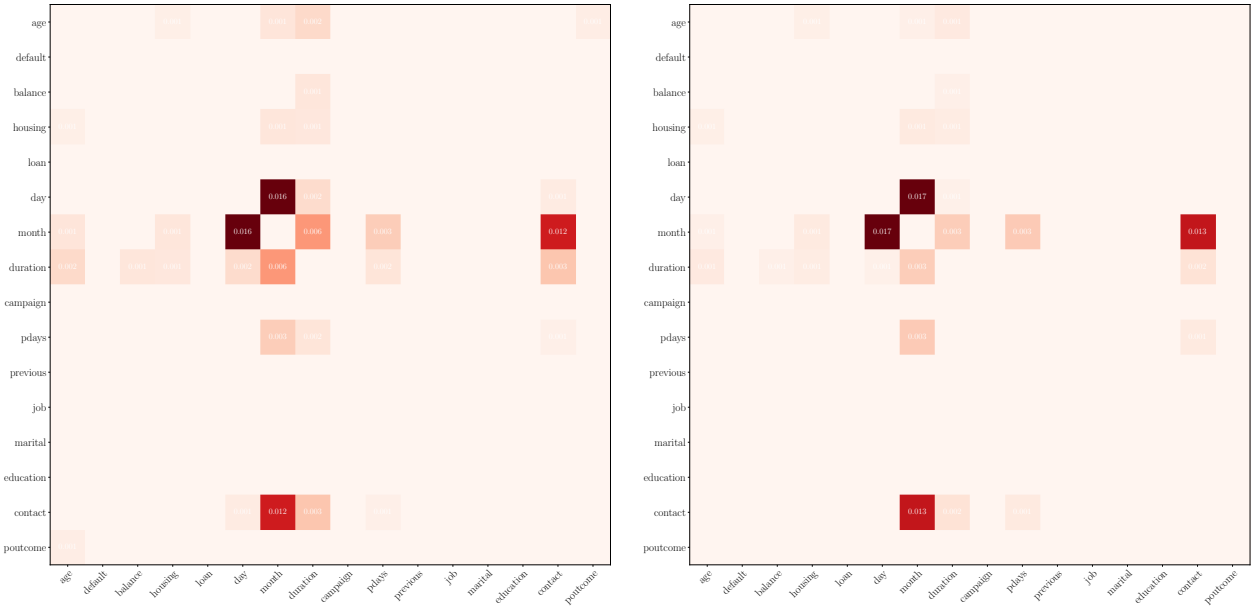


(a) RF (b) GBT

Figure 6.4 Interaction Indices on Adult.

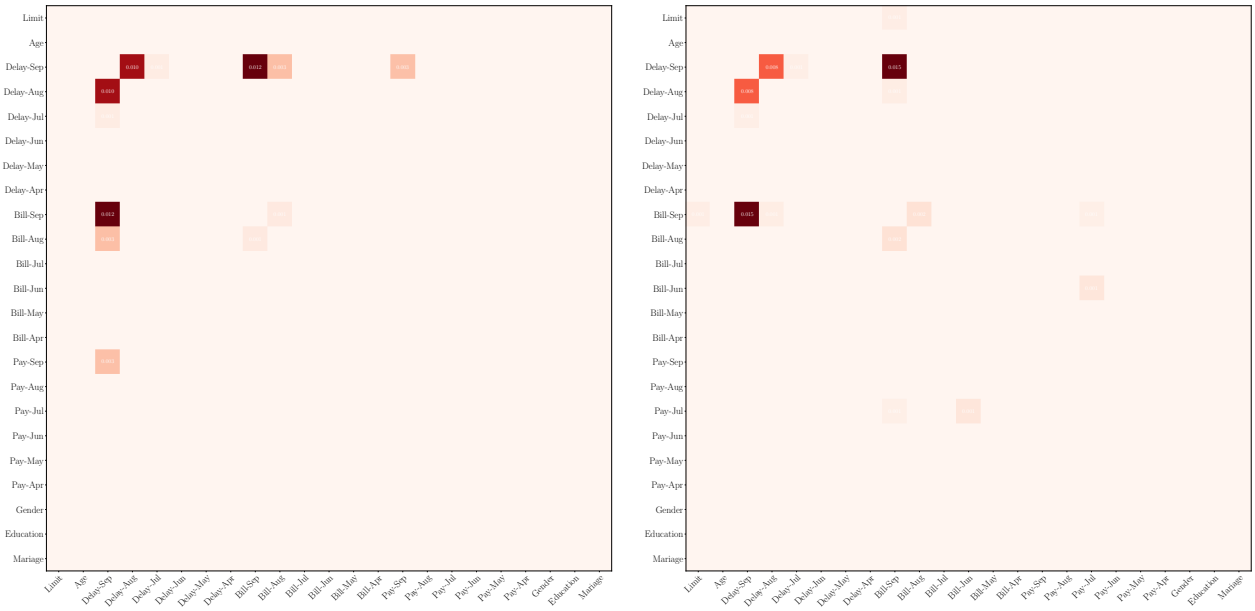
(a) RF (b) GBT

Figure 6.5 Interaction Indices on BikeSharing.



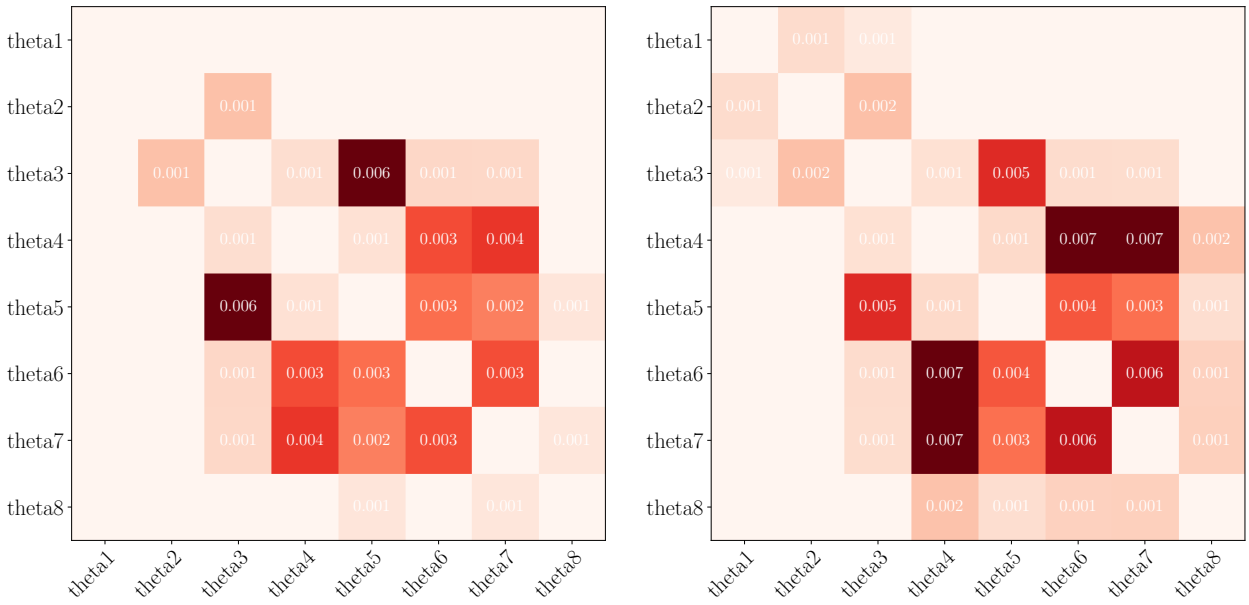
(a) RF (b) GBT

Figure 6.6 Interaction Indices on Marketing.



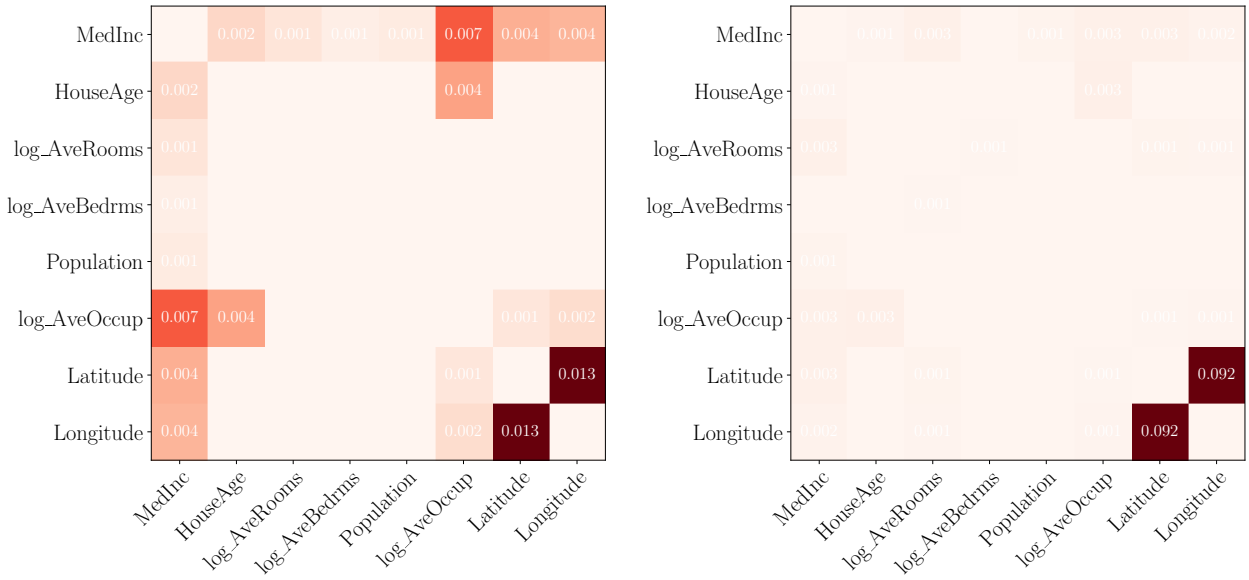
(a) RF (b) GBT

Figure 6.7 Interaction Indices on Default-Credit.



(a) RF (b) GBT

Figure 6.8 Interaction Indices on Kin8nm.



(a) RF (b) GBT

Figure 6.9 Interaction Indices on California.

Subsample Size FD-Trees additionally require precomputing the matrices $\{\boldsymbol{H}^k\}_{k=1}^d$ of size $N \times N$. Since we employed tree-based models, these matrices were easily computable from Algorithm 4. Yet, because N is considerably large for the datasets involved, it was necessary to subsample $M \ll N$ datapoints and utilize $M \times M$ submatrices instead. However, this subsampling introduced stochasticity in the training of FD-Trees because different subsamples led to different optimal trees. Ideally, M should be large enough to lead to stable partitions $(\Omega^{[1]}, \Omega^{[2]}, \ldots, \Omega^{[M]})$ on multiple reruns. Yet, it should also not be too large to avoid the $\mathcal{O}(N^2)$ time and space complexities

To assess the stability of partitions, we repeat the following methodology 10 times: 1) subsample M data points, 2) compute the $M \times M$ sub-matrices $\{\boldsymbol{H}^k\}_{k=1}^d$, 3) grow a full FD-Tree and return its partition. The Rand Index [Hubert and Arabie, 1985] can then be used to measure the *similarity* between any pairs of partitions resulting from these repeated reruns. The Rand Index considers all pairings of data points $\boldsymbol{x}^{(i)}$ and $\boldsymbol{x}^{(j)}$. The two partitions are said to agree on the pair if either 1) $\boldsymbol{x}^{(i)}$ and $\boldsymbol{x}^{(j)}$ are in the same group for both partitions, 2) $\boldsymbol{x}^{(i)}$ and $\boldsymbol{x}^{(j)}$ are in separate groups for both partitions. The Rand Index is then the ratio of agreeing pairs to the total number of pairs. However, it is best to employ the Adjusted Rand Index [Hubert and Arabie, 1985] which normalizes the metric to account for chance. Thus, random partitions will have an index close to zero.

In Figure 6.10, we present the Adjusted Rand Index as a function of subsample size. A general trend is that partitions stabilize the more and more samples are used to train FD-Trees. Importantly, the partitions of depth-1 FD-trees are extremely stable on Adults and BikeSharing. For the other datasets, it takes multiple samples before the first split stabilizes. To keep the experimental setup simple, we advocate subsampling $M = 600$ datapoints, which is reasonable for most datasets.

6.4.2 Quantitative Results

Disagreement between the PDP/SHAP/PFI explainers are indicators of explanation unfaithfulness. Accordingly, we investigated whether FD-Trees could significantly reduce said disagreements. Before addressing this question, note that the disagreement metrics $D(\boldsymbol{\phi}, \boldsymbol{\phi}')$ and $D(\boldsymbol{\Phi}, \boldsymbol{\Phi}')$ are scale-sensitive: for any $\epsilon \in]0, 1[$ we have $D(\epsilon\boldsymbol{\phi}, \epsilon\boldsymbol{\phi}') < D(\boldsymbol{\phi}, \boldsymbol{\phi}')$ and $D(\epsilon\boldsymbol{\Phi}, \epsilon\boldsymbol{\Phi}') < D(\boldsymbol{\Phi}, \boldsymbol{\Phi}')$. This introduces a bias since reducing the explanation norm can reduce their disagreements. Remember that PDP/SHAP/PFI all describe the model predictions $h(\boldsymbol{x})$ relative to the mean $\mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}_\Omega}[h(\boldsymbol{z})]$. So, if the background $\mathcal{B}_\Omega$ is very local, the model may deviate less from its average and explanations will naturally be smaller. Consequently, even a random tree can identify regions with reduced disagreements, a fact that we consis-

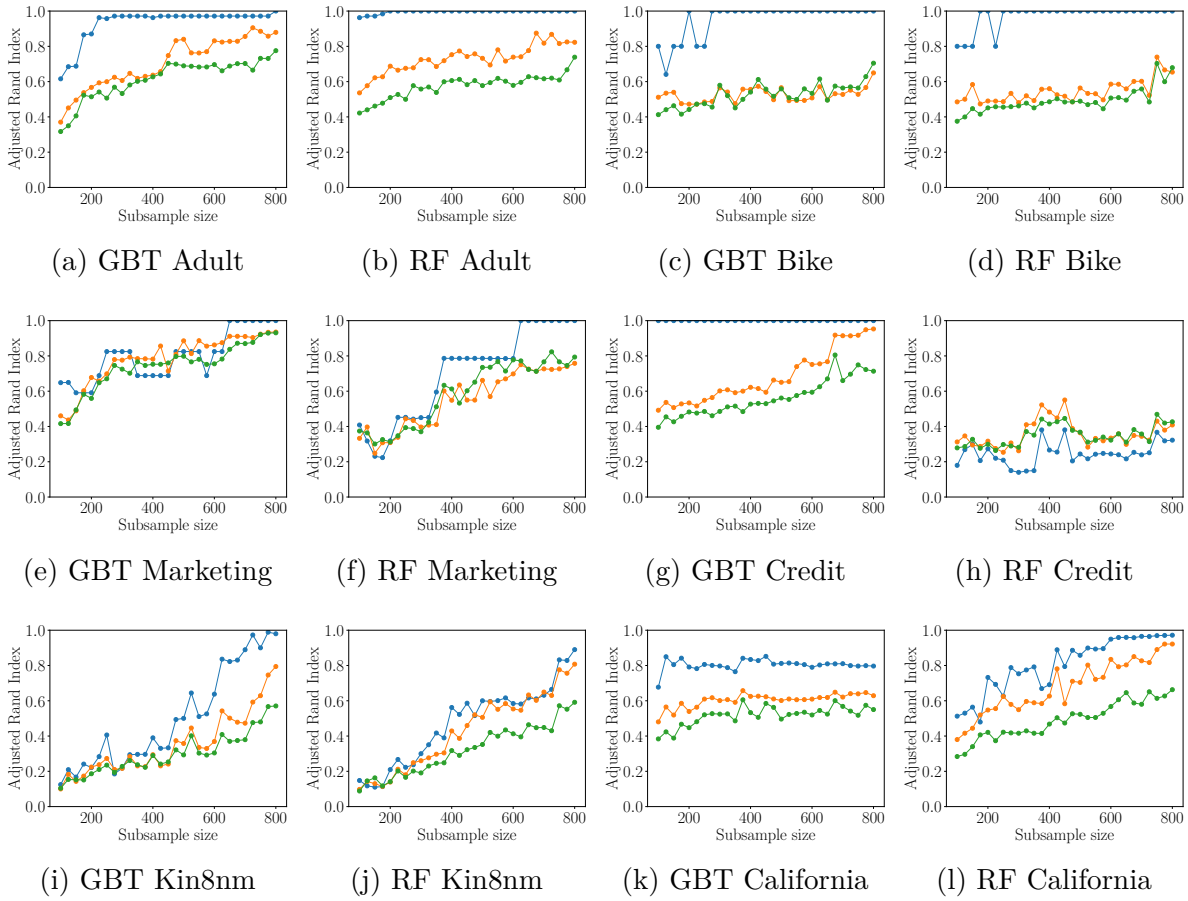


Figure 6.10 Stability of the Partitions given by FD-Trees as a function of the subsample size. The colors blue, orange, and green refer to FD-Trees of depth 1, 2, and 3 respectivelly.

tently observed empirically. A basic sanity check for FD-Trees is to compare the reduction in explanation disagreements to those induced by random trees. A stronger sanity check is comparing FD-Trees to regions yielded by a Classification And Regression Tree (CART) fitted on the model output. CART minimizes the deviation $\mathbb{E}_{\boldsymbol{x}\sim\mathcal{B}_\Omega}[\,(\,h(\boldsymbol{x}) - \mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}_\Omega}[h(\boldsymbol{z})]\,)^2\,]$ at each leaf Ω, and so directly minimizes the norms of the explanations.

Figure 6.11 (a)&(b) compares the baselines (Random, CART) to various FD-Trees (GADGET-PDP, CoE, PDP-PFI). Each box-plot represents the distribution of explanation disagreements across model types, random seeds, and datasets. On the left, the local disagreements are between PDP and SHAP : $D(\boldsymbol{\phi}^{\text{PDP}}, \boldsymbol{\phi}^{\text{SHAP}})$. On the right, the global disagreements $D(\boldsymbol{\Phi}, \boldsymbol{\Phi}')$ are averaged over the three pairings PDP-SHAP, PDP-PFI, and SHAP-PFI. FD-Trees reduce local/global explanation disagreements more than both baselines. These improvements are statistically significant according to paired Student-t tests.

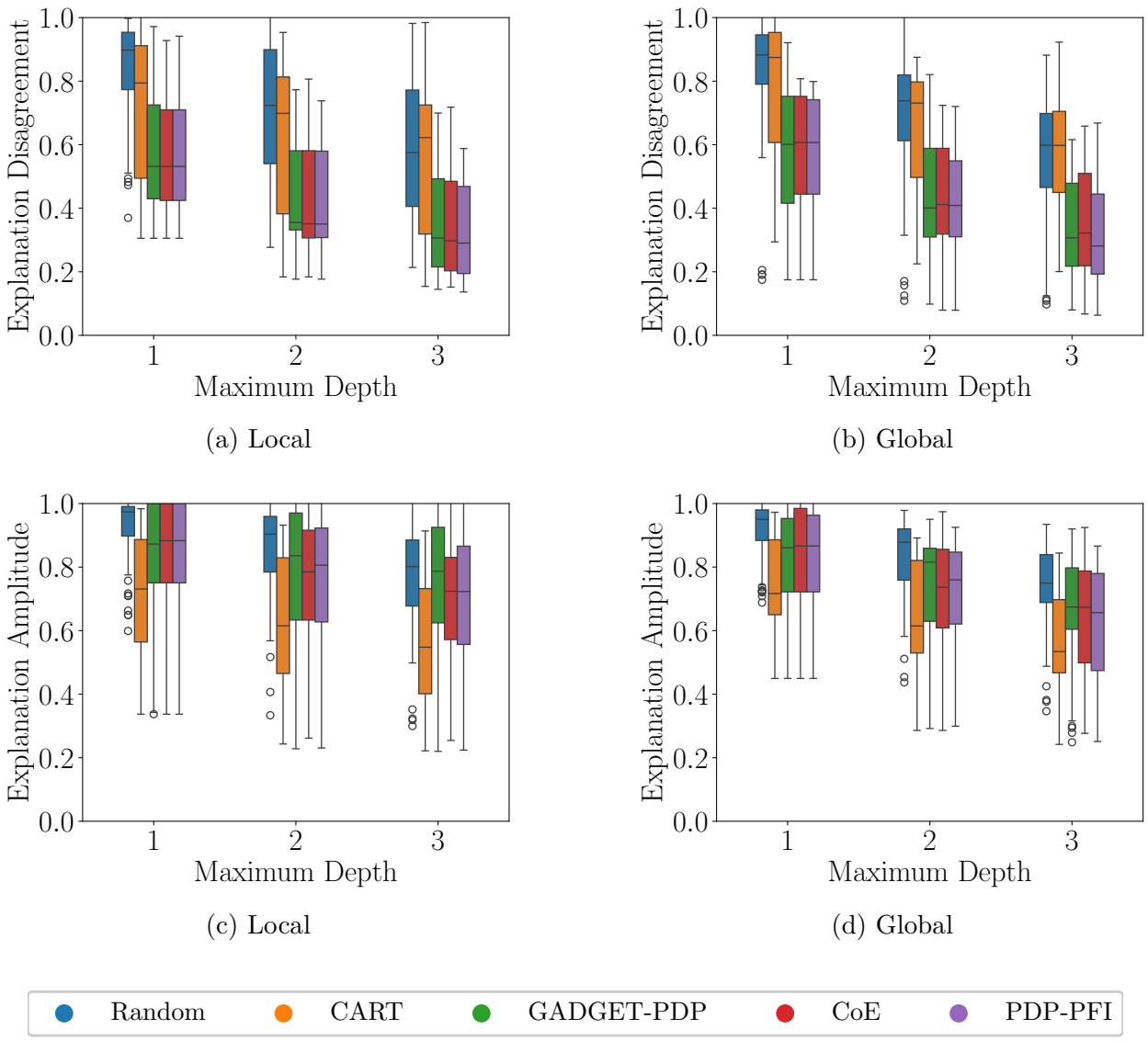


Figure 6.11 Explanation disagreements and amplitudes for two baselines (Random, CART) and FD-Trees (GADGET-PDP, CoE, PDP-PFI) of depth 1, 2, and 3. Left column are local feature attributions while right column is global feature importance. The disagreements/amplitudes were normalized w.r.t disagreements/amplitudes obtained when the whole data is considered as the background.

Still, is there more agreement simply because explanations are smaller? Looking at the explanation amplitudes in Figure 6.11(c)&(d), CART is by far the method that leads to the smallest explanation amplitudes, locally and globally. Yet, CART did not manage to reduce explanation disagreements as much as FD-Trees could. This demonstrates that increasing alignment between post-hoc explainers is more complicated than simply making explanations smaller, and that feature interactions are a key quantity to minimize.

Table 6.1 P-values of the Repeated-Measure-ANOVA tests comparing the explanation disagreements between the GADGET-PDP, CoE, and PDP-PFI objectives. For each p-value lower than 0.05, we also show the objective leading to the least disagreements : (1) GADGET-PDP (2) CoE (3) PDP-PFI.

Locality	Default-Credit	Adult	Marketing	Kin8nm	BikeSharing	California
Local	**0.003 (3)**	0.31	**0.015 (3)**	0.42	0.06	**0 (3)**
Global	**0.01 (3)**	**0.03 (3)**	0.25	**0 (3)**	**0.001 (3)**	0.7

Given that FD-Trees (GADGET-PDP, CoE, PDP-PFI) can identify useful regions, we compared them more thoroughly via Repeated-Measure-ANOVA tests, see Table 6.1. Repeated-Measure-ANOVA aims to identify if there are significant differences in "outcome" for various "treatments" applied to recurring "subjects". In our setting, the "outcome" is the explanation disagreement, the "subjects" are the 30 combinations of model type, random seed, and depth of the FD-Tree, while the "treatment" is the objective employed when growing the tree. According to Table 6.1, there are often no significant differences between GADGET-PDP, CoE, and PDP-PFI. When the differences are significant, it is systematically the PDP-PFI objective that leads to the least disagreements.

6.4.3 Qualitative Results

Adult-Income The Adult-Income task is to predict if someone makes more than 50k USD based on demographic attributes. The first step of the analysis was to compute post-hoc explanations using the whole dataset as the background distribution. According to Figure 8.13 (a)&(c), the resulting PDP local attributions are a poor estimate of the SHAP values. This warns us that strong feature interactions make the local attributions of `age` and `educational-num` unreliable. To reduce disagreements, we studied the regional explanations over the leaves of a FD-Tree. All FD-Trees that were trained on RFs and Adult-Income identified the same first split: separating married from unmarried people. Figure 8.13 (b)&(d) shows the corresponding regional explanations. Our first observation is that errors between PDP and SHAP suddenly decrease. Secondly, the model has very distinctive behaviors between married and unmarried people. From Figure 8.13 (b), the attribution of `age` tends to be negative for younger people, but to a larger extent if you are married. Also, the attribution of `age` becomes negative for older married people, while it remains positive for older unmarried ones. Similar observations can be made for `education-num`.

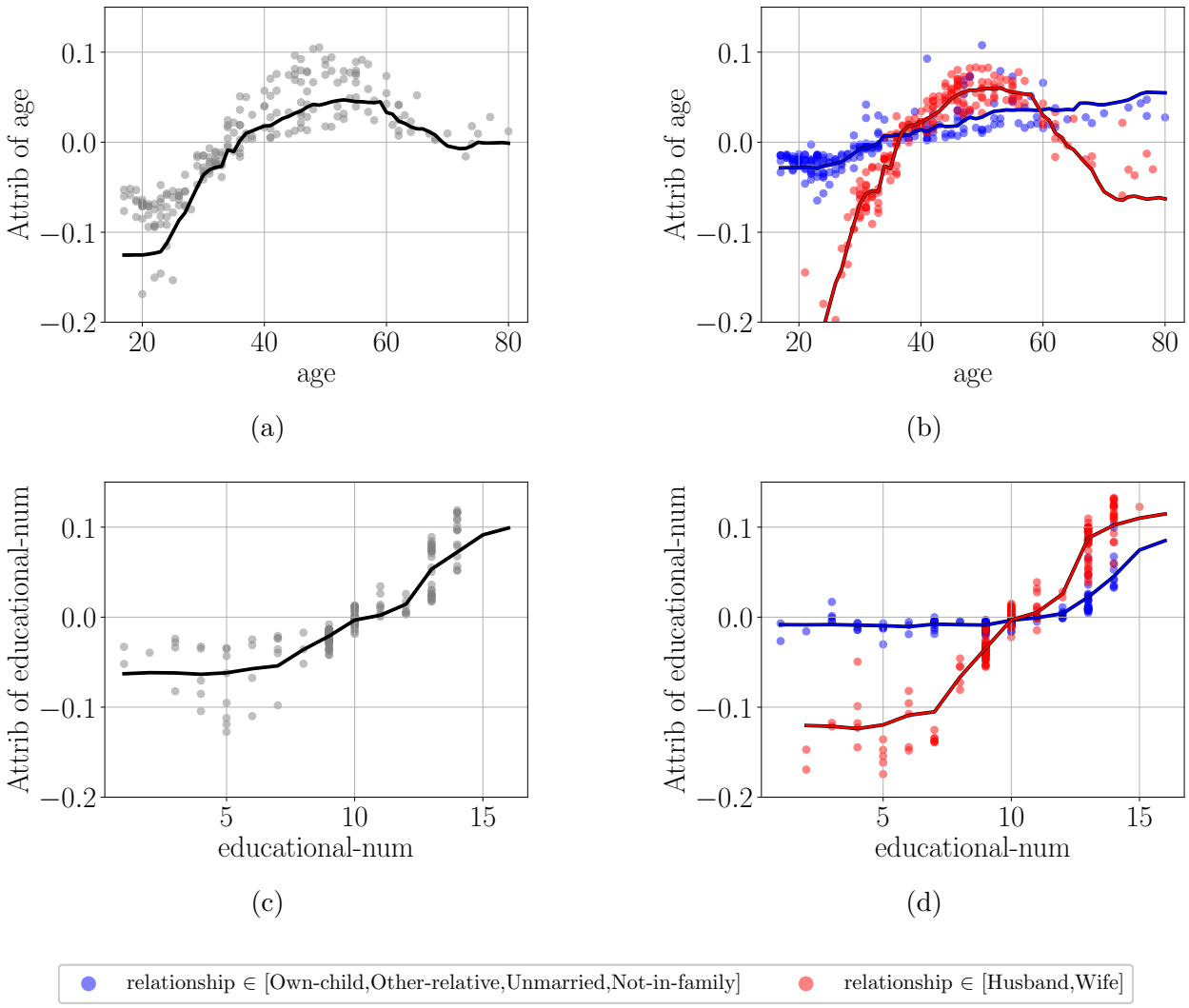


Figure 6.12 Adult Income. Lines are PDPs while points are SHAP values. (a)&(c) represent the SHAP and PDP explanations when the background is set to the whole dataset. (b)&(d) plot regional explanations with backgrounds restricted to the two regions indicated in red/blue colors.

California The California dataset available on the Statlib Repository[2] consists in predicting the median house value in a California block from 1990. The input features involve demographic characteristics aggregated over each block as well as the `longitude` and `lattitude` of the respective blocks. We present the results of fitting a GBT on this dataset.

As highlighted in Figure 6.9, the strongest interactions involve features `longitude` and `lattitude`. In Figure 6.13 (a), we observe stronger disagreements regarding the global importance of these two features. Additionally, according to Figure 6.13 (b)&(c), the PDP

[2] `http://lib.stat.cmu.edu/datasets/`

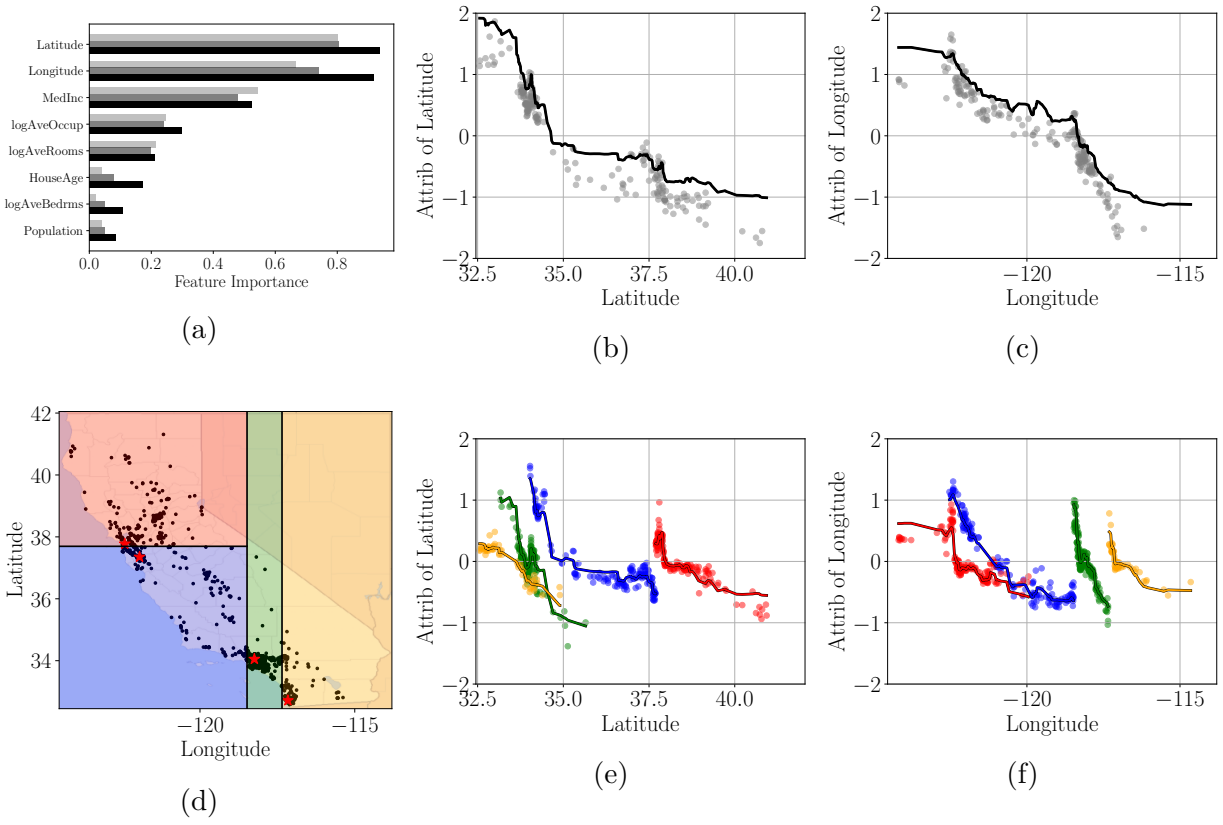


Figure 6.13 California. The top row shows the global (a) and local (b)&(c) explanations when the background is set to the whole data distribution. Lines are the local PDP while points are the local SHAP values. (d) The state of California is split by a FD-Tree into four regions shown in color. The major cities of Los Angeles, San Francisco, San Diego, and San Jose are shown as red stars. (e)&(f) The local PDP/SHAP explanations extracted from these four regions.

local explanation is a poor estimate of the SHAP values. Based on all of these observations, it is difficult to explain the effect of each separate coordinate on the model. The impact of varying `longitude` depends on `lattitude` and vice versa. Nonetheless, we hypothesize that FD-Trees can be used to split up California into regions where the role of both coordinates is more additive.

We fitted a depth-2 FD-Tree with the PDP-PFI objective because it dominates the other objectives (Table 6.1). All the splits conducted by this tree were applied to the `longitude` and `lattitude` features. Consequently, the tree leaves can be visualized over a map of California, see Figure 6.13 (d). Figure 6.13 (e)&(f) presents the PDP and SHAP explanations whose background distribution is restricted to a single leaf. The disagreements between the two explainers are greatly reduced as a result. As a final note, the large cities of Los Angeles,

San Francisco, San Diego, and San Jose are shown on the California map as red stars. Interestingly, each city belongs to a separate FD-Tree leaf. This result is not a coincidence, since these splits were consistent across various FD-Trees trained on GBTs with the CoE and PDP-PFI objectives.

6.5 Discussion

Providing regional explanations induces a higher cognitive load on users because they must understand the regions description, as well as the model behavior on each separate region. For example, instead of providing a single ranking of global feature importance, one such ranking must be reported for each FD-Tree leaf. We view this as a necessary price to pay in order to gain faithful insight into model behavior. Nevertheless, practitioners can still use the whole dataset as background as long as the explanations of various techniques are shown in tandem to reveal potential interactions.

FD-Trees are currently grown in a greedy fashion : we split each node along a feature that is locally optimal, and we never consider future impacts of a split, nor do we backtrack on any previous choice. Greedy strategies are common in tree induction because of the considerable search space [Louppe, 2014]. Even so, investing more time to find a better solution (*e.g.* via look-ahead strategies [Esmeir and Markovitch, 2007]) may prove beneficial for FD-Trees.

Contributions

Understanding that disagreements between PDP/SHAP/PFI are induced by feature interactions, we proposed a general family of interaction quantifiers (*i.e.* Lack of Additivity (LoA) functions). These LoA functions are minimized by partitioning the input space into regions using a FD-Tree (Functional Decomposition Tree). Restricting post-hoc explainers to the leaves of a FD-Tree was shown to increase PDP/SHAP/PFI alignment more than alternative partitions. Finally, on a more qualitative note, we demonstrated that FD-Trees provide granular information about model behavior. For instance, we saw that the effect of age/education on high-income predictions depends on whether an individual is married or not.

CHAPTER 7 SUBSAMPLING DISAGREEMENT

7.1 Motivation

Additive Explanations (either ante-hoc or post-hoc) are always relative to the average prediction $\mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{z})]$ over a background distribution $\mathcal{B}$. In fact, Section 2.3 highlighted how the many additive explanations proposed in the literature all involve expectations $\mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}$ in their definition. If the distribution is a Dirac measure $\delta_{\boldsymbol{z}}$ centered at a single point $\boldsymbol{z}$ (*i.e.* if we want to compare the model predictions at $\boldsymbol{x}$ and $\boldsymbol{z}$) these expectations are tractable.

However, in XAI is it common to set $\mathcal{B}$ to the empirical distribution over the full dataset

$$\mathcal{B} := \frac{1}{N}\sum_{i=1}^{N}\delta_{\boldsymbol{x}^{(i)}}, \tag{7.1}$$

or a regional restriction $\mathcal{B}_\Omega$ thereof. In either case, computing the additive explanations scales linearly with N which can be prohibitive for ML use-cases that typically involve 10K-100K data points. To avoid $\mathcal{O}(N)$ complexity, one must estimate the expectations $\mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}$ by subsampling $M \ll N$ background points $\{\boldsymbol{z}^{(j)}\}_{j=1}^{M} \sim \mathcal{B}^M$ and computing the average

$$\frac{1}{M}\sum_{j=1}^{M} f(\boldsymbol{z}^{(j)}) \xrightarrow{p} \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[f(\boldsymbol{z})]. \tag{7.2}$$

Subsampling background instances requires choosing M points uniformly at random from the dataset. For example, the `SHAP` library will subsample $M = 100$ instances uniformly at random from the dataset if it is too large[1]. Subsampling the dataset introduces possible disagreement additive explanations across reruns of the same code.

7.1.1 Disagreement Measure

Subsampling Disagreement

Additive Explanations require estimating expectations $\mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}$ by subsampling M instances $\{\boldsymbol{z}^{(j)}\}_{j=1}^{M} \sim \mathcal{B}^M$ from the data and averaging the results. This introduces potential disagreement between reruns since running the same code twice can lead to two different subsamples $\{\boldsymbol{z}^{(j)}\}_{j=1}^{M} \sim \mathcal{B}^M$ and $\{\boldsymbol{z}'^{(j)}\}_{j=1}^{M} \sim \mathcal{B}^M$ with different additive explanations.

[1] https://github.com/slundberg/shap/blob/0662f4e9e6be38e658120079904899cccda59ff8/shap/maskers/_tabular.py#L54-L55

How to reduce Disagreement?

Fortunately, quantifying the error of between $\mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[f(\boldsymbol{z})]$ and $\frac{1}{M}\sum_{j=1}^{M} f(\boldsymbol{z}^{(j)})$ is at the heart of Statistical Theory and Monte-Carlo methods [Owen, 2013]. These work advocate reporting Confidence Intervals (CIs) around the estimate $\widehat{\phi}$ to capture the ground-truth ϕ with probability $1-\delta$. Examples of Confidence Intervals applicable to additive explanations include Theorems 4.1.1 & 5.4.3 that leverage the Central Limit Theorem. Other examples are Theorems 4.3.1 & 5.4.4 based on the asymptotic normality of U-statistics [Lee, 2019].

The width of the CIs typically decreases with $M^{\frac{1}{2}}$ meaning that having 100 times more samples lead to an improvement of factor 10 on the Subsampling Disagreement. Thus, the recommendation to control these disagreements is to increase the number of subsamples M until the CIs widths become small enough.

Confidence Intervals are suitable for diagnosing and controlling the Subsampling Disagreements. Still, they all rely on the null hypothesis $H_0 : \{\boldsymbol{z}^{(j)}\}_{j=1}^{M} \sim \mathcal{B}^M$ which assumes that each datum is equally likely to be picked in the subsample. This null hypothesis excludes the possibility of cherry-picking a specific datum to support a preconceived narrative about the model.

Assuming you do not trust the function `sklearn.utils.resample`[2] used by `SHAP` to subsample your data, how would you verify that the null hypothesis H_0 is true? Well, given that you have access to the full dataset, you can plot histograms of each feature x_j and compare the full dataset to the subsample. Or you can compare certain statistics such as the feature's mean/median/quantiles across the full and subsampled data. If differences are significant according to a statistical test, then you can claim that the subsamples send to `SHAP` are not representative and reject the estimate and its CIs.

But, what would you do if you did not have access to the full data? What if the dataset was too big. What if the dataset was owned by someone else and they provided you the subsample? How then would you be sure that H_0 holds and that Confidence Intervals characterize well the subsampling disagreements? These questions are at the heart of this Chapter.

We now present a real-world scenario where assuming that the null hypothesis H_0 holds can have dire consequences.

[2] `https://scikit-learn.org/stable/modules/generated/sklearn.utils.resample.html`

7.1.2 Audit Scenario

A company has a private dataset $D = \{(\boldsymbol{x}^{(i)}, y^{(i)})\}_{i=1}^{N}$ and a proprietary model $h : \mathcal{X} \to [0, 1]$ that is meant to be deployed in society. The binary feature with index s (*i.e.* $x_s \in \{0, 1\}$) represents a sensitive feature with respect to which the model should not explicitly discriminate (*e.g.* gender, religion etc.). Both data D and model h are highly private, so the company is very careful when providing information about them to anyone.

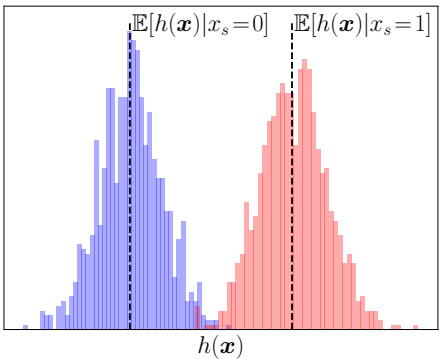
Before deploying the model, the company must go through an auditing process where an external auditor characterizes the societal impacts of the model. The model h is a black box from the point of view of the auditor and they are only able to call it through an API provided by the company. At first, the auditor asks the company for the necessary data to compute fairness metrics *e.g.* the Demographic Parity [Dwork et al., 2012], the Predictive Equality [Corbett-Davies et al., 2017], or the Equal Opportunity [Hardt et al., 2016]. For simplicity, the auditor decides to simply compute the Demographic Parity

$$\mathbb{E}[h(\boldsymbol{x})|x_s = 0] - \mathbb{E}[h(\boldsymbol{x})|x_s = 1], \tag{7.3}$$

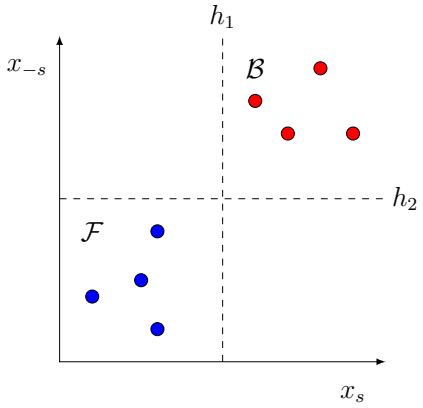
and therefore demands access to the model outputs for all inputs with different values of the sensitive feature : $h(D_0)$ and $h(D_1)$, where $D_0 = \{\boldsymbol{x}^{(i)} : x_s^{(i)} = 0\}$ and $D_1 = \{\boldsymbol{x}^{(i)} : x_s^{(i)} = 1\}$ are subsets of the input data of sizes N_0 and N_1 respectively. Doing so does not force the company to share features values in the data nor does it require direct access to the architecture of the proprietary model. Hence, this demand respects privacy requirements and the company will accept to share the model outputs across all instances, see Figure 7.1 (a).

At this point, the audit confirms the model is biased in favor of $x_s = 1$ and puts in question its deployment. Now, the company argues that, although the model exhibits a disparity in outcomes, it does not mean that the model explicitly uses the feature x_s to make its decision. If such is the case, then the disparity could be explained by other features statistically associated with x_s. Some of these other features may be acceptable grounds for decisions. To verify such a claim, the auditor decides to employ post-hoc techniques to explain the disparity. Since the model is a black-box, the auditor shall compute the Fair-SHAP attributions (cf. Equation 2.92), where the foreground $\mathcal{F}$ and background $\mathcal{B}$ are the data conditioned on $x_s = 0$ and $x_s = 1$ respectively

$$\mathcal{F} := \frac{1}{N_0} \sum_{\boldsymbol{x}^{(i)} \in D_0} \delta_{\boldsymbol{x}^{(i)}} \qquad \mathcal{B} := \frac{1}{N_1} \sum_{\boldsymbol{z}^{(j)} \in D_1} \delta_{\boldsymbol{z}^{(j)}}. \tag{7.4}$$



(a) The data initially provided to the audit is $h(D_0)$ and $h(D_1)$ *i.e.* the model predictions for all instances in the private dataset for different values of x_s. This dataset can later be used by the audit to assess whether the subsets S_0', S_1' provided by the company were cherry-picked.



(b) Models h_1 and h_2 (decision boundaries in dashed lines) with perfect accuracy exhibit a disparity in outcomes w.r.t $x_s < 0$ and $x_s > 0$. Here, $\Phi_s^{\text{Fair}}(h_1, \mathcal{F}, \mathcal{B}) = -1$ while $\Phi_s^{\text{Fair}}(h_2, \mathcal{F}, \mathcal{B}) = 0$ so h_2 is **indirectly** unfair toward x_s because of correlations in the data.

Figure 7.1 Illustrations of the audit scenario.

By definition of Fair-SHAP, the resulting attributions $\boldsymbol{\Phi}^{\text{Fair}}(h, \mathcal{F}, \mathcal{B})$ will sum up to the demographic parity. If the sensitive feature has a large negative $\Phi_s^{\text{Fair}}(h, \mathcal{F}, \mathcal{B})$, then this would mean that the model is **explicitly** relying on x_s to make its decisions and the company would be forbidden from deploying it. If the Fair-SHAP attribution has a small amplitude, however, the company could still argue in favor of deploying the model in spite of having disparate outcomes. Indeed, the difference in model outcomes could be attributed to more acceptable features. See Figure 7.1 (b) for an illustration of this reasoning.

To compute the exact Fair-SHAP attributions, the auditor would need access to the full datasets D_0 and D_1. Still, because of privacy concerns on sharing the data, and because exact Fair-SHAP values are too costly, both parties agree that the company shall only provide subsets $S_0 \subset D_0$ and $S_1 \subset D_1$ of size M to the auditor so they can compute a Monte-Carlo estimate

$$\widehat{\boldsymbol{\Phi}}^{\text{Fair}}(h, S_0, S_1) = \frac{1}{M^2} \sum_{\boldsymbol{x}^{(i)} \in S_0} \sum_{\boldsymbol{z}^{(j)} \in S_1} \boldsymbol{\phi}^{\text{SHAP-int}}(h, \boldsymbol{x}^{(i)}, \boldsymbol{z}^{(j)}). \tag{7.5}$$

According to Theorem 5.4.4, $\widehat{\boldsymbol{\Phi}}^{\text{Fair}}(h, S_0, S_1)$ is a consistent estimate of the Fair-SHAP score $\boldsymbol{\Phi}^{\text{Fair}}(h, \mathcal{F}, \mathcal{B})$ if the subsets are sampled uniformly at random *i.e.* $S_0 \sim \mathcal{F}^M$ and $S_1 \sim \mathcal{B}^M$.

Before sharing subsamples, the company first estimates Fair-SHAP on its own by sampling $S_0 \sim \mathcal{F}^M$ and $S_1 \sim \mathcal{B}^M$ uniformly at random. They observe that $\widehat{\Phi}_s^{\text{Fair}}$ indeed has a large negative value so they must carefully select which data points will be sent, otherwise, the

auditor may observe the direct bias toward $x_s = 1$ and the model will not be deployed. Moreover, the company understands that the auditor currently has access to the data $h(D_0)$ and $h(D_1)$ representing model predictions on the whole dataset (see Figure 7.1 (a)). Therefore, if the company does not share subsets S_0, S_1 that were select uniformly at random from D_0, D_1, it is possible for the audit to detect this fraud by doing a statistical test comparing $h(S_0)$ to $h(D_0)$ and $h(S_1)$ to $h(D_1)$. The company wants a method to select **misleading subsets** S_0', S_1' whose Fair-SHAP values are manipulated in their favor while remaining undetected by the auditor. Such a method is illustrated on a toy example before being described in details.

7.1.3 Toy Example

Let's imagine the company trains a model with the aim for pre-screening applicants of a job that requires carrying heavy objects. They fit a classification model using historical data of individuals who succeeded ($Y = 1$) and failed ($Y = 0$) the job in the past. The causal graph behind this synthetic historical data is presented in Figure 7.2. We observe that sex (S) influences height (H), and that both these features influence the Muscular Mass (M). In the end, past successes/failures (Y) were only based on the two attributes relevant to the job: H and M. Also, two noise features $N1, N2$ were added.

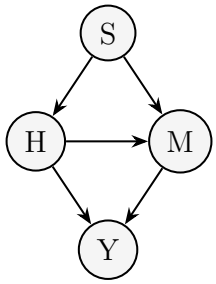


Figure 7.2 Graphical Model Generating the Toy Example.

We create a synthetic dataset following the causal graph in Figure 7.2 while ensuring that the feature distributions matched those in the following empirical study, comparing skeletal mass distributions between men and women [Janssen et al., 2000]. First, the sex feature was sampled from a Bernoulli $S \sim \text{Bernoulli}(0.5)$. According to Table 1 of [Janssen et al., 2000], the average height of women participants was 163 cm while it was 177cm for men. Both height distributions had the same standard deviation of 7cm. Hence, we sampled height via $H|S = \text{man} \sim \mathcal{N}(177, 49)$ and $H|S = \text{woman} \sim \mathcal{N}(163, 49)$. Janssen et al. [2000] noted an approximately linear relationship between height and skeletal muscle mass for both sexes. Therefore, we computed the muscle mass M as $M|\{H = h, S = \text{man}\} = 0.186h + 5\epsilon$ and $M|\{H = h, S = \text{woman}\} = 0.128h + 4\epsilon$ where $\epsilon \sim \mathcal{N}(0, 1)$ The values of coefficients 0.186, 0.128 and noise levels 5 and 4 were chosen so the distributions of $M|S$ would approximately

match that of Table 1 in [Janssen et al., 2000]. Finally, the target was chosen following

$$\begin{aligned} &Y|\{H=h, M=m\} \sim \text{Bernoulli}(\, P(H,M)\,) \\ &\text{with } \; P(H,M) = \Big[1+\exp\{100\times\mathbb{1}(H<160) - 0.3(M-28)\}\Big]^{-1}. \end{aligned} \tag{7.6}$$

Simply put, the chances of succeeding on the job ($Y=1$) were null for individuals with a smaller height than 160cm. Individuals higher than 160cm with a higher skeletal mass had the highest success rate. Yet, tall individuals with less muscle mass could still succeed if they displayed sufficient determination. In the end, $N=6000$ samples were taken leading to the following disparity

$$\mathbb{P}(Y=1|S=\text{man}) = 0.733 \qquad \mathbb{P}(Y=1|S=\text{woman}) = 0.110, \tag{7.7}$$

which is induced by correlations between strength and sex, not by past direct discrimination. Yet, if a predictive model is fitted on this data and used to screen applicants, there is a risk that the model learns the shortcut "women are less likely to succeed" and so directly uses the Sex feature for prediction. This model would then transform *indirect* discrimination into *direct* discrimination, and become illegal as a result. To explain the disparities, the company runs Fair-SHAP without cheating and observes a large attribution for the feature Sex, see the blue bar Figure 7.3. This current model is unlikely to pass the audit! The company has two options : retrain the model, or fake its way to success. Retraining a model is expensive and could take months on real datasets. Whereas cherry-picking subsets S'_0, S'_1 of size $M=100$ takes only a few minutes and leaves the model intact. Figure 7.3 illustrates the cherry-picking results using three methods: brute-force, genetic, and FoolSHAP presented in Section 7.2. Unlike competing approaches, FoolSHAP removed all the importance of Sex while increasing the importance of the meritocratic feature Height.

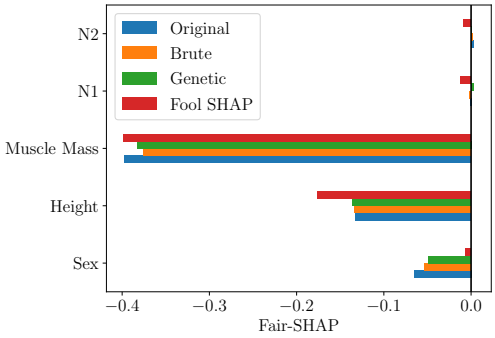
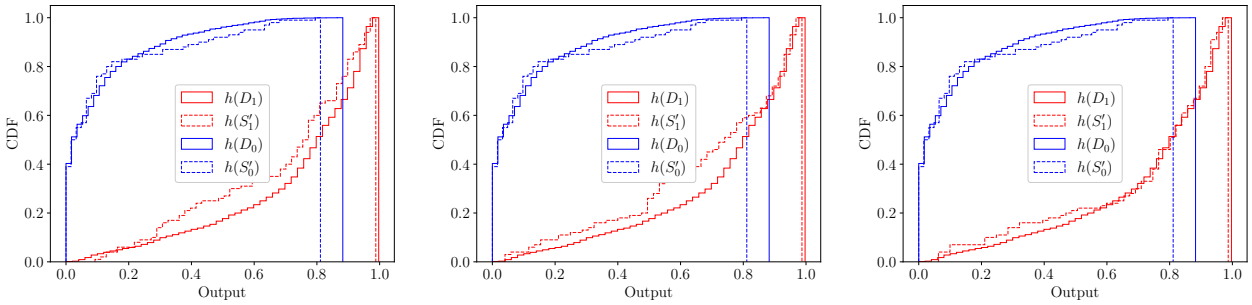


Figure 7.3 (Blue bar) the correct Fair-SHAP estimate obtained by sampling subsets uniformly at random. The importance given to Sex is unacceptable. (Other bars) are the results of the cherry-picking algorithms.

Is the auditor able to detect the cherry-picking of the subsets S'_0 and S'_1 send to them? Figure 7.4 compares the CDFs of $h(S'_0), h(D_0)$ and $h(S'_1), h(D_1)$. Note that FoolSHAP is the cherry-picking algorithm whose CDFs are closest to the full dataset. Consequently, the auditor cannot detect the fraud using statistical tests.



(a) CDFs for genetic algorithm. (b) CDFs for brute-force. (c) CDFs for FoolSHAP.

Figure 7.4

7.2 Methodology

7.2.1 Cherry-Picking

To fool the audit, the company can decide to indeed subsample S_0' uniformly at random $S_0' \sim \mathcal{F}^M$. Then, given this choice of foreground data, they can repeatedly subsample $S_1' \sim \mathcal{B}^M$, and choose the set S_1' leading to the smallest $|\widehat{\Phi}_s^{\text{Fair}}(h, S_0', S_1')|$. We shall call this method "brute-force". Its issue is that, by subsampling S_1' from $\mathcal{B}$, it will take an enormous number of repetitions to reduce the attribution since the score $\widehat{\Phi}_s^{\text{Fair}}(h, S_0', S_1')$ is concentrated on the ground-truth $\Phi_s^{\text{Fair}}(h, \mathcal{F}, \mathcal{B})$.

A more clever method is to re-weight the background distribution before sampling from it

$$\mathcal{B}_{\boldsymbol{\omega}} := \sum_{\boldsymbol{z}^{(j)} \in D_1} \omega_j \delta_{\boldsymbol{z}^{(j)}} \quad \text{where} \quad \omega_j \geq 0 \ \text{ and } \ \sum_j \omega_j = 1 \tag{7.8}$$

and then subsample $S_1' \sim \mathcal{B}_{\boldsymbol{\omega}}^M$. Data points $\boldsymbol{z}^{(j)}$ with a large weight ω_j have a higher probability of being chosen, hence the terminology *cherry-picking*. To make the model look fairer, the company needs the Fair-SHAP attributions computed with these cherry-picked points to have a small magnitude.

Proposition 7.2.1. *Let S_0' be **fixed**, and let $\xrightarrow{p}$ represent convergence in probability as the size M of the set $S_1' \sim \mathcal{B}_{\boldsymbol{\omega}}^M$ increases, we have*

$$\begin{aligned} \widehat{\Phi}_s^{Fair}(h, S_0', S_1') &\xrightarrow{p} \sum_{\boldsymbol{z}^{(j)} \in D_1} \omega_j \left(\frac{1}{M} \sum_{\boldsymbol{x}^{(i)} \in S_0'} \phi_s^{SHAP\text{-}int}(h, \boldsymbol{x}^{(i)}, \boldsymbol{z}^{(j)}) \right) \\ &:= \sum_{\boldsymbol{z}^{(j)} \in D_1} \omega_j a_j \end{aligned} \tag{7.9}$$

The proof is given in Appendix D.1.1.

The Fair-SHAP estimate with a weighted background concentrates on a linear function of the weights $\sum_j \omega_j a_j$. Note that the coefficients $a_j := \frac{1}{M} \sum_{\boldsymbol{x}^{(i)} \in S_0'} \phi_s^{\text{SHAP-int}}(h, \boldsymbol{x}^{(i)}, \boldsymbol{z}^{(j)})$ can be computed efficiently with Algorithm 5 when h is a tree ensemble. If the model is not a tree ensemble, the coefficients are still tractable but computing them might take a lot more time. Still, no matter how long it takes to compute the a_j coefficients, the company would only need to run these computations once and store the results on their server.

An additional requirement for the company to fool the audit is that the non-uniform distribution $\mathcal{B}_{\boldsymbol{\omega}}$ remains *similar* to the original $\mathcal{B}$. Otherwise, the fraud could be detected. Since the auditor can only access data through the output of the model $h(D)$, similarity between distributions can be defined with the Wasserstein distance in output space.

Definition 7.2.1 (Wassertein Distance)**.** *Any probability measure π over $D_1 \times D_1$ is called a coupling measure between $\mathcal{B}$ and $\mathcal{B}_{\boldsymbol{\omega}}$, denoted $\pi \in \Delta(\mathcal{B}, \mathcal{B}_{\boldsymbol{\omega}})$, if $1/N_1 = \sum_j \pi_{ij}$ and $\omega_j = \sum_i \pi_{ij}$. The Wassertein distance between $\mathcal{B}$ and $\mathcal{B}_{\boldsymbol{\omega}}$ mapped to the output-space is defined as*

$$\mathcal{W}(h(\mathcal{B}), h(\mathcal{B}_{\boldsymbol{\omega}})) = \min_{\pi \in \Delta(\mathcal{B}, \mathcal{B}_{\boldsymbol{\omega}})} \sum_{i,j} |h(\boldsymbol{z}^{(i)}) - h(\boldsymbol{z}^{(j)})| \pi_{ij}, \tag{7.10}$$

a.k.a the cost of the optimal transport plan that distributes the mass from one distribution to the other.

We propose Algorithm 6 to compute the weights $\boldsymbol{\omega}$ by minimizing the magnitude of the Fair-SHAP score while maintaining a small Wasserstein distance. The trade-off between attribution manipulation and proximity to the data is tuned via a hyper-parameter $\lambda > 0$. We show in the Appendix D.1.2 that the optimization problem at line 5 of Algorithm 6 can be reformulated as a Minimum Cost Flow (MCF) and hence can be solved in polynomial time (more precisely $\widetilde{\mathcal{O}}(N_1^{2.5})$ as in [Fukuchi et al., 2020]).

Algorithm 6 Compute non-uniform weights

1: **procedure** COMPUTE_WEIGHTS($D_1, \{a_j\}_j, \lambda$)
2: $\quad \beta \;:= \text{sign}[\sum_{\boldsymbol{z}^{(j)} \in D_1} a_j]$
3: $\quad \mathcal{B} \;:= \frac{1}{N_1} \sum_{\boldsymbol{z}^{(j)} \in D_1} \delta_{\boldsymbol{z}^{(j)}}$ ▷ Unmanipulated background
4: $\quad \mathcal{B}_{\boldsymbol{\omega}} := \sum_{\boldsymbol{z}^{(j)} \in D_1} \omega_j \delta_{\boldsymbol{z}^{(j)}}$ ▷ Manipulated background as a function of $\boldsymbol{\omega}$
5: $\quad \boldsymbol{\omega} = \text{argmin}_{\boldsymbol{\omega}} \;\; \beta \sum_{\boldsymbol{z}^{(j)} \in D_1} \omega_j a_j + \lambda \mathcal{W}(h(\mathcal{B}), h(\mathcal{B}_{\boldsymbol{\omega}}))$ ▷ Optimization Problem
6: $\quad$ **return** $\boldsymbol{\omega}$;

7.2.2 Detection

We now discuss ways the audit can detect manipulation of the sampling procedure. Recall that the auditor has previously been given access to $h(D_0), h(D_1)$ representing the model

outputs across all instances in the private dataset. The auditor will then be given subsamples S'_0, S'_1 of D_0, D_1 on which they can compute the output of the model and compare with $h(D_0), h(D_1)$. To assess whether the subsamples provided by the company were sampled uniformly at random, the audit has to conduct statistical tests. The null hypothesis of these tests will be that S'_0, S'_1 were sampled uniformly at random from D_0, D_1.

KS test A first test that can be conducted is a two-samples Kolmogorov-Smirnov (KS) test [Massey Jr, 1951]. We let

$$\widehat{F}_S(x) = \frac{1}{|S|} \sum_{z \in S} \mathbb{1}(z \leq x) \tag{7.11}$$

be the empirical CDF of observations in the finite set S. Given two sets S and S', the KS statistic is

$$\mathrm{KS}(S, S') = \sup_{x \in \mathbb{R}} |\widehat{F}_S(x) - \widehat{F}_{S'}(x)|. \tag{7.12}$$

Under the null-hypothesis $H_0 : S \sim \mathcal{R}^{|S|}, S' \sim \mathcal{R}^{|S'|}$ for some univariate distribution $\mathcal{R}$, this statistic is expected to not be too large with high probability. Hence, when the company provides the subsets S'_0, S'_1, the audit can sample their own two subsets $h(S_0), h(S_1)$ uniformly at random from $h(D_0), h(D_1)$ and compute the statistics $\mathrm{KS}(h(S_0), h(S'_0))$ and $\mathrm{KS}(h(S_1), h(S'_1))$ to detect a fraud.

Wald test An alternative is the Wald test, which is based on the central limit theorem. Under the null hypothesis $H_0 : S \sim \mathcal{R}^M$ for some distribution $\mathcal{R}$, the empirical average of the model outputs $h(S)$ is asymptotically normally distributed as M increases. We have

$$\mathrm{Wald}(h(S), h(\mathcal{R})) := \frac{\frac{1}{M} \sum_{z \in h(S)} z - \mu}{\sigma / \sqrt{M}} \rightsquigarrow \mathcal{N}(0, 1), \tag{7.13}$$

where $\mu := \mathbb{E}_{z \sim h(\mathcal{R})}[z]$ and $\sigma^2 := \mathbb{V}_{z \sim h(\mathcal{R})}[z]$. The test rejects the null hypothesis for set S' if

$$|\,\mathrm{Wald}(h(S'), h(\mathcal{R}))\,| > F^{-1}_{\mathcal{N}(0,1)}(1 - \alpha/2), \tag{7.14}$$

where $F^{-1}_{\mathcal{N}(0,1)}$ is the inverse CDF of a standard normal variable.

The detection Algorithm 7 with significance α uses both the Kolmogorov-Smirnov and Wald tests with Bonferonni corrections (*i.e.* the $\alpha/4$ terms in the Algorithm).

Algorithm 7 Detection with significance α

procedure DETECT_FRAUD($h(D_0), h(D_1), h(S'_0), h(S'_1), \alpha,\ M$)
  for $i = 0, 1$ **do**
    Sample M instances uniformly at random from $h(D_i)$ and store in $h(S_i)$;
    p-value-KS = KS($h(S_i), h(S'_i)$)
    p-value-Wald = Wald($h(S'_i), h(D_i)$)
    if p-value-KS $< \alpha/4$ **or** p-value-Wald $< \alpha/4$ **then** ▷ Reject the null hypothesis
      return 1
  return 0;

7.2.3 FoolSHAP

The procedure returning the subsets S'_0, S'_1 is presented in Algorithm 8. It conducts a log-space search between $\lambda_{\min}$ and $\lambda_{\max}$ for the λ hyper-parameter (line 6) in order to explore the possible attacks. For each value of λ, the attacker runs Algorithm 6 to obtain $\mathcal{B}_{\boldsymbol{\omega}}$ (line 7), then repeatedly samples $S'_1 \sim \mathcal{B}^M_{\boldsymbol{\omega}}$ (line 10) and attempts to detect the fraud (line 11). The attacker will choose $\mathcal{B}_{\boldsymbol{\omega}}$ that minimizes the magnitude of $\widehat{\Phi}_s$ while having a detection rate below some threshold τ (line 12). An example of search over λ on a real-world dataset is presented in Figure 7.5.

Algorithm 8 FoolSHAP

1: **procedure** FOOL_SHAP($h, D_0, D_1, M, \lambda_{\min}, \lambda_{\max}, \tau, \alpha$)
2: Subsample M instances S'_0 from D_0 without cheating
3: Compute $a_j := \frac{1}{M} \sum_{\boldsymbol{x}^{(i)} \in S'_0} \phi_s^{\text{SHAP-int}}(h, \boldsymbol{x}^{(i)}, \boldsymbol{z}^{(j)})$
4: Initialize $\omega_j^\star = 1/N_1 \ \ \forall j$
5: Initialize $\Phi_s^\star = \sum_{\boldsymbol{z}^{(j)} \in D_1} \omega_j^\star\, a_j$
6: **for** $\lambda = \lambda_{\max}, \ldots, \lambda_{\min}$ **do**
7: $\boldsymbol{\omega}$ = COMPUTE_WEIGHTS($D_1, \{a_j\}_j, \lambda$)
8: `Detection` $= 0$
9: **for** rep $= 1, \ldots, 100$ **do** ▷ Detect the manipulation
10: $S'_1 \sim \mathcal{B}^M_{\boldsymbol{\omega}}$
11: `Detection` += DETECT_FRAUD($h(D_0), h(D_1), h(S'_0), h(S'_1), \alpha,\ M$)
12: **if** $|\sum_{\boldsymbol{z}^{(j)} \in D_1} \omega_j\, a_j| < |\Phi_s^\star|$ **and** `Detection` $< 100\tau$ **then**
13: $\boldsymbol{\omega}^\star = \boldsymbol{\omega}$
14: $\Phi_s^\star = \sum_{\boldsymbol{z}^{(j)} \in D_1} \omega_j\, a_j$ ▷ Update the solution
15: $S'_1 \sim \mathcal{B}^M_{\boldsymbol{\omega}^\star}$ ▷ Cherry-pick by sampling from the non-uniform background
16: **return** S'_0, S'_1

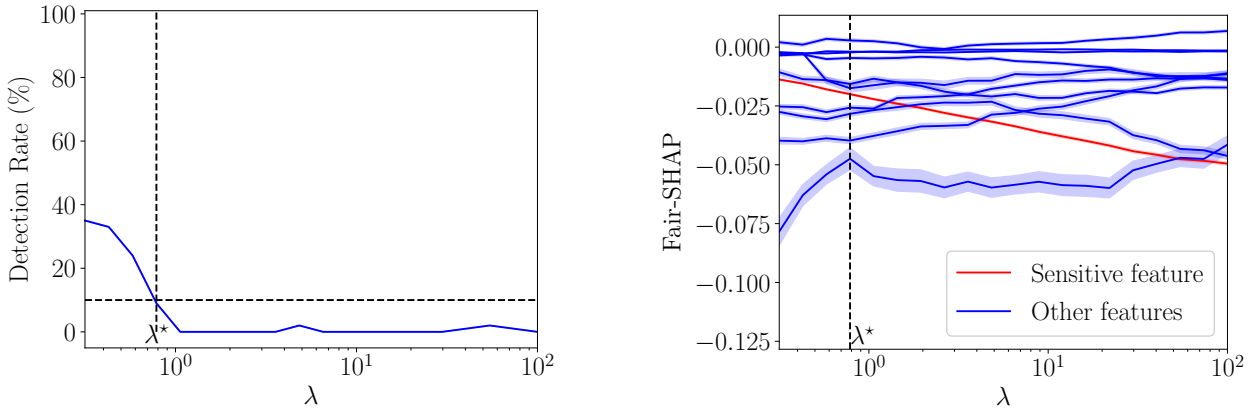


Figure 7.5 Example of log-space search over values of λ using an XGBoost classifier fitted on Adults. (a) The detection rate as a function of the parameter λ of the attack. The attacker uses a detection rate threshold $\tau = 10\%$. (b) For each value of λ, the vertical slice of the 11 curves is the Fair-SHAP obtained with the resulting $\mathcal{B}_\omega$. The goal here is to reduce the amplitude of the sensitive feature (red curve).

7.2.4 Contributions

The first technique to fool `SHAP` with perturbations of the background distribution was a genetic algorithm [Baniecki and Biecek, 2022]. Although promising, the cross-over and mutation operations it employs to perturb data do not take into account feature correlations and can therefore generate unrealistic data. Moreover, the objective to minimize does not enforce similarity between the original and manipulated backgrounds. We show in Appendix D.2 that these limitations lead to systematic fraud detections. Hence, our contributions are two-fold. First, by perturbing the background via non-uniform weights over pre-existing instances rather than a genetic algorithm, we avoid the issue of non-realistic data. Second, by considering the Wasserstein distance, we can control the similarity between the original and fake backgrounds.

The FoolSHAP Algorithm is an extension of the Stealthily Biased Sampling method introduced by [Fukuchi et al., 2020]. In their work, of Fukuchi et al. minimize the Wasserstein distance in input space while enforcing a hard constraint on the number of instances that land on the four different bins $(x_s, y) \in \{0, 1\}^2$. As a result, they can set the Demographic Parity to any given value while staying close to the original data. In the FoolSHAP setting of manipulating model explanations, we leave the Demographic Parity intact and instead manipulate its feature attributions. In terms of optimization objective, we now minimize a Shapley value with a soft constraint on the Wasserstein distance in output space. The Wasserstein distance is minimized in output space rather than input space to ensure that the Demographic Parity being explained remains intact.

7.3 Experiments

The following experiments compare the Brute-Force, Genetic, and FoolSHAP cherry-picking algorithms. The source code is available online[3].

7.3.1 Datasets

We consider four standard datasets from the FAccT literature, namely COMPAS, Adult-Income, Marketing, and Communities.

- **COMPAS** regroups 6,150 records from criminal offenders in Florida collected from 2013-2014. This binary classification task consists in predicting who will re-offend within two years. The sensitive feature s is `race` with values $x_s = 0$ for African-American and $x_s = 1$ for Caucasian.
- **Adult Income** contains demographic attributes of 48,842 individuals from the 1994 U.S. census. It is a binary classification problem with the goal of predicting whether a particular person makes more than 50K USD per year. The sensitive feature s in this dataset is `gender`, which took values $x_s = 0$ for female, and $x_s = 1$ for male.
- **Marketing** involves information on 41,175 customers of a Portuguese bank and the binary classification task is to predict who will subscribe to a term deposit. The sensitive attribute is `age` and took values $x_s = 0$ for age `30-60`, and $x_s = 1$ for age `not30-60`
- **Communities & Crime** contains per-capita violent crimes for 1994 different communities in the US. The binary classification task is to predict which communities have crimes below the median rate. The sensitive attribute is `PercentWhite` and took values $x_s = 0$ for `PercentWhite<90%`, and $x_s = 1$ for `PercentWhite>=90%`.

Three models were considered: Multi-Layered Perceptrons (MLP), Random Forests (RF), and eXtreme Gradient Boosted trees (XGB). One model of each type was fitted on each dataset with a $4:1$ train/test split ratio, and for 5 different train/test split seeds, resulting in 60 models total. All categorical features for COMPAS, Adult, and Marketing were one-hot-encoded which resulted in a total of 11, 40, and 61 columns for each dataset. A simple 50 steps random search was conducted to fine-tune the hyper-parameters with five-fold cross-validation on the training set. The resulting test set performance and demographic parities, aggregated over the 5 random data splits, are reported in Tables 7.1 and 7.2.

[3] `https://github.com/gablabc/Fool_SHAP`

Table 7.1 Models Test Accuracy % (mean ± stddev).

	mlp	rf	xgb
COMPAS	68.2 ± 0.9	67.7 ± 0.8	68.6 ± 0.8
Adult	85.6 ± 0.3	86.3 ± 0.2	87.1 ± 0.1
Marketing		91.1 ± 0.1	91.4 ± 0.3
Communities		83 ± 2	82 ± 2

Table 7.2 Models Demographic Parity (mean ± stddev).

	mlp	rf	xgb
COMPAS	-0.12 ± 0.01	-0.11 ± 0.01	-0.11 ± 0.02
Adult	-0.20 ± 0.01	-0.19 ± 0.01	-0.192 ± 0.004
Marketing		-0.104 ± 0.005	-0.11 ± 0.01
Communities		-0.50 ± 0.01	-0.54 ± 0.02

7.3.2 Detector Calibration

Detector calibration refers to the assessment that, assuming the null hypothesis to be true, the probability of rejecting it (*i.e.* false positive) should be bounded by the significance level α. Remember that the null hypothesis of the audit detector is that the sets S'_0, S'_1 provided by the company are sampled uniformly from D_0, D_1. Hence, to test the detector, the auditor can sample their own subsets $h(S_0), h(S_1)$ uniformly from at random from $h(D_0), h(D_1)$, run the detection algorithm, and count the number of detection over 1000 repeats. Table 7.3 shows the false positive rates over the five train-test splits using a significance level $\alpha = 5\%$. We observe that the false positive rates are indeed bounded by α for all model types and datasets implying that the detector employed by the auditor is calibrated.

Table 7.3 False Positive Rates (%) of the detector *i.e.* the frequency at which S_0, S_1 are considered cherry-picked when they are not. No rate should be above 5%.

	mlp	rf	xgb
COMPAS	4.0	4.6	4.0
Adult	4.3	4.3	4.2
Marketing		4.9	5.0
Communities		3.8	4.2

7.3.3 Attack Results

The first step of the attack (line 3 of Algorithm 8) requires that the company run `SHAP` on their own and compute the necessary coefficients to run Algorithm 6. For the COMPAS and Adults datasets, the `ExactExplainer` of `SHAP` was used. Since Marketing and Communities contain more than 15 features, and since the `ExactExplainer` scales exponentially with the number of features, we were restricted to using the tree-based optimizations (cf. Algorithm 5). This efficient algorithm avoids the exponential cost of Shapley values but is only applicable to tree-based models such as RFs and XGBs. Therefore, we could not conduct the attack on MLPs fitted on Marketing and Communities.

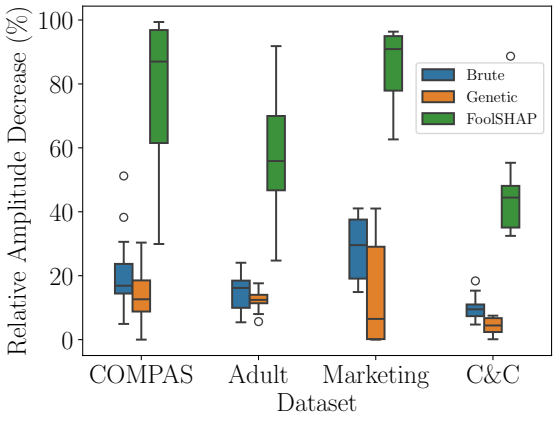


Figure 7.6 Relative decrease in amplitude of the sensitive feature attribution induced by the various attacks on `SHAP`.

The following step is to solve the MCF for various values of λ (line 7 of Algorithm 8). As stated previously, solving the MCF can be done in polynomial time in terms of N_1, which was feasible for small datasets like COMPAS and Communities, but not for larger datasets like Adult and Marketing. To solve this issue, as was done in [Fukuchi et al., 2020], we compute the manipulated weights multiple times using 5 bootstrap subsamples of D_1 of size 2000 to obtain a set of weights $\boldsymbol{\omega}^{[1]}, \boldsymbol{\omega}^{[2]}, \ldots, \boldsymbol{\omega}^{[5]}$ which we average to obtain the final weights $\boldsymbol{\omega}$.

Results of 46 attacks with $M = 200$ are shown in Figure 7.6. As a point of reference, we also show results for the brute-force and genetic algorithms. To make comparisons to our attack more meaningful, the brute-force method was only allowed to run for the same amount of time it took to search for the non-uniform weights $\boldsymbol{\omega}$ (about 30-180 seconds). Also, the genetic algorithm ran for 400 iterations and was stopped early if there were 10 consecutive detections. We note that, across all datasets, FoolSHAP leads to greater reductions of the sensitive feature attribution compared to brute-force search and the genetic perturbations of the background.

Now focusing on FoolSHAP, for the datasets COMPAS and Marketing, we observe median reductions in amplitudes of about 90%. This means that our attack can considerably reduce the apparent importance of the sensitive attribute. For the Adult and Communities datasets, the median reduction in amplitude is about 50% meaning that we typically reduce by half the importance of the sensitive feature. Still, looking at the maximum reduction in amplitude for

Adult-Income and Communities, we note that one attack managed to reduce the amplitude by 90%. Therefore, luck can play a part in the degree of success of FoolSHAP, which is to be expected from data-driven attacks. Finally, the audit was consistently unable to detect the fraud using statistical tests.

We end this section by presenting 8 examples of FoolSHAP attacks, two per dataset. The two main takeaways of these 8 Figures are 1) The manipulated CDFs $h(S'_1)$ are similar to the true ones $h(D_1)$. 2) The Confidence Interval shown around the fake Fair-SHAP estimates (orange bars) do not cover the correct Fair-SHAP estimate (the blue bar). This is because the null hypothesis $H_0 : S'_1 \sim \mathcal{B}^M$ is not respected and so Confidence Intervals do not provide a relevant measure of uncertainty.

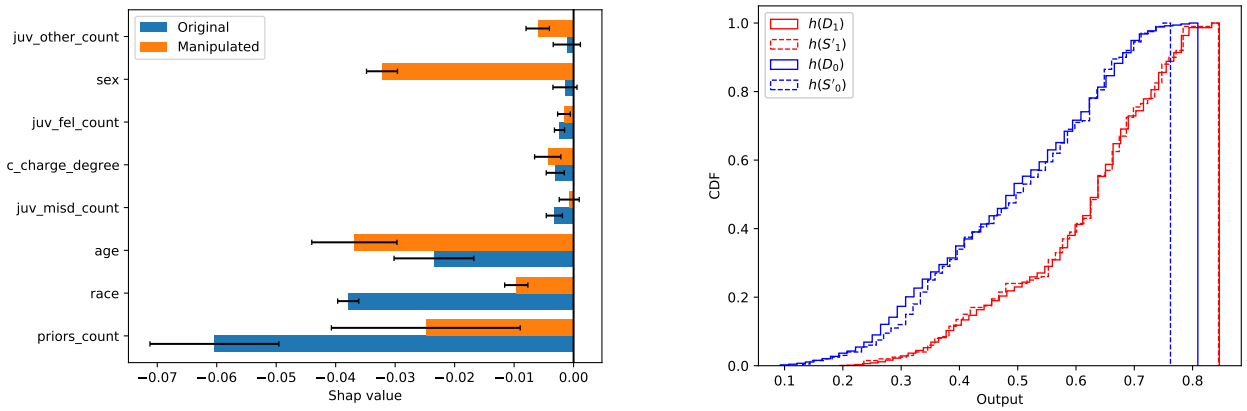


Figure 7.7 Attack of RF fitted on COMPAS. Left: Fair-SHAP before and after the attack with $M = 200$. As a reminder, the sensitive attribute is `race`. Right: Comparison of the CDF of the misleading subsets $h(S'_0), h(S'_1)$ and the CDF over the whole data. $h(D_0), h(D_1)$.

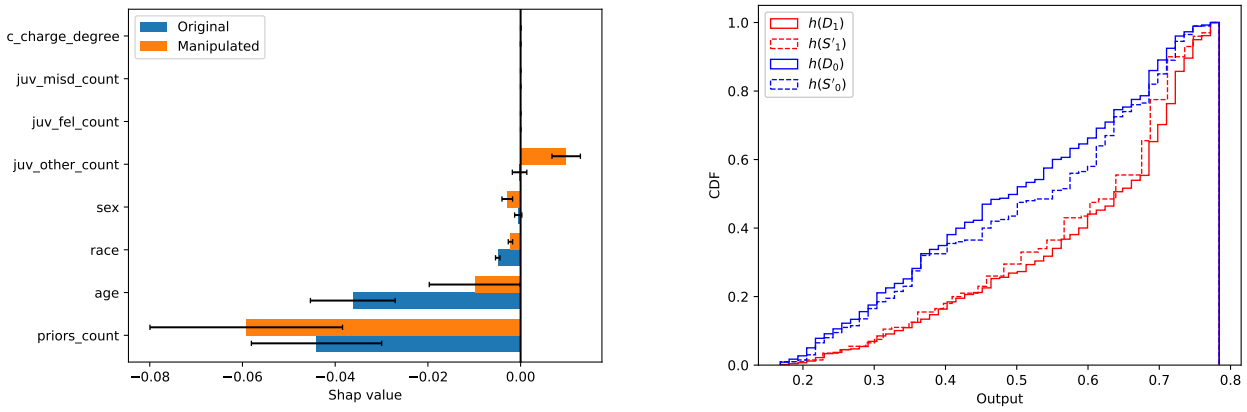


Figure 7.8 Attack of XGB fitted on COMPAS. Left: Fair-SHAP before and after the attack with $M = 200$. As a reminder, the sensitive attribute is `race`. Right: Comparison of the CDF of the misleading subsets $h(S'_0), h(S'_1)$ and the CDF over the whole data. $h(D_0), h(D_1)$.

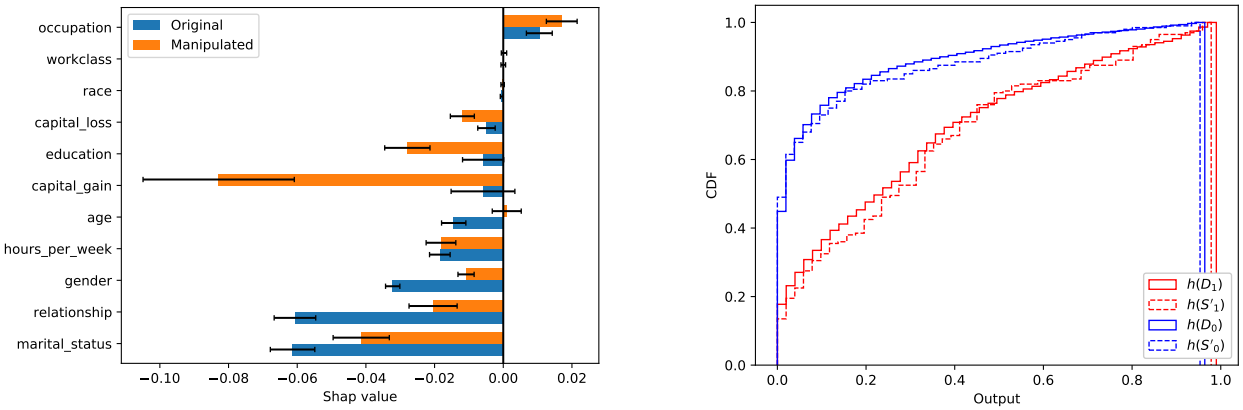


Figure 7.9 Attack of RF fitted on Adults. Left: Fair-SHAP before and after the attack with $M = 200$. As a reminder, the sensitive attribute is `gender`. Right: Comparison of the CDF of the misleading subsets $h(S'_0), h(S'_1)$ and the CDF over the whole data. $h(D_0), h(D_1)$.

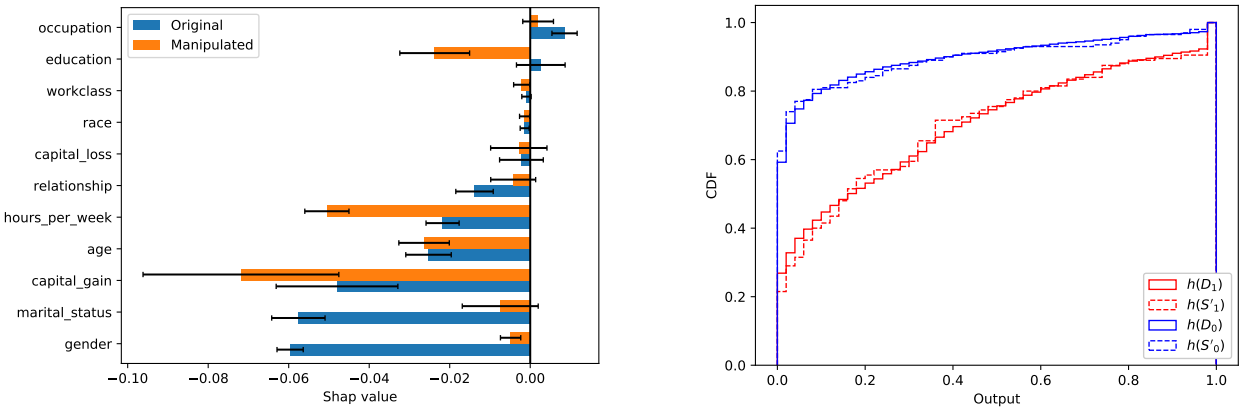


Figure 7.10 Attack of XGB fitted on Adults. Left: Fair-SHAP before and after the attack with $M = 200$. As a reminder, the sensitive attribute is `gender`. Right: Comparison of the CDF of the misleading subsets $h(S'_0), h(S'_1)$ and the CDF over the whole data. $h(D_0), h(D_1)$.

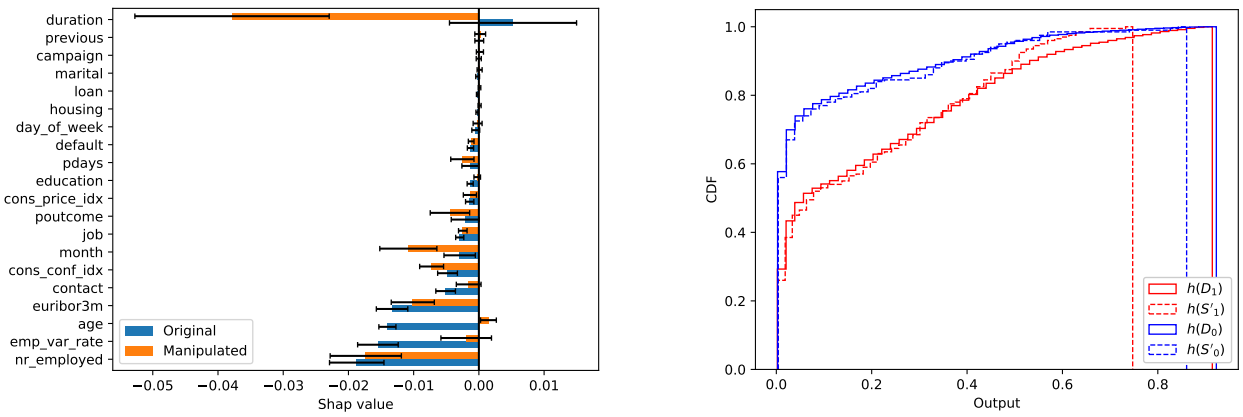


Figure 7.11 Attack of RF fitted on Marketing. Left: Fair-SHAP before and after the attack with $M = 200$. As a reminder, the sensitive attribute is `age`. Right: Comparison of the CDF of the misleading subsets $h(S'_0), h(S'_1)$ and the CDF over the whole data. $h(D_0), h(D_1)$.

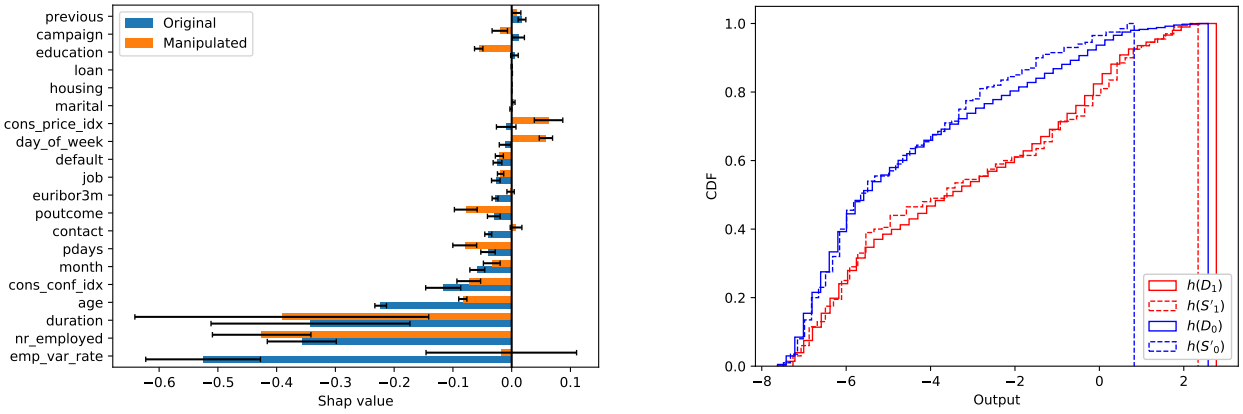


Figure 7.12 Attack of XGB fitted on Marketing. Left: Fair-SHAP before and after the attack with $M = 200$. As a reminder, the sensitive attribute is `age`. Right: Comparison of the CDF of the misleading subsets $h(S'_0), h(S'_1)$ and the CDF over the whole data. $h(D_0), h(D_1)$.

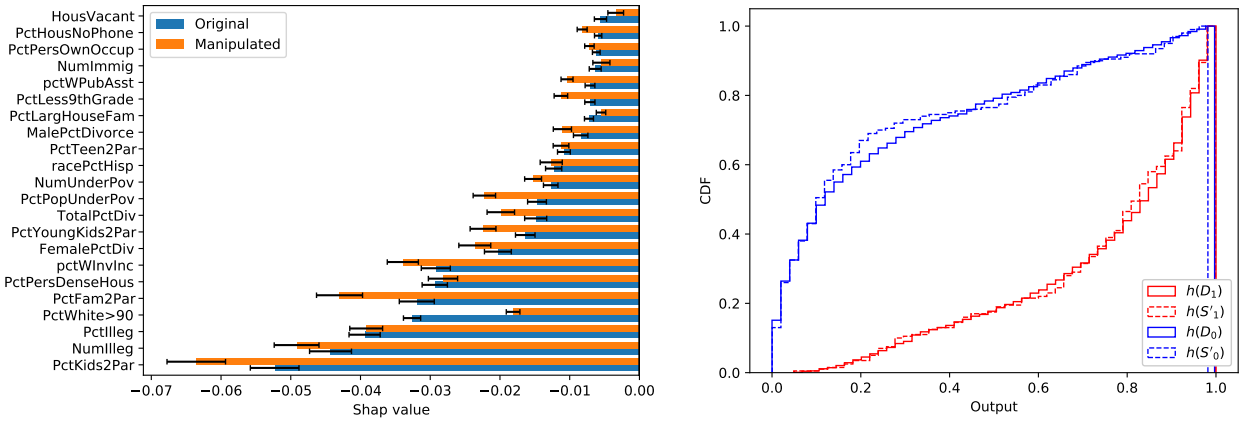


Figure 7.13 Attack of RF fitted on Communities. Left: Fair-SHAP before and after the attack with $M = 200$. As a reminder, the sensitive attribute is `PctWhite>90`. Right: Comparison of the CDF of the misleading subsets $h(S'_0), h(S'_1)$ and the CDF over the whole data. $h(D_0), h(D_1)$.

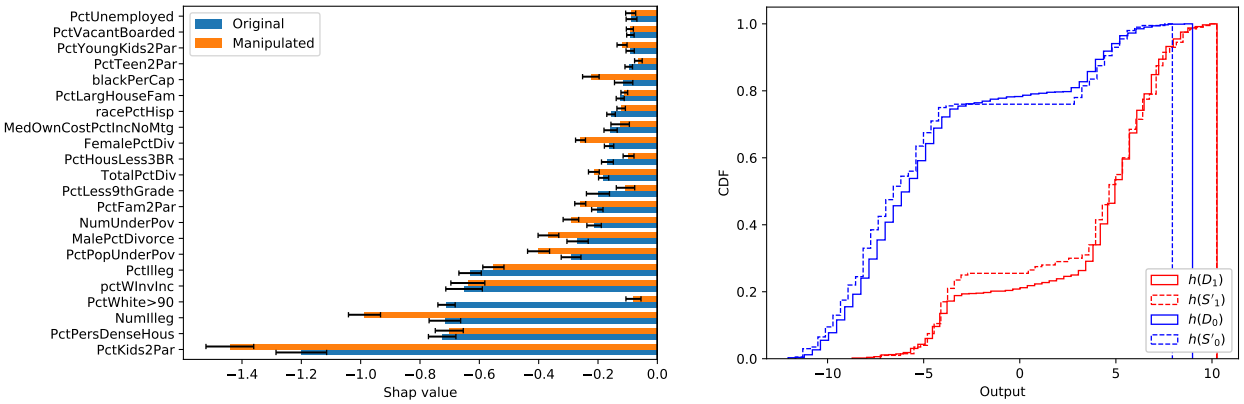


Figure 7.14 Attack of XGB fitted on Communities. Left: Fair-SHAP before and after the attack with $M = 200$. As a reminder, the sensitive attribute is `PctWhite>90`. Right: Comparison of the CDF of the misleading subsets $h(S'_0), h(S'_1)$ and the CDF over the whole data. $h(D_0), h(D_1)$.

Contributions

We identified the random subsampling of data points as a source of disagreements between post-hoc additive explainers. Indeed, if the subsample size is too low, running the same code twice could result in different explanations. Fortunately, statistical Confidence Intervals offer a principled mean of characterizing this uncertainty.

These Confidence Intervals assume that the data is subsampled uniformly at random, which will hold in most practical applications. Nonetheless, we have presented a novel Audit-Scenario, where assuming the data was subsampled uniformly at random leads to the wrong conclusion that a biased model is unbiased.

CHAPTER 8 UNDERSPECIFICATION DISAGREEMENT

8.1 Motivation

The Rashomon Effect [Breiman, 2001b], also known as model under-specification [D'Amour et al., 2020] or model multiplicity [Marx et al., 2020] refers to the observation that there often exists a large diversity of models that fit empirical data well. This phenomenon is both a blessing and a curse.

It is a blessing in settings where practitioners are only interested in finding *some* model with satisfactory performance. The Rashomon Effect consequently implies the existence of a very large pool of models to choose from. Some of these candidate models might possess additional desirable properties such as fairness or interpretability [Semenova et al., 2022].

However, the Rashomon Effect becomes a curse when one wishes to extract insights from model explanations. What is one expected to conclude if a large set of good models yield contradictory local explanations for the same Gap, or different rankings of global feature importance? In the words of Leo Breiman [2001b] "*The multiplicity problem and its effect on conclusions drawn from models needs serious attention.*"

We illustrate the underspecification of local feature attributions on a toy regression problem. We sampled 1000 4-dimensional points $\boldsymbol{x} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$ where $\boldsymbol{\Sigma}$ is identity, except for $\Sigma_{1,2} = \Sigma_{2,1} = 0.75$, labelled them via $y := f(\boldsymbol{x}) + \Delta$, with $f(\boldsymbol{x}) = -8\cos(x_1 - x_2)\cos(x_1 + x_2) + 1.5x_3$ (x_4 is a dummy variable) and Δ is Gaussian noise with standard deviation $\sigma = 0.1$. We then independently trained five Multi-Layered Perceptrons (MLP) with layerwidths$= 50, 20, 10$ and ReLU activations. All models ended up having test set Root-Mean-Squared-Error (RMSE) between 0.47 and 0.62, while the target had a standard deviation of 4.91. After conducting paired Student-t tests between the model with RMSE 0.47 and the four others, we concluded that the one with error 0.62 was significantly worst and should be discarded. The other three models did not have a significantly worst test RMSE and so we kept them.

We analyzed the predictions of the four remaining models at the input $\boldsymbol{x} = (\frac{\pi}{2}, \frac{\pi}{2}, \frac{\pi}{2}, \frac{\pi}{2})$ which ranged from 9.05 to 10.05 (the ground truth being $f(\boldsymbol{x}) = 0.75\pi + 8 \approx 10.36$). Specifying the background distribution $\mathcal{B}$ to be the whole training set, we computed the output baselines $\mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[\, h_k(\boldsymbol{z})\,]$ which ranged from -1.00 to -1.07 across the four models. Therefore, for all four models, the prediction Gap $G(h_k, \boldsymbol{x}, \mathcal{B})$ was positive meaning that running SHAP or EG on all models would answer the same contrastive question: why is the prediction at

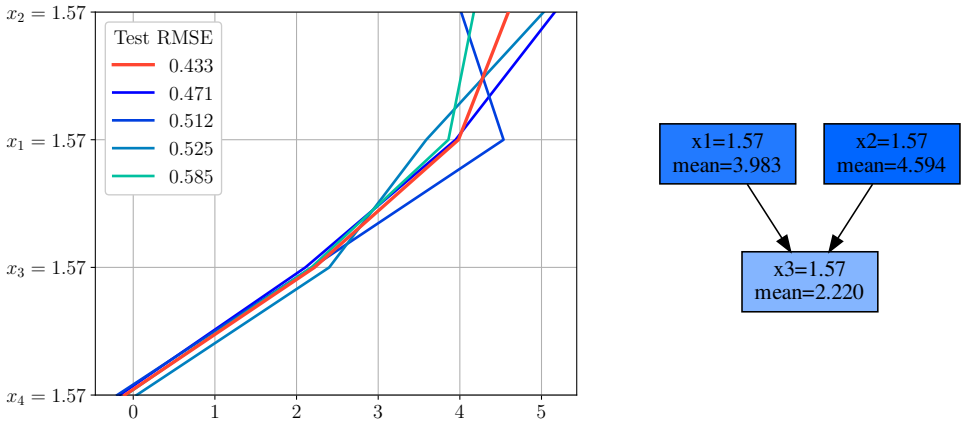


Figure 8.1 Left: local feature attributions for the average model $\overline{h}$ (orange line) and each individual model (blue lines). Right: Partial order of local feature importance. There is a directed path from feature x_i to feature x_j if **all good models** agree that feature x_i is more important than x_j.

$\boldsymbol{x}$ so much higher-than-average? To provide insight into why the Gaps at $\boldsymbol{x}$ are positive, Figure 8.1 (Left) presents the SHAP local feature attributions for all four models as blue lines. We see that the various MLPs lead to different interpretations.

Previous work tackles these disagreements by aggregating the feature attributions of multiple independently trained models. Consensus scores are provided in tandem with the aggregated attributions as means to convey how confident the feature attributions are.

For instance, Shaikhina et al. [2021] aggregate local feature attributions by defining the average model $\overline{h} = \frac{1}{M}\sum_{i=1}^{M} h_k$ and computing its feature attributions $\boldsymbol{\phi}(\overline{h}, \boldsymbol{x}, \mathcal{B})$. Following their methodology, we average the predictions of our four models, leading to a single predictor $\overline{h}$ with a test RMSE of 0.43. The resulting SHAP feature attribution is shown as an orange line in Figure 8.1 (Left). The total order of local feature importance for this average model is represented in the first column of Table 8.1. In particular, this explanation suggests that x_2 is more important than x_1, which given our knowledge of the symmetry of the ground truth seems somewhat spurious. Indeed, since the underlying data-generating distribution, the target function f, and the point $\boldsymbol{x}$ to explain are all symmetric w.r.t x_1 and x_2, an ideal explanation would certainly not support that x_2 is more important than x_1. Shaikhina et al. [2021] advocate reporting the variance $\frac{1}{M}\sum_{k=1}^{M}(\,\phi_j(h_k, \boldsymbol{x}, \mathcal{B}) - \phi_j(\overline{h}, \boldsymbol{x}, \mathcal{B})\,)^2$ as a consensus score for feature attributions, see the second column of Table 8.1. Note that variance is higher for the attributions of features x_1 and x_2, suggesting that their contribution toward the output is more uncertain. Still, it is unclear what variance values are low/high enough to label attributions as trustworthy/untrustworthy. Moreover, despite their higher variance,

Feature	Attribution $\overline{h}$	Variance	Mean rank	Ordinal Consensus
$x_2 = 1.57$	4.59	0.50	2.75	0.83
$x_1 = 1.57$	3.98	0.35	2.25	0.83
$x_3 = 1.57$	2.22	0.10	1.0	1.00
$x_4 = 1.57$	-0.10	0.09	0.0	1.00

Table 8.1 Aggregated feature attributions and consensus scores following previous methods.

features x_1 and x_2 are locally more important than features x_3 and x_4 for all models. Thus, the variance can lead to an overly pessimistic picture of the insights one can gather from feature attributions of multiple models

In a similar effort, Schulz et al. [2021] aggregate local explanations by averaging the ranks of the feature importance across models $\frac{1}{M}\sum_{k=1}^{M}\mathbf{r}[\,|\boldsymbol{\phi}(h_k, \boldsymbol{x}, \mathcal{B})|\,]$, where $\mathbf{r}:\mathbb{R}_+^d \to [d]$ is the rank function that maps each component of a vector to its rank among the other components. The results on the toy example are shown in the third column of Table 8.1. This aggregation also advocates that x_2 is locally more important than x_1, which is again spurious. As the consensus metric, Schulz et al. [2021] use the ordinal consensus that takes values between 0 and 1 and measures the consistency between the various rankings. This score is presented in the fourth column of Table 8.1. It highlights that all feature importance ranks are confident. Indeed, both x_1 and x_2 have an Ordinal Consensus of 0.83 seeing that there is only a single model for which the ranks of these two features are switched. Nonetheless, looking at Figure 8.1 (Left), the model that contradicts all others has a test RMSE of 0.512, which is the second best of the whole ensemble. Simply put, this model offers a different but still valid perspective on the data. However, its opinion is "washed out" by the other three models in the computation of the Ordinal Consensus. Hence, we argue that the Ordinal Consensus offers a view of consensus that is too optimistic.

As we have just highlighted, the methods of [Shaikhina et al., 2021] and [Schulz et al., 2021] share the same limitations:

- It is unclear what statements one can/cannot make using these frameworks. For instance, is x_2 really more important than x_1 for explaining the gap? Both approaches return a total order of local feature importance, which suggests one statement of relative importance for every pair of features *i.e.* feature i is locally less/more important than feature j. As we have seen, the consensus metrics provided in tandem with the total orders (Variance or Ordinal Consensus) do not help deciding what statements on relative importance are trustworthy.

- It is unclear what is the impact of model performances on the insights provided by these two methods. For instance, the second-best model in the ensemble contradicts all others regarding the relative importance of x_1 and x_2. However, its opinions are diluted when aggregating all explanations.

In light of those takeaways, we characterize feature attribution's disagreements by investigating *statements* about relative feature importance, and whether all good models agree on them. For instance, how can we confirm that feature x_2 is locally more important than x_1? As noted earlier, one model considers, contrary to the other four, that x_1 is more important than x_2. Given that this model is as good as any other, we can simply **abstain** from claiming any relation of importance between x_1 and x_2. In this case, abstention seems indeed a cautious position given the symmetry of the ground truth. Following this logic, for every other pair of features, we check if all four models agree on their relative importance. For instance, all four models agree that x_1 is more important than x_3. We record this consensus as a trustworthy statement and we represent it with an arrow from x_1 to x_3 in Figure 8.1 (Right). Furthermore, while all four models agree that x_1, x_2 and x_3 have a positive attribution, this is not the case for x_4 (our dummy variable). Based on this observation, we keep only the variables for which all models agree on the sign and exclude x_4 from our final explanation. All relations of importance between pairs of features for which there is consensus among the four models form a *partial order*. The partial order is visualized with a Hasse Diagram in Figure 8.1 (Right).

8.2 Disagreement Measure

8.2.1 Rashomon Set

Underspecification is characterized theoretically using Rashomon Sets [Fisher et al., 2019]

Definition 8.2.1 (Rashomon Set)**.** *Given a hypothesis space $\mathcal{H}$, a loss function ℓ, a data set S, and a tolerance threshold $\epsilon > 0$, the Rashomon set is defined as*

$$\mathcal{R}(\mathcal{H}, \epsilon) := \Big\{ h \in \mathcal{H} : \widehat{\mathcal{L}}_S(h) \leq \epsilon \Big\}, \tag{8.1}$$

where we leave the dependence in S and ℓ implicit from the context.

These sets contain all models that have an acceptable performance ϵ on empirical data. Although they have an appealing and simple interpretation, their exact computation is intractable unless $|\mathcal{H}|$ is small or $\mathcal{H}$ is the set of linear models fitted with squared loss. Hence,

in general settings, the Rashomon Sets have to be estimated, which can be done by sampling models and keeping the ones with satisfactory performance [Dong and Rudin, 2019, Semenova et al., 2022]. However, this method can be time-consuming and requires extensive memory to store thousands of models.

Other approaches work implicitly with the Rashomon Set by solving optimization problems over $\mathcal{H}$ under the constraint that $\widehat{\mathcal{L}}_S(h) \leq \epsilon$. In doing so, one can explore the different characteristics of models in the Rashomon Set without ever needing to represent the set explicitly. Such optimization problems have been studied to characterize the under-specification of model predictions [Coker et al., 2021, Hsu and Calmon, 2022, Marx et al., 2020], and global feature importance [Fisher et al., 2019].

We now present how to study local feature attributions of all models in the Rashomon Set.

8.2.2 Local Feature Attributions

Let $s : \mathcal{H} \times \mathcal{X} \times \mathcal{P}(\mathcal{X}) \to \{0, 1\}$ be a statement supported by h about the local feature attributions at $\boldsymbol{x}$ relative to $\mathcal{B}$. Given a performance threshold $\epsilon > 0$, end-users will only be presented statements on which there is a perfect consensus for all models in the Rashomon Set

$$\forall h \in \mathcal{R}(\mathcal{H}, \epsilon) \quad s(h, \boldsymbol{x}, \mathcal{B}) = 1. \tag{8.2}$$

We now present various statements about local feature attributions.

Definition 8.2.2 (Positive (Negative) Gap)**.** *The gap $G(h, \boldsymbol{x}, \mathcal{B})$ is positive (resp. negative) according to h if $G(h, \boldsymbol{x}, \mathcal{B}) > 0$ (resp. $G(h, \boldsymbol{x}) < 0$). Formally, the statements take the form $s(h, \boldsymbol{x}, \mathcal{B}) = \mathbb{1}[G(h, \boldsymbol{x}, \mathcal{B}) > 0]$ and $s(h, \boldsymbol{x}, \mathcal{B}) = \mathbb{1}[G(h, \boldsymbol{x}, \mathcal{B}) < 0]$.*

Before running SHAP or EG, it is primordial to understand the sign of the gap as it is the basis behind the contrastive question we attempt to answer. There may exist instances $\boldsymbol{x}^{(i)}$ in the data where there is no consensus on the sign of the gap. Therefore, we let

$$\mathrm{SG}(\epsilon, \mathcal{B}) := \Big\{ i \in [N] : \forall h_1, h_2 \in \mathcal{R}(\mathcal{H}, \epsilon) \ \ \mathrm{sign}[G(h_1, \boldsymbol{x}^{(i)}, \mathcal{B})] = \mathrm{sign}[G(h_2, \boldsymbol{x}^{(i)}, \mathcal{B})] \Big\}, \tag{8.3}$$

be the sets of data instances on which a contrastive question makes sense. If two models disagree on the sign of the Gap, then it is useless to run SHAP or EG on them since these techniques would not end up answering the same contrastive question. If a contrastive question has been formulated without ambiguity, we can run SHAP or EG and analyze the local feature attributions.

Definition 8.2.3 (Positive (Negative) Attribution)**.** *Feature j has positive (resp. negative) attribution according to h if $\phi_j(h, \boldsymbol{x}, \mathcal{B}) > 0$ (resp. $\phi_j(h, \boldsymbol{x}, \mathcal{B}) < 0$). More formally, the statements are $s(h, \boldsymbol{x}, \mathcal{B}) = \mathbb{1}[\phi_j(h, \boldsymbol{x}, \mathcal{B}) > 0]$ and $s(h, \boldsymbol{x}, \mathcal{B}) = \mathbb{1}[\phi_j(h, \boldsymbol{x}, \mathcal{B}) < 0]$.*

We can now define the sets

$$\mathrm{SA}(\epsilon, \boldsymbol{x}, \mathcal{B}) := \Big\{ j \in [d] : \forall h_1, h_2 \in \mathcal{R}(\mathcal{H}, \epsilon) \ \ \mathrm{sign}[\phi_j(h_1, \boldsymbol{x}, \mathcal{B})] = \mathrm{sign}[\phi_j(h_2, \boldsymbol{x}, \mathcal{B})] \Big\}, \quad (8.4)$$

which store the features whose attribution has a consistent sign across all good models. After identifying the sign of the local feature attributions, it makes sense to order them according to their magnitude.

Definition 8.2.4 (Local Relative Importance)**.** *Feature i is locally less important than j (or equivalently j is locally more important than i) according to h if $|\phi_i(h, \boldsymbol{x}, \mathcal{B})| \leq |\phi_j(h, \boldsymbol{x}, \mathcal{B})|$. Formally, the statements take the form $s(h, \boldsymbol{x}, \mathcal{B}) := \mathbb{1}[|\phi_i(h, \boldsymbol{x}, \mathcal{B})| \leq |\phi_j(h, \boldsymbol{x}, \mathcal{B})|]$.*

Model consensus on local relative importance leads to a partial order $\preceq_{\epsilon, \boldsymbol{x}, \mathcal{B}}$ on $\mathrm{SA}(\epsilon, \boldsymbol{x}, \mathcal{B})$ defined by

$$i \preceq_{\epsilon, \boldsymbol{x}, \mathcal{B}} j \iff \forall\, h \in \mathcal{R}(\mathcal{H}, \epsilon) \quad |\phi_i(h, \boldsymbol{x}, \mathcal{B})| \leq |\phi_j(h, \boldsymbol{x}, \mathcal{B})|\,, \quad (8.5)$$

$\forall i, j \in \mathrm{SA}(\epsilon, \boldsymbol{x}, \mathcal{B})$. By requiring a perfect consensus on the Rashomon Set, we guarantee that the order relations will be transitive. Partial orders differ from the common total orders by allowing some pairs of features to be incomparable when there exist two models with conflicting evidence on relative importance.

Recall that asserting the consensus on a statement over the Rashomon Set (*i.e.* verifying that $\forall h \in \mathcal{R}(\mathcal{H}, \epsilon),\ s(h, \boldsymbol{x}, \mathcal{B}) = 1$) can require checking that uncountably many hypotheses h satisfy that statement. Fortunately, for the specific statements that are of interest to us, this can be rephrased as an optimization problem.

Definition 8.2.5 (Local Feature Attribution Consensus)**.** *Given a tolerance level $\epsilon > 0$, a Rashomon Set $\mathcal{R}(\mathcal{H}, \epsilon)$, and a local feature attribution $\boldsymbol{\phi} : \mathcal{H} \times \mathcal{X} \times \mathcal{P}(\mathcal{X}) \to \mathbb{R}^d$, consensus on statements are asserted via the following optimization problems.*

1. ***Positive (Negative) Gap :*** *There is consensus that the gap $G(h, \boldsymbol{x}, \mathcal{B})$ is positive (resp. negative) if $\inf_{h \in \mathcal{R}(\mathcal{H}, \epsilon)} G(h, \boldsymbol{x}, \mathcal{B}) > 0$ (resp. $\sup_{h \in \mathcal{R}(\mathcal{H}, \epsilon)} G(h, \boldsymbol{x}, \mathcal{B}) < 0$).*
2. ***Positive (Negative) Attribution :*** *There is consensus that feature j has a positive (resp. negative) attribution if $\inf_{h \in \mathcal{R}(\mathcal{H}, \epsilon)} \phi_j(h, \boldsymbol{x}, \mathcal{B}) > 0$ (resp. $\sup_{h \in \mathcal{R}(\mathcal{H}, \epsilon)} \phi_j(h, \boldsymbol{x}, \mathcal{B}) < 0$).*

3. ***Local Relative Importance :*** *Let there be a consensus that the attribution of features i and j have signs s_i and s_j. Under this assumption, the local feature importance becomes $|\phi_i(h, \boldsymbol{x}, \mathcal{B})| = s_i \phi_i(h, \boldsymbol{x}, \mathcal{B})$ for any $h \in \mathcal{R}(\mathcal{H}, \epsilon)$, and similarly for feature j. Consequently, there is a consensus that i is locally less important than j if*

$$\sup_{h \in \mathcal{R}(\mathcal{H}, \epsilon)} s_i \phi_i(h, \boldsymbol{x}, \mathcal{B}) - s_j \phi_j(h, \boldsymbol{x}, \mathcal{B}) \leq 0.$$

These optimization problems may potentially be intractable depending on the hypothesis set $\mathcal{H}$ and loss functions ℓ. Nonetheless, we will see that they can be solved exactly and efficiently for Parametric Additive Regression, Kernel Ridge Regression, and Random Forests.

8.2.3 Global Feature Importance

We can also consider global model statements $s : \mathcal{H} \times \mathcal{P}(\mathcal{X}) \to \{0, 1\}$, which are no longer specific to any input $\boldsymbol{x}$, and assert a consensus over them. When interpreting models globally, there is no need to define the notions of Gap or even sign of the attribution. Indeed, since global feature importance are already positive, we only need to study statements of relative importance.

Definition 8.2.6 (Global Relative Importance)**.** *We say that feature i is globally less important than j (or equivalently, j is globally more important than i) according to h if $\Phi_i(h, \mathcal{B}) \leq \Phi_j(h, \mathcal{B})$. Formally, the statements take the form $s(h, \mathcal{B}) := \mathbb{1}[\Phi_i(h, \mathcal{B}) \leq \Phi_j(h, \mathcal{B})]$.*

Model consensus on global relative importance defines a partial order $\preceq_{\epsilon, \mathcal{B}}$ on $[d]$:

$$i \preceq_{\epsilon, \mathcal{B}} j \iff \forall\, h \in \mathcal{R}(\mathcal{H}, \epsilon) \quad \Phi_i(h, \mathcal{B}) \leq \Phi_j(h, \mathcal{B}). \tag{8.6}$$

As with local feature attributions, consensus assertion over the Rashomon Set can be rephrased as an optimization problem.

Definition 8.2.7 (Global Feature Importance Consensus)**.** *Given a tolerance level $\epsilon > 0$, a Rashomon Set $\mathcal{R}(\mathcal{H}, \epsilon)$, and a Global Feature Importance $\boldsymbol{\Phi} : \mathcal{H} \times \mathcal{P}(\mathcal{X}) \to \mathbb{R}^d$, there is a consensus that i is globally less important than j if and only if*

$$\sup_{h \in \mathcal{R}(\mathcal{H}, \epsilon)} \Phi_i(h, \mathcal{B}) - \Phi_j(h, \mathcal{B}) \leq 0.$$

8.2.4 Relation To Prior Work

Prior methods for characterizing the effect of model underspecification on local feature attributions have mainly focused on explaining an ensemble of models $E = \{h_k\}_{k=1}^{M}$ trained with the same stochastic learning algorithm $h_k \sim \mathcal{A}(S)$ [Schulz et al., 2021, Shaikhina et al., 2021]. We go a step further by studying the feature attributions of all models in the Rashomon Set. For this reason, it may not be immediately clear how our method compares to prior work. The following proposition shows that what we propose is a more conservative alternative to both existing methods.

Proposition 8.2.1. *Let $\boldsymbol{\phi}(\cdot, \boldsymbol{x}, \mathcal{B})$ be a linear local feature attribution functional, and $E = \{h_k\}_{k=1}^{M}$ be an ensemble of M models from $\mathcal{H}$ trained with the same stochastic learning algorithm $h_k \sim \mathcal{A}(S)$. Said local feature attribution and ensemble will be employed in the methods of [Schulz et al., 2021, Shaikhina et al., 2021]. Moreover, let $\epsilon \geq \max\{\widehat{\mathcal{L}}_S(h_k)\}_{k=1}^{M}$ be an error tolerance, and let $\preceq_{\epsilon,\boldsymbol{x},\mathcal{B}}$ be the consensus order relation on $SA(\epsilon, \boldsymbol{x}, \mathcal{B})$ (cf. Equation 8.5). If the relation $i \preceq_{\epsilon,\boldsymbol{x},\mathcal{B}} j$ holds, we have that i is locally less important than j in the two total orders of prior work [Schulz et al., 2021, Shaikhina et al., 2021]. The proof is presented in Appendix E.1.2.*

This proposition is key as it implies that our framework will not provide users with statements that are not supported by existing approaches. In a way, all we do is abstain from making statements whose uncertainty is highest.

8.2.5 Recommendations for Error Tolerance

It remains to address the specification of the error tolerance ϵ. This is a critical choice because the tolerance controls the size of the Rashomon Set and therefore the number of statements on which consensus is attained. Assuming that the empirical loss minimizer h_S of Equation 2.4 is unique, setting ϵ to its minimum value will explain a single model h_S and lead to total orders of local/global feature importance. As ϵ increases, the Rashomon Set will inflate and contradicting explanations will lead to partial orders of feature importance. The number of statements present in these partial orders will diminish and eventually become null for a sufficiently high ϵ. Thus, varying the error tolerance influences *how many* statements about the empirical loss minimizer we abstain from making.

But why would we ever want to abstain from making certain statements supported by h_S? Isn't it the model that is going to be deployed anyway? The risk is that some explanations of h_S might be contradicted by another model with *slightly worst empirical loss*. When this occurs, we argue that the explanations of h_S are not trustworthy and we advocate for

abstention. Determining the right notion of *slightly worst empirical loss* is a difficult problem. Here we suggest two approaches 1) one based on statistical guarantees 2) a heuristic based on relative error increases.

Capture Bounds Assume we can find $\epsilon_{\max}$ such that any model with a larger empirical loss can be shown to be suboptimal in terms of population loss $\mathcal{L}_{\mathcal{D}}(h)$. More precisely, with probability $1-\delta$, $\widehat{\mathcal{L}}_S(h) > \epsilon_{\max}$ implies that $\mathcal{L}_{\mathcal{D}}(h) > \mathcal{L}_{\mathcal{D}}(h^\star)$. Then it is not relevant to set $\epsilon > \epsilon_{\max}$ since the Rashomon Set would include models that are likely suboptimal. Assuming the solution $h^\star$ of Equation 2.1 is unique, this $\epsilon_{\max}$ is the smallest value that respects

$$\mathop{\mathbb{P}}_{S\sim\mathcal{D}^N}[\widehat{\mathcal{L}}_S(h^\star) \leq \epsilon_{\max}] = \mathop{\mathbb{P}}_{S\sim\mathcal{D}^N}[h^\star \in \mathcal{R}(\mathcal{H}, \epsilon_{\max})] > 1-\delta. \tag{8.7}$$

We shall refer to such statistical guarantees as "Capture Bounds" since they guarantee that the Rashomon Set will "capture" the best-in-class model. By setting $\epsilon = \epsilon_{\max}$, with high probability, any statement on which there is a consensus on the Rashomon Set will also hold for the unknown $h^\star$. That is, we explain the best model without knowing which one it is. We now present three capture bounds

First, if $\mathcal{H}$ is finite and small (*e.g.* $|\mathcal{H}| \leq 100$), we recommend using Model Set Selection [Kissel and Mentch, 2021]. We define the subset $E \subseteq \mathcal{H}$ of all models that are not significantly worse than the empirical risk minimizer h_S according to a statistical test *e.g.* paired Student-t tests with significance $1-\delta$. Setting $\epsilon_{\max} = \max\{\widehat{\mathcal{L}}_S(h)\}_{h\in E}$ guarantees that Equation 8.7 holds. This capture bound was previously applied to the ensemble of five MLPs from Section 8.1.

Second, if strong assumptions can be made on how the target was generated, then the following capture bound can be used.

Proposition 8.2.2. *Under the assumption that the data were generated by the optimal model $h^\star$ plus iid zero-mean Gaussian noise*

$$y = h^\star(\boldsymbol{x}) + \Delta, \quad \textit{where} \quad \Delta \sim \mathcal{N}(0, \sigma^2), \tag{8.8}$$

and using the squared loss $\ell(y', y) = (y'-y)^2$, we have that

$$\mathop{\mathbb{P}}_{S\sim\mathcal{D}^N}[\widehat{\mathcal{L}}_S(h^\star) > \epsilon_{max}] = 1 - F_{\chi^2_N}\left(\frac{N}{\sigma^2}\epsilon_{max}\right), \tag{8.9}$$

where $F_{\chi^2_N}$ is the CDF of a chi-2 random variable with N degrees of freedom. The proof is provided in Appendix E.1.1.

Solving $\delta := 1 - F_{\chi^2_N}(\frac{N}{\sigma^2}\epsilon_{\max})$ for $\epsilon_{\max}$ yields the desired tolerance. If the residuals Δ follow another law than Gaussian, one could replace the χ^2_N CDF by the CDF of the distribution of $1/N \sum_{i=1}^{N}(\Delta^{(i)})^2$. The assumption that the data was generated by $h^\star$ plus symmetric noise is very strong, but it is ubiquitous in Statistics and Linear Regression (See for instance [Hastie et al., 2009, Section 3.2] and [Wasserman, 2004, Section 13.5]). Therefore, we think this capture bound is *at-least* worth investigating in any regression problem.

We suggest a third capture bound when a good reference hypothesis can be chosen a priori *i.e.* before seeing the dataset S on which the empirical loss is computed.

Proposition 8.2.3. *Let ℓ be the 0-1 loss, $S \sim \mathcal{D}^N$ be a dataset, $h_{ref} \in \mathcal{H}$ be a reference model that is independent of S, and $h^\star$ be a best in-class hypothesis, for any $\epsilon' \in \mathbb{R}^+$, we have*

$$\mathop{\mathbb{P}}_{S\sim\mathcal{D}^N}[\widehat{\mathcal{L}}_S(h^\star) \geq \epsilon' + \widehat{\mathcal{L}}_S(h_{ref})] \leq \exp\left\{ - \frac{N\epsilon'^2}{2}\right\}. \tag{8.10}$$

The proof is provided in Appendix E.1.1

Solving $\delta := \exp\left\{ - \frac{N(\epsilon_{\max} - \widehat{\mathcal{L}}_S(h_{\text{ref}}))^2}{2}\right\}$ for $\epsilon_{\max}$ yields the error tolerance.

Relative Increase Heuristic Capture bounds rely on very strong assumptions and therefore cannot be used out-of-the-box for all problems. When they are inapplicable, we recommend the heuristic

$$\epsilon = (1 + \epsilon_{\text{rel}}) \times \widehat{\mathcal{L}}_S(h_S), \tag{8.11}$$

for a ϵ_{rel} typically fixed to 5% [Coker et al., 2021, Dong and Rudin, 2019], although smaller values could be used. Setting ϵ based on this heuristic does not provide any statistical guarantee. Consequently, any alternative model $h' \in \mathcal{R}(\mathcal{H}, \epsilon)$ that is highlighted by the practitioner should be compared to h_S using a paired Student-t test on fresh data. For example, if a model in the Rashomon Set is found to contradict h_S on a statement of interest, then one should assert that the test error of this alternative model is not significantly worse than that of the empirical loss minimizer.

Having introduced how to assert consensus over the additve explanations in the Rashomon Set, we can describe our quantitative measure of Underspecification Disagreements.

Underspecification Disagreement

Underspecification refers to the existence of a Rashomon Set containing infinitely many models with good empirical performance. These competing models might disagree in terms of LFA and GFI, which makes it hard to derive insights from them. To quantify these disagreements, we report the Cardinality of local partial orders

$$| \preceq_{\epsilon,\boldsymbol{x}^{(i)},\mathcal{B}} | := \left(\frac{1}{2}d(d+1)\right)^{-1} \mathbb{1}[\boldsymbol{x}^{(i)} \in \mathrm{SG}(\epsilon,\mathcal{B})] \times |\{(j,k) \in \mathrm{SA}(\epsilon,\boldsymbol{x}^{(i)},\mathcal{B})^2 : j \preceq_{\epsilon,\boldsymbol{x}^{(i)},\mathcal{B}} k\}| \tag{8.12}$$

as a function of ϵ. Cardinality is the ratio of statements on which consensus is attained to the total number of statements $\frac{1}{2}d(d+1) = \frac{1}{2}d(d-1)$ (local relative importance) $+$ d (attribution sign). It goes from 0 (when there is no consensus on the sign of the gap) to 1 (when we have total order over d features.)

How to reduce Disagreement?

Cardinalities $\{ | \preceq_{\epsilon,\boldsymbol{x}^{(i)},\mathcal{B}} | \}_{i=1}^{N}$ can be increased in two ways.

- First, correlated features (x_i, x_j) can drastically increase the Underspecification Disagreement because competing models can rely more on one feature or the other. The proposed solution is to treat correlated features as a single group. For instance, if `age` and `salary` are strongly correlated, the input can be re-formulated as

$$\boldsymbol{x} := [(\texttt{age}, \texttt{salary}), \texttt{gender}, \texttt{education-num}]^T. \tag{8.13}$$

  In that case, the replace-function (cf. Equation 3.3) would replace `age` and `salary` simultaneously. For model-specific implementations, an embedding $\boldsymbol{\xi} : \prod_{j=1}^{d} \mathcal{X}_j \to \mathbb{R}^{[d']}$ must map the feature $(\texttt{age}, \texttt{salary})$ to two different columns before it is fed to the ML model *i.e.* $h(\boldsymbol{x}) = (h^{\mathrm{ML}} \circ \boldsymbol{\xi})(\boldsymbol{x})$.

- Second, local feature attributions are always relative to a baseline $\mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{z})]$. If the Gap between $h(\boldsymbol{x})$ and the baseline is small, the explanation will likely be underspecified. The solution is to increase the Gap by changing $\mathcal{B}$.

8.3 Parametric Additive Models

8.3.1 Methodology

As discussed in Section 5.2, Parametric Additive Models are the composition of an embedding and a linear model

$$h_{\boldsymbol{\omega}}^{\text{add}} = h_{\boldsymbol{\omega}}^{\text{lin}} \circ \boldsymbol{\xi}. \tag{8.14}$$

The components of the embedding store the bases of the functions $h_j(x_j)$. Letting, $\boldsymbol{H}$ be the $N \times d'$ matrix whose i^{th} row stores the embedding $\boldsymbol{\xi}(\boldsymbol{x}^{(i)})$, the Rashomon Set with squared loss has a closed form

Definition 8.3.1 (Rashomon Set for Parametric Additive Regression)**.** *Let $\mathcal{H}$ be the set of Parametric Additive Regression models (cf Equation 8.14), ℓ be the squared loss, S be a dataset of size N, and $\boldsymbol{\omega}_S = \text{argmin}_{h\in\mathcal{H}} \widehat{\mathcal{L}}_S(h)$ be the least-square estimate. If one uses the performance threshold $\epsilon \geq \widehat{\mathcal{L}}_S(\boldsymbol{\omega}_S)$, then the Rashomon set $\mathcal{R}(\mathcal{H}, \epsilon)$ consists of all parameters $\boldsymbol{\omega}$ s.t.*

$$(\boldsymbol{\omega} - \boldsymbol{\omega}_S)^T \frac{\boldsymbol{H}^T\boldsymbol{H}}{N} (\boldsymbol{\omega} - \boldsymbol{\omega}_S) \leq \epsilon - \widehat{\mathcal{L}}_S(\boldsymbol{\omega}_S). \tag{8.15}$$

The Rashomon Set is an ellipsoid in parameter space. Moreover, if we let $\epsilon < \widehat{\mathcal{L}}_S(\boldsymbol{\omega}_S)$, then the Rashomon Set is empty.

According to Definition 8.2.5, asserting consensus on local feature attribution statements amounts to optimization problems that are linear with respect to the attributions $\boldsymbol{\phi}$. Moreover, Section 5.2 previously demonstrated that local feature attributions $\boldsymbol{\phi}$ are linear functions of the weights $\boldsymbol{\omega}$

$$\phi_j^{\text{LFA}}(h_{\boldsymbol{\omega}}^{\text{add}}, \boldsymbol{x}, \mathcal{B}) = \sum_{k \in \mathcal{I}_{\boldsymbol{\xi}}^{-1}(\{j\})} \omega_k \Big(\xi_k(\boldsymbol{x}) - \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[\xi_k(\boldsymbol{z})] \Big) := \boldsymbol{\omega}_j^T \, \overline{\boldsymbol{\xi}}_j(\boldsymbol{x}), \tag{8.16}$$

where $\boldsymbol{\omega}_j := (\omega_k)_{k\in\mathcal{I}_{\boldsymbol{\xi}}^{-1}(\{j\})}$ regroups all weights modeling the dependency w.r.t feature j. Similarly, we defined $\overline{\boldsymbol{\xi}}_j(\boldsymbol{x})$. Therefore, asserting a consensus on the Rashomon Set of Parametric Additive models requires maximizing/minimizing a linear function over an ellipsoid

$$\begin{aligned} \min_{\boldsymbol{\omega}}/\max_{\boldsymbol{\omega}} \quad & \boldsymbol{a}^T\boldsymbol{\omega} \\ \text{with} \quad & (\boldsymbol{\omega} - \boldsymbol{\omega}_S)^T \boldsymbol{A} (\boldsymbol{\omega} - \boldsymbol{\omega}_S) \leq \epsilon - \widehat{\mathcal{L}}_S(\boldsymbol{\omega}_S), \end{aligned} \tag{8.17}$$

with $\boldsymbol{A} := \frac{\boldsymbol{H}^T\boldsymbol{H}}{N}$ and assuming $\epsilon \geq \widehat{\mathcal{L}}_S(\boldsymbol{\omega}_S)$. The choice of objective depends on the type of statement and the instance $\boldsymbol{x}^{(i)}$ being explained. For Positive (Negative) Gap : $\boldsymbol{a}^T\boldsymbol{\omega} := \overline{\boldsymbol{\xi}}(\boldsymbol{x}^{(i)})^T\boldsymbol{\omega}$. For Positive (Negative) Attributions of feature j : $\boldsymbol{a}^T\boldsymbol{\omega} := \overline{\boldsymbol{\xi}}_j(\boldsymbol{x}^{(i)})^T\boldsymbol{\omega}_j$. For Local

Relative Importance of feature j and k : $\boldsymbol{a}^T\boldsymbol{\omega} := s_j\overline{\boldsymbol{\xi}}_j(\boldsymbol{x}^{(i)})^T\boldsymbol{\omega}_j - s_k\overline{\boldsymbol{\xi}}_k(\boldsymbol{x}^{(i)})^T\boldsymbol{\omega}_k$. The optimum values of Equation 8.17 can be expressed using the Cholesky decomposition $\boldsymbol{A} = \boldsymbol{C}\boldsymbol{C}^T$

$$\pm \sqrt{\epsilon - \widehat{\mathcal{L}}_S(\boldsymbol{\omega}_S)}\ \|\boldsymbol{a}'\| + \boldsymbol{a}^T\boldsymbol{\omega}_S, \tag{8.18}$$

where $\boldsymbol{a}' = \boldsymbol{C}^{-1}\boldsymbol{a}$. See Appendix E.2.1 for more details. Equation 8.18 reveals that the minimum and maximum values of any linear functional evaluated on the Rashomon Set are a deviation of $\sqrt{\epsilon - \widehat{\mathcal{L}}_S(\boldsymbol{\omega}_S)}\|\boldsymbol{a}'\|$ from $\boldsymbol{a}^T\boldsymbol{\omega}_S$ the value of the functional evaluated on the least-square. Since the deviation is an explicit function of the tolerance ϵ, a consensus on local feature attribution statements can be asserted at any tolerance level.

Regarding Global Feature Importance measures, Section 5.2 previously proved that they take a quadratic form $\Phi_j^{\text{GFI},[2]}(h_{\boldsymbol{\omega}}^{\text{add}}, \mathcal{B}) = \boldsymbol{\omega}_j^T\boldsymbol{B}_j\boldsymbol{\omega}_j$. As a result, asserting a consensus on Global Relative Importance statements in the Rashomon Set of Additive Regression (*i.e.* solving Definition 8.2.7) requires optimizing a quadratic form over an ellipsoid

$$\begin{aligned} \min_{\boldsymbol{\omega}}\!/\!\max_{\boldsymbol{\omega}} \quad & \boldsymbol{\omega}_i^T\boldsymbol{B}_i\boldsymbol{\omega}_i - \boldsymbol{\omega}_j^T\boldsymbol{B}_j\boldsymbol{\omega}_j \\ \text{with} \quad & (\boldsymbol{\omega} - \boldsymbol{\omega}_S)^T\boldsymbol{A}(\boldsymbol{\omega} - \boldsymbol{\omega}_S) \leq \epsilon - \widehat{\mathcal{L}}_S(\boldsymbol{\omega}_S), \end{aligned} \tag{8.19}$$

which is known as the Trust-Region-Subproblem (TRS). Impressively, by Corollary 7.2.2 of [Conn et al., 2000, Section 7.2] this problem has necessary optimality conditions for the global optimum, even when the quadratic form is non-convex. We describe our TRS solver in Appendix E.2.1.

The following applies the proposed methodology on house price prediction. Notably, we demonstrate the benefits of grouping correlated features and increasing the Gap by changing $\mathcal{B}$.

8.3.2 House Price Prediction

The Kaggle-Houses[1] dataset consists of predicting the logarithm of the selling price of 2919 houses based on 79 numerical and categorical features. The training set S contains the first 1460 houses which are labeled, while the test set regroups the remaining 1459 houses whose selling prices are hidden by Kaggle. The only way to measure test performance is to submit predictions on the Kaggle Website.

For simplicity, we only selected numerical features and removed time-related features since we are only interested in the physical properties of the houses. Moreover, features that were per-

[1] `https://www.kaggle.com/c/house-prices-advanced-regression-techniques`

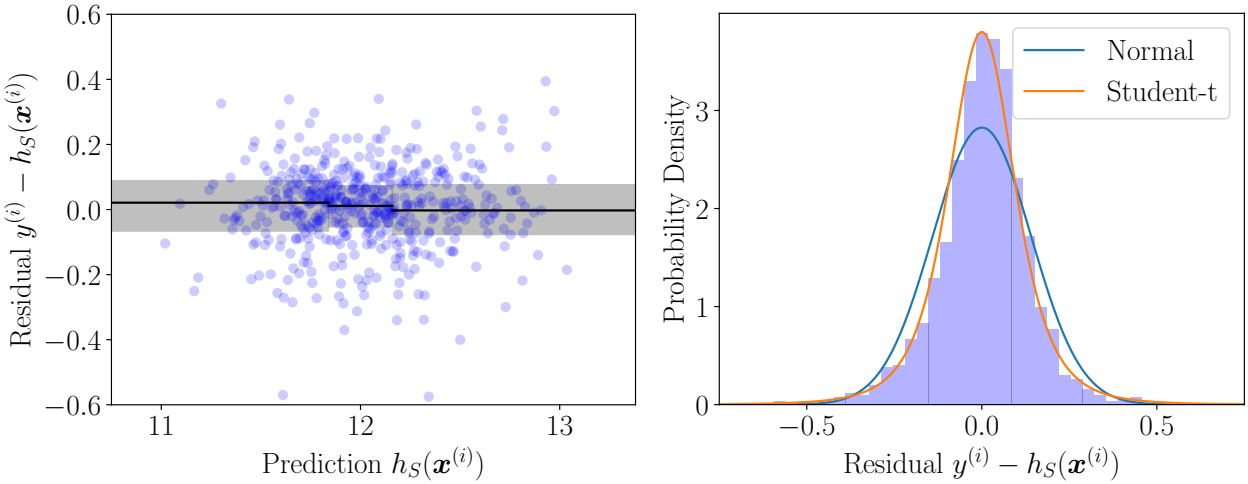


Figure 8.2 Residuals Analysis of h_S. (Left) Residual as a function of the prediction to assess homogeneity. The horizontal lines represent the 25$^{\text{th}}$, 50$^{\text{th}}$, and 75$^{\text{th}}$ percentiles for three different prediction bins. (Right) Histogram of the residuals and fitted densities.

fectly collinear with others were ignored since they would render the matrix $\boldsymbol{H}^T\boldsymbol{H}$ singular. We were left with 19 numerical features which were non-redundant, although some had a very high Spearman correlation : `GarageArea/GarageCars`, `BsmtPercFin/BsmtFullBath`, and `BedroomAbvGrd/TotRmsAbvGrd`. We decided to keep correlated features to see how they impact model underspecification.

Additive Regression requires deciding which h_j to parametrize with spline bases and which to parametrize as linear functions of the input $h_j(x_j) = \omega_j x_j$. For each feature, we fitted the target with a depth-3 decision tree using only that feature as input and selected the k features with the lowest RMSE for spline parametrization. We tuned the hyperparameter k, the polynomial degree of the splines, the number of knots, and their positions via five-fold cross-validation. The resulting least-square models had a train RMSE of 0.141. As references for test performance, predicting the average training set target yields a RMSE of 0.426 on Kaggle while Gradient Boosting[2] leads to an error of 0.127. In the case of Additive Regression, we got a test error of 0.150.

To quantify the under-specification of our hypothesis class, we computed the Rashomon Set containing all good models on the training set. We could not use the test set since labels are not available. To fix a reasonable value of tolerance ϵ, we investigated whether the assumptions behind the capture bound of Proposition 8.2.2 were reasonable on this dataset. That is, could the labels have been provided by the best-in-class $h^\star$ plus iid noise Δ? We

[2] `https://www.kaggle.com/code/eesuck/xgboost-regressor`

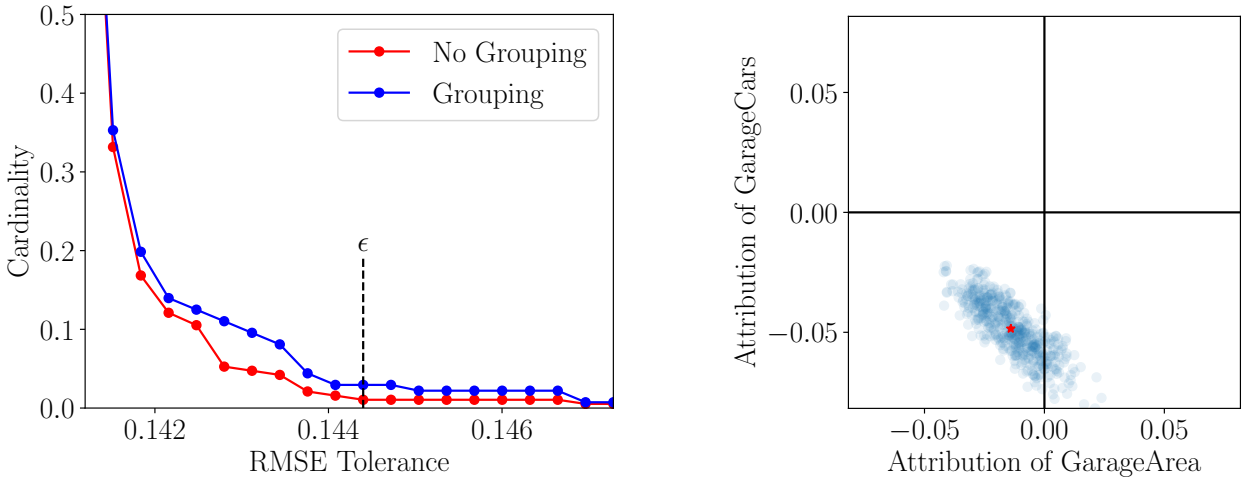


Figure 8.3 (Left) Median partial order Cardinalities as a function of the tolerance on training RMSE. The two curves represent whether we group correlated features together. (Right) Local Feature Attributions of models sampled from the Rashomon Set boundary.A trade-off between local attributions of correlated features is apparent.

first assumed that h_S and $h^\star$ make similar enough predictions on training data to view the residuals $\{y^{(i)} - h_S(\boldsymbol{x}^{(i)})\}_{i=1}^N$ as noise samples $\{\Delta^{(i)}\}_{i=1}^N$. Figure 8.2 (Left) supports that the residuals are homogeneous but Figure 8.2 (Right) reveals they are not Gaussian and are better modeled with a Student-t. Supported by these observations, we modeled the noise Δ with a Student-t distribution fitted on the residuals. Afterward, we approximated the distribution of $\widehat{\mathcal{L}}_S(h^\star) = \frac{1}{N}\sum_{i=1}(\Delta^{(i)})^2$ with the empirical distribution resulting from sampling $\{\Delta^{(i)}\}_{i=1}^N \sim t_\nu^N$ a total of 2×10^5 times. Taking the 95^{th} percentile of this empirical distribution yielded the tolerance $\epsilon_{\text{max}} = 0.1444$. Under our assumptions, by fixing $\epsilon = \epsilon_{\text{max}} = 0.1444$ we have an approximate 95% chance that the Rashomon Set will include the best-in-class model.

Local feature attributions were computed on all houses in the training set using Equation 8.16. The background employed $\mathcal{B}$ was the empirical distribution over the training data. We report the partial order cardinalities (cf. Equation 8.12) at several tolerance values, see the red curve in Figure 8.3(Left). Note that the cardinalities are stable with respect to small perturbations of ϵ. However, the cardinalities are rather small, which we suspect is partly due to feature correlations. To test this hypothesis, we sampled models from the Rashomon Set boundary and compared their local attributions for correlated features, see Figure 8.3(Right). A trade-off is apparent: the more models rely on one feature, the less they rely on the other.

To deal with this under-specification, we *group* correlated features x_i and x_j into a a single feature (x_i, x_j). This means that the replace-function (cf. Equation 3.3) will perturbate both features simultaneously. We group `GarageArea/GarageCars` into `Garage`,

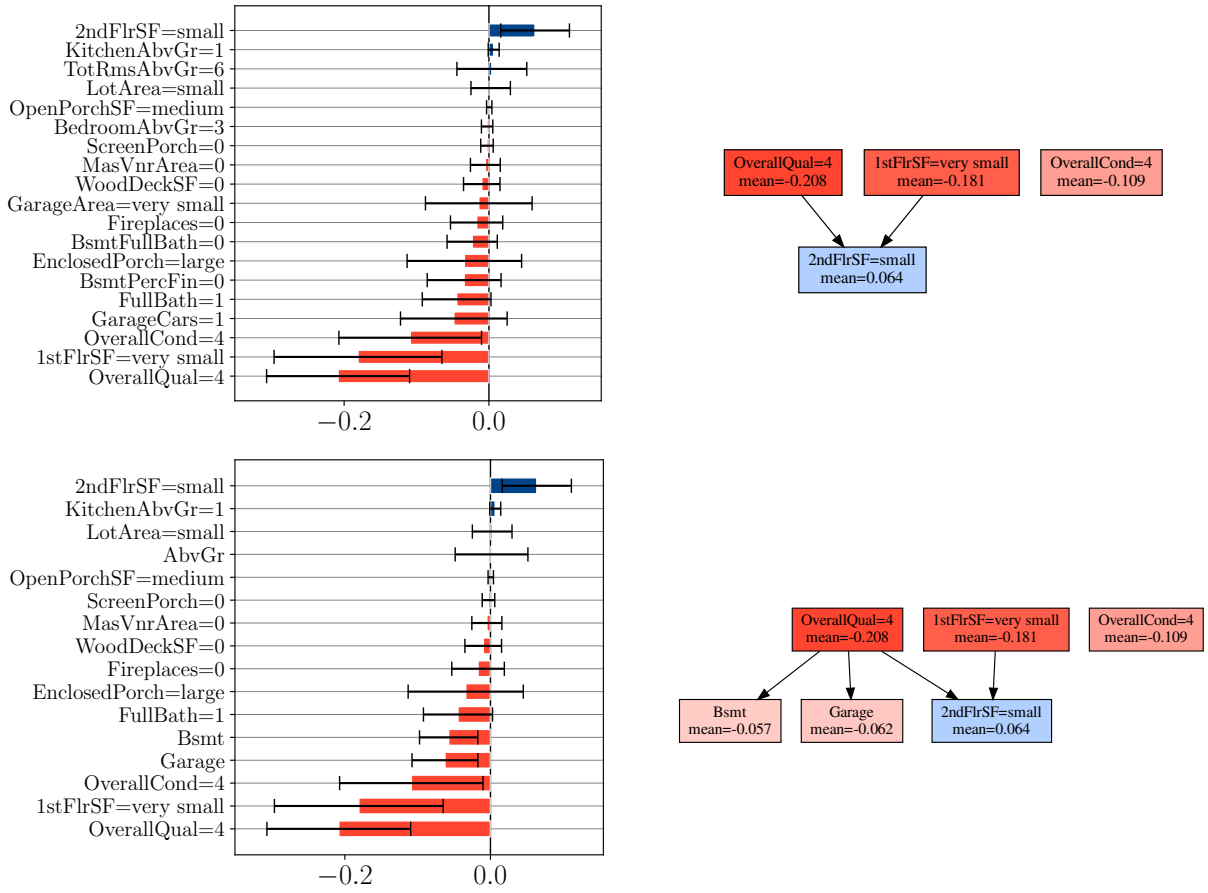


Figure 8.4 Local feature attributions of a house with a below-average price. (Top) Without grouping. (Bottom) With grouping.

`BsmtPercFin/BsmtFullBath` into `Bsmt` and `BedroomAbvGrd/TotRmsAbvGrd` into `AbvGrd`. In doing so, one obtains partial orders with higher cardinalities as evidenced by the blue curve in Figure 8.3(Left), suggesting that grouping correlated features can reduce explanation under-specification. In the sequel, we will present local/global feature attributions with and without grouping.

We explained the predictions on the house with the fifth-smallest selling price: 40K USD. Said predictions ranged from 70K to 100K in the Rashomon Sets and there was a consensus that the gap was negative. Figure 8.4 shows the local feature attribution on this instance and the partial orders that summarize all the statements good models agree on. Observe that features `OverallQual=4` (quality of materials and finish of the house from a scale of 1 to 10) and `1stFlrSF=very small` have maximal importance when explaining the drop in price relative to the mean. These statements are robust to the choice of model within the Rashomon Set. `OverallQual=4` also has maximal importance but, because it is incomparable to any

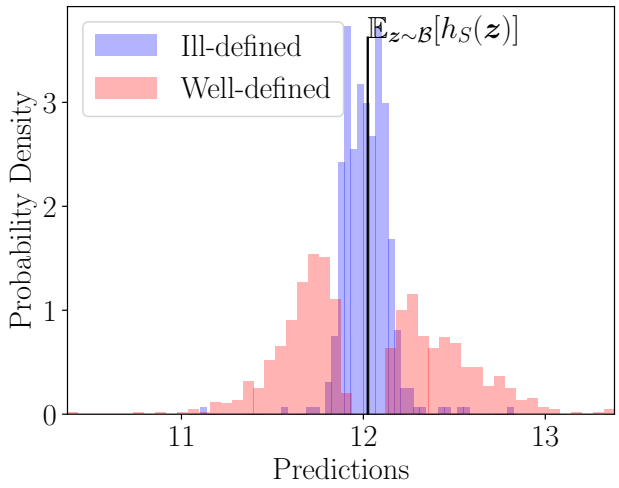


Figure 8.5 Distributions of predictions for houses with ill-defined and well-defined gaps across the Rashomon Set of Kaggle-Houses. The background $\mathcal{B}$ is the empirical distribution over the whole training data.

other feature, we find it safer to simply ignore it. Moreover, we note that there are no garage-related and basement-related features in the Hasse diagram without Grouping. As illustrated in Figure 8.4 (Top-Left), the attributions of highly correlated features such as `GarageArea/GarageCars` and `BsmtPercFin/BsmtFullBath` do not have a consistent sign. This is because competing models can rely on one feature or the other, which prohibits a consensus on which feature leads to a decrease in selling price. By grouping correlated features, the attributions of the groups `Garage` and `Bsmt` become consistently negative, see Figure 8.4 (Bottom-Left).

Finally, at tolerance $\epsilon = 0.1444$, the sign of the gaps is well-defined for 68% of the houses. For about one-third of houses, there exists two models $h_1, h_2 \in \mathcal{R}(\mathcal{H}, \epsilon)$ which assign gaps $G(h_1, \boldsymbol{x}^{(i)}, \mathcal{B}) < 0$ and $G(h_2, \boldsymbol{x}^{(i)}, \mathcal{B}) > 0$. When this occurs, it does not make sense to compute local feature attributions at $\boldsymbol{x}^{(i)}$ since the different models end up answering different contrastive questions. What can we do about those instances? As a reminder, the background $\mathcal{B}$ employed was the empirical distribution over the training data. Figure 8.5 shows the distributions of predictions for instance whose gap is well-defined or ill-defined across the Rashomon Set. Instances whose gap does not have a consistent sign tend to have predictions $h_S(\boldsymbol{x}^{(i)})$ near the baseline $\mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[h_S(\boldsymbol{z})]$ so that the Gap $G(h_S, \boldsymbol{x}^{(i)}, \mathcal{B})$ is very small. This could explain why models with similar performance can assign different signs to the gap. Importantly, model underspecification warns us that the contrastive question is not well-posed on these houses and it would be better to use another background $\mathcal{B}'$ when explaining them. By redefining $\mathcal{B}'$ to be the empirical distribution over all houses with a predicted

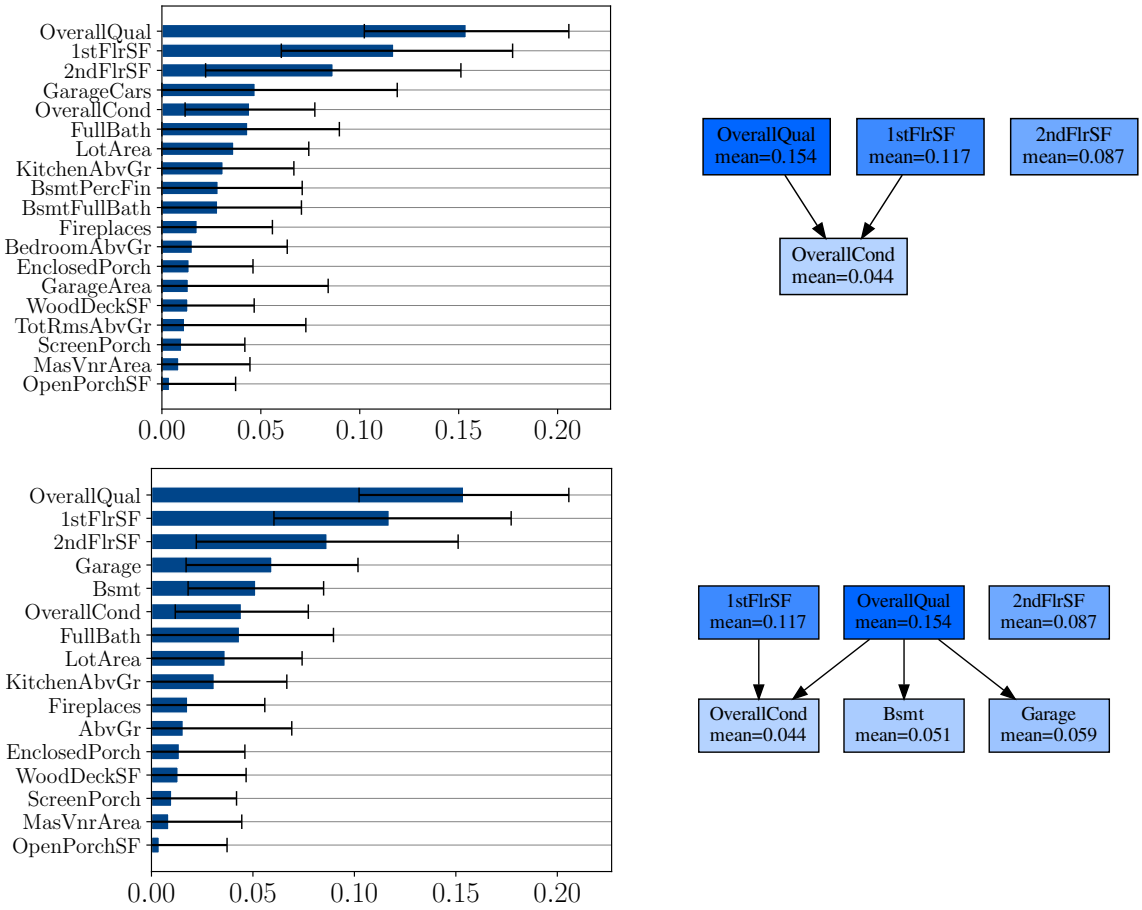


Figure 8.6 Global Feature Importance of the Kaggle-Houses dataset. (Top) Without grouping. (Bottom) With grouping.

price below the first quartile, the prediction gaps increased and 97% of the houses that were previously unexplainable suddenly became explainable.

We end this use-case by presenting GFI in Figure 8.6. For simplicity, we only include in the Hasse diagrams the features whose global importance is non-null across the whole Rashomon set. Such features appear to be necessary in the sense that every model in the Rashomon Set relies on them. As seen previously, the partial order without Grouping does not contain features related to the basement and the garage. We believe that this can be again attributed to strong feature correlations. By grouping correlated features, we discover that the joint effects of `Garage` and `Bsmt` are important for all good models.

8.4 Kernel Methods

8.4.1 Methodology

As a quick reminder, kernel models take the form

$$h_{\boldsymbol{\alpha}}^{\text{kernel}}(\boldsymbol{x}) = \sum_{\ell=1}^{R} \alpha_\ell k(\boldsymbol{x}, \boldsymbol{r}^{(\ell)}), \tag{8.20}$$

where $k(\cdot, \cdot)$ is a PDS kernel and $D = \{\boldsymbol{r}^{[\ell]}\}_{\ell=1}^{R}$ is a dictionary of reference inputs. The parameters $\boldsymbol{\alpha}$ are trained by minimizing the regularized loss

$$\boldsymbol{\alpha}_D = \underset{\boldsymbol{\alpha}\in\mathbb{R}^R}{\text{argmin}}\, \widehat{\mathcal{L}}_D(h_{\boldsymbol{\alpha}}^{\text{kernel}}) + \lambda \boldsymbol{\alpha}^T \boldsymbol{K} \boldsymbol{\alpha}, \tag{8.21}$$

where $\boldsymbol{K}$ is a $R \times R$ matrix containing elements $K_{ij} = k(\boldsymbol{r}^{[i]}, \boldsymbol{r}^{[j]})$. Since kernel methods are heavily regularized, it makes sense to define the Rashomon Set

$$\mathcal{R}(\mathcal{H}, \epsilon) := \Big\{ h_{\boldsymbol{\alpha}}^{\text{kernel}} \in \mathcal{H} : \widehat{\mathcal{L}}_D(h_{\boldsymbol{\alpha}}^{\text{kernel}}) + \lambda \boldsymbol{\alpha}^T \boldsymbol{K} \boldsymbol{\alpha} \leq \epsilon \Big\}, \tag{8.22}$$

that now accepts worst predictors if they are sufficiently smooth. When the squared loss is employed, the Rashomon Set has a closed-form.

Definition 8.4.1 (Rashomon Set for Kernel Ridge Regression)**.** *Let $\mathcal{H}$ be the hypothesis space of kernel k and dictionary D, ℓ be the squared loss, $\lambda > 0$ be a regularization hyper-parameter, and $\boldsymbol{\alpha}_D$ be the solution of the regularized least-square. If one uses the performance threshold $\epsilon \geq \widehat{\mathcal{L}}_D(h_{\boldsymbol{\alpha}_D}^{kernel}) + \lambda \boldsymbol{\alpha}_D^T \boldsymbol{K} \boldsymbol{\alpha}_D$, then the Rashomon set $\mathcal{R}(\mathcal{H}, \epsilon)$ regroups all models $h_{\boldsymbol{\alpha}}^{kernel}$ s.t.*

$$(\boldsymbol{\alpha} - \boldsymbol{\alpha}_D)^T (\boldsymbol{K}/R + \lambda \boldsymbol{I}) \boldsymbol{K} (\boldsymbol{\alpha} - \boldsymbol{\alpha}_D) \leq \epsilon - \widehat{\mathcal{L}}_D(h_{\boldsymbol{\alpha}_D}^{kernel}) - \lambda \boldsymbol{\alpha}_D^T \boldsymbol{K} \boldsymbol{\alpha}_D. \tag{8.23}$$

The Rashomon Set is an ellipsoid in $\mathbb{R}^R$.

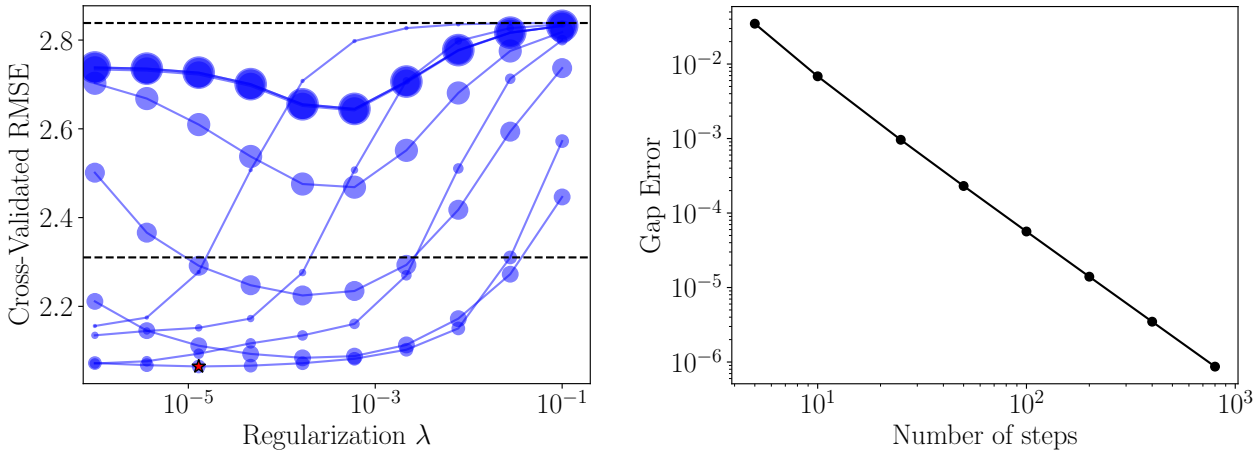
The proof is mutatis mutandis like the proof for Ridge Regression in [Semenova et al., 2022] but with Kernel Ridge instead.

It was demonstrated in Section 5.3 that local feature attributions of kernel methods are linear functions of $\boldsymbol{\alpha}$. similarly, global feature importance are quadratic functions of $\boldsymbol{\alpha}$. Now, since the Rashomon Set is an ellipsoid, asserting consensus involves the same optimization problems as Section 8.3.1.

8.4.2 Criminal Recidivism Prediction

COMPAS is a proprietary model currently employed in the United States to predict the risk of recidivism from individuals that were recently arrested. These risks are encoded as integers going from 1 (low-risk) to 10 (high-risk). The use of this automated tool in the justice system is driven by the promise of providing objective information to judges based on empirical data, thus circumventing human biases. Still, the strong reliance of models on historical data means they can reproduce/perpetuate past injustices. To test such claims, ProPublica has collected several thousands of COMPAS scores from 2013-2014 in the Florida Broward County [Larson et al., 2016]. In the resulting article, several pairs of Caucasian and African-American defendants are presented along with their COMPAS scores, the former often being lower than the latter despite the Caucasian defendant having a longer criminal history. These examples of pairs along with the subsequent analysis from the article seem to imply that the proprietary model depends on race. However, the methodology of ProPublica has been heavily criticized alongside the claim that COMPAS depends explicitly on race [Rudin et al., 2018]. Hence, there may exist alternative explanations besides race for the discrepancy between scores, so it is pertinent to study the local/global additive explanations over the whole Rashomon Set of reasonable models when predicting COMPAS scores.

To analyze the dependencies of risk scores on the various features, we reproduced the experiments of [Fisher et al., 2019] where a Kernel Ridge Regression model was fitted directly on the 1-10 scores from the ProPublica dataset. The same features were employed while adding two additional ones related to juvenile misdemeanors and felonies. The dataset was split in train and test sets with ratios of 0.8 and 0.2. The training samples were used to define the dictionary of reference inputs D. We utilized the polynomial kernel $k(\boldsymbol{x}, \boldsymbol{x'}) = (\gamma\langle \boldsymbol{x}, \boldsymbol{x'}\rangle + 1)^p$ with degree $p = 3$ and the Gaussian kernel $k(\boldsymbol{x}, \boldsymbol{x'}) = \exp(-\gamma\|\boldsymbol{x} - \boldsymbol{x'}\|^2)$. The kernel scale hyper-parameter γ and the regularization factor λ were fine-tuned with 5-fold cross-validation on the training set, see the results for Gaussian Kernels in Figure 8.7 (a). Similar results were obtained with Polynomial Kernels. The test set RMSE of the final model was 2.11 for Gaussian Kernels and 2.12 for Polynomial Kernels. We note that the performances are worse than [Fisher et al., 2019] because, unlike them, we predict the recidivism risk scores and not the risk scores for *violent* recidivism, which could be easier to predict. We decided to study the recidivism risk scores instead since these are the ones that were actually discussed in the ProPublica article.



(a) Tuning of γ (dot sizes) and λ with Gaussian kernels. The top horizontal line shows the error of the predictor returning the mean, while the bottom line shows the error of a Random Forest with default hyperparameters.

(b) Convergence of the Gap Error w.r.t the number of steps in the quadrature. Second-order convergence is occurring, so augmenting the number of steps by a factor 10 reduced the error by a factor 100.

Figure 8.7

Input	Name	Score	Race	Age	Priors	Charge
$\boldsymbol{x}$	Robert Cannon	6	African-American	22	0	Misdemeanor
$\boldsymbol{z}$	James Rivelli	3	Caucasian	54	3	Felony

Table 8.2 Comparison of the COMPAS scores of two individuals.

After fitting the models, we identified a pair of Caucasian/African-American individuals who were highlighted in the ProPublica piece and applied our explainability framework on them. More specifically, we compared Robert Cannon and James Rivelli, see Table 8.2. James Rivelli is a 54-year-old Caucasian man who was arrested for shoplifting. Despite having a criminal record with three priors, he was assigned a low COMPAS score. In contrast, Robert Cannon, a 22-year-old African-American charged with petit theft, was assigned a high risk of recidivism. Letting Robert be the input of $\boldsymbol{x}$ and James be the input $\boldsymbol{z}$, the differences in COMPAS scores are also present in model predictions: $h^{\text{kernel}}_{\boldsymbol{\alpha}_D}(\boldsymbol{x}) = 4.9$ and $h^{\text{kernel}}_{\boldsymbol{\alpha}_D}(\boldsymbol{z}) = 2.5$ for Gaussian Kernels, and $h^{\text{kernel}}_{\boldsymbol{\alpha}_D}(\boldsymbol{x}) = 4.9$ and $h^{\text{kernel}}_{\boldsymbol{\alpha}_D}(\boldsymbol{z}) = 2.4$ for Polynomial Kernels. Therefore, the prediction gap $G(h^{\text{kernel}}_{\boldsymbol{\alpha}_D}, \boldsymbol{x}, \boldsymbol{z}) = h^{\text{kernel}}_{\boldsymbol{\alpha}_D}(\boldsymbol{x}) - h^{\text{kernel}}_{\boldsymbol{\alpha}_D}(\boldsymbol{z})$ is positive. Given the historical racism in the United States, it is very tempting to look at these two individuals and say that Robert Cannon is predicted to have a higher risk "because of his race". Still, there may exist a diversity of alternative explanations for this discrepancy, which we can

study by exploring the Rashomon Set of our Kernel Ridge models. The Integrated Gradient was employed using Robert as the input of interest $\boldsymbol{x}$ and James as the reference input $\boldsymbol{z}$ to obtain feature attributions. Since computing the IG feature attributions requires estimating the integrals of Equation 5.14 with quadratures, we ended up with estimates $\widehat{\boldsymbol{\phi}}^{\text{IG}}(h^{\text{kernel}}_{\boldsymbol{\alpha}_D}, \boldsymbol{x}, \boldsymbol{z})$ of the real attributions $\boldsymbol{\phi}^{\text{IG}}(h^{\text{kernel}}_{\boldsymbol{\alpha}_D}, \boldsymbol{x}, \boldsymbol{z})$. We characterize the errors of Trapezoid Quadratures by reporting the Gap Error (cf Equation 5.16), see Figure 8.7 (b).

Now, can we use a capture bound to set the tolerance ϵ? Unfortunately, the empirical loss $\widehat{\mathcal{L}}_D(h^{\text{kernel}}_{\boldsymbol{\alpha}}) + \lambda \boldsymbol{\alpha}^T \boldsymbol{K} \boldsymbol{\alpha}$ involves regularization so we cannot guarantee that the Rashomon Set (cf. Equation 8.22) contains $\boldsymbol{\alpha}^\star$ unless we make a strong (unverifiable) smoothness assumption $\boldsymbol{\alpha}^{\star T} \boldsymbol{K} \boldsymbol{\alpha}^\star \leq B$. Without knowledge of B, we instead resort to a relative increase heuristic $\epsilon = 1.01 \times \left[\widehat{\mathcal{L}}_D(h^{\text{kernel}}_{\boldsymbol{\alpha}_D}) + \lambda \boldsymbol{\alpha}_D \boldsymbol{K} \boldsymbol{\alpha}_D\right]$ (an increase of $\epsilon_{\text{rel}} = 1\%$ of the minimum objective value).

Figure 8.8 presents the local feature attributions across the Rashomon Sets of Gaussian and Polynomial Kernels. Since the results are consistent across the two types of Kernels, we only discuss Gaussian Kernels. In the top bar plot, the Integrated Gradient of the empirical loss minimizer is plotted as the blue/red bars. We see that the features `Age=22` and `Race=Black` have positive attributions while the features `Charge=Misdemeanor` and `Prior=0` have negative attributions. This suggests that one of the possible explanations for the high risk of Robert relative to James is racial discrimination toward African-Americans. However, when we additionally consider the opinion of models with slightly worst performance on the training data, some of our previous statements on feature attribution cease to hold. Importantly, there exists a competing model h' that yield a null attribution to the feature `Race=Black`, and whose test error is not significantly worse than $h^{\text{kernel}}_{\boldsymbol{\alpha}_D}$ according to a paired Student-t test with $\delta = 0.05$. Therefore, there are reasonable explanations for the disparity between Robert and James that do not rely on Robert being African-American. Even when considering the whole Rashomon Set, there remain statements on which models reach consensus. Notably, the attribution of the feature `Age=22` remains positive and has maximum importance.

Finally, to extract more general insights, we investigated the Global Feature Importance of the Kernel Ridge Regression models. To remain consistent with the experimental setup of Fisher et al. [2019], we utilized the PFI-O global importance functional using the targets y instead of $h(\boldsymbol{x})$ (cf. Equation 3.43). This functional compares the performance on the original data to the performance on synthetic data where the j^{th} feature is replaced by a sample from the marginal. Fisher et al. [2019] have previously proven that the PFI-O can be expressed as a quadratic form $\boldsymbol{\alpha}^T \boldsymbol{Q} \boldsymbol{\alpha} + \boldsymbol{b}^T \boldsymbol{\alpha}$ involving the $\boldsymbol{\alpha}$ coefficients. As a result,

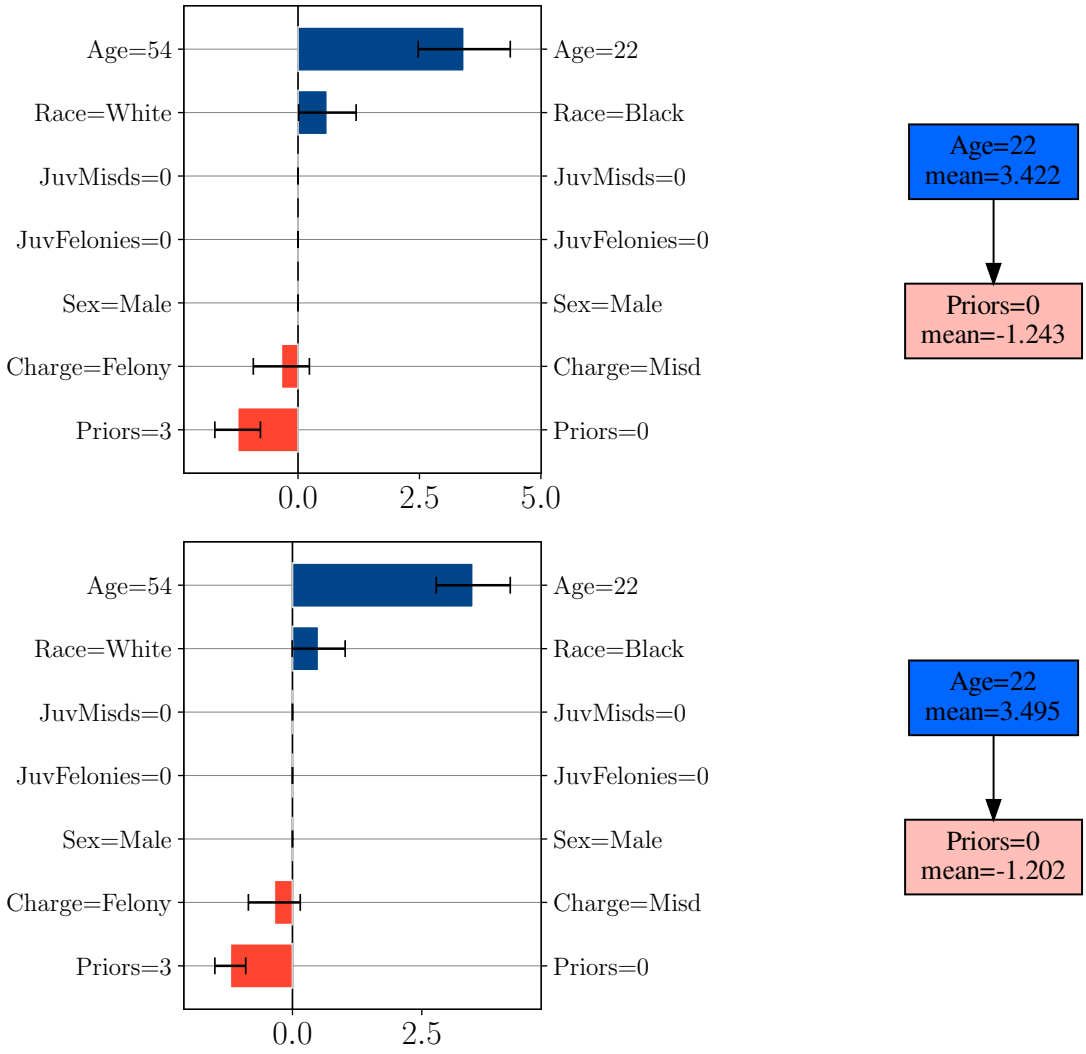


Figure 8.8 Local feature attributions comparing Robert Cannon to James Rivelli. (Top) Gaussian Kernels. (Bottom) Polynomial Kernels. The features on the left of the bar charts represent James while the values on the right represent Robert.

asserting consensus on Global Relative Importance over the Rashomon Set requires solving a TRS.

The results for Gaussian and Polynomial kernels are shown in Figure 8.9. Observe that all models in the Rashomon Set consider the features `Age` and `Priors` more important than the remaining ones. This is true even though their interval of min-max importance $\left[\min_{h\in\mathcal{R}(\mathcal{H},\epsilon)}\Phi_j(h), \max_{h\in\mathcal{R}(\mathcal{H},\epsilon)}\Phi_j(h)\right]$ intersects with the min-max interval of `JuvMisds`. This observation implies that looking at min-max importance intervals, as suggested by Fisher et al. [2019], provides an incomplete view of the Rashomon Set. On the contrary, asserting consensus on Relative Global Importance (*i.e.* $\Phi_i(h) < \Phi_j(h)\ \forall h \in \mathcal{R}(\mathcal{H},\epsilon)$) highlights that

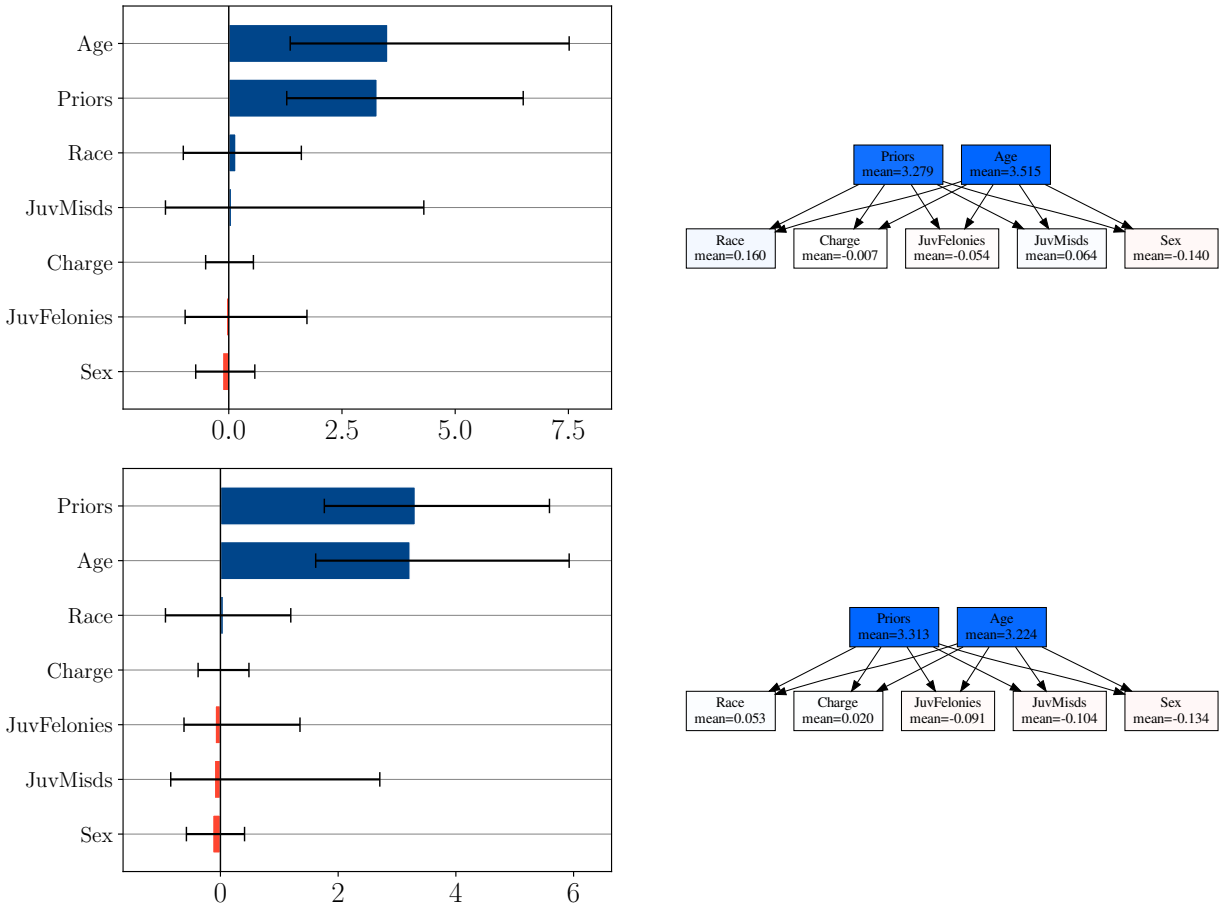


Figure 8.9 Original Permutation Feature Importance (PFI-O) of Kernel Ridge Regression fitted on COMPAS. (Top) Gaussian Kernels. (Bottom) Polynomial Kernels.

`Age` and `Priors` have maximal importance

These observations are concordant with previous work of Rudin et al. [2018] who hypothesize that COMPAS depends strongly on age, priors, and not on race. Nonetheless, our local/-global additive explanations must not be taken as absolute facts about the proprietary model COMPAS. This is because we do not have access to the model and we are surrogating it with Kernel Ridge models fitted on 7 features. The original COMPAS model, in opposition, takes 137 different factors into consideration to produce a score [Rudin et al., 2018]. Our analysis is more of a proof of concept that we can make sense of the LFA/GFI of competing models.

8.5 Random Forests

8.5.1 Methodology

We recap the definition of Random Forest (RF) that was presented in Section 2.1.2. A RF is an ensemble of independently trained decision trees whose predictions are averaged to yield the final predictions [Breiman, 2001a]. Let $r \in \mathbb{N}$ represent the seed encoding all pseudo-random processes in the training of a single tree $h^{\text{tree},[r]}$. Moreover, define $U([M])$ the uniform distribution over all M possible seeds on a computer. A RF is trained by sampling m seeds uniformly at random $R \sim U([M])^m$, and averaging the resulting trees

$$h_R^{\text{rf}}(\boldsymbol{x}) = \frac{1}{|R|} \sum_{r \in R} h^{\text{tree},[r]}(\boldsymbol{x}). \tag{8.24}$$

Since sampling m seeds out of M with/without replacement assigns a non-zero probability to any subset of m seeds, we conceptualize the space of all **possible** RFs as the collection of all subsets of trees.

Definition 8.5.1. *Given a large set $[M]$ of random seeds, the set of all possible RFs of m trees is*

$$\mathcal{H}_m := \left\{ \frac{1}{|R|} \sum_{r \in R} h^{tree,[r]} \;\; : \;\; R \subseteq [M] \;\; \textit{and} \;\; |R| = m \right\}, \tag{8.25}$$

i.e. all averages of subsets of m trees picked randomly. Moreover, we define $\mathcal{H}_{m:} := \cup_{k=m}^{M} \mathcal{H}_k$ as all RFs with least m trees. We interpret $\mathcal{H}_{1:}$ as the set of all possible RF that can ever appear in practice on a given dataset, regardless of the choice of m.

Figure 8.10 illustrates an example of space $\mathcal{H}_m$ which accentuates their combinatoric nature. We finally note the monotonic relation $m < m' \Rightarrow \mathcal{H}_{m:} \supset \mathcal{H}_{m':}$.

Since we interpret $\mathcal{H}_{1:}$ as the set of all possible RFs that can ever appear in practice on a dataset, we aim to characterize its Rashomon Set $\mathcal{R}(\mathcal{H}_{1:}, \epsilon)$. Such a Rashomon Set cannot be explicitly represented because if its exponential size ($|\mathcal{H}_{1:}| = 2^M - 1$). Still, we will see that studying the space $\mathcal{H}_{m:}$ for a carefully chosen m can help us characterize a large subset of the Rashomon Set. The reason we want to work with hypotheses $\mathcal{H}_{m:}$ is that they have a desirable property: optimizing a linear functional over them is tractable, as highlighted by the following proposition.

Proposition 8.5.1. *Let $\mathcal{T} := \{h^{tree,[r]}\}_{r=1}^{M}$ be a set of M trees, $\mathcal{H}_{m:}$ be the set of all RFs with at least m trees from $\mathcal{T}$, and $\phi : \mathcal{H}_{m:} \to \mathbb{R}$ be a linear functional, then $\min_{h \in \mathcal{H}_{m:}} \phi(h)$ amounts to averaging the m smallest values of $\phi(h^{tree,[r]})$ for $r = 1, 2, \ldots M$.*

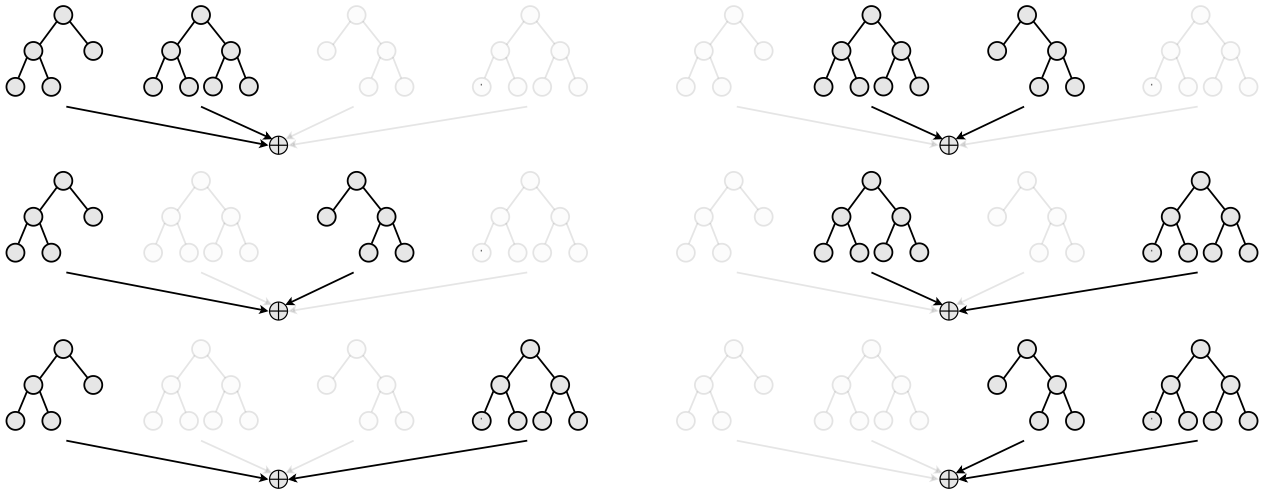


Figure 8.10 Example of the space $\mathcal{H}_2$ representing all possible Random Forests resulting from the groupings of 2 decision trees out of $M = 4$.

The proof is presented in Appendix E.1.3. Examples of linear functionals $\phi : \mathcal{H}_{m:} \to \mathbb{R}$ include the model prediction at fixed input $h(\boldsymbol{x})$, $\boldsymbol{z}$-Anchored Decompositions computable with Algorithm 4, and treeSHAP feature attributions computable with Algorithm 5.

At this point, we assume that the desired tolerance on error ϵ has been fixed and so we wish to identify a value $m(\epsilon)$ that guarantees that $\mathcal{H}_{m(\epsilon):} \subseteq \mathcal{R}(\mathcal{H}_{1:}, \epsilon)$, or equivalently, that $\max_{h \in \mathcal{H}_{m(\epsilon):}} \widehat{\mathcal{L}}_S(h) \leq \epsilon$. This value $m(\epsilon)$ should be as small as possible so that the space $\mathcal{H}_{m(\epsilon):}$ is as large as possible. With this goal in mind, we restrict ourselves to losses $\ell(y', y)$ that are monotonically increasing w.r.t $|y' - y|$. This includes the 0 -1 loss and the squared loss for example. Such losses are of interest because $\max_{y' \in \mathcal{Y}'} \ell(y', y) = \max\{\ell(\min_{y' \in \mathcal{Y}'} y', y)\,,\, \ell(\max_{y' \in \mathcal{Y}'} y', y)\}$ for any set $\mathcal{Y}'$. This means that the worst loss on a point must be attained by either of the two most extreme predictions at that point. Remembering that model predictions are linear functionals of the trees, Proposition 8.5.1 can be used to efficiently identify the min/max predictions at any input. Therefore, it makes sense to define the upper bound

$$\begin{aligned}\max_{h \in \mathcal{H}_{m:}} \widehat{\mathcal{L}}_S(h) \leq & \frac{1}{N} \sum_{i=1}^{N} \max_{h \in \mathcal{H}_{m:}} \ell(h(\boldsymbol{x}^{(i)}), y^{(i)}), \\ = & \frac{1}{N} \sum_{i=1}^{N} \max \left\{ \ell\Big(\min_{h \in \mathcal{H}_{m:}} h(\boldsymbol{x}^{(i)}), y^{(i)} \Big)\,,\, \ell\Big(\max_{h \in \mathcal{H}_{m:}} h(\boldsymbol{x}^{(i)}), y^{(i)} \Big) \right\} := \epsilon^{+}(m),\end{aligned} \tag{8.26}$$

which can be computed efficiently at any $m \leq M$ in time $\mathcal{O}(NM \log M)$. Because of the scalability of this process w.r.t M, the total number of tree M must be reasonable, but still large enough so that $\mathcal{T} = \{h^{\text{tree},[r]}\}_{r=1}^{M}$ is representative of all trees that would be produced

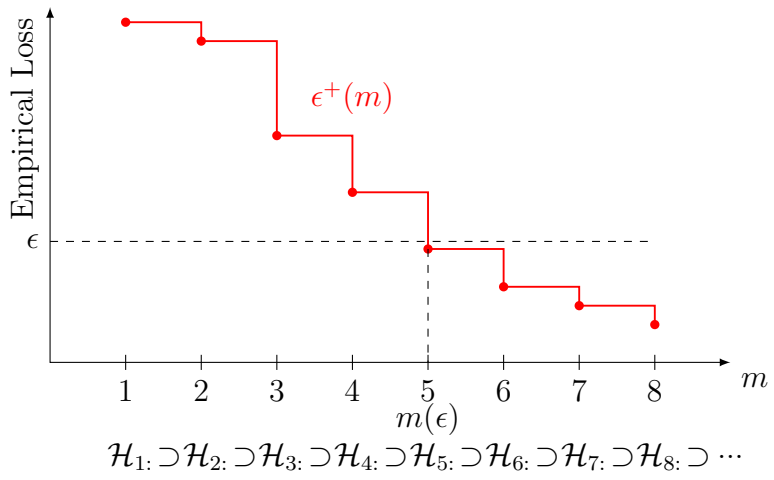


Figure 8.11 Choosing m based on the error tolerance ϵ.

with all possible seeds on a computer. We will see in Section 8.5.2 that setting $M = 1000$ can be representative of all trees fitted on real-world data.

Now, given an absolute tolerance ϵ on the empirical loss, we search for the smallest number of trees m we can keep while ensuring that $\epsilon^+(m) \leq \epsilon$

$$m(\epsilon) := \min\{m \in [M] : \epsilon^+(m) \leq \epsilon\}. \tag{8.27}$$

The intuition behind the computation of $m(\epsilon)$ is presented in Figure 8.11. Since setting $m = m(\epsilon)$ guarantees that $\max_{h \in \mathcal{H}_{m:}} \widehat{\mathcal{L}}_S(h) \leq \epsilon^+(m) \leq \epsilon$, we have $\mathcal{H}_{m(\epsilon):} \subseteq \mathcal{R}(\mathcal{H}_{1:}, \epsilon)$. Hence, we are going to employ $\mathcal{H}_{m(\epsilon):}$ as an under-estimate of the Rashomon Set over which we can efficiently optimize linear functionals such as model predictions and local feature attributions.

Asserting consensus over $\mathcal{H}_{m(\epsilon):}$ on local feature attribution statements is done via optimization problems (cf. Definition 8.2.5). These problems are solved efficiently with Proposition 8.5.1. For example, to compute $\min_{h \in \mathcal{H}_{m:}} \phi_j(h, \boldsymbol{x}, \mathcal{B})$, we calculate the vector of feature attributions of all trees $[\phi_j(h^{\text{tree},[1]}, \boldsymbol{x}, \mathcal{B}), \phi_j(h^{\text{tree},[2]}, \boldsymbol{x}, \mathcal{B}), \ldots, \phi_j(h^{\text{tree},[M]}, \boldsymbol{x}, \mathcal{B})]^T$, then we sort it and average its m smallest values.

Asserting model consensus on global feature importance statements is a lot more complicated since the functionals $\boldsymbol{\Phi}(\cdot, \mathcal{B})$ are not linear w.r.t the model. We refer to Appendix E.2.2 for the full details of how we deal with global feature importance. In short, we employ the functional

$$\Phi_j^{\text{SHAP},[1]}(h, \mathcal{B}) := \mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mathcal{B}}[\,|\phi_j^{\text{SHAP-int}}(h, \boldsymbol{x}, \mathcal{B})|\,] \tag{8.28}$$

and create an ensemble E containing

1. Approximates of $\text{argmin/max}_h \Phi_j^{\text{SHAP},[1]}(h, \mathcal{B})$ for $1 \leq j \leq d$.
2. Approximates of $\text{argmin/max}_h \Phi_i^{\text{SHAP},[1]}(h, \mathcal{B}) - \Phi_j^{\text{SHAP},[1]}(h, \mathcal{B})$ for $1 \leq i < j \leq d$.

After, we assert a consensus among all models in $E \subset \mathcal{H}_{m(\epsilon)}$ leading to the partial order

$$i \mathrel{\widehat{\preceq_{\epsilon,\mathcal{B}}}} j \iff \forall\, h \in E \quad \Phi_i(h, \mathcal{B}) \leq \Phi_j(h, \mathcal{B}). \tag{8.29}$$

We underestimate the model diversity but the partial order is guaranteed to be transitive.

8.5.2 Income Prediction

The Adult-Income dataset[3] contains the census data of 48,842 individuals collected in 1994. It consists of a binary classification task with the goal of predicting if a person makes more ($y = 1$) or less ($y = 0$) than 50k USD per year based on 14 attributes. The data was split into train and test sets with ratios 0.8 and 0.2 respectively. The training set was used to obtain the set $\mathcal{T}$ of M iid trees. For the model, we utilized Scikit-Learn's `RandomForestClassifiers` whose hyperparameters were tuned with a 100 steps random search and 5-fold cross-validation. Then, we trained $M = 1000$ trees in order to generate a set $\mathcal{T}$. The training was actually repeated 5 times so that we ended up with 5 distinct sets of 1000 trees $\mathcal{T}_i$ with $i = 1, 2, \ldots, 5$. This was done to verify our assumption that $\mathcal{T}$ is representative of all trees trained with bootstrapped data and random splits.

After obtaining tree collections, we estimated the Rashomon Set containing all RFs that perform well on the test set. The loss employed was the 0-1 loss meaning the Rashomon Set contains all models with a Misclassification Rate below some threshold ϵ. The tolerance ϵ was set via the capture bound of Proposition 8.2.3 using $h_{\text{ref}} = 1/M \sum_{r=1}^{M} h^{\text{tree},[r]}$ as the reference model. This proposition is applicable since we compute the Rashomon Set on test data that is independent of the hypothesis h_{ref} fitted on training data. Under confidence $\delta = 1\%$, the proposition led to an error tolerance $\epsilon = \sqrt{-2\log(1\%)/N} + \widehat{\mathcal{L}}_S(h_{\text{ref}}) \approx 3\% + \widehat{\mathcal{L}}_S(h_{\text{ref}})$. By computing the upper bound $\epsilon^+(m)$ on test samples, we set the minimum number of trees $m(\epsilon) = 815$, see Figure 8.12.

[3] `https://archive.ics.uci.edu/ml/datasets/adult`

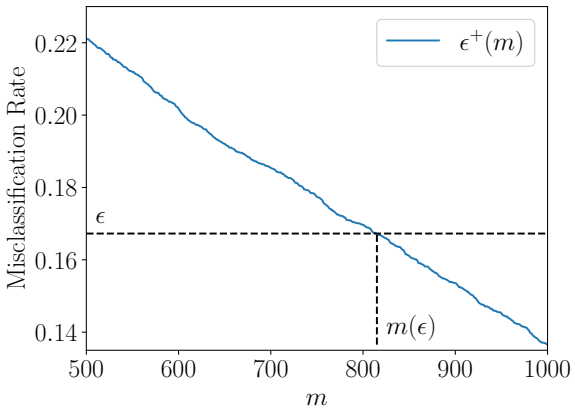


Figure 8.12 Setting $m(\epsilon)$ the minimum number of trees to keep given the tolerance ϵ on the Missclassification Rate. We advocate for keeping at least 815 trees out of 1000.

At this tolerance level, and using the 500 training data as the background $\mathcal{B}$, the sign of the gap was consistent for 90.8% of the individuals. Therefore, under-specification prohibits us from explaining one-tenth of the data. Figure 8.13 (a) shows the distributions of predictions for individuals whose gap is well-defined or ill-defined across the Rashomon Set. Again, individuals whose gap is ill-defined tend to have predictions $h_{\text{ref}}(\boldsymbol{x}^{(i)})$ near the baseline $\mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[h_{\text{ref}}(\boldsymbol{z})]$ so that the Gap $G(h_{\text{ref}}, \boldsymbol{x}^{(i)}, \mathcal{B})$ is small. Once more, we replace the background and re-explain those inputs. Letting $\mathcal{B}'$ be 500 uniformly-chosen adults who were predicted to make more than 50K (*i.e.* $h_{\text{ref}}(\boldsymbol{x}^{(i)}) > 0.5$), the gaps became highly negative and 100% of the previously unexplainable individuals were suddenly explainable.

Since the model outputs $h(\boldsymbol{x}) \in [0, 1]$ are estimated conditional probabilities of $y|\boldsymbol{x}$ and not as hard 0/1 predictions, the local feature attributions should sum up to a difference in conditional probabilities. We computed local feature attributions with TreeSHAP (cf. Algorithm 5). As discussed previously, by storing the local feature attributions of each individual tree $\boldsymbol{\phi}^{\text{SHAP-int}}(h^{\text{tree},[r]}, \boldsymbol{x}, \mathcal{B})$, we can efficiently assert consensus over the Rashomon Set of RFs. Figure 8.13 (b) presents the mean partial-order cardinality as a function of tolerance on test error. We observe that the five curves are very similar which suggests that fitting $M = 1000$ trees can be representative of all trees possibly generated for RFs. For error tolerances smaller than the ϵ employed, the mean cardinality decreases very rapidly. This means that our partial orders abstain from making many statements supported by $h_{\text{ref}} = 1/M \sum_{r=1}^{M} h^{\text{tree},[r]}$, but which are contradicted by other RFs with slightly worst test performance. We now discuss two instances that were explained with our framework.

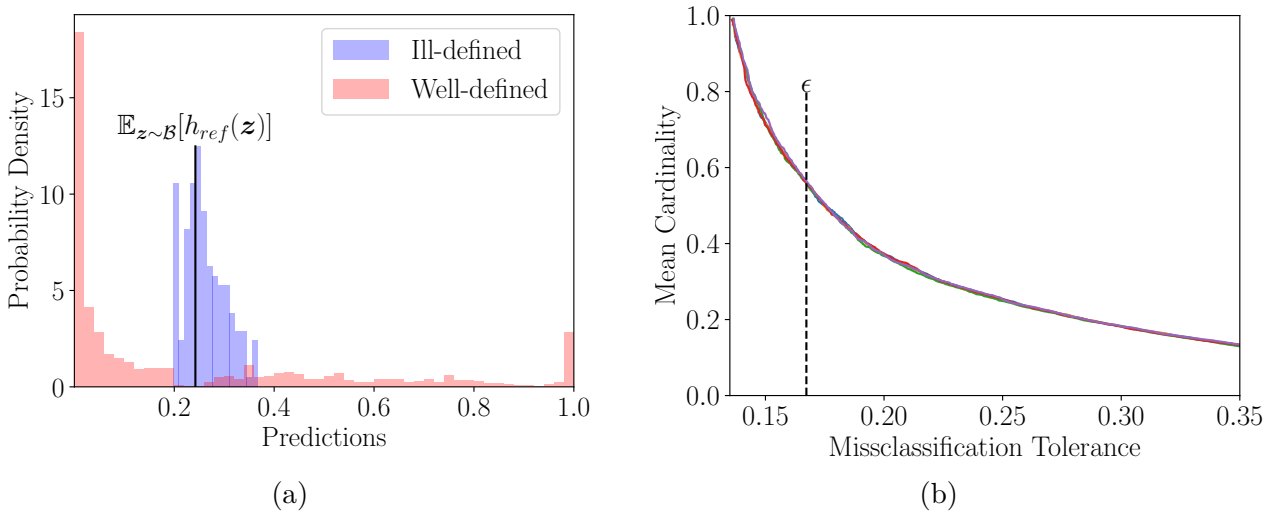


Figure 8.13 Underspecification of RF explanations. (a) Distributions of predictions for instance with ill-defined and well-defined gaps across the Rashomon Set for Adult-Income. The background $\mathcal{B}$ is the empirical distribution over 500 uniform samples from the training data. (b) Partial Order cardinalities for various error tolerances. Each curve is associated with a different tree collection $\mathcal{T}_i$.

The first instance is an individual who makes more than 50k per year and whose predictions range from 0.69 to 0.74 across $\mathcal{H}_{815:}$. The average prediction on the background for all trees is 0.23 so this individual has a positive gap, which we aim to explain with TreeSHAP. Figure 8.14 (Top) illustrates this person's local feature attribution and the resulting partial order that encodes the statements on which there is a consensus in $\mathcal{H}_{815:}$. We observe that the features `educational-num=large` and `marital-status=Married` have maximal positive importance for understanding why this individual has higher-than-average predictions. At the second rank is the feature `age=large`, which is also important but to a lesser extent. Looking at the bar char on the top left, we note that the feature `gender=Male` is given a small yet consistently positive attribution across all models. It appears that all RFs with at least 815 trees exhibit a small gender bias. We will come back to this in our analysis of global feature importance.

The second instance is a person who makes more than 50k and whose predictions range from 0.30 to 0.50. The prediction gap is still positive in that case but it is smaller than the previous example. Figure 8.14 (Bottom) shows how our framework would explain the positive gaps. We focus on the two features `capital-gain=0` and `workclass=Self-Employed` which both have a negative attribution according to the reference model. Looking at the error bars on the bar chart, we observe that the model underspecification is higher for `workclass`

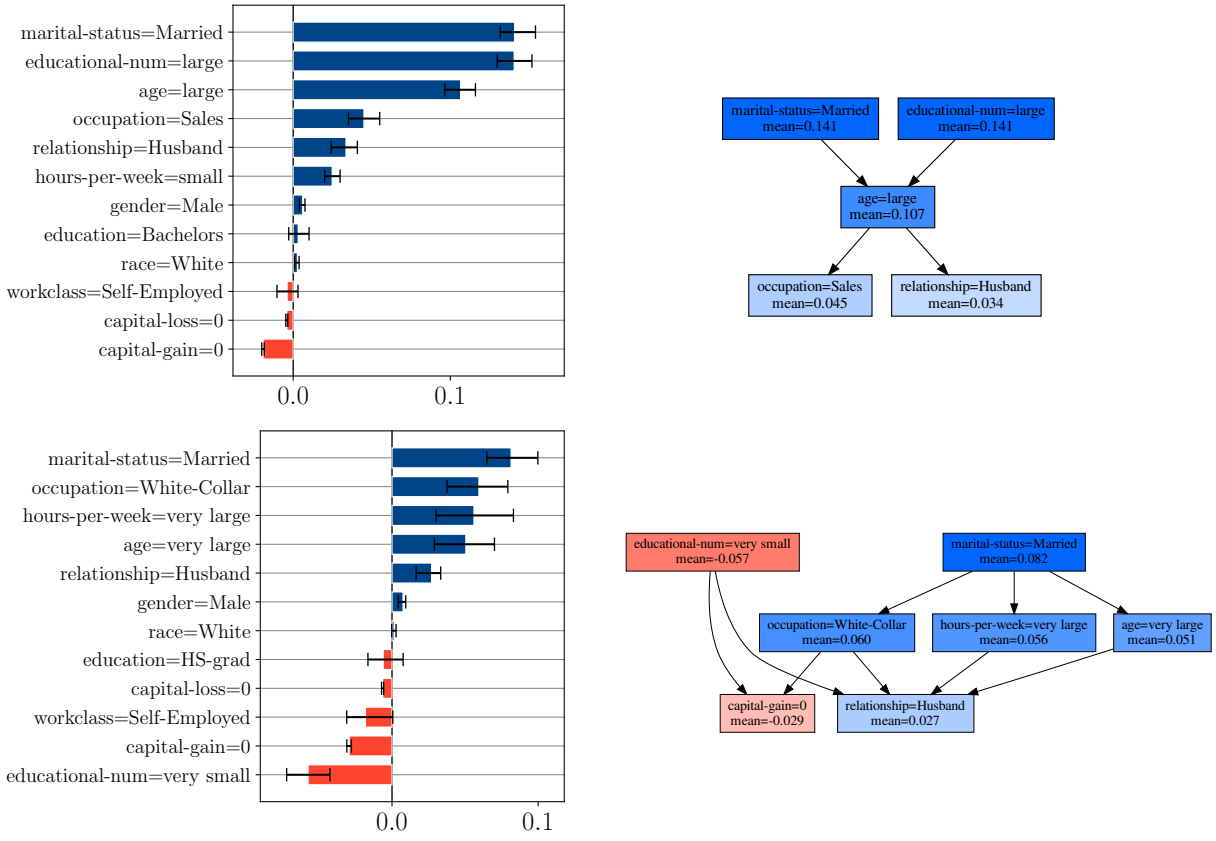


Figure 8.14 Local feature attributions on two individuals (Top) A person with a high prediction, (Bottom) Individual near the decision boundary. The Hasse Diagrams only show the first three ranks.

than with `capital-gain`. This means that there is more agreement among RFs that `capital-gain=0` reduced the model output. For `workclass=Self-Employed`, the model underspecification is so high that the min-max interval crosses the origin, which implies the existence of RFs with satisfactory performance that yield a positive attribution to this feature. Our framework identified this ambiguity and hence removed the feature `workclass` from the partial order despite it having a negative attribution according to the reference model.

Figure 8.15 presents the global feature importance. The associated Hasse diagram is not shown because the feature ordering is a total order. Indeed, the rankings are consistent across all RFs with at least 815 trees. Interestingly, there were more disagreements when looking at local feature attributions. This highlights that combining local attributions $\boldsymbol{\phi}$ into global ones $\boldsymbol{\Phi}$ can result in information loss. Hence, it is primordial to investigate explanations under-specification both globally and locally.

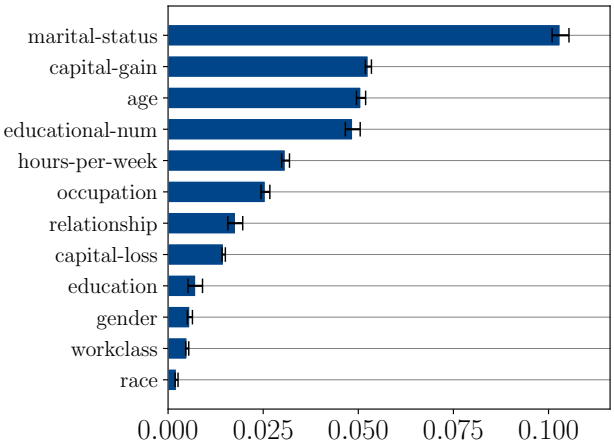


Figure 8.15 Global Feature Importance on Adult-Income.

Notice that all features have non-null importance across the Rashomon Set. This was not true for the hypothesis class of Parametric Additive Models, see Figure 8.6. We suspect that this is due to the training procedure of RFs. Indeed, when growing trees, a random subset of candidate features is chosen at each internal node. The optimal split is then chosen among these features. Hence, even if a feature is irrelevant for predicting y, there is a non-zero probability it will be used by some of the trees in the forest. This is unfortunate in the context of biases because any good RF uses the `gender` features for prediction.

8.6 Discussion

As suggested by our experiments, model under-specification has an important impact on feature attributions on real data, and taking into account this uncertainty seems necessary to derive reliable insight from machine learning models. Our conservative approach only retains the information on features attributions on which all models agree and still succeeds in finding partial order in this chaos. This in itself is an important observation because one could have expected the partial orders to be trivial and contain no interesting structure (no arrows).

Our approach requires a perfect consensus among all good models. However, when employing our methodology with a finite ensemble of models, one may wonder why not also consider statements on which a majority of models agree (or at least 90% of the models agree). As a more extreme example, a practitioner may have 1000 models and 999 of these models state something while a single one states the opposite. Our approach would abstain from making any statement in that case, which may seem unnecessarily strict. An important argument

for requiring a perfect consensus is that it ensures the transitivity of the order relations. This property is crucial for the interpretability of the feature orderings. We note that some prior work has produced partial orders from the consensus of at least $\alpha\%$ of the models via the transitive closure and fine-tuning of α to avoid cycles [Cheng et al., 2010]. Nonetheless, in our context of local explainability, this method has two issues. First, it would require fine-tuning α for each instance $\boldsymbol{x}^{(i)}$ and therefore the interpretation of order relations would change on an instance-by-instance basis. Second, because they rely on transitive closure, the resulting Hasse diagrams could be misinterpreted seeing as the existence of a directed path between two features would not imply a consensus among at least $\alpha\%$ of the models that one feature is more important. Our diagrams, on the other hand, remain simple to interpret: for any instance $\boldsymbol{x}^{(i)}$, a directed path between two features means that all models agree on the relative importance statement and the absence of such path means that at least one model disagrees on that statement. Still, we think that imperfect consensus is a pertinent future work direction, especially for extending our framework to Bayesian methods.

On a more philosophical note, a justification for perfect consensus is that, given that the error threshold ϵ was fixed at a value that represents a satisfactory performance, any single model that disagrees with the rest is still a good model, and its mere existence is enough to put into question the claim supported by the others. Going back to the extreme scenario of 999 models disagreeing with a single one, if this solitary model had the worst performance of the whole ensemble, slightly reducing the error tolerance would remove this model from the Rashomon Set and we would reach a consensus.

Speaking of tuning the error tolerance ϵ, similar to prior work [Fisher et al., 2019, Hsu and Calmon, 2022, Marx et al., 2020], we explore a range of tolerance values and inspect the effect of under-specification on conclusions drawn from models. Nonetheless, it is not clear what is the right value for ϵ, however, we argue that this is a limitation shared by multiple studies on the Rashomon Set [Coker et al., 2021, D'Amour et al., 2020, Dong and Rudin, 2019, Semenova et al., 2022]. It is well understood that the ϵ parameter should be "small enough" to represent negligible performance differences. But, there is still no agreement on what "small enough" means depending on the ML task and hypothesis space. We think the most promising directions in tackling this limitation are Proposition 7 from Fisher et al. [2019], Profile Likelihoods [Coker et al., 2021, Appendix C.1], Model Set Selection [Kissel and Mentch, 2021], and our Propositions 8.2.2 & 8.2.3. All these statistical guarantees suggest defining the "set of all good models" as a set that contains the best-in-class $h^\star$ with high probability. Future work should investigate these theoretical results jointly.

Contributions

We have developed a framework for asserting whether all models within the Rashomon Set support that feature j is important locally/globally, or support that feature j is more important than k locally/globally. Asserting consensus across the Rashomon Set is a challenging task since it contains an infinite amount of models. Crucially, by leveraging optimization problems, we were able to efficiently assert consensus on real-world datasets and models.

The collection of all statement over which consensus is attained form a *partial order*, which we visualized as a Hasse Diagram to get insight from model explanations in-spite of their underspecification. Finally, the cardinality of these partial orders was defined as a measure of disagreement. Disagreements were subsequently reduced by either increasing the prediction Gap, or treating correlated features as a single group in the explanation.

CHAPTER 9 TOUR OF THE PYFD PACKAGE

A variety of open-source packages have recently been developed by the XAI community: `shap`[Lundberg and Lee, 2017], `interpret`[Nori et al., 2019], `innvestigate`[Alber et al., 2019], `captum`[Kokhlikyan et al., 2020], `dalex`[Baniecki et al., 2021], `alibi`[Klaise et al., 2021], `aix360`[Arya et al., 2021], and `xplique`[Fel et al., 2022]. These libraries provide high-level APIs to compute a variety of post-hoc additive explanations.

```
from library import Explainer
# Create an Explainer object
explainer = Explainer(model, data)
# Explain your model predictions
explanations = explainer.explain(data)
```

Such APIs are versatile and easy to use, but it might become harder to understand what is really happening under the hood. As a result, new practitioners may be tempted to treat explainers as black-boxes on top of the model h. Moreover, the lack of ground-truth for the "best" additive explanation makes it difficult to decide what technique (and what package) to trust. There are ongoing efforts to develop faithfulness metrics for XAI methods [Agarwal et al., 2022b, Boisnard et al., 2023] but finding the right faithfulness metrics is still an open question.

This Thesis tackles the lack of ground-truths from a different angle: rather than benchmarking explanation technique, we advocate *aligning* them until they all converge to the ante-hoc explanation of an additive model. This change in philosophy requires a new package with a different workflow.

Part I of this manuscript tackled the unification post-hoc additive explanations and understanding the root cause of their disagreements. We now know that techniques are fundamentally based on $\boldsymbol{z}$-Anchored Decompositions and disagree because they handle feature interactions differently. These discoveries have practical implications : a universal XAI package can be built by focusing on functional decompositions. These decompositions can then be leveraged to build various additive explanations.

```
from pyfd.decompositions import get_components_brute_force

# Compute Anchored Decompositions
decomposition = get_components_brute_force(model, foreground, background)

# Compute Global PDP and PFI Importance
I_PDP = [np.mean( np.mean(decomposition[(i,)], axis=1)**2 ) for i in range(d)]
I_PFI = [np.mean( np.mean(decomposition[(i,)], axis=0)**2 ) for i in range(d)]
```

In this code, the function `get_components_brute_force` is used to compute the $\boldsymbol{H}^u$ matrices storing $\boldsymbol{z}$-Anchored Decompositions (cf. Equation 4.3). These matrices are then used to build the PDP-[2]/PFI Global Feature Importance (cf. Equation 4.6).

Part II introduced measures of disagreement of post-hoc additive explanations. First, the *Interaction Disagreement* reports the impact of feature interaction on the disagreements between PDP/SHAP/PFI. These disagreement were attenuated by explaining the model on well-chosen regions. Second, we advocated reporting Confidence Intervals to quantify the *Subsampling Disagreement*. Finally, the *Underspecification Disagreements* caused by the existence of a Rashomon Set were characterized by asserting consensus over all good models. These disagreements were later diminished by grouping correlated features together, or increasing the prediction Gap $G(h, \boldsymbol{x}, \mathcal{B})$.

What do these three notions of disagreement have in common? They are not quality metrics aimed at comparing explanation techniques, nor are they specific to any additive explanation method. Rather, these disagreements characterize the quality of the contrastive question : *which features are causing the model prediction* $h(\boldsymbol{x})$ *to be higher/lower than the average* $\mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{z})]$. Minimizing the Interaction Disagreements using a regional background $\mathcal{B}_\Omega$ guarantees that the contrastive question can be answered by any post-hoc additive explanation (since they all agree). Reducing the Subsampling and Underspecification Disagreements ensures that the answer to the contrastive question is not sensitive to arbitrary decisions : the choice of subsample or model within the Rashomon Set.

Because explanation disagreements relate to the quality of the contrastive question, we do not aim at benchmarking additive explanation techniques. Instead, disagreement are presented to the user as an incentive to *reframe the contrastive question* so that disagreements are negligible. We implement this new paradigm of *aligning instead of benchmarking*, in the Package `Python Function Decompositions` (`PyFD` for short). The rest of this Chapter is dedicated to illustrating the library's API and its workflow.

We study the toy model of [Bénard et al., 2021, Appendix 1], which we previously investigated in Section 4.1.3. The input $\boldsymbol{x} \in \mathbb{R}^5$ follows a multivariate Gaussian $\mathcal{B} := \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$ with a covariance matrix that is 1 on the diagonal, and 0 everywhere except for $\Sigma_{1,2} = \Sigma_{2,1} = \rho_{1,2}$ and $\Sigma_{4,5} = \Sigma_{5,4} = \rho_{4,5}$. The true model which generated the data is:

$$h(\boldsymbol{x}) = \alpha x_1 x_2 \, \mathbb{1}[x_3 \geq 0] + \beta x_4 x_5 \, \mathbb{1}[x_3 < 0]. \tag{9.1}$$

```
def generate_problem(N, seed, rho_12, rho_45, alpha, beta):
    # Generate input
    np.random.seed(seed)
    cov = np.eye(5)
    cov[0, 1] = rho_12; cov[1, 0] = rho_12
    cov[3, 4] = rho_45; cov[4, 3] = rho_45
    X = np.random.multivariate_normal(np.zeros(5), cov=cov, size=(N,))
    # Model to explain
    def h(X):
        return alpha * X[:, 0] * X[:, 1] * (X[:, 2]>=0).astype(np.int64) +\
               beta * X[:, 3] * X[:, 4] * (X[:, 2]<0).astype(np.int64)

    return X, h

X, h = generate_problem(1000, 42, 0.2, 0.5, 1, 2)
```

9.1 Setup

The first step in `PyFD` is to create a `Features` object that represents the input domain $\mathcal{X} = \prod_{j=1}^{d} \mathcal{X}_j$ of the model $h : \mathcal{X} \to \mathbb{R}$ being explained. To create this object, one must pass the data matrix $\boldsymbol{X}$, the feature names, and their types to the constructor.

```
from pyfd.features import Features
features = Features(X, feature_names=[f"x{i}" for i in range(1, 6)], feature_types=["num"]*5)
```

The various feature types supported by `PyFD` are

Feature Type	Domain	Example
`num`	$\mathcal{X}_j = \mathbb{R}$	`FloorArea=100.23`
`sparse_num`	$\mathcal{X}_j = \mathbb{R}$ with $\mathbb{P}[x_j = 0] > 0$	`Revenue=0,Revenue=35K`
`num_int`	$\mathcal{X}_j = \mathbb{N}$	`Age=26`
`bool`	$\mathcal{X}_j = \{0, 1\}$	`Workingday,notWorkingday`
`percent`	$\mathcal{X} = [0, 1]$	`Score=65%`
`ordinal,nominal`	$\mathcal{X}_j = [C]$	`Animal=cat`

In the current example, all features are `num`. The `.summary()` method provides an overview of each input feature.

```
features.summary()
>>>
|Idx|        Name        |        Type        |    Card    |     I^-1({i})     |
-------------------------------------------------------------------------------
| 0 | x1                 | num                | inf        | [0]               |
| 1 | x2                 | num                | inf        | [1]               |
| 2 | x3                 | num                | inf        | [2]               |
| 3 | x4                 | num                | inf        | [3]               |
| 4 | x5                 | num                | inf        | [4]               |
-------------------------------------------------------------------------------
```

By our design choice, features are actually abstract features, possibly representing a set of low level input features. By default, the identity embedding is assumed which means that feature j is simply the super feature $[j]$ corresponding to the single jth column of the input.

9.2 Additive Explanations

9.2.1 Anchored Components

When computing $\boldsymbol{z}$-Anchored Decompositions, one must provide a list of anchors $\{\boldsymbol{z}^{(j)}\}_{j=1}^{M}$ and a list of evaluation points $\{\boldsymbol{x}^{(i)}\}_{i=1}^{N}$ for the decompositions. The components h_u are stored in the $N \times M$ matrix $\boldsymbol{H}^u$

$$H_{ij}^u := h_{u,\boldsymbol{z}^{(j)}}(\boldsymbol{x}^{(i)}). \tag{9.2}$$

In `PyFD`, the matrices are obtained with a `get_components_` function and are stored in a dictionary `decomposition` whose keys are tuples representing the subsets u

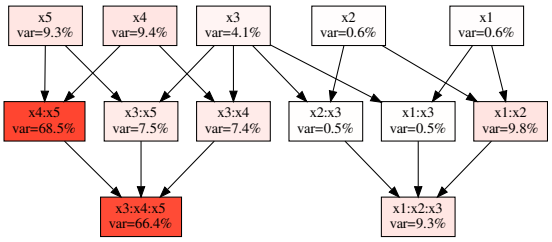
$$\begin{aligned} \texttt{decomposition[()]} &= [h(\boldsymbol{z}^{(1)}), \ldots, h(\boldsymbol{z}^{(M)})]^T \\ \texttt{decomposition[(0,)]} &= \boldsymbol{H}^{\{0\}} \\ \texttt{decomposition[(0, 1)]} &= \boldsymbol{H}^{\{0,1\}}. \end{aligned} \tag{9.3}$$

The `PyFD` package offers many `get_components_` primitives : some are Model-Agnostic (M-A) and others are Model-Specific (M-S).

Function	Description
`get_components_brute_force`	M-A computation of $\{\boldsymbol{H}^u\}_{\lvert u\rvert \leq D}$
`get_components_adaptive`	M-A computation of $\{\boldsymbol{H}^u\}_{u \in U}$ with Algorithm 2
`get_components_linear`	M-S for linear models (cf. Equation 5.8)
`get_components_ebm`	M-S for EBMs [Nori et al., 2019].
`get_components_tree`	M-S for tree ensembles with Algorithm 4

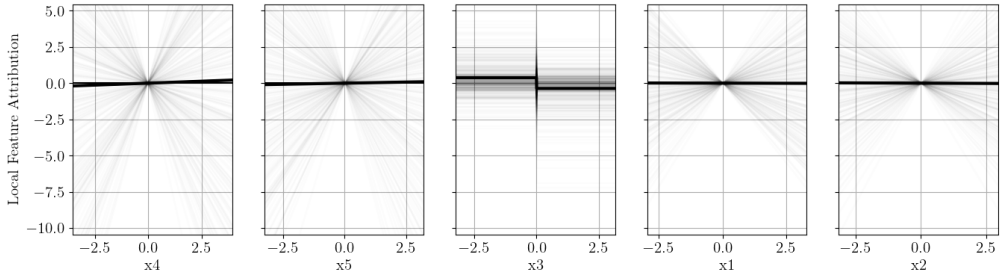
In the current example, to remain Model-Agnostic, we use the `get_components_adaptive` primitive. We then visualize the lattice-space of the decomposition.

```
from pyfd.decompositions import get_components_adaptive
from pyfd.plots import decomposition_graph

# Compute the functional decomposition with foreground=background=X
decomposition = get_components_adaptive(h, X, tolerance=1e-2)
dot = decomposition_graph(decomposition, features.names())
```



According to this plot, the model involves interactions between features $\{x_1, x_2, x_3\}$ and $\{x_3, x_4, x_5\}$. Given the `decomposition` dictionary, we can plot the main effects.

```
from pyfd.plots import partial_dependence_plot
partial_dependence_plot(decomposition, X, X, features)
```



The PDPs are shown as dark lines while the ICE curves $h_{j,\boldsymbol{z}}(\boldsymbol{x})$ are colored gray. The PDPs are essentially flat while the ICE curves are not, which is indicative that feature interactions make the PDP unreliable.

9.2.2 Shapley Values

In addition to computing $\boldsymbol{z}$-Anchored Decompositions, the `PyFD` package provides Model-Agnostic and Model-Specific implementations of the Interventional Shapley Values.

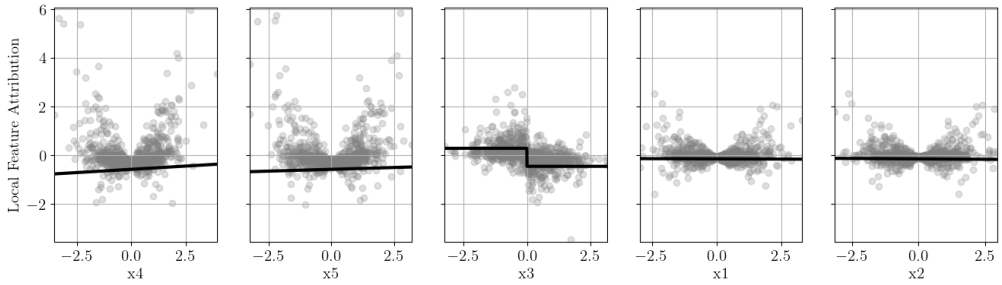
Function	**Description**
`permutation_shap`	M-A permutation estimate (cf. Equation 4.22)
`lattice_shap`	M-A lattice estimate (cf. Equation 4.24)
`interventional_treeshap`	M-S for tree ensembles (cf. Algorithm 5)

The results of these algorithms are stored in $\{\boldsymbol{\Phi}^k\}_{k=1}^d$ matrices

$$\Phi_{ij}^k := \phi_k^{\text{SHAP-int}}(h, \boldsymbol{x}^{(i)}, \boldsymbol{z}^{(j)}). \tag{9.4}$$

Going back to the example, since anchored decompositions have already been fully computed, we need not estimate Shapley Values and can simply use the `shap_from_decomposition` primitive that redistributes interactions evenly between features involved (cf. Equation 3.31)

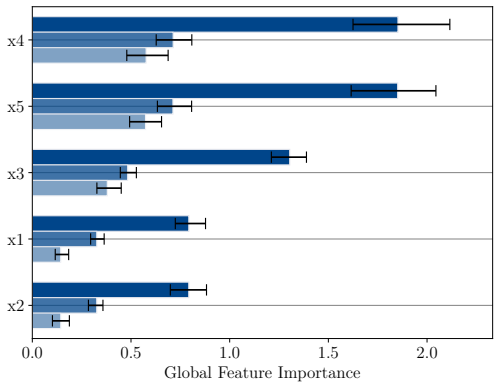
```
from pyfd.shapley import shap_from_decomposition
from pyfd.plots import attrib_scatter_plot
# Compute Shapley Values
shap_values = shap_from_decomposition(decomposition)
# Plot the results
attrib_scatter_plot(decomposition, shap_values, foreground=X, features=features)
```



In this plot, the dark line is again the PDP while the gray dots are the Shapley Values. Because of strong feature interactions, it is hard to see a general trend appear from these local feature attributions.

To complement our local analysis of model behavior, we can compute various measures of Global Feature Importance: the function `get_PDP_PFI_importance` returns the PDP-[2] and PFI importance using the $\{\boldsymbol{H}^k\}_{k=1}^d$ matrices (cf. Equation 4.6) and the function `get_SHAP_importance` aggregates Shapley Values leveraging Equation 3.42. All these functions can return bootstrap confidence intervals to account for the Subsampling Disagreements.

```
from pyfd.decompositions import get_PDP_PFI_importance
from pyfd.shapley import get_SHAP_importance
from pyfd.plots import bar
# Compute Feature Importance
I_PDP, I_PFI, Err_PDP, Err_PFI = get_PDP_PFI_importance(decomposition, bootstrap_error=True)
I_SHAP, Err_SHAP = get_SHAP_importance(shap_values, bootstrap_error=True)
# Plot the results
bar([I_PFI, I_SHAP, I_PDP], features.names(), xerr=[Err_PFI, Err_SHAP, Err_PDP])
```



The various GFI techniques do not agree and, since their bootstrap confidence intervals do not intersect, such disagreements are unlikely caused by the subsampling error. These significant disagreements suggest that the model exhibits feature interactions and so additive explanations of this model given our choice of background/contrastive questions are not reliable.

To resume, `PyFD` computes $\boldsymbol{z}$-Anchored Decompositions, Interventional Shapley Values, and the user aggregates them to their liking to generate post-hoc additive explanations.

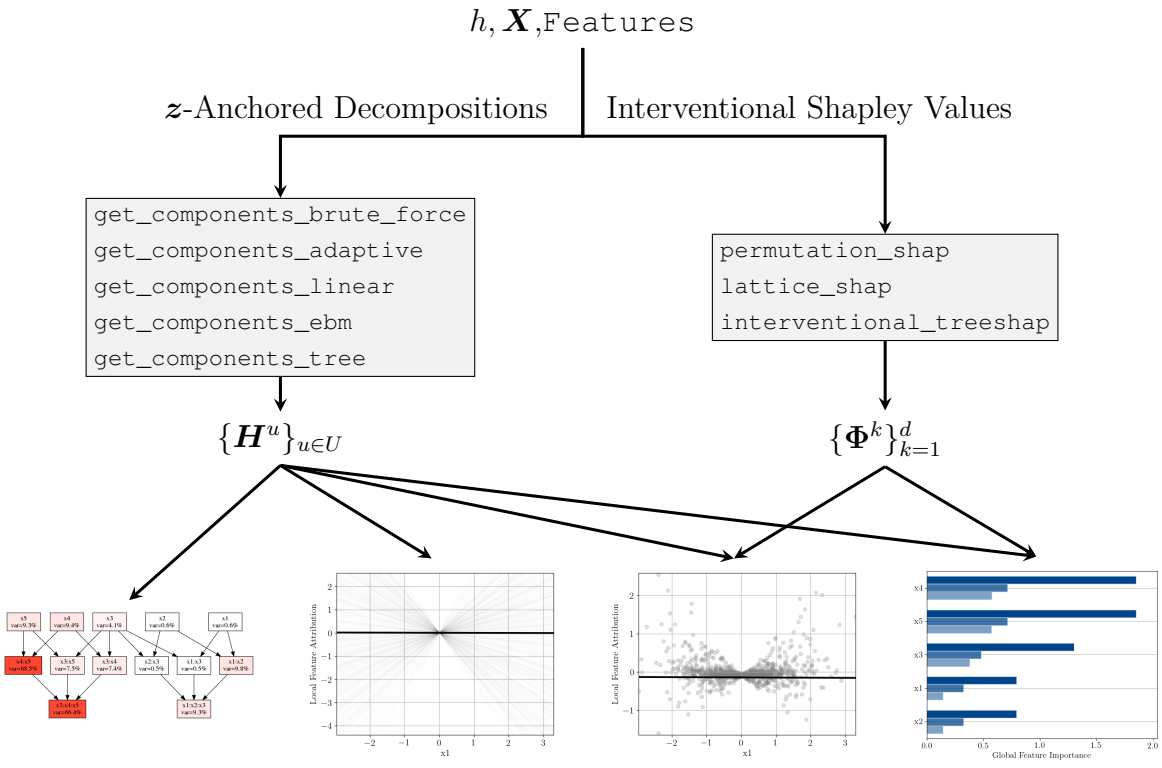


Figure 9.1 `PyFD` workflow to compute post-hoc additive explanations.

9.3 Minimizing Feature Interactions

`PyFD` lets users easily and transparently compute additive explanations, but it also highlights the difficulties in observing general trends when features interact. As seen previously, we could not conclude anything by looking at local feature attributions (PDP and SHAP), and there were considerable disagreements regarding the PDP/SHAP/PFI global importance. To address this situation, the second pillar of `PyFD` is the implementation of feature interaction quantifiers, and methodologies to reduce said interactions.

Feature interaction are quantified with the Lack of Additivity (LoA) metrics introduced in Chapter 6. Notably, the normalized Cost of Exclusion (CoE)

$$100\% \times \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}[\,(\,h(\boldsymbol{x}) - \sum_{\substack{u\subset[d]\\|u|\leq 1}} h_{u,\mathcal{B}}(\boldsymbol{x})\,)^2\,] / \mathop{\mathbb{V}}_{\boldsymbol{x}\sim\mathcal{B}}[h(\boldsymbol{x})] \tag{9.5}$$

measures the error between the model and its Interventional additive decomposition. The CoE is null whenever the model is additive and is bounded above by 100% when features are independent.

```
from pyfd.decompositions import get_CoE
print(get_CoE(decomposition))
>>> 160.05
```

Here, CoE exceeds 100% which indicates that are features are correlated. This is not dramatic because the exact value of the CoE is not of critical importance. The true utility of the CoE is to quantify the remaining room for improvement. We now present two methodologies built-in `PyFD` for reducing feature interactions : grouping features and FD-Trees.

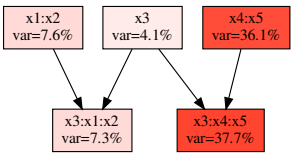
9.3.1 Grouping Features

Chapter 8 previously demonstrated that treating correlated features as a single group can reduce Underspecification Disagreements. In the current example, grouping the correlated features $\{x_1, x_2\}$ and $\{x_4, x_5\}$ also reduces the Interaction Disagreements because these features happen to interact. In `PyFD`, features are grouped by calling the `.group()` method.

```
grouped_features = features.group([[0, 1], [3, 4]])
print(grouped_features.summary())
>>>
|Idx|        Name        |        Type        |    Card    |     I^-1({i})     |
-------------------------------------------------------------------------------
| 0 | x3                 | num                | inf        | [2]               |
| 1 | x1:x2              | num:num            | inf:inf    | [0, 1]            |
| 2 | x4:x5              | num:num            | inf:inf    | [3, 4]            |
-------------------------------------------------------------------------------
```

In the new feature object `grouped_features`, x_1 and x_2 are treated as a single feature $x_1{:}x_2$ which is part of the domain $\mathcal{X}_j = \mathbb{R}^2$ and similarly for x_4 and x_5. Because multiple columns of the input matrix correspond to the same feature, it is now crucial to understand the preimage $\mathcal{I}_{\boldsymbol{\xi}}^{-1}$. According to the `.summary()` method, the feature $x_1 : x_2$ maps to the first and second columns of $\boldsymbol{X}$ while $x_4 : x_5$ maps to the last two columns. From this point on, it is important to pass the preimage to the `PyFD` primitives to inform them that multiple columns of $\boldsymbol{X}$ are treated as a single feature.

```
# Pass the preimage Imap_inv
grouped_decomposition = get_components_adaptive(h, X, Imap_inv=grouped_features.Imap_inv)
dot = decomposition_graph(grouped_decomposition, grouped_features.names())
```



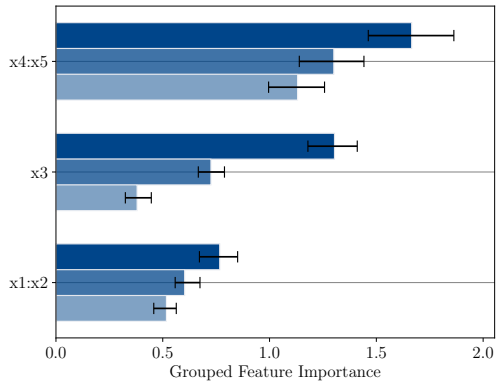
In this new lattice-space, there are no longer individual main effects for x_1, x_2, x_4, x_5 since these features are now part of a group. The grouped features $x_1 : x_2$ and $x_4 : x_5$ have their respective main effects and interactions involving x_3.

We can compute the Shapley Values by directly passing the decomposition as argument to `shap_from_decomposition`

```
grouped_shap_values = shap_from_decomposition(grouped_decomposition)
```

and aggregate the results to get GFIs.

```
I_PDP, I_PFI, err_PDP, err_PFI = get_PDP_PFI_importance(grouped_decomposition,
                                                        bootstrap_error=True)
err_SHAP, err_SHAP = get_SHAP_importance(grouped_shap_values, bootstrap_error=True)
bar([I_PFI, I_SHAP, I_PDP], grouped_features.names(), xerr=[err_PFI, err_SHAP, err_PDP])
```



The remaining disagreements between the three feature importance imply that feature x_3 still interacts with others. This prohibits comparisons between the importance of features x_3 and $x_1 : x_2$ because PFI and PDP disagree on their relative importance. Again, the CoE is computed.

```
print(get_CoE(grouped_decomposition))
>>>44.70
```

By grouping features, the CoE has diminished from 160.05 to 44.70. This is a considerable improvement but there is clearly room for more.

9.3.2 FD-Trees

Chapter 6 has shown that restricting the background $\mathcal{B}_\Omega$ to the leaves Ω of a FD-Tree can reduce feature interactions. In `PyFD`, FD-Trees are very simple to fit.

```
from pyfd.fd_trees import CoE_Tree

# The original features are passed to the constructor
tree = CoE_Tree(features, max_depth=1)
# The fit method requires the background=X and the functional decomposition
tree.fit(X, grouped_decomposition)
tree.print(verbose=True)
>>>
Before CoE 44.7
Samples 1000
if x3 <= 0.0224:
|   CoE 1.56
|   Samples 500
|   Region 0
else:
|   CoE 0.00
|   Samples 500
|   Region 1
Final CoE 1.56
```

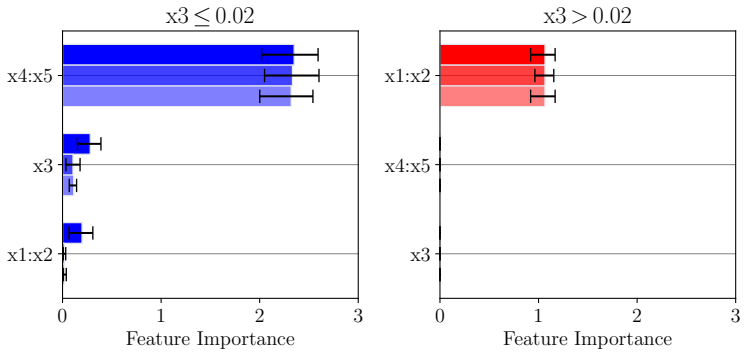
By partitioning the input space into the two disjoint regions $\Omega_- := \{\boldsymbol{x} \in \mathbb{R}^5 : x_3 \leq 0.022\}$ and $\Omega_+ := \{\boldsymbol{x} \in \mathbb{R}^5 : x_3 > 0.022\}$, the CoE was further reduced from 44.7 to 1.56. This means that the model is "almost" additive when restricted to those regions and when grouping features $x_1 : x_2$ and $x_4 : x_5$. The regional decompositions and Shapley Values on the background $\mathcal{B}_{\Omega_-}$ and $\mathcal{B}_{\Omega_+}$ are computed as follows.

```
# Predict which sample is part of which leaf
regions = tree.predict(X)
# Get rule description of the leaves
rules = tree.rules(use_latex=True)
```

```
# Iterate over both regions
regional_backgrounds = [[], []]
regional_decomposition = [[], []]
regional_shap = [[], []]
for r in range(2):
    regional_backgrounds[r] = X[regions==r]
    # Decomposition
    regional_decomposition[r] = get_components_adaptive(h, regional_backgrounds[r],
                                                        Imap_inv=grouped_features.Imap_inv,
                                                        tolerance=1e-2)
    # Shapley values
    regional_shap[r] = shap_from_decomposition(regional_decomposition[r])
```

From these decompositions and Shapley Values, one can compute the regional feature importance as follows.

```
from pyfd.plots import COLORS
fig, axes = plt.subplots(1, 2, figsize=(8, 4))
for r in range(2):
    I_PDP, I_PFI, Err_PDP, Err_PFI = get_PDP_PFI_importance(regional_decomposition[r],
                                                            bootstrap_error=True)
    I_SHAP, Err_SHAP = get_SHAP_importance(regional_shap[r], bootstrap_error=True)
    bar([I_PFI, I_SHAP, I_PDP], grouped_features.names(),
        xerr=[Err_PFI, Err_SHAP, Err_PDP],
        ax=axes[r], color=COLORS[r])
    axes[r].set_title(rules[r])
```



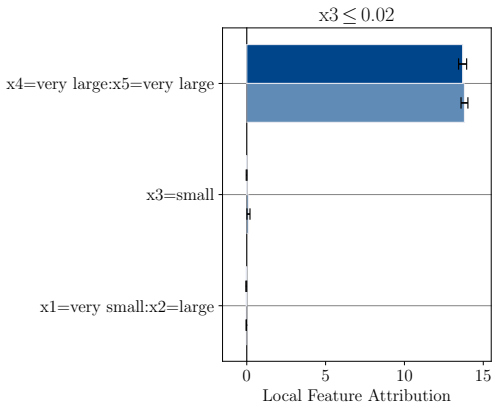
As a result of grouping features and splitting the input space in half, there is almost perfect agreement between the three explainability techniques. Agreement implies that additive explanations are more faithful to the model as argued in Chapter 6.

Since post-hoc additive explanations are aligned, we can use them to answer a contrastive question. To this end, we first compute the PDP/SHAP local feature attributions on region 0 (*i.e.* the region Ω_-).

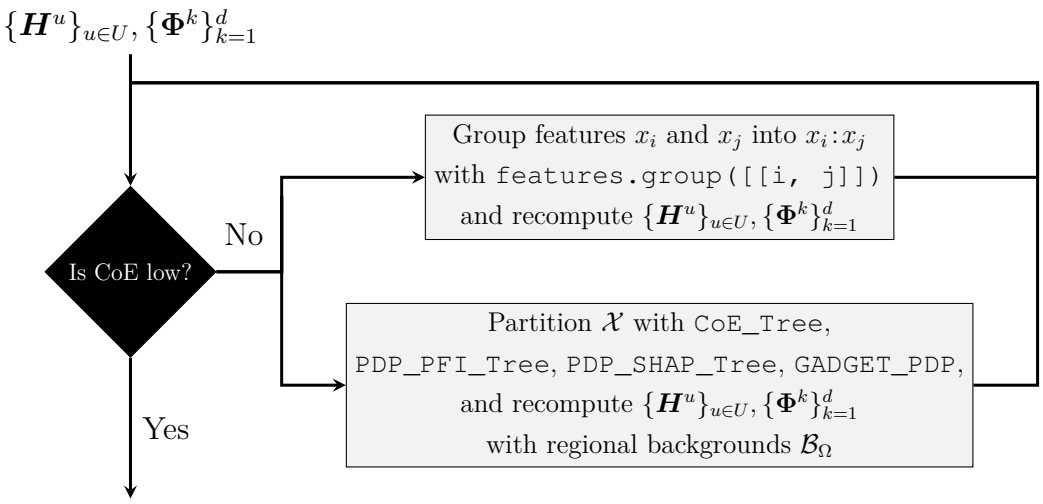
```
from scipy.stats import norm

# The multiplicative factor for CLT-based 95% confidence intervals
std_factor = norm.ppf(1-0.025) / np.sqrt(len(regional_backgrounds[0]))

### Explain predictions on the group 0 ###
# PDP attribution
PDP_attribution = np.column_stack([regional_decomposition[0][(i,)].mean(1)
                                                    for i in range(3)])
Err_PDP = np.column_stack([regional_decomposition[0][(i,)].std(1) * std_factor
                                                    for i in range(3)])
# Shapley attribution
SHAP_attribution = regional_shap[0].mean(1)
Err_SHAP = regional_shap[0].std(0) * std_factor

print(PDP_attribution.shape)
>>> (500, 3)
print(SHAP_attribution.shape)
>>> (500, 3)
```

Note that we estimate the error of PDP and SHAP using the Central Limit Theorem, as suggested by Theorems 4.1.1 & 5.4.3. We now present the local feature attributions explaining the point $\boldsymbol{x} \in \Omega_-$ with the largest Gap.

```
# Explain the point with the largest prediction on group 0
idx = np.argmax( h(regional_backgrounds[0]) )
# Plot
bar([PDP_attribution[idx], SHAP_attribution[idx]],
    grouped_features.print_value( regional_backgrounds[0][idx] ),
    xerr=[Err_PDP[idx], Err_SHAP[idx]])
plt.title(rules[0])
```



Both PDP and SHAP agree that x_1 and x_2 being large is the main factor pushing this prediction away from the mean over $\mathcal{B}_{\Omega_-}$. To resume, `PyFD` offers tools to reduce the feature interactions (*i.e.* increase alignment between PDP/SHAP/PFI): one can 1) group correlated/interacting features, 2) partition the input space with a FD-Tree.



Additive Explanations are Faithful.

Figure 9.2 PyFD workflow to reduce feature interactions.

Contributions

The PyFD Python Package is a new framework for computing additive explanations of black-boxes. The library provides several Model-Agnostic and Model-Specific implementations for computing $\boldsymbol{z}$-Anchored Decompositions and Interventional Shapley Values. Once these quantities are available, they can be used to make various models diagnostics *e.g.* visualize the lattice space, plot the PDP/SHAP local feature attributions, and plot the PDP/SHAP/PFI global feature importance.

Since these explanations are unified through the lens of $\boldsymbol{z}$-Anchored Decompositions, disagreements between the various explanation techniques can be sourced back to feature interactions. Consequently, PyFD offers tools for quantifying and reducing feature interactions. The Cost of Exclusion is used as a reference metric to show the "room for improvement" in terms of interaction reduction. Also, users are allowed to group features together and fit FD-Trees to reduce the strength of interactions. To keep the Subsampling Disagreements in check, PyFD provides confidence intervals based on bootstrapping or the Central Limit Theorem. Implementing the Underspecification Disagreements in PyFD is part of future work since efficiently exploring the Rashomon Set is still an open challenge.

CHAPTER 10 PYFD IN PRACTICE

PyFD is a unified framework for computing post-hoc additive explanations and reporting/reducing their Interaction and Subsampling Disagreements. The last chapter presented the framework on a simplified toy example. The current chapter applies it on two real-world datasets : the BikeSharing and Marketing data from the UCI repository.

10.1 Bike Rentals Prediction

The BikeSharing dataset[1] aims to predict the hourly count of bike rentals between years 2011 and 2012 in Washington state, based on time and weather features. It involves 17K instances and 10 features.

Name	Type	Domain	Description
yr	ordinal	{2011, 2012}	The current year
mnth	ordinal	{Jan,Feb,...,Nov,Dec}	The current month
hr	num_int	$\{0, 1, \ldots, 10, 11\}$	The current hour
holiday	bool	{False, True}	Is it a holiday?
weekday	ordinal	{Sun, Mon, ..., Fri, Sat}	The current weekday
workingday	bool	{False, True}	Is it a workinday?
weathersit	num_int	$\{1, 2, 3, 4\}$	The weather situation (4=bad)
temp	num	$[0.82, 41]$	The temperature in Celcius
hum	num	$[0, 1]$	The humidity index
windspeed	num	$[0, 0.85]$	The windspeed

In PyFD, the feature object is constructed as follows.

```
from pyfd.features import Features
feature_names = ["yr", "mnth", "hr", "holiday", "weekday", "workingday",
                                "weathersit", "temp", "hum", "windspeed"]
feature_types = [
        ["ordinal", "2011", "2012"],
        ["ordinal",
            "Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul",
            "Aug", "Sep", "Oct", "Nov", "Dec",
        ],
        "num_int", "bool",
        ["ordinal", "Sun", "Mon", "Thue", "Wed", "Thur", "Fri", "Sat"],
        "bool", "num_int", "num", "num", "num",
    ]
features = Features(X, feature_names, feature_types)
```

[1] https://archive.ics.uci.edu/dataset/275/bike+sharing+dataset

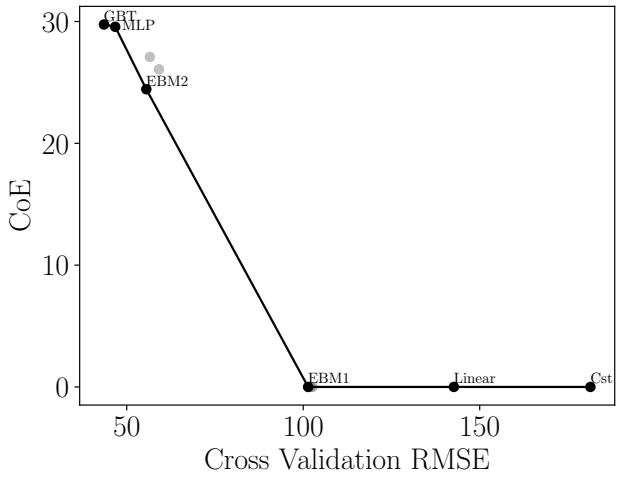


Figure 10.1 Pareto front showing the optimal tradeoffs between performance and interaction strength on the Bike-Sharing use-case.

```
features.summary()
>>>
|Idx|        Name        |        Type        |    Card    |     I^-1({i})     |
-------------------------------------------------------------------------------
| 0 | yr                 | ordinal            | 2          | [0]               |
| 1 | mnth               | ordinal            | 12         | [1]               |
| 2 | hr                 | num_int            | 24         | [2]               |
| 3 | holiday            | bool               | 2          | [3]               |
| 4 | weekday            | ordinal            | 7          | [4]               |
| 5 | workingday         | bool               | 2          | [5]               |
| 6 | weathersit         | num_int            | 4          | [6]               |
| 7 | temp               | num                | inf        | [7]               |
| 8 | hum                | num                | inf        | [8]               |
| 9 | windspeed          | num                | inf        | [9]               |
-------------------------------------------------------------------------------
```

The first step of the pipeline was to split the dataset into train/test sets with ratios 0.8:0.2. The training set was utilized to select the final model and the test set provided an unbiased measure of its performance. Seeing as predicting bike rentals is a regression problem, the Root-Mean-Squared-Error (RMSE) was used to report performance.

One of the takeaways of Chapter 6 is that feature interactions prohibit consensus among post-hoc explainers. Consequently, interactions should only be included in the model if they are necessary to attain high performance. To see if interactions are necessary for model performance, we first investigated additive models (Linear, Splines, EBM1, GBTs with `max_depth=1`) as well as models with pair-wise interactions (EBM2, GBTs with `max_depth=2`), before employing full complexity models (GBTs, RFs, MLPs). Figure 10.1 shows the Pareto front of the five-fold cross-validation RMSE and the Cost of Exclusion.

This curve regroups the models with the most competitive tradeoffs between performance and interaction strength. Going from the best additive model (EBM1) to the best model with pair-wise interactions (EBM2) leads to a RSME reduction from 100 to 55, while the CoE increases from 0% to 25%. Going from pairwise interactions to a full complexity model (GBT, MLP) reduces the RMSE from 55 to 45 and increases the CoE from 25% to 30%. Because the GBTs and MLPs attain higher performance than models with pair-wise interactions, we conclude that the Bike-Sharing dataset involves high-order interactions. The GBT and MLP models were selected, leading to test set errors of 42.7 and 43.8 respectively. The remainder of the analysis will only present results for GBT because the results for MLPs were identical.

Given the GBT model, we subsample 1000 training points to define the background distribution $\mathcal{B}$. Then we compute its $\boldsymbol{z}$-Anchored Decompositions via `get_components_tree`, which implements Algorithm 4.

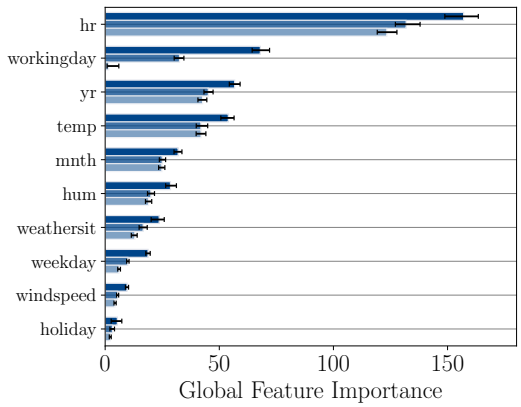
```
from pyfd.decompositions import get_components_tree, get_CoE
# Subsample the data
background = X_train[:1000]
# Compute the Anchored Decomposition with foreground=background
decomposition = get_components_tree(model, background, background, anchored=True)
print(get_CoE(decomposition))
>>> 30.45
```

We can also compute the Shapley Values using `interventional_treeshap`, which implements Algorithm 5.

```
from pyfd.shapley import interventional_treeshap
shap_values = interventional_treeshap(model, background, background, anchored=True)
```

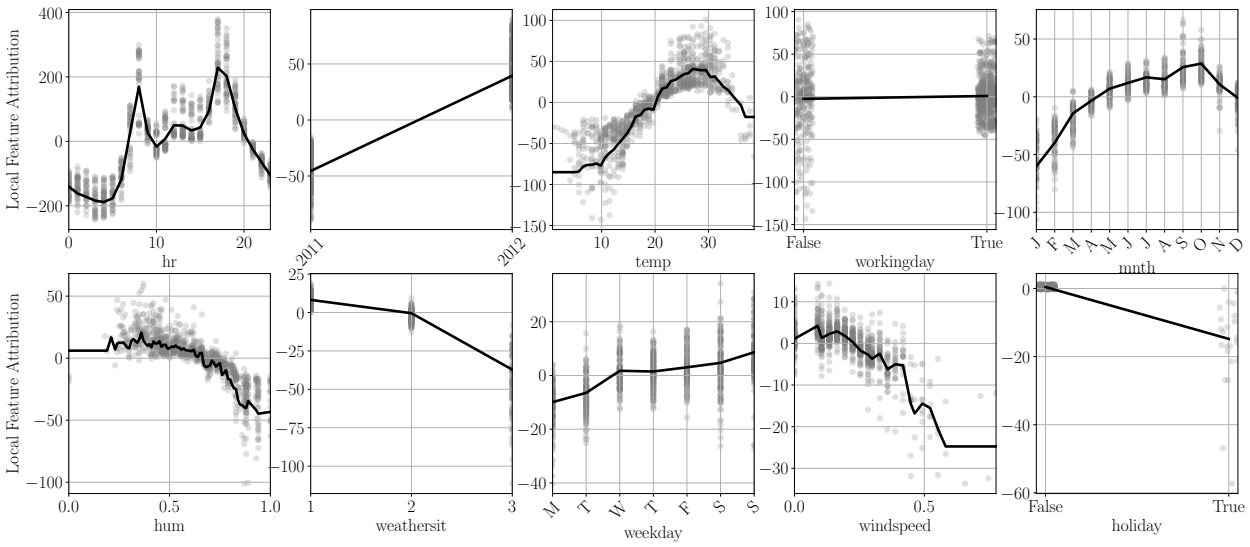
The results are aggregated to yield PDP/SHAP/PFI feature importance, along with their bootstrap confidence intervals to characterize the subsampling disagreement.

```
from pyfd.decompositions import get_PDP_PFI_importance
from pyfd.shapley import get_SHAP_importance
from pyfd.plots import bar

I_PDP, I_PFI, Err_PDP, Err_PFI = get_PDP_PFI_importance(decomposition, bootstrap_error=True)
I_SHAP, Err_SHAP = get_SHAP_importance(shap_values, bootstrap_error=True)
bar([I_PFI, I_SHAP, I_PDP], features.names(), xerr=[Err_PFI, Err_SHAP, Err_PDP])
```



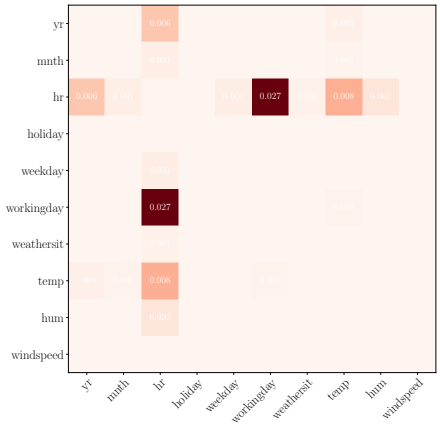
There are strong disagreements between the three methods regarding the importance of the feature `workingday`: PDP gives it no importance, SHAP ranks it fourth, and PFI ranks it second. According to the bootstrap confidence intervals, these disagreements are unlikely attributable to the subsampling, so the model must contain strong interactions involving `workingday`. Feature interactions are also apparent when plotting the PDP alongside the SHAP values.

```
from pyfd.plots import attrib_scatter_plot
attrib_scatter_plot(decomposition, shap_values, background, features, n_cols=5)
```



Shapley Values are poor estimates of the PDP, especially for `hr`, `yr`, `temp`, and `workingday`. This is again indicative of strong feature interactions. To quantify said interactions, we compute the Shapley-Taylor indices using an algorithm inspired by Algorithm 5

```
from pyfd.shapley import taylor_treeshap
from pyfd.plots import interactions_heatmap
# This implementation needs a small background
small_background = X_train[:500]
shap_taylor = taylor_treeshap(model, small_background, small_background)
interactions_heatmap(shap_taylor, features.names())
```



The model contains strong interactions between `hr-workingday`, `hr-temp`, and `hr-yr`. We do not advocate reducing these interactions with grouping. The reason being that `hr` interacts with three different features, so we would need to group `hr-workingday-temp-yr`. Interpreting such a large group might be difficult. We recommend reducing these interactions by fitting a FD-Tree with the CoE objective and using only the features `yr`, `hr`, `workingday`, and `temp` as the split candidates.

```
from pyfd.fd_trees import CoE_Tree

interacting_features = [0, 2, 5, 7]
tree = CoE_Tree(max_depth=2, features=features.select(interacting_features), alpha=0.02)
tree.fit(background[:, interacting_features], decomposition)
tree.print()
>>>
LoA 30.43
if workingday <= 0:
|   if hr <= 8:
|   |   Region 0
|   else:
|   |   Region 1
else:
|   If hr <= 6:
|   |   Region 2
|   else:
|   |   Region 3
Final LoA 9.14
```

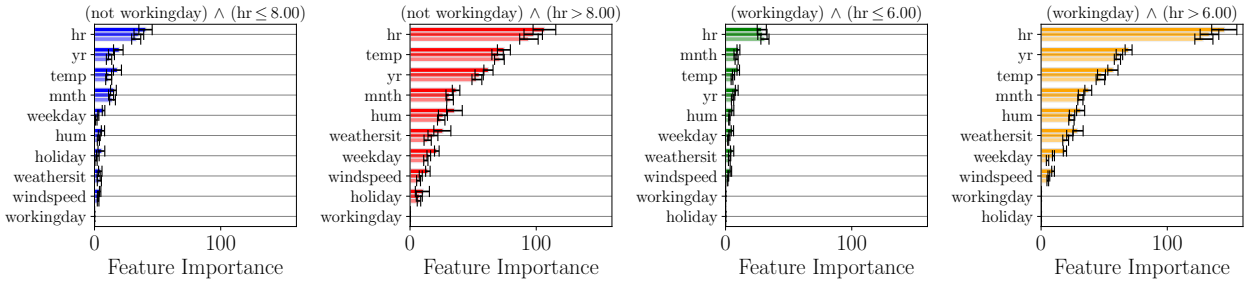
The FD-Tree is able to reduce the CoE from 30% to 9% by separating workingdays from non-workingdays and early-time hours from day-time hours. The functional decompositions and shapley values can then be computed separately on each region Ω using the regional background $\mathcal{B}_\Omega$.

```
# Map each background sample to its region
regions = tree.predict(background[:, interacting_features])
rules = tree.rules(use_latex=True)
```

```
# Iterate over all regions
regional_backgrounds = [[], [], [], []]
regional_decomposition = [[], [], [], []]
regional_shap = [[], [], [], []]
for r in range(4):
    regional_backgrounds[r] = background[regions==r]
    # Regional Decomposition
    regional_decomposition[r] = get_components_tree(model,
                                    regional_backgrounds[r], regional_backgrounds[r])
    # Shapley values
    regional_shap[r] = interventional_treeshap(model,
                                    regional_backgrounds[r], regional_backgrounds[r])
```

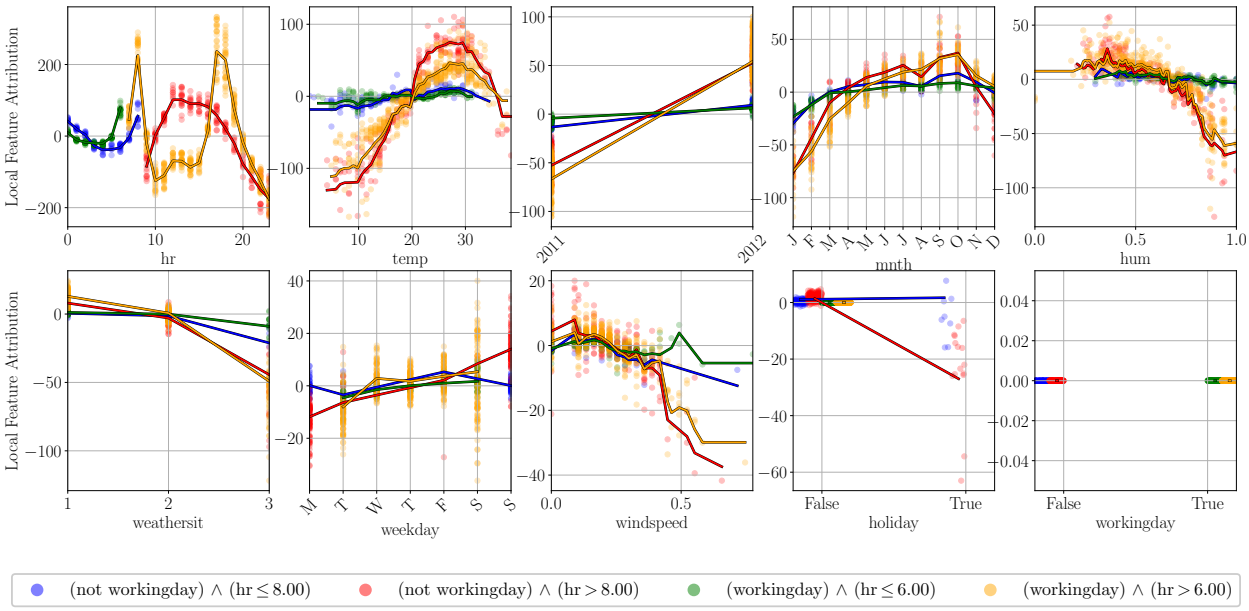
The Global Feature Importance are plotted for each separate region.

```
fig, axes = plt.subplots(1, 4, figsize=(16, 4))
for r in range(4):
    I_PDP, I_PFI, Err_PDP, Err_PFI = get_PDP_PFI_importance(regional_decomposition[r],
                                                            bootstrap_error=True)
    I_SHAP, Err_SHAP = get_SHAP_importance(regional_shap[r], bootstrap_error=True)
    bar([I_PFI, I_SHAP, I_PDP], features.names(),
    xerr=[Err_PFI, Err_SHAP, Err_PDP], ax=axes[r], color=COLORS[r])
    axes[r].set_title(rules[r])
```



Our first observation is that the PDP/SHAP/PFI feature importance agree meaning that the model is "almost additive" on each region. As a result, these additive explanations must be more faithful to the model since they all fall back to ante-hoc/intrinsic explanations for additive models (cf. Equation 2.44). Secondly, the feature `workingday` is no longer given any importance by the three methods. This does not mean that `workingday` is not an important feature! Rather, assigning a score to this feature is ill-posed because it interacts strongly with others. According to the FD-Tree, it is better to interpret `workinday` as a feature that affects the behavior of others. Thirdly, all features have a reduced importance when restricted to early-time hours (blue and green bars). This suggests that the model has less variability within these regions, which is confirmed by plotting the PDP and SHAP local feature attributions over each region.

```
attrib_scatter_plot(regional_decomposition, regional_shap, regional_backgrounds, features)
```



This plot contains many interesting patterns. First, the PDP/SHAP local attributions of `hr` depend on whether it is a working day or not. On workingdays, the local attributions of `hr` peaks at rush hours, while they peak in the afternoon on non-workingdays. This could imply that bikes are mostly rented for work-related purposes on working days, while they are rented for recreational purposes during non-workingdays.

Second, the PDPs of features `temp`, `yr`, `mnth`, `hum`, and `weathersit` are flat whenever the background is restricted to early-time hours (*i.e.* the blue and green curves). Additionally, the Shapley Values are very small in both green and blue regions. This implies that these features have little effect on the model response whenever it is early in the morning. In the case of `temp`, the PDP local attribution is non-flat in the yellow and red regions. This suggests that increasing temperature increases the model output, on average. This effect is more pronounced on non-workingday (red) compared to a workingday (yellow). Note that such behavior is a three-way interaction : the effect of temperature on the model response depends on both the `hr` and `workingay`.

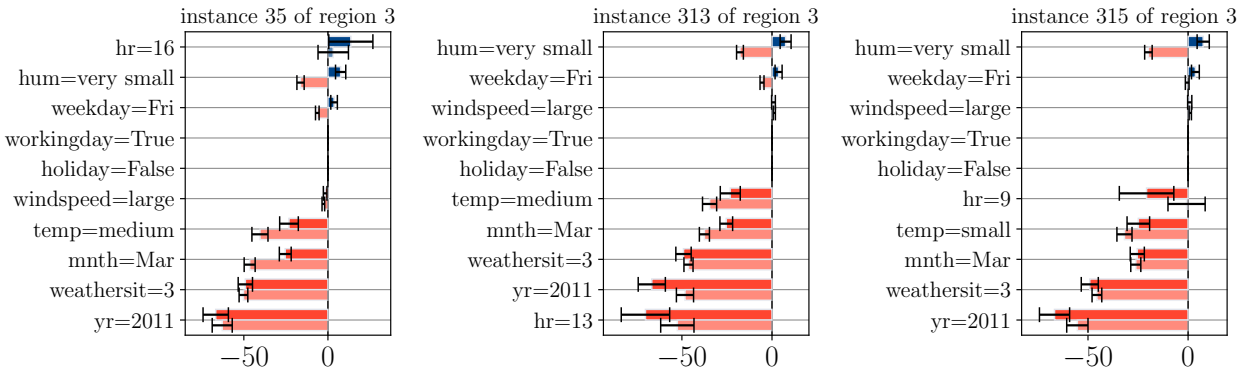
Third, the local attributions of `holiday` present some peculiar behaviors. Whenever it is a working day, the feature has no attribution because it does not vary (it is stuck at `holiday=False`). However, since non-workingdays can be holidays or not, the feature does have an effect on the model in the blue and red regions. In the case of the red region (non-workingdays during daytime hours), the attribution of `holiday=True` is negative, meaning that there are less bike rentals on holidays compared to non-holidays.

We end this analysis of Bike-Sharing by investigating the local feature attributions of a few instances of interests. Looking at the PDP/SHAP attributions for feature `hum`, we note that there are a few yellow points with `hum=0`, a positive PDP attribution, and a negative SHAP attribution. These points are only yellow, so they are all part of region 3. We therefore compute the local feature attributions on this region.

```
from scipy.stats import norm

std_factor = norm.ppf(1-0.025) / np.sqrt(len(regional_backgrounds[3]))
# PDP attribution
PDP_attribution = np.column_stack([regional_decomposition[3][(i,)].mean(1)
                                   for i in range(10)])
Error_PDP = np.column_stack([regional_decomposition[3][(i,)].std(1) * std_factor
                                   for i in range(10)])
# Shapley attribution
SHAP_attribution = regional_shap[3].mean(1)
Error_SHAP = regional_shap[3].std(0) * std_factor

print(PDP_attribution.shape)
>>> (482, 10)
print(SHAP_attribution.shape)
>>> (482, 10)
```

We explain all the points that have `hum=0` in this region.

```
# Explain the point with hum=0 in region 3
idxs = np.where(regional_backgrounds[3][:, 8]==0)[0]
fig, axes = plt.subplots(1, len(idxs), figsize=(12, 4))
for i, idx in enumerate(idxs):
    # Plot
    bar([PDP_attribution[idx], SHAP_attribution[idx]],
        features.print_value( regional_backgrounds[3][idx] ),
        xerr=[Error_PDP[idx], Error_SHAP[idx]],
        ax = axes[i])
    axes[i].set_title(f"instance {idx} of region 3")
```



The fact that the PDP and SHAP attributions do not agree on the sign for `hum=very small` suggests remaining interactions involving `hum`. Still, PDP and SHAP agree that the

features `yr=2011`, `weathersit=3`, and `mnth=Mar` have an important negative attribution on these three instances. Looking at the dataset more closely, we realized that these inputs are all part of the 10th of March 2011, a day with bad weather conditions, few bike rentals, and `hum=0` for all its entries. In fact, this is the only day with `hum=0`, so we suspect it is a typo. Looking at the PDP for feature `hum`, it appears that lower humidity is associated with higher bike rentals. Yet, on the 10th of March 2011, there was `hum=0` and a low amount of bike rentals. Since this is a very local pattern, the statistical association between `hum=0` and low bike rentals was likely modeled by a high-order interaction, which would explain why PDP and SHAP still disagree.

10.2 Predicting Marketing Campaign Success

The Marketing dataset[2] aims to predict the success of a marketing campaign conducted by a Portuguese banking institution. Each of the 45K instances is phone call and the binary target y represents whether the subject has accepted ($y = 1$) or refused ($y = 0$) to make a term deposit. Every single phone call is characterized by 16 features.

Name	Type	Domain	Description
`age`	`num_int`	$\{18, 19, \ldots, 94, 95\}$	Subject's age
`default`	`bool`	{False, True}	Has credit in default?
`balance`	`num`	$[-8019, 102K]$	Average yearly balance
`housing`	`bool`	{False, True}	Has housing loan?
`loan`	`bool`	{False, True}	Has personal loan?
`day`	`num_int`	$\{1, 2, \ldots, 30, 31\}$	Contact's Day
`month`	`ordinal`	{Jan, Feb,..., Nov, Dec}	Contact's Month
`duration`	`num`	$[0, 4918]$	Contact's Duration
`campaign`	`num`	$[0, 275]$	# Contacts in Current Campaign
`pdays`	`num`	$[-1, 871]$	# Days since previous Contact
`previous`	`num`	$[0, 0.85]$	# Contacts in Previous Campaign
`job`	`nominal`	{?,admin,...}	Subject's job
`marital`	`nominal`	{divorced,married,single}	Subject's marital status
`education`	`nominal`	{?,primary,secondary,tertiary}	Subject's education
`contact`	`nominal`	{?,cellular,telephone}	Contact communication type
`poutcome`	`nominal`	{?,other,failure,success}	Outcome of previous campaign

Categorical features with values "?" are missing. Because we do not know the data collection process, we cannot determine with certainty whether these features are *missing completely at random* (MCAR). Features are MCAR if their missingness is induced by Bernoulli latent

[2] `https://archive.ics.uci.edu/dataset/222/bank+marketing`

variables that are independent of any observed and missing data. To assert if features are MCAR, it is recommended to model missing values as unique categories "?", and check if these categories are important to the predictive model h [Hastie et al., 2009, Section 9.6]. Categories "?" that are MCAR should not important to h since their missingness is independent of the target y and other features.

In the interest of space, we will not show how to construct the features object in `PyFD`. We will simply show the result of the `.summary()` methods.

```
features.summary()
>>>
|Idx|        Name        |        Type        |    Card    |     I^-1({i})     |
-------------------------------------------------------------------------------
| 0 | age                | num_int            | 78         | [0]               |
| 1 | default            | bool               | 2          | [1]               |
| 2 | balance            | num                | inf        | [2]               |
| 3 | housing            | bool               | 2          | [3]               |
| 4 | loan               | bool               | 2          | [4]               |
| 5 | day                | num_int            | 31         | [5]               |
| 6 | month              | ordinal            | 12         | [6]               |
| 7 | duration           | num                | inf        | [7]               |
| 8 | campaign           | num                | inf        | [8]               |
| 9 | pdays              | num                | inf        | [9]               |
| 10| previous           | num                | inf        | [10]              |
| 11| job                | nominal            | 12         | [11]              |
| 12| marital            | nominal            | 3          | [12]              |
| 13| education          | nominal            | 4          | [13]              |
| 14| contact            | nominal            | 3          | [14]              |
| 15| poutcome           | nominal            | 4          | [15]              |
-------------------------------------------------------------------------------
```

The first step of the pipeline was splitting the dataset into train/test sets with ratios 0.8:0.2. The training set was used to select the final model and the test set yielded an unbiased measure of its performance. Because the dataset is unbalanced (89% of the data has $y = 0$), the F1-Score was employed to assess performance.

As done with the BikeSharing use-case, interactions were only be included in the model if they were necessary to attain high performance. Like previously, we first investigated additive models (Linear, Splines, EBM1, GBTs with `max_depth=1`) as well as models with pair-wise interactions (EBM2, GBTs with `max_depth=2`), before using full complexity models (GBTs, RFs, MLPs). Figure 10.3 illustrates the Pareto front of the five-fold cross-validation F1-Score and the Cost of Exclusion. This front highlights the optimal tradeoffs between performance and interactions. Going from the best additive model (EBM1) to the best model with pair-wise interactions (EBM2) results in a F1-Score increase from 50% to 54%, while the CoE increases from 0% to 37%. Going from pairwise interactions to a GBT marginally increases the F1-Score from 54% to 54.5% and doubles the magnitude of the CoE (going from 37% to

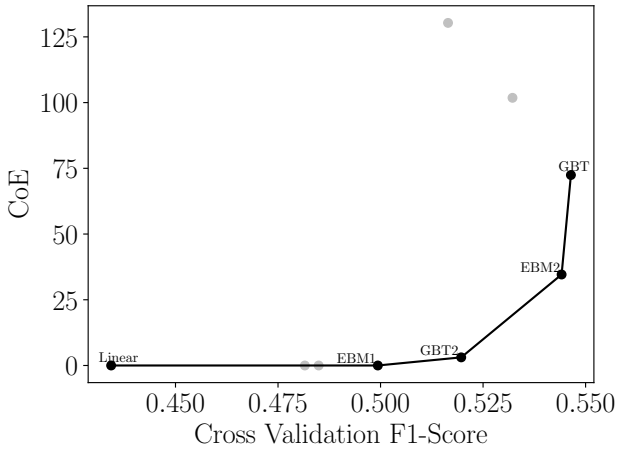


Figure 10.3 Pareto front showing the optimal tradeoffs between performance and interaction strength on the Marketing use-case.

72.5%). While the GBT does perform better than the EBM2 with pair-wise interactions, its CoE is twice as large meaning that it will be harder to extract faithful additive explanations from it. For instance, a FD-Tree fitted on the GBT might require more leaves to attain low CoE. For this reason, the remainder of the analysis was applied to the EBM2 model whose test set F1-Score was 55.1%.

Given a EBM2, we subsample 2000 training points to define the background distribution $\mathcal{B}$. Then we compute its $\boldsymbol{z}$-Anchored Decompositions via `get_components_ebm`, a model-specific implementation optimized for EBMs with pair-wise interactions

```
from pyfd.decompositions import get_components_ebm, get_CoE
# Compute Anchored Decompositions with a subsample of data
background = X_train[:2000]
decomposition = get_components_ebm(model, background, background, anchored=True)
print(get_CoE(decomposition))
>>> 37.76
```

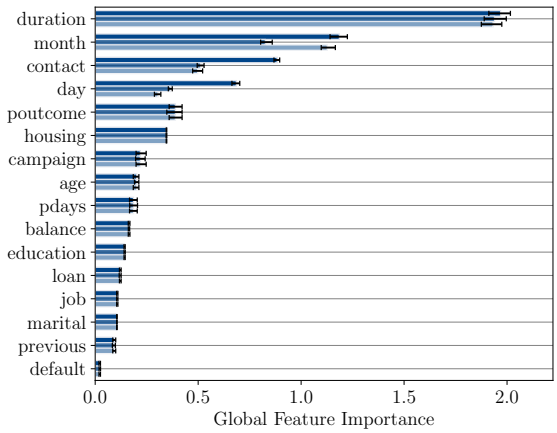
We can also compute the Shapley Values using `shap_from_decomposition` since we are able to compute the full $\boldsymbol{z}$-Anchored Decompositions for EBMs.

```
from pyfd.shapley import shap_from_decomposition
shap_values = shap_from_decomposition(decomposition)
```

The results are aggregated to yield PDP/SHAP/PFI feature importance, alongside their bootstrap confidence intervals that characterize the subsampling disagreement.

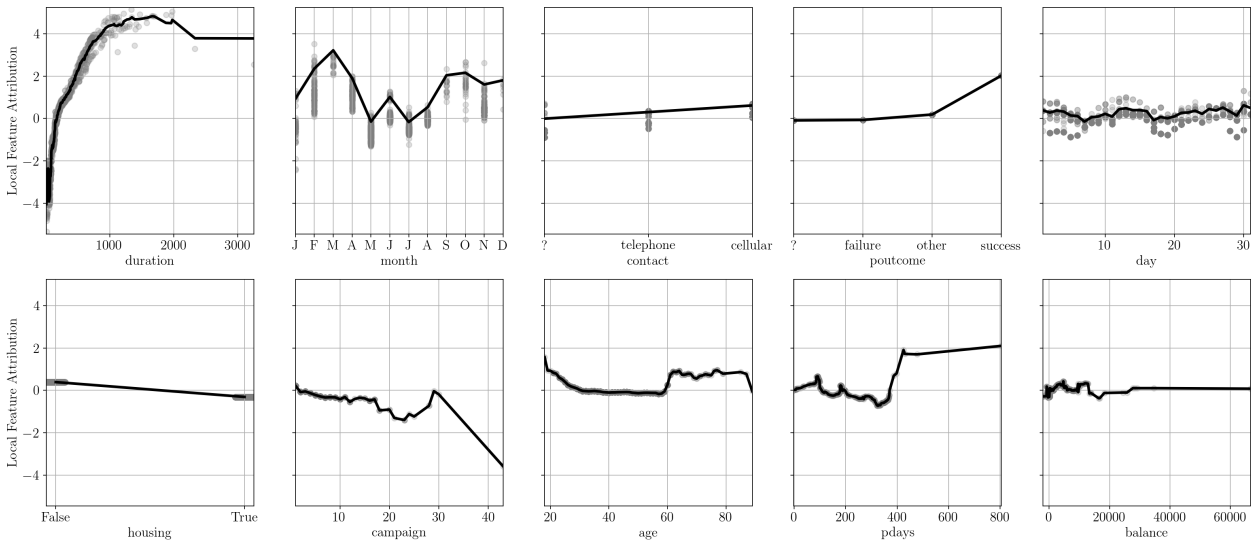
```
from pyfd.decompositions import get_PDP_PFI_importance
from pyfd.shapley import get_SHAP_importance
from pyfd.plots import bar

I_PDP, I_PFI, Err_PDP, Err_PFI = get_PDP_PFI_importance(decomposition, bootstrap_error=True)
```

```
I_SHAP, Err_SHAP = get_SHAP_importance(shap_values, bootstrap_error=True)
bar([I_PFI, I_SHAP, I_PDP], features.names(), xerr=[Err_PFI, Err_SHAP, Err_PDP])
```



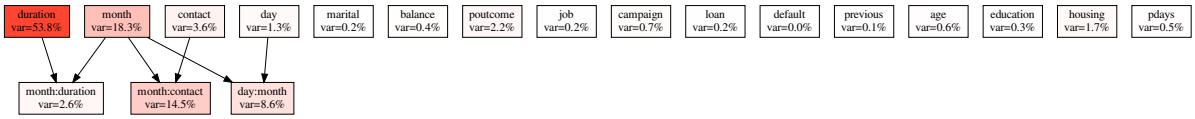
PDP, SHAP, and PFI disagree regarding the importance of the features `month`, `contact`, and `day`. Disagreements are also apparent when plotting the PDP/SHAP local attributions.

```
from pyfd.plots import attrib_scatter_plot
# idxs = 10 only plots the top-10 features
attrib_scatter_plot(decomposition, shap_values, background, features, idxs=10)
```



In these plots, we see that the PDP of `month` is a poor estimate of the Shapley values. Visualizing the lattice space confirms that the features `month-duration`, `month-contact`, and `day-month` are involved in strong pair-wise interactions.

```
from pyfd.plots import decomposition_graph
dot = decomposition_graph(decomposition, features.names())
```



How can we reduce these interactions? By lucky coincidence, the two features `day`-`month` that interact also happen to share a semantic meaning. They both relate to the concept of "current date". As a result, it makes sense to treat them as a single feature `day:month`. So, instead of attributing a separate importance to `day=12` and `month=jun`, we can provide a single attribution to the date `day=12:month=jun`.

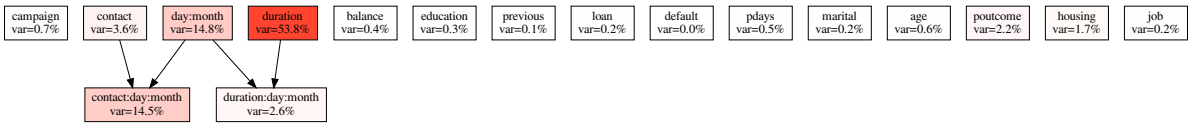
```
grouped_features = features.group([[5, 6]])
grouped_features.summary()
>>>
|Idx|        Name         |        Type         |    Card     |     I^-1({i})     |
-------------------------------------------------------------------------------------
| 0 | age                 | num_int             | 78          | [0]               |
| 1 | default             | bool                | 2           | [1]               |
| 2 | balance             | num                 | inf         | [2]               |
| 3 | housing             | bool                | 2           | [3]               |
| 4 | loan                | bool                | 2           | [4]               |
| 5 | duration            | num                 | inf         | [7]               |
| 6 | campaign            | num                 | inf         | [8]               |
| 7 | pdays               | num                 | inf         | [9]               |
| 8 | previous            | num                 | inf         | [10]              |
| 9 | job                 | nominal             | 12          | [11]              |
| 10| marital             | nominal             | 3           | [12]              |
| 11| education           | nominal             | 4           | [13]              |
| 12| contact             | nominal             | 3           | [14]              |
| 13| poutcome            | nominal             | 4           | [15]              |
| 14| day:month           | num_int:ordinal     | 31:12       | [5, 6]            |
-------------------------------------------------------------------------------------
```

The pre-image $\mathcal{I}_{\xi}^{-1}$ shows that the new feature `day:month` relates to the columns 5 and 6 of the data. This pre-image must be passed to the `PyFD` primitives to inform them that both columns are treated as a single feature.

```
# Compute the functional decomposition
grouped_decomposition = get_components_ebm(model, background, background,
                                           Imap_inv=grouped_features.Imap_inv,
                                           anchored=True)
print(get_CoE(grouped_decomposition))
>>>17.78
```

Grouping `day` and `month` has reduced the CoE from 37% to 17%, but there are still some interactions left in model.

```
dot = decomposition_graph(grouped_decomposition, grouped_features.names())
```



Notably, the feature `contact` is still involved in interactions with `day:month`. These interactions can be minimized by partitioning the input space with a FD-Tree.

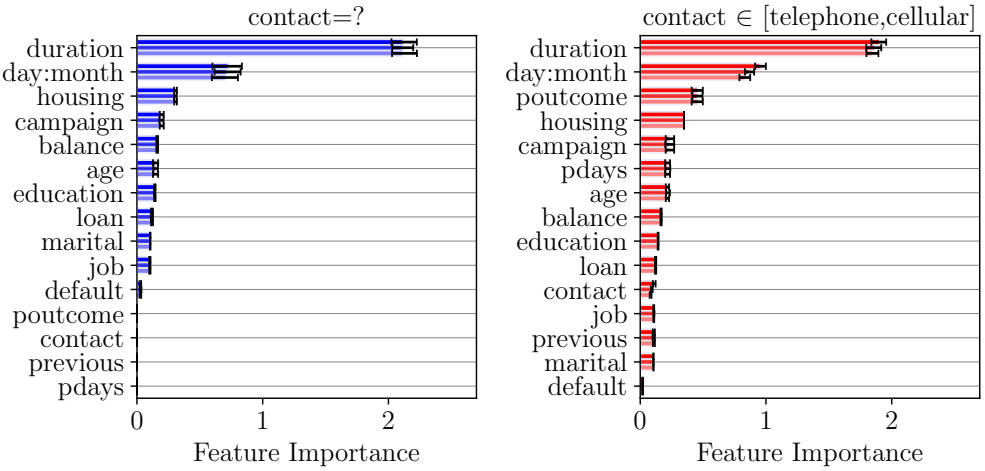
```
from pyfd.fd_trees import CoE_Tree
from pyfd.plots import plot_legend, COLORS

# Fit a FDTree by passing the `grouped_decomposition` as argument.
tree = CoE_Tree(max_depth=1, features=features, alpha=0.01)
tree.fit(background, grouped_decomposition)
tree.print()
>>>
CoE 17.78
If contact <= 0.0000:
|   Region 0
else:
|   Region 1
Final CoE 2.34
```

The FD-Tree made its split along the feature `contact` at value zero leading to the regions `contact` =? and `contact` $\in$ {telephone,cellular}, and a CoE of 2.34%. We now explain the model on each separate region.

```
# Get the region of each background sample
regions = tree.predict(background)
# Get the rules describing the regions
rules = tree.rules(use_latex=True)

# Iterate over both regions
regional_backgrounds = [[], []]
regional_decomposition = [[], []]
regional_shap = [[], []]
for r in range(2):
    regional_backgrounds[r] = background[regions==r]
    # Regional Decomposition
    regional_decomposition[r] = get_components_ebm(model,
                                            regional_backgrounds[r],
                                            regional_backgrounds[r],
                                            Imap_inv=grouped_features.Imap_inv,
                                            anchored=True)
    # Shapley values
    regional_shap[r] = shap_from_decomposition(regional_decomposition[r])
```

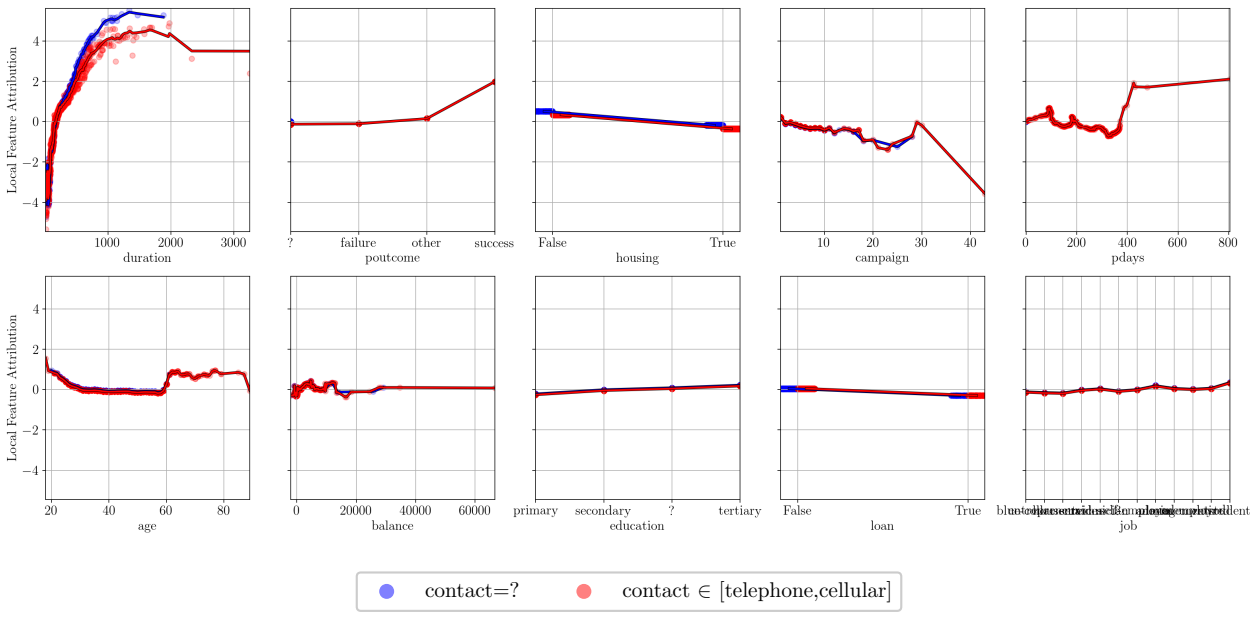
The global feature importance are plotted for each separate region

```
fig, axes = plt.subplots(1, 2, figsize=(8, 4))
for r in range(2):
    I_PDP, I_PFI, Err_PDP, Err_PFI = get_PDP_PFI_importance(regional_decomposition[r],
                                                            bootstrap_error=True)
    I_SHAP, Err_SHAP = get_SHAP_importance(regional_shap[r], bootstrap_error=True)
    bar([I_PFI, I_SHAP, I_PDP], grouped_features.names(),
        xerr=[Err_PFI, Err_SHAP, Err_PDP], ax=axes[r], color=COLORS[r])
    axes[r].set_xlim(0, 2.7)
    axes[r].set_xlabel("Feature Importance")
    axes[r].set_title(rules[r], fontsize=15)
```



in addition to the local feature attributions.

```
attrib_scatter_plot(regional_decomposition, regional_shap,
                    regional_backgrounds, grouped_features, idxs=10)
```



We can make several observations from these regional additive explanations.

First, the feature `duration` is the most important one in both regions, demonstrating that it is the most important for the EBM2 model. Looking at the PDP/SHAP local attributions in both regions, there appears to be an increasing relationship between the call duration and the probability of success. This seems trivial in hindsight. Phone calls rejecting the offer would be short and sweet "No thanks, I am not interested (Hangs up)". Successful phone calls would be longer since, upon acceptance, the bank would need to discuss logistic details regarding the term deposit. Although the `duration` feature is highly predictive of the target variable, its causal relationship is backward: success y is a cause of `duration`, rendering the model inappropriate for planning future campaigns. For example, it is nonsensical to encourage longer phone calls to increase the probability of successes.

Second, the feature `campaign` is important in both regions and increasing it decreases the model output according to both PDP and SHAP. Perhaps clients which were more frequently called became annoyed and were more prone to reject the bank proposition. Nevertheless, the model is non-causal and so be careful when recommending employees to call clients less frequently on future campaigns.

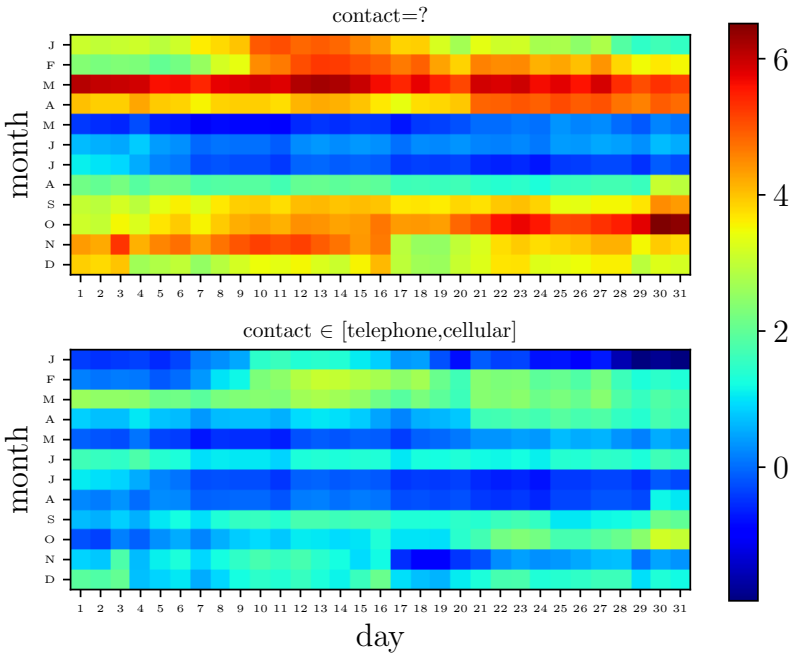
Third, the feature `poutcome` (which measures the success of a previous campaign) is important on the red region, while it is not important in the blue region. Looking at the local feature attributions, in the red region, there is a large positive attribution for `poutcome=success` implying that persons who previously made a deposit based on phone calls are more likely to do it again. In the blue region, however, the feature `poutcome` is consistently missing (equal to "?"). Thus, the missingness of features `contact` and `poutcome` are statistically associated.

Is `contact` missing completely at random (MCAR)? The fact that FD-Trees minimized feature interaction via the regions `contact` =? and `contact` $\in$ {telephone,cellular} suggests that missigness impacts the effects of other features response. To confirm it, we plot the main effects $h_{j,\mathcal{B}_\Omega}(\boldsymbol{x})$ of feature `day:month` over the blue and red regions. By setting the foreground $\mathcal{F} \neq \mathcal{B}$ to a 31×12 grid, we can provide the most detailed plot possible.

```
# Foreground as a grid
xx, yy = np.meshgrid(np.arange(1, 32), np.arange(12))
foreground = np.column_stack((xx.ravel(), yy.ravel()))
# Main effect of day:month on Region 0
decomp = get_components_ebm(model, foreground, regional_backgrounds[0],
                            Imap_inv=[[5, 6]], anchored=False)
zz_0 = decomp[(0,)].reshape(xx.shape)
# Main effect of day:month on Region 1
decomp = get_components_ebm(model, foreground, regional_backgrounds[1],
                            Imap_inv=[[5, 6]], anchored=False)
zz_1 = decomp[(0,)].reshape(xx.shape)
```

Since the feature `day:month` is a group, its main effect $h_{j,\mathcal{B}_\Omega}(\boldsymbol{x})$ is plotted as a heatmap on each region.

```
from matplotlib.colors import Normalize
import matplotlib.cm as cm
cmap = cm.get_cmap('jet')
normalizer = Normalize(min(zz_0.min(), zz_1.min()), max(zz_0.max(), zz_1.max()))

fig, axes = plt.subplots(2, 1)
# Region 0
axes[0].set_title(rules[0], fontsize=10)
axes[0].imshow(zz_0, cmap=cmap, norm=normalizer)
axes[0].set_ylabel("month")
axes[0].set_xticks(np.arange(31), labels=range(1, 32), fontsize=5)
axes[0].set_yticks(np.arange(12), labels=features.feature_objs[6].cats, fontsize=5)

# Region 1
axes[1].set_title(rules[1], fontsize=10)
axes[1].imshow(zz_1, cmap=cmap, norm=normalizer)
axes[1].set_xlabel("day")
axes[1].set_ylabel("month")
axes[1].set_xticks(np.arange(31), labels=range(1, 32), fontsize=5)
axes[1].set_yticks(np.arange(12), labels=features.feature_objs[6].cats, fontsize=5)
fig.colorbar(ax=axes.ravel().tolist(), mappable=cm.ScalarMappable(norm=normalizer, cmap=cmap))
```



According to this plot, the impact of varying `month` on model output depends on whether the `contact` is known or not. When `contact` =?, the model output drops significantly on summer months compared to other months. Such large drops on summer months are not apparent when the feature `contact` is not missing. Given this observation, the fea-

ture `contact` cannot be MCAR. If it were, missingness would not affect the statistical associations between `day`, `month` and the target y.

Given our limited understanding of how the Marketing data was collected, we cannot address the distortion induced by missing the `contact` feature. The best we can do is separate instances with missing and non-missing `contact` values. Doing so, we can compute regional additive explanations that are more aligned, as evidenced with the low Cost of Exclusion.

Contributions

The `PyFD` package is more than a collection of primitives for computing $\boldsymbol{z}$-Anchored Decompositions and additive explanations. It is a systematic methodology for explaining models involving feature interactions. As seen in the two use-cases, `PyFD` lets users measure the strength of interactions by either computing the CoE, or by reporting disagreements on the PDP/SHAP/PFI global feature importance or PDP/SHAP local feature attributions. Moreover, the package provides tools for minimizing interactions : grouping interacting features and partitioning the input space with a FD-Tree.

In the BikeSharing use-case, we saw that fitting a depth-2 FD-Tree would separate workingdays from non-workingdays and early-time hours from day-time hours. The additive explanations exhibited heterogeneous behaviors over the different regions. Most notably, the local attribution of the temperature was small whenever it was early in the morning. However, during the day, the local attribution of temperature would increase as it got hotter, and this effect was more pronounced on non-workingdays compared to workingdays. This is possibly because hot weather encourages recreational biking more than work-related biking.

In the Marketing dataset, grouping the features `day` and `month`, which are both semantically related to the concept of "current date" led to a considerable reduction in interaction strength. Going further, FD-Trees made use realize that the `contact` feature was not missing completely at random. Consequently, separating samples with and without missing contact decreased interaction strength even more.

CHAPTER 11 CONCLUSION

11.1 Contributions

As machine learning models grow in complexity, so does the demand for transparency and explainability of their decisions. As a result, the subfield of eXplainable Artificial Intelligence (XAI) has recently risen under the promise of shedding light on the reasoning behind complex model predictions. Despite considerable progress since its inception, the XAI field is facing multiple roadblocks.

One of them is the so-called *Disagreement Problem*, where different explainability methods provide contradictory descriptions of model behavior [Krishna et al., 2022]. Given the absence of *ground-truths* representing the correct explanation, there is no systematic methodology practitioners can follow to address said disagreements. While a considerable amount of the literature is currently focusing on developing *faithfulness metrics* to benchmark explanation methods, it was shown that these said metrics are inconsistent : an explanation can be ranked as the most faithful according to a metric and as the least faithful according to another [Tomsett et al., 2020]. Consequently, faithfulness metrics are currently not sufficient to help practitioners decide which explanation to trust whenever there are disagreements.

To address the lack of ground-truth in explainability, we propose a new direction : *instead of benchmarking explanations, we should increase their alignment.* To this end, the first part of the Thesis has unified post-hoc additive explanations through the lens of functional decompositions. More precisely, we have shown that all additive explanations techniques can be expressed as an aggregation of $\boldsymbol{z}$-Anchored Decompositions. Crucially, unifying additive explanations sheds light on why disagreements occur : explanation techniques follow different strategies to redistribute feature interactions among the individual features involved. For example, Partial Dependence Plots (PDP) ignore feature interactions altogether, SHapley Additive exPlanations (SHAP) share interactions evenly between all features involved, and Permutation Feature Importance (PFI) counts the same interaction multiple times. Moreover, we have proven that, when all explainability techniques converge to the same result, then they coincide with the ante-hoc (built-in) explanation of an additive model. Since ante-hoc explanations are by definition faithful, the goal of computing faithful explanations is equivalent to the goal of aligning post-hoc explainers.

Given that we should minimize the disagreements between explanations, we defined the *Interaction*, *Subsampling*, and *Underspecification* Disagreements and demonstrated how to

minimize them.

- The *Interaction Disagreement* reports how strong interactions prohibit agreement among the PDP/SHAP/PFI additive explanations. To reduce feature interactions, we advocate partitioning the input space using a FD-Tree (Functional Decomposition Tree) and then restrict the post-hoc explanations to the leaves of the tree.

- The *Subsampling Disagreement* reports the instability induced by the need to subsample data in order to efficiently compute post-hoc additive explanations. We proposed reporting confidence intervals alongside additive explanation to justify whether more samples are required. In most practical cases, subsamples are chosen uniformly at random and confidence intervals should capture the actual explanation with high-probability. Nonetheless, we also presented an audit scenario where the subsample can be cherry-picked by an ill-intentioned company. In such settings, the confidence intervals fail to capture the actual explanation, unbeknownst to the auditor.

- The *Underspecification Disagreement* tackles the existence of an equivalence class of models with good empirical performance (*i.e.* a Rashomon Set). These models might provide contradicting local/global additive explanations, which makes it hard to derive insights from them. We advocated aggregating the additive explanations of competing models into *partial orders* by asserting consensus across all models in the Rashomon Set. The agreement rate between models was proposed as a disagreement score, and was reduced by either treating correlated features as a group, or by increasing the Gap for inputs with predictions near the mean.

Finally, we developed a package `PyFD` (Python Functional Decomposition), which regroups all previous contributions into a unique framework. This library is the culmination of parts I and II of the manuscript.

- `PyFD` implements the algorithms presented in part I in order to efficiently compute $\boldsymbol{z}$-Anchored Decompositions for a variety of models (Linear Models, Additive Models, Tree Ensembles, Neural Networks, etc.). These fine-grained functional decompositions are then aggregated by the user to compute the additive explanation of their choice. Within this framework, computing additive explanations is no longer about calling various high-level APIs, but requires making conscious decisions such as how to aggregate anchored decompositions, which features to treat as a group, and what backgrounds to use : the full data $\mathcal{B}$ or a regional distribution $\mathcal{B}_\Omega$.

- Part II introduced disagreement scores that are not specific to any additive explainer. Rather, they quantify the quality of the contrastive question: *what features cause the prediction $h(\boldsymbol{x})$ to be higher/lower than the average prediction $\mathbb{E}_{z\sim\mathcal{B}}[h(\boldsymbol{z})]$ over the background $\mathcal{B}$?*. Consequently, `PyFD` provides many tools for increasing alignment to ensure that the contrastive question is well posed. The Interaction Disagreements prohibit alignment among the additive explanations, so users cannot know for certain what features explain the gap. `PyFD` minimizes feature interactions by treating interacting features as a group or by partitioning the input space with a FD-Tree. Doing so ensures that all additive explainers agree. The Subsampling Uncertainty addresses fragility of the contrastive question w.r.t the choice of data subsample. The library supports confidence intervals to report the impact of this uncertainty. The Underspecification Disagreements remain to be implemented in the package.

The `PyFD` workflow illustrated on a toy example and two real-world datasets. `PyFD` allowed us to gain several insights from the models and data : 1) the impact of hotter temperatures on bike rentals depends on the time and whether it is a workingday or not 2) partitioning the input space can determine whether a feature is missing completely at random or not.

11.2 Future Work

There remains many challenges to tackle before providing trustworthy and explainable machine learning systems. Here, we offer our perspective on which research directions should be investigated next.

FD-Box In chapter 6, partitioning the input space $\mathcal{X}$ via a FD-Tree was shown to considerably reduce the strength of feature interactions. To keep these trees interpretable, however, we had to limit their depth to 3. Consequently, FD-Trees cannot identify very fine-grained regions. Future work could remedy this issue by dropping the constraints of partitioning the whole input space, and instead focus on fitting a *single* hyper-box $\Omega = \prod_{j=1}^{d} \Omega_j$ with $\Omega_j \subseteq \mathcal{X}_j$ that contains an instance of interest $\boldsymbol{x} \in \Omega$. We envision using a variant of the PRIM algorithm [Friedman and Fisher, 1999] to rapidly find the hyper-box. Once it is obtained, the hyper-box could provide algorithm recourse to user $\boldsymbol{x}$. In fact, assuming the model is approximately additive over Ω, then main effects $h_{j,\mathcal{B}_\Omega}(\boldsymbol{x})$ would faithfully describe the behavior of h and user $\boldsymbol{x}$ could use them to infer the necessary changes $\boldsymbol{\Delta}$ to get a different prediction $h(\boldsymbol{x} + \boldsymbol{\Delta}) \neq h(\boldsymbol{x})$.

Fool SHAP 2.0 Chapter 7 demonstrated that the FoolSHAP algorithm can identify which data points to cherry-pick in order to minimize the importance of a sensitive feature, while remaining representative of the whole dataset from the point of view of an external auditor. FoolSHAP has several limitations, which we aim to address in future work. First, our experiments considered a single sensitive features whose attribution we wished to decrease. For example, in Adult-Income we reduced the importance of the sensitive `gender` feature. Nonetheless, there are two other features in Adult-Income that share information with `gender`: `relationship` and `marital-status`. In fact, `relationship` can take the value `widowed` and `marital-status` can take the value `wife`, which are both proxies of `gender=female`. For this reason, these two other features may be considered sensitive and decision-making that relies strongly on them may not be acceptable. Hence, future implementations should reduce the total attributions of all three features.

Second, it was assumed for simplicity that S'_0 was sampled uniformly at random and then fixed. Afterward, the cherry-picking was applied on the other set S'_1. This simplification allowed to formulate the optimization of the cherry-picking weights $\boldsymbol{\omega}$ as a Linear Program with optimality guarantees. Still, future work should cherry-pick the foreground and background simultaneously. This will lead to a Bilinear Program optimizing weights $\boldsymbol{\omega}^{\mathrm{f}}$ and $\boldsymbol{\omega}^{\mathrm{b}}$ for the foreground and background respectively. Bilinear Programs are non-convex optimization problem that can be solved to local optima via the coordinate descent algorithm [Nahapetyan, 2009].

Third, FoolSHAP requires that the auditor can only access data through the model output. That is, the auditor can only detect a fraud by comparing histograms $h(D_0)$,$h(D_1)$ with $h(S'_0)$,$h(S'_1)$. This assumption was made because sharing model outputs respects privacy concerns regarding the dataset D. Yet, there may exist information on D that can be shared without revealing private information. For instance, the company could be required to share mean and variance of certain input features. FoolSHAP would then need to take those extra constraints into account when cherry-picking data that "looks like" the whole data. As future work, we envision an automated process that maps any statistical test the auditor wishes to conduct into hard constraints of a Bilinear Program.

Rashomon Set Chapter 8 asserted consensus on additive explanations over the Rashomon Set by solving optimization problems over $\mathcal{H}$ under the constraint that $\widehat{\mathcal{L}}_S(h) \leq \epsilon$. When investigating the local attributions of d features on N inputs, there are $\mathcal{O}(Nd^2)$ such optimization problems to solve. This becomes quickly impractical unless these problems have closed-form solutions, which was the case for Parametric Additive Regression, Kernel Ridge Regression, and Random Forest. For alternative hypothesis spaces, we propose two solutions

that should be explored in future work : 1) Ensembles and 2) Ellipsoid Estimates.

First, practitioners could train an ensemble of diverse models $E := \{h_k\}_{k=1}^M$ and keep the ones with an empirical loss below ϵ. They would consequently underestimate the Rashomon Set $\mathcal{R}(E, \epsilon) \subset \mathcal{R}(\mathcal{H}, \epsilon)$ but asserting consensus would become as simple as checking each individual model. This approach is straight-forward but it introduces an undesirable uncertainty : *what if I train one more model? Would it suddenly disagree on the importance of feature j?*. This uncertainty is not addressed by our statistical guarantees from Section 8.2.5, which only ensure that models with $\widehat{\mathcal{L}}_S(h) > \epsilon_{\text{max}}$ are likely suboptimal and can be safely excluded. These results do not guarantee that any new model added to the ensemble would agree with the others. Additionally, there is a risk that models in the ensemble are not diverse enough so consensus will be artificially high. Training diverse MLPs is possible by initializing them with different parameters since their loss is non-convex. Also, one can train diverse ensembles of GBTs and EBMs by fitting each model on bootstrapped data and introducing additional stochasticity in the tree growth [Friedman, 2002].

Second, it might be possible to extend Ellipsoid Rashomon Sets to other hypothesis spaces. For instance, the Rashomon Sets of Additive Logistic Regression and Kernel Logistic Regression could be approximated using a second-order Taylor Expansion of the loss, that would lead to an Ellipsoid Rashomon Set estimate $\widehat{\mathcal{R}}(\mathcal{H}, \epsilon)$. Future work could derive the corresponding capture bound $\mathbb{P}_{S\sim\mathcal{D}^N}[h^\star \in \widehat{\mathcal{R}}(\mathcal{H}, \epsilon_{\text{max}})] > 1 - \delta$ using the asymptotic theory of Maximum Likelihoods Estimates [Wood, 2017, Appendix A].

Sensitivity vs Attribution This thesis has focused on a form of local explanations $\boldsymbol{\phi}^{\text{LFA}}(h, \boldsymbol{x}, \mathcal{B})$ called "Local Feature Attributions" (LFA). These local explanations provide one score per feature and are meant to convey how influential said feature is toward the Gap $h(\boldsymbol{x}) - \mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{z})]$. A large positive attribution is interpreted as stating that feature j increases the model output relative to the mean. Similarly, a large negative attribution implies that feature j decreases the model output relative to the mean. One of the main takeaway of this manuscript it that LFAs have a natural form $\phi_j^{\text{LFA}}(h, \boldsymbol{x}, \mathcal{B}) = h_j(x_j) - \mathbb{E}_{z_j\sim\mathcal{B}_j}[h_j(z_j)]$ whenever the underlying model is additive *i.e.* $h(\boldsymbol{x}) = \omega_0 + \sum_{j=1}^d h_j(x_j)$. Following this definition, feature j contributes greatly to the Gap if the value $h_j(x_j)$ is far from its average $\mathbb{E}_{z_j\sim\mathcal{B}_j}[h_j(z_j)]$.

While a LFA $h_j(x_j) - \mathbb{E}_{z_j\sim\mathcal{B}_j}[h_j(z_j)]$ conveys the influence of feature j toward the Gap, it does not tell you what happens when x_j increases/decreases. Such a measure of local importance would be better conveyed by the *Sensitivity* $\phi_j^{\text{Sensitivity}}(h, \boldsymbol{x}) = \frac{d}{dx_j} h_j(x_j)$ that returns the slope of the model at the point of interest. In that case, a positive slope informs the user

that slightly increasing x_j increases the model output. These form of local explanations could be useful to provide Algorithmic Recourse to users who wish to change the model outcome.

Very recent work was shown that local *Attribution* and *Sensitivity* are complementary approaches to local explainability [Bilodeau et al., 2024]. The main Theorem of the paper states that, whenever the hypothesis space $\mathcal{H}$ is rich enough, LFAs and Sensitivity explanations can freely be positive/negative/null regardless of the other. This makes intuitive sense when the model is additive because one method returns a difference $h_j(x_j) - \mathbb{E}_{z_j \sim \mathcal{B}_j}[h_j(z_j)]$ while the other returns a slope $\frac{d}{dx_j} h_j(x_j)$. If $\mathcal{H}$ is a rich enough, then there are no constraints between the value $h_j(x_j)$ and the slope $\frac{d}{dx_j} h_j(x_j)$. The slope can be null while $h_j(x_j)$ is far from the average, or the slope can be large while $h_j(x_j)$ is near the average.

Since Sensitivity methods are complementary to LFAs, future work should include them in our unified theory. It would also be crucial to understand how the Interaction, Subsampling, Underspecification Disagreements affect Sensitivity-based local explanations.

Understanding the Learning Algorithm We have tackled the challenge of understanding models h that contain feature interactions. Yet, throughout the manuscript, we have not cared *how* that model was chosen. The main characteristic of Machine Learning is that predictive models h are yielded by a *learning algorithm* $h = \mathcal{A}_\Psi(S)$ applied on training data S using hyperparameters Ψ. It is therefore of practical interest to not only explain the selected model h, but also the algorithm $\mathcal{A}$ that learned it. Explaining learning algorithms can take multiple forms.

First, running a learning algorithm requires specifying the input space $\mathcal{X} = \prod_{j \in J} \mathcal{X}_j$ in which the data exists. The choice of the feature subset $J \subseteq [d]$ is referred to as feature selection. Within this context, a relevant question is: *does the learning algorithm $\mathcal{A}$ need feature j to produce an accurate predictor?*. If the answer is no, then the feature could be removed from J without inhibiting the algorithm from producing good models. This notion of global feature importance is different from the one investigated in this manuscript : *does a fixed model h use feature j locally/globally?*. To understand the subtle difference between both notions, imagine a scenario where features x_i and x_j are redundant *i.e.* $\mathbb{P}[x_i = x_j] = 1$. Then, some model h learned on this data might rely solely on feature j to make its predictions. Yet, the learning algorithm $\mathcal{A}$ does not *need* feature j to produce good predictors since feature i could be used in its stead. The difference between both notions of importance is reminiscent of what Fisher et al. [2019] label the Model Reliance and the Algorithmic Reliance.

Existing work has investigated the Algorithmic Importance of feature j by training one model on the full feature set $[d]$ and another on the reduced feature set $[d] \setminus \{j\}$ [Williamson

et al., 2021]. Under the assumption that the squared loss is used, $\mathcal{H}$ is rich enough, and that the data size grows to infinity, [Hooker et al., 2021, Theorem 2] has demonstrated that this importance score converges to the Total Sobol Index [Bénard et al., 2021]. This importance measure conveys whether feature j is *necessary* to produce a good predictor. Future work should investigate additional feature subsets $[d] \setminus \{i, j\}$ in order to highlight potential interactions.

Second, the critical role of the hyperparameters Ψ in the learning algorithm cannot be understated. A bad choice of hyperparameters often leads to subpar performance. Nonetheless, the choice of hyperparameters is often made through an exhaustive search because their impact on the algorithm is opaque. In fact, through-out all experiments in this thesis, the hyperparameters were selected via a random search $\Psi \sim \mathcal{B}$ whenever the grid-search was too costly. This led to models h with good generalization performance, but it remains frustrating to not understand this part of the ML pipeline. Since the hyperparameters are sampled via a random distribution, it might be possible to extend the notions of Interventional/ANOVA functional decompositions to decompose the function that maps Ψ to any quantity of interest $\phi(\mathcal{A}_\Psi(S)) \in \mathbb{R}$ about the resulting model (*e.g.* , the cross-validation performance). Decomposing this complex function of Ψ would shed light on which hyperparameters are important/unimportant and involved in interactions.

Third, the learning algorithm is fed data $S = \{(\boldsymbol{x}^{(i)}, y^{(i)})\}_{i=1}^N$ to produce a predictor h. It is therefore natural to ask : *how important are certain samples toward the selection of the model? Or toward a certain characteristic of the model?*. These questions require generalizing feature attribution/importance to datum attribution/importance. This task is called *Data Valuation* in the literature, and its main promises are to let practitioners debug their dataset by identifying outliers, correcting mislabels etc [Koh and Liang, 2017].

An obvious way to compute the responsibility of a data point would be to run the learning algorithm with ($\mathcal{A}(S)$) and without ($\mathcal{A}(S \setminus \{i\})$) the data point, and compare the quantity of interest. However, retraining from scratch for every data point is intractable for modern Machine Learning applications. Early attempts at tackling Data Valuation studied the *Influence Function*, which is a continuous relaxation of $\mathcal{A}(S \setminus \{i\})$ where sample i is down-weighted instead of being removed. When employing ML models whose loss function is convex with respect to the parameter, this continuous relaxation has an exact solution for any differentiable quantity of interest on h [Barshan et al., 2020, Koh and Liang, 2017]. Still, the assumption of convex loss w.r.t model parameters breaks for Deep Learning and so Influence Function are not guaranteed to describe the real relationship between datum and models. Alternative work has defined datum importance measures based on the Shapley Values, and relaxations

thereof [Ghorbani and Zou, 2019, Kwon and Zou, 2021, Wang and Jia, 2023].

Non-Tabular Data All use-cases studied in the manuscript involved tabular data, that is, data with a fixed set of features that each has a clear semantic meaning (*e.g.* `Age`, `gender`, `Salary`.). However, many tasks where Deep Learning has been successful do not involve features with a clear interpretation : speech recognition, natural language processing, and image classification. In such scenarios, the interpretation of "$x_i = 1.34$ has attribution 1.3 toward the gap" is less clear. What does it mean for a pixel to be important? Or the word `bat` in a sentence? The truth is that these features are meaningless when studied individually. The interpretation of a pixel/word is derived from its context and not just its value. A dark pixel can represent hair or the night sky, depending on its neighborhood pixels. The word `bat` can mean a baseball bat or the bat animal, depending on the rest of the sentence.

Deep Learning is believed to be successful at solving tasks with non-semantic inputs because the networks learn the high-level semantics in their hidden layers [Goodfellow et al., 2016]. Consequently, applying the `PyFD` framework to such use-cases would first require extracting concepts from hidden layers. If concept activations could be quantified, then `PyFD` could explain the function h that maps concepts activations $\boldsymbol{x}$ to the network output $h(\boldsymbol{x})$. The CRAFT technique [Fel et al., 2023] is evidence that identifying concepts from images (*e.g.* pants, dirt, shovel) might be possible by factorizing the activations of hidden layers. Nevertheless, it is still an open-debate whether Deep Neural Networks really learn the same concepts as humans [Freiesleben and König, 2023].

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Chirag Agarwal, Nari Johnson, Martin Pawelczyk, Satyapriya Krishna, Eshika Saxena, Marinka Zitnik, and Himabindu Lakkaraju. Rethinking stability for attribution-based explanations. *arXiv preprint arXiv:2203.06877*, 2022a.

Chirag Agarwal, Satyapriya Krishna, Eshika Saxena, Martin Pawelczyk, Nari Johnson, Isha Puri, Marinka Zitnik, and Himabindu Lakkaraju. Openxai: Towards a transparent evaluation of model explanations. *Advances in Neural Information Processing Systems*, 35: 15784–15799, 2022b.

Sushant Agarwal, Shahin Jabbari, Chirag Agarwal, Sohini Upadhyay, Steven Wu, and Himabindu Lakkaraju. Towards the unification and robustness of perturbation and gradient based explanations. In *International Conference on Machine Learning*, pages 110–119. PMLR, 2021.

Maximilian Alber, Sebastian Lapuschkin, Philipp Seegerer, Miriam Hägele, Kristof T Schütt, Grégoire Montavon, Wojciech Samek, Klaus-Robert Müller, Sven Dähne, and Pieter-Jan Kindermans. innvestigate neural networks! *Journal of machine learning research*, 20(93): 1–8, 2019.

Salim I Amoukou and Nicolas JB Brunel. Rethinking counterfactual explanations as local and regional counterfactual policies. *arXiv preprint arXiv:2209.14568*, 2022.

Daniel W Apley and Jingyu Zhu. Visualizing the effects of predictor variables in black box supervised learning models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 82(4):1059–1086, 2020.

Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, et al. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information fusion*, 58:82–115, 2020.

Vijay Arya, Rachel KE Bellamy, Pin-Yu Chen, Amit Dhurandhar, Michael Hind, Samuel C Hoffman, Stephanie Houde, Q Vera Liao, Ronny Luss, Aleksandra Mojsilović, et al. Ai

explainability 360 toolkit. In *Proceedings of the 3rd ACM India Joint International Conference on Data Science & Management of Data (8th ACM IKDD CODS & 26th COMAD)*, pages 376–379, 2021.

Hubert Baniecki and Przemyslaw Biecek. Manipulating shap via adversarial data perturbations (student abstract). 2022.

Hubert Baniecki, Wojciech Kretowicz, Piotr PiÄ, Jakub WiĹ, et al. Dalex: responsible machine learning with interactive explainability and fairness in python. *Journal of Machine Learning Research*, 22(214):1–7, 2021.

Elnaz Barshan, Marc-Etienne Brunet, and Gintare Karolina Dziugaite. Relatif: Identifying explanatory training samples via relative influence. In *International Conference on Artificial Intelligence and Statistics*, pages 1899–1909. PMLR, 2020.

Sara Beery, Grant Van Horn, and Pietro Perona. Recognition in terra incognita. In *Proceedings of the European conference on computer vision (ECCV)*, pages 456–473, 2018.

Tom Begley, Tobias Schwedes, Christopher Frye, and Ilya Feige. Explainability for fair machine learning. *arXiv preprint arXiv:2010.07389*, 2020.

Vaishak Belle and Ioannis Papantonis. Principles and practice of explainable machine learning. *Frontiers in big Data*, page 39, 2021.

Clément Bénard, Sébastien Da Veiga, and Erwan Scornet. Mda for random forests: inconsistency, and a practical solution via the sobol-mda. *arXiv preprint arXiv:2102.13347*, 2021.

Umang Bhatt, Adrian Weller, and José MF Moura. Evaluating and aggregating feature-based model explanations. *arXiv preprint arXiv:2005.00631*, 2020.

Blair Bilodeau, Natasha Jaques, Pang Wei Koh, and Been Kim. Impossibility theorems for feature attribution. *Proceedings of the National Academy of Sciences*, 121(2):e2304406120, 2024.

Frédéric Boisnard, Ryma Boumazouza, Mélanie Ducoffe, Thomas Fel, Estèle Glize, Lucas Hervier, Vincent Mussot, Agustin Martin Picard, Antonin Poché, and David Vigouroux. Guidelines to explain machine learning algorithms. 2023.

Sebastian Bordt and Ulrike von Luxburg. From shapley values to generalized additive models and back. In *International Conference on Artificial Intelligence and Statistics*, pages 709–745. PMLR, 2023.

Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001a.

Leo Breiman. Statistical modeling: The two cultures (with comments and a rejoinder by the author). *Statistical science*, 16(3):199–231, 2001b.

Nadia Burkart and Marco F Huber. A survey on the explainability of supervised machine learning. *Journal of Artificial Intelligence Research*, 70:245–317, 2021.

Laurent Candillier and Vincent Lemaire. Nomao. UCI Machine Learning Repository, 2012. DOI: https://doi.org/10.24432/C53G79.

Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1721–1730, 2015.

Chun-Hao Chang, Sarah Tan, Ben Lengerich, Anna Goldenberg, and Rich Caruana. How interpretable and trustworthy are gams? In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pages 95–105, 2021.

Hugh Chen, Joseph D Janizek, Scott Lundberg, and Su-In Lee. True to the model or true to the data? *arXiv preprint arXiv:2006.16234*, 2020.

Weiwei Cheng, Michaël Rademaker, Bernard De Baets, and Eyke Hüllermeier. Predicting partial orders: ranking with abstention. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 215–230. Springer, 2010.

Yoichi Chikahara, Shinsaku Sakaue, Akinori Fujino, and Hisashi Kashima. Learning individually fair classifier with path-specific causal-effect constraint. In *International Conference on Artificial Intelligence and Statistics*, pages 145–153. PMLR, 2021.

Alexandra Chouldechova. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big data*, 5(2):153–163, 2017.

Beau Coker, Cynthia Rudin, and Gary King. A theory of statistical inference for ensuring the robustness of scientific results. *Management Science*, 67(10):6174–6197, 2021.

Andrew R Conn, Nicholas IM Gould, and Philippe L Toint. *Trust region methods*. SIAM, 2000.

Sam Corbett-Davies, Emma Pierson, Avi Feller, Sharad Goel, and Aziz Huq. Algorithmic decision making and the cost of fairness. In *Proceedings of the 23rd acm sigkdd international conference on knowledge discovery and data mining*, pages 797–806, 2017.

Ian Covert, Scott M Lundberg, and Su-In Lee. Understanding global feature contributions with additive importance measures. *Advances in Neural Information Processing Systems*, 33:17212–17223, 2020.

Jessica Dai, Sohini Upadhyay, Ulrich Aivodji, Stephen H Bach, and Himabindu Lakkaraju. Fairness via explanation quality: Evaluating disparities in the quality of post hoc explanations. In *Proceedings of the 2022 AAAI/ACM Conference on AI, Ethics, and Society*, pages 203–214, 2022.

Alexander D'Amour, Katherine Heller, Dan Moldovan, Ben Adlam, Babak Alipanahi, Alex Beutel, Christina Chen, Jonathan Deaton, Jacob Eisenstein, Matthew D Hoffman, et al. Underspecification presents challenges for credibility in modern machine learning. *arXiv preprint arXiv:2011.03395*, 2020.

Susanne Dandl, Giuseppe Casalicchio, Bernd Bischl, and Ludwig Bothmann. Interpretable regional descriptors: Hyperbox-based local explanations. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 479–495. Springer, 2023.

MaryBeth Defrance and Tijl De Bie. Maximal fairness. In *Proceedings of the 2023 ACM Conference on Fairness, Accountability, and Transparency*, pages 851–880, 2023.

Jiayun Dong and Cynthia Rudin. Variable importance clouds: A way to explore variable importance for the set of good models. *arXiv preprint arXiv:1901.03209*, 2019.

Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference*, pages 214–226, 2012.

Gabriel Erion, Joseph D Janizek, Pascal Sturmfels, Scott M Lundberg, and Su-In Lee. Improving performance of deep learning models with axiomatic attribution priors and expected gradients. *Nature machine intelligence*, 3(7):620–631, 2021.

Saher Esmeir and Shaul Markovitch. Anytime learning of decision trees. *Journal of Machine Learning Research*, 8(5), 2007.

Thomas Fel, Lucas Hervier, David Vigouroux, Antonin Poche, Justin Plakoo, Remi Cadene, Mathieu Chalvidal, Julien Colin, Thibaut Boissin, Louis Bethune, Agustin Picard, Claire Nicodeme, Laurent Gardes, Gregory Flandin, and Thomas Serre. Xplique: A deep learning explainability toolbox. *Workshop on Explainable Artificial Intelligence for Computer Vision (CVPR)*, 2022.

Thomas Fel, Agustin Picard, Louis Bethune, Thibaut Boissin, David Vigouroux, Julien Colin, Rémi Cadène, and Thomas Serre. Craft: Concept recursive activation factorization for explainability. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2711–2721, 2023.

Aaron Fisher, Cynthia Rudin, and Francesca Dominici. All models are wrong, but many are useful: Learning a variable's importance by studying an entire class of prediction models simultaneously. *JMLR*, 20(177):1–81, 2019.

Timo Freiesleben and Gunnar König. Dear xai community, we need to talk! fundamental misconceptions in current xai research. In *World Conference on Explainable Artificial Intelligence*, pages 48–65. Springer, 2023.

Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

Jerome H Friedman. Stochastic gradient boosting. *Computational statistics & data analysis*, 38(4):367–378, 2002.

Jerome H Friedman and Nicholas I Fisher. Bump hunting in high-dimensional data. *Statistics and computing*, 9(2):123–143, 1999.

Jerome H Friedman and Bogdan E Popescu. Predictive learning via rule ensembles. 2008.

Christopher Frye, Damien de Mijolla, Tom Begley, Laurence Cowton, Megan Stanley, and Ilya Feige. Shapley explainability on the data manifold. *arXiv preprint arXiv:2006.01272*, 2020.

Kazuto Fukuchi, Satoshi Hara, and Takanori Maehara. Faking fairness via stealthily biased sampling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 412–419, 2020.

Amirata Ghorbani and James Zou. Data shapley: Equitable valuation of data for machine learning. In *International conference on machine learning*, pages 2242–2251. PMLR, 2019.

Alex Goldstein, Adam Kapelner, Justin Bleich, and Emil Pitkin. Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation. *journal of Computational and Graphical Statistics*, 24(1):44–65, 2015.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

Alicja Gosiewska and Przemyslaw Biecek. Do not trust additive explanations. *arXiv preprint arXiv:1903.11420*, 2019.

Brandon M Greenwell, Bradley C Boehmke, and Andrew J McCarthy. A simple and effective model-based variable importance measure. *arXiv preprint arXiv:1805.04755*, 2018.

Baptiste Gregorutti, Bertrand Michel, and Philippe Saint-Pierre. Correlation and variable importance in random forests. *Statistics and Computing*, 27:659–678, 2017.

Cheng Guo and Felix Berkhahn. Entity embeddings of categorical variables. *arXiv preprint arXiv:1604.06737*, 2016.

Moritz Hardt, Eric Price, and Nati Srebro. Equality of opportunity in supervised learning. *Advances in neural information processing systems*, 29, 2016.

John C Harsanyi. A simplified bargaining model for the n-person cooperative game. *International Economic Review*, 4(2):194–220, 1963.

Trevor Hastie, Robert Tibshirani, and Jerome H Friedman. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Julia Herbinger, Bernd Bischl, and Giuseppe Casalicchio. Decomposing global feature effects based on feature interactions. *arXiv preprint arXiv:2306.00541*, 2023.

Margot Herin, Marouane Il Idrissi, Vincent Chabridon, and Bertrand Iooss. Proportional marginal effects for global sensitivity analysis. *arXiv preprint arXiv:2210.13065*, 2022.

Andrew Herren and P Richard Hahn. Statistical aspects of shap: Functional anova for model interpretation. *arXiv preprint arXiv:2208.09970*, 2022.

Munir Hiabu, Joseph T Meyer, and Marvin N Wright. Unifying local and global model explanations by functional decomposition of low dimensional structures. In *International Conference on Artificial Intelligence and Statistics*, pages 7040–7060. PMLR, 2023.

Giles Hooker. Discovering additive structure in black box functions. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 575–580, 2004.

Giles Hooker, Lucas Mentch, and Siyu Zhou. Unrestricted permutation forces extrapolation: variable importance requires at least one more model, or there is no free variable importance. *Statistics and Computing*, 31(6):1–16, 2021.

Hsiang Hsu and Flavio du Pin Calmon. Rashomon capacity: A metric for predictive multiplicity in probabilistic classification. *arXiv preprint arXiv:2206.01295*, 2022.

Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of classification*, 2: 193–218, 1985.

Ian Janssen, Steven B Heymsfield, ZiMian Wang, and Robert Ross. Skeletal muscle mass and distribution in 468 men and women aged 18–88 yr. *Journal of applied physiology*, 2000.

Dominik Janzing, Lenon Minorics, and Patrick Blöbaum. Feature relevance quantification in explainable ai: A causal problem. In *International Conference on artificial intelligence and statistics*, pages 2907–2916. PMLR, 2020.

Neil Jethani, Mukund Sudarshan, Ian Connick Covert, Su-In Lee, and Rajesh Ranganath. Fastshap: Real-time shapley value estimation. In *International conference on learning representations*, 2021.

Niki Kilbertus, Mateo Rojas Carulla, Giambattista Parascandolo, Moritz Hardt, Dominik Janzing, and Bernhard Schölkopf. Avoiding discrimination through causal reasoning. *Advances in neural information processing systems*, 30, 2017.

Nicholas Kissel and Lucas Mentch. Forward stability and model path selection. *arXiv preprint arXiv:2103.03462*, 2021.

Janis Klaise, Arnaud Van Looveren, Giovanni Vacanti, and Alexandru Coca. Alibi explain: Algorithms for explaining machine learning models. *Journal of Machine Learning Research*, 22(181):1–7, 2021.

Allan R. Klumpp. Apollo lunar-descent guidance. 1971. URL `https://www.nasa.gov/wp-content/uploads/static/history/alsj/ApolloDescentGuidnce.pdf`.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pages 1885–1894. PMLR, 2017.

Narine Kokhlikyan, Vivek Miglani, Miguel Martin, Edward Wang, Bilal Alsallakh, Jonathan Reynolds, Alexander Melnikov, Natalia Kliushkina, Carlos Araya, Siqi Yan, et al. Captum: A unified and generic model interpretability library for pytorch. *arXiv preprint arXiv:2009.07896*, 2020.

Satyapriya Krishna, Tessa Han, Alex Gu, Javin Pombra, Shahin Jabbari, Steven Wu, and Himabindu Lakkaraju. The disagreement problem in explainable machine learning: A practitioner's perspective. *arXiv preprint arXiv:2202.01602*, 2022.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

F Kuo, I Sloan, Grzegorz Wasilkowski, and Henryk Woźniakowski. On decompositions of multivariate functions. *Mathematics of computation*, 79(270):953–966, 2010.

Yongchan Kwon and James Zou. Beta shapley: a unified and noise-reduced data valuation framework for machine learning. *arXiv preprint arXiv:2110.14049*, 2021.

Yongchan Kwon and James Y Zou. Weightedshap: analyzing and improving shapley based feature attributions. *Advances in Neural Information Processing Systems*, 35:34363–34376, 2022.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles, 2017.

Jeff Larson, Surya Mattu, Lauren Kirchner, and Julia Angwin. How we analyzed the compas recidivism algorithm. *ProPublica*, May 2016. URL `https://www.propublica.org/article/how-we-analyzed-the-compas-recidivism-algorithm`.

A J Lee. *U-statistics: Theory and Practice*. Routledge, 2019.

Miguel Lerma and Mirtha Lucas. Symmetry-preserving paths in integrated gradients. *arXiv preprint arXiv:2103.13533*, 2021.

Yin Lou, Rich Caruana, and Johannes Gehrke. Intelligible models for classification and regression. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 150–158, 2012.

Yin Lou, Rich Caruana, Johannes Gehrke, and Giles Hooker. Accurate intelligible models with pairwise interactions. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 623–631, 2013.

Gilles Louppe. Understanding random forests: From theory to practice. *arXiv preprint arXiv:1407.7502*, 2014.

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in neural information processing systems*, pages 4765–4774, 2017.

Scott M Lundberg, Gabriel G Erion, and Su-In Lee. Consistent individualized feature attribution for tree ensembles. *arXiv preprint arXiv:1802.03888*, 2018.

Scott M Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee. From local explanations to global understanding with explainable ai for trees. *Nature machine intelligence*, 2(1): 56–67, 2020.

Charles Marx, Flavio Calmon, and Berk Ustun. Predictive multiplicity in classification. In *International Conference on Machine Learning*, pages 6765–6774. PMLR, 2020.

Frank J Massey Jr. The kolmogorov-smirnov test for goodness of fit. *Journal of the American statistical Association*, 46(253):68–78, 1951.

Luke Merrick and Ankur Taly. The explanation game: Explaining machine learning models using shapley values. In *International Cross-Domain Conference for Machine Learning and Knowledge Extraction*, pages 17–38. Springer, 2020.

Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. MIT press, 2018.

Christoph Molnar, Gunnar König, Bernd Bischl, and Giuseppe Casalicchio. Model-agnostic feature importance and effects with dependent features: a conditional subgroup approach. *Data Mining and Knowledge Discovery*, pages 1–39, 2023.

Artyom G Nahapetyan. Bilinear programming., 2009.

Bitya Neuhof and Yuval Benjamini. Confident feature ranking. In *International Conference on Artificial Intelligence and Statistics*, pages 1468–1476. PMLR, 2024.

Harsha Nori, Samuel Jenkins, Paul Koch, and Rich Caruana. Interpretml: A unified framework for machine learning interpretability. *arXiv preprint arXiv:1909.09223*, 2019.

Art B. Owen. *Monte Carlo theory, methods and examples.* 2013. URL `https://artowen.su.domains/mc/`.

Judea Pearl and Dana Mackenzie. *The book of why: the new science of cause and effect*. Basic books, 2018.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Vitali Petsiuk, Abir Das, and Kate Saenko. Rise: Randomized input sampling for explanation of black-box models. *arXiv preprint arXiv:1806.07421*, 2018.

Kaivalya Rawal and Himabindu Lakkaraju. Beyond individualized recourse: Interpretable and interactive summaries of actionable recourses. *Advances in Neural Information Processing Systems*, 33:12187–12198, 2020.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Cynthia Rudin, Caroline Wang, and Beau Coker. The age of secrecy and unfairness in recidivism prediction. *arXiv preprint arXiv:1811.00731*, 2018.

Jorge Sánchez and Florent Perronnin. High-dimensional signature compression for large-scale image classification. In *CVPR 2011*, pages 1665–1672. IEEE, 2011.

Jonas Schulz, Rafael Poyiadzi, and Raul Santos-Rodriguez. Uncertainty quantification of surrogate explanations: an ordinal consensus approach. *arXiv preprint arXiv:2111.09121*, 2021.

Lesia Semenova, Cynthia Rudin, and Ronald Parr. On the existence of simpler machine learning models. In *2022 ACM Conference on Fairness, Accountability, and Transparency*, pages 1827–1858, 2022.

Torgyn Shaikhina, Umang Bhatt, Roxanne Zhang, Konstantinos Georgatzis, Alice Xiang, and Adrian Weller. Effects of uncertainty on the quality of feature importance explanations. *AAAI Workshop on Explainable Agency in Artificial Intelligence*, 2021.

Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

Lloyd S Shapley. A value for n-person games. *Contributions to the Theory of Games*, pages 307–317, 1953.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017.

Mateusz Staniak and Przemyslaw Biecek. Explanations of model predictions with live and breakdown packages. *arXiv preprint arXiv:1804.01955*, 2018.

Carolin Strobl, Anne-Laure Boulesteix, Thomas Kneib, Thomas Augustin, and Achim Zeileis. Conditional variable importance for random forests. *BMC bioinformatics*, 9:1–11, 2008.

Erik Strumbelj and Igor Kononenko. An efficient explanation of individual classifications using game theory. *The Journal of Machine Learning Research*, 11:1–18, 2010.

Erik Štrumbelj and Igor Kononenko. Explaining prediction models and individual predictions with feature contributions. *Knowledge and information systems*, 41:647–665, 2014.

Mukund Sundararajan and Amir Najmi. The many shapley values for model explanation. In *International conference on machine learning*, pages 9269–9278. PMLR, 2020.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. *arXiv preprint arXiv:1703.01365*, 2017.

Mukund Sundararajan, Kedar Dhamdhere, and Ashish Agarwal. The shapley taylor interaction index. In *International conference on machine learning*, pages 9259–9268. PMLR, 2020.

Muhammad Faaiz Taufiq, Patrick Blöbaum, and Lenon Minorics. Manifold restricted interventional shapley values. In *International Conference on Artificial Intelligence and Statistics*, pages 5079–5106. PMLR, 2023.

Richard Tomsett, Dan Harborne, Supriyo Chakraborty, Prudhvi Gurram, and Alun Preece. Sanity checks for saliency metrics. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 6021–6029, 2020.

Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harv. JL & Tech.*, 31:841, 2017.

Jiachen T Wang and Ruoxi Jia. Data banzhaf: A robust data valuation framework for machine learning. In *International Conference on Artificial Intelligence and Statistics*, pages 6388–6421. PMLR, 2023.

Larry Wasserman. *All of Statistics: A concise course in statistical inference*. Springer, 2004.

Brian D Williamson, Peter B Gilbert, Noah R Simon, and Marco Carone. A general framework for inference on algorithm-agnostic variable importance. *Journal of the American Statistical Association*, pages 1–14, 2021.

Simon N Wood. *Generalized additive models: an introduction with R*. chapman and hall/CRC, 2017.

Chih-Kuan Yeh, Cheng-Yu Hsieh, Arun Suggala, David I Inouye, and Pradeep K Ravikumar. On the (in) fidelity and sensitivity of explanations. *Advances in neural information processing systems*, 32, 2019.

Chih-Kuan Yeh, Kuan-Yun Lee, Frederick Liu, and Pradeep Ravikumar. Threading the needle of on and off-manifold value functions for shapley explanations. In *International Conference on Artificial Intelligence and Statistics*, pages 1485–1502. PMLR, 2022.

APPENDIX A SUPPLEMENTARY ON LITERATURE REVIEW

A.1 Integrated Gradient

As a reminder, the gap $h^{\text{lin}}_{\boldsymbol{\omega}}(\boldsymbol{x}) - h^{\text{lin}}_{\boldsymbol{\omega}}(\boldsymbol{z})$ between the outputs of a linear model evaluated at $\boldsymbol{x}, \boldsymbol{z}$ can easily be explained:

$$\phi_j^{\text{LFA}}(h^{\text{lin}}_{\boldsymbol{\omega}}, \boldsymbol{x}, \boldsymbol{z}) = \omega_j(x_j - z_j). \tag{A.1}$$

If the model is not linear, however, Taylor expansions could be leveraged to get approximate feature attributions. Letting $\boldsymbol{\Delta} := \boldsymbol{x} - \boldsymbol{z}$, the Taylor expansion of h centered at $\boldsymbol{z}$ and evaluated at $\boldsymbol{x}$ is

$$h(\boldsymbol{z} + \boldsymbol{\Delta}) = h(\boldsymbol{z}) + \langle \nabla h(\boldsymbol{z}), \boldsymbol{\Delta} \rangle + o(\|\boldsymbol{\Delta}\|), \tag{A.2}$$

where $o(\|\boldsymbol{\Delta}\|)$ denotes the set of function that are negligible compared to $\|\boldsymbol{\Delta}\|$ near $\mathbf{0}$

$$g \in o(\|\boldsymbol{\Delta}\|) \iff \lim_{\|\Delta\| \to 0} \frac{g(\|\boldsymbol{\Delta}\|)}{\|\boldsymbol{\Delta}\|} = 0. \tag{A.3}$$

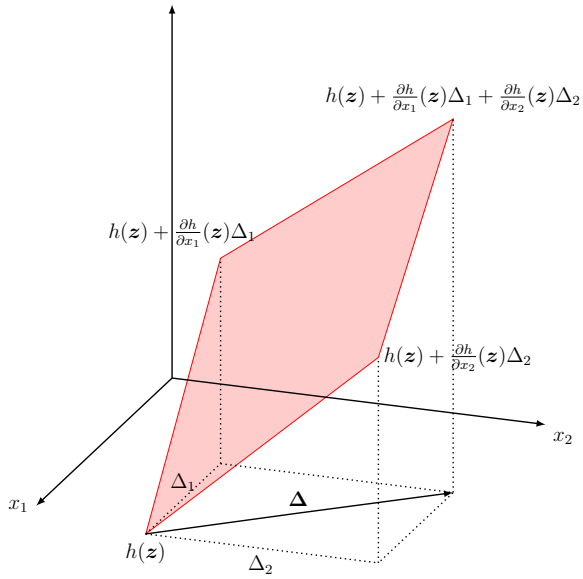


Figure A.1 Attribution of a Taylor expansion centered at $\boldsymbol{z}$ and evaluated at $\boldsymbol{x} = \boldsymbol{z} + \boldsymbol{\Delta}$.

Since the Taylor expansion (cf. Equation A.2) is linear, the feature attributions to explain the gap $h(\boldsymbol{x}) - h(\boldsymbol{z})$ are $\frac{\partial h}{\partial x_j}(\boldsymbol{z})(x_j - z_j)$. See Figure A.1 for the visual intuition. Nonetheless, because the Taylor expansion only describes h locally, these feature attributions might not be faithful when $\|\boldsymbol{\Delta}\|$ is large (*i.e.* $\boldsymbol{x}$ and $\boldsymbol{z}$ are far apart). The solution put forward by Integrated Gradients is 1) follow a smooth path between $\boldsymbol{z}$ and $\boldsymbol{x}$, 2) perform multiple Taylor expansions of h along the path, 3) compute infinitesimal feature attributions using said Taylor expansions, 4) aggregate all feature attributions.

The path between $\boldsymbol{z}$ and $\boldsymbol{x}$ is represented via a parametric curve, which is a continous function $\boldsymbol{\gamma} : [0, 1] \to \mathbb{R}^d$ such that $\boldsymbol{\gamma}(0) = \boldsymbol{z}$ and $\boldsymbol{\gamma}(1) = \boldsymbol{x}$, see Figure A.2 (a). Each component of the output vector is a continuous function γ_j

$$\boldsymbol{\gamma}(t) = (\gamma_1(t), \gamma_2(t), \ldots, \gamma_d(t)). \tag{A.4}$$

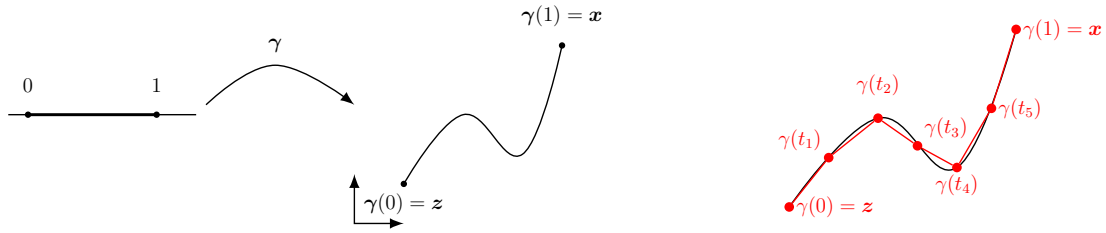


Figure A.2 (a) Example of parametric curve. (b) Discretizing a parametric curve with $T = 6$.

The unit interval $[0, 1]$ can be discretized into T intervals with boundary knots $t_k := k/T$ for $k = 0, 1, \ldots, T$. Such knots can be mapped though $\boldsymbol{\gamma}$ leading to a set of points $\{\boldsymbol{\gamma}(t_k)\}_{k=0}^{T}$ along the parametric curve, see Figure A.2 (b). After discretization, the gap between $h(\boldsymbol{x})$ and $h(\boldsymbol{z})$ is expressed as a telescopic sum

$$h(\boldsymbol{x}) - h(\boldsymbol{z}) = \sum_{k=1}^{T} h(\boldsymbol{\gamma}(t_k)) - h(\boldsymbol{\gamma}(t_{k-1})). \tag{A.5}$$

The key insight from Integrated Gradient now follows: as $T \to \infty$, the non-linear model h will become approximately linear along the linear segments connecting points $\boldsymbol{\gamma}(t_k)$ and $\boldsymbol{\gamma}(t_{k-1})$. Consequently, one can extract feature attributions of linear models along these tiny segments

$$\begin{aligned}
h(\boldsymbol{x}) - h(\boldsymbol{z}) &= \sum_{k=1}^{T} h(\boldsymbol{\gamma}(t_k)) - h(\boldsymbol{\gamma}(t_{k-1})) \\
&= \sum_{k=1}^{T} \langle \nabla h(\boldsymbol{\gamma}(t_{k-1})),\, \boldsymbol{\gamma}(t_k) - \boldsymbol{\gamma}(t_{k-1}) \rangle + o(\|\boldsymbol{\gamma}(t_k) - \boldsymbol{\gamma}(t_{k-1})\|) \\
&= \sum_{k=1}^{T} \langle \nabla h(\boldsymbol{\gamma}(t_{k-1})),\, \boldsymbol{\gamma}(t_k) - \boldsymbol{\gamma}(t_{k-1}) \rangle + \sum_{k=1}^{T} o(\|\boldsymbol{\gamma}(t_k) - \boldsymbol{\gamma}(t_{k-1})\|) \\
&= \sum_{i=1}^{T} \left\langle \nabla h(\boldsymbol{\gamma}(t_{k-1})),\, \frac{\boldsymbol{\gamma}(t_k) - \boldsymbol{\gamma}(t_{k-1})}{1/T} \right\rangle \frac{1}{T} + \sum_{k=1}^{T} o(\|\boldsymbol{\gamma}(t_k) - \boldsymbol{\gamma}(t_{k-1})\|).
\end{aligned} \tag{A.6}$$

Under the regularity assumptions on h and $\boldsymbol{\gamma}$ presented in [Lerma and Lucas, 2021, Proposition 1], taking the limit $T \to \infty$, the first term will converge to

$$\lim_{T\to\infty} \sum_{k=1}^{T} \left\langle \nabla h(\boldsymbol{\gamma}(t_{k-1})),\, \frac{\boldsymbol{\gamma}(t_k) - \boldsymbol{\gamma}(t_{k-1})}{1/T} \right\rangle \frac{1}{T} = \int_0^1 \left\langle \nabla h(\boldsymbol{\gamma}(t)), \frac{d\boldsymbol{\gamma}(t)}{dt} \right\rangle dt \tag{A.7}$$

and the second one to zero *i.e.* $\lim_{T\to\infty}\sum_{k=1}^{T} o(\|\boldsymbol{\gamma}(t_k)-\boldsymbol{\gamma}(t_{k-1})\|)=0$. Thus,

$$\begin{aligned} h(\boldsymbol{x})-h(\boldsymbol{z}) &= \int_0^1 \left\langle \nabla h(\boldsymbol{\gamma}(t)), \frac{d\boldsymbol{\gamma}(t)}{dt} \right\rangle dt \\ &= \int_0^1 \sum_{j=1}^{d} \frac{\partial h}{\partial x_j}(\boldsymbol{\gamma}(t))\frac{d\gamma_j}{dt}dt \\ &= \sum_{j=1}^{d} \underbrace{\int_0^1 \frac{\partial h}{\partial x_j}(\boldsymbol{\gamma}(t))\frac{d\gamma_j}{dt}dt}_{\text{IG Attribution}}, \end{aligned} \tag{A.8}$$

which suggests defining the IG feature attribution

$$\phi_j^{\text{IG-}\gamma}(h,\boldsymbol{x},\boldsymbol{z}) = \int_0^1 \frac{\partial h}{\partial x_j}(\boldsymbol{\gamma}(t))\frac{d\gamma_j}{dt}dt. \tag{A.9}$$

Taking a linear path between $\boldsymbol{z}$ and $\boldsymbol{x}$ (*i.e.* $\boldsymbol{\gamma}(t)=(1-t)\boldsymbol{z}+t\boldsymbol{x}$) leads to the definition of Integrated Gradients often seen in the literature

$$\phi_j^{\text{IG}}(h,\boldsymbol{x},\boldsymbol{z}) = (x_j-z_j)\int_0^1 \frac{\partial h}{\partial x_j}((1-t)\boldsymbol{z}+t\boldsymbol{x})dt. \tag{A.10}$$

If the baseline $\boldsymbol{z}\sim\mathcal{B}$ is a random variable, then returning the average IG attribution leads to the Expected Gradient

$$\phi_j^{\text{EG}}(h,\boldsymbol{x},\mathcal{B}) = \mathop{\mathbb{E}}_{\substack{\boldsymbol{z}\sim\mathcal{B}\\ t\sim U(0,1)}}\left[(x_j-z_j)\frac{\partial h}{\partial x_j}((1-t)\boldsymbol{z}+t\boldsymbol{x})\right]. \tag{A.11}$$

A.2 More on Shapley values

Proposition A.2.1 (**Proposition 2.3.1**). *Let $d = 2$, $h^{lin}_{\boldsymbol{\omega}}$ be a linear model, and $\mathcal{B} = \mathcal{N}(\mathbf{0}, (1-\rho)\boldsymbol{I} + \rho\mathbf{1})$ be a distribution over two correlated Gaussian variables. Computing Shapley values of the Interventional and Observational games yields the feature attributions*

$$\begin{aligned}\boldsymbol{\phi}^{\textit{SHAP-int}}(h^{lin}_{\boldsymbol{\omega}}, \boldsymbol{x}, \mathcal{B}) &= [\omega_1 x_1, \omega_2 x_2]^T \\ \boldsymbol{\phi}^{\textit{SHAP-obs}}(h^{lin}_{\boldsymbol{\omega}}, \boldsymbol{x}, \mathcal{B}) &= [\omega_1 x_1 + \frac{\rho}{2}(\omega_2 x_1 - \omega_1 x_2), \omega_2 x_2 + \frac{\rho}{2}(\omega_1 x_2 - \omega_2 x_1)]^T.\end{aligned} \tag{A.12}$$

Proof. To simplify notation, we will write $h \equiv h^{\text{lin}}_{\boldsymbol{\omega}}$. Let us start by deriving the Interventional Shapley Values. Given a subset $S \subseteq [d] \setminus \{j\}$, the marginal contribution

$$\begin{aligned}\nu^{\text{int}}_{h,\boldsymbol{x},\mathcal{B}}(S \cup \{j\}) - \nu^{\text{int}}_{h,\boldsymbol{x},\mathcal{B}}(S) &= \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_{S\cup\{j\}}, \boldsymbol{z}_{\overline{S\cup\{j\}}})] - \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_S, \boldsymbol{z}_{\overline{S}})] \\ &= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}[h(\boldsymbol{x}_{S\cup\{j\}}, \boldsymbol{z}_{\overline{S\cup\{j\}}}) - h(\boldsymbol{x}_S, \boldsymbol{z}_{\overline{S}})] \\ &= \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}\left[\omega_0 + \sum_{j\in S\cup\{j\}} \omega_j x_j + \sum_{j\in\overline{S}\setminus\{j\}} \omega_j z_j - \omega_0 - \sum_{j\in S} \omega_j x_j - \sum_{j\in\overline{S}} \omega_j z_j\right] \\ &= \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}\left[\sum_{j\in S\cup\{j\}} \omega_j x_j - \sum_{j\in S} \omega_j x_j + \sum_{j\in\overline{S}\setminus\{j\}} \omega_j z_j - \sum_{j\in\overline{S}} \omega_j z_j\right] \\ &= \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}\Big[\omega_j x_j - \omega_j z_j\Big] = \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}\Big[\omega_j (x_j - z_j)\Big] \\ &= \omega_j\Big(x_j - \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[z_j]\Big) = \omega_j x_j\end{aligned} \tag{A.13}$$

is independent on S. The Shapley Values, which are the weighted marginal contributions, are therefore

$$\boldsymbol{\phi}^{\text{SHAP-int}}(h, \boldsymbol{x}, \mathcal{B}) = [\omega_1 x_1, \omega_2 x_2]^T. \tag{A.14}$$

Now investigating the Observational Shapley Values

$$\begin{aligned}\nu^{\text{obs}}_{h,\boldsymbol{x},\mathcal{B}}(S) &= \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{z})|\boldsymbol{z}_S = \boldsymbol{x}_S] \\ &= \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}\left[\omega_0 + \sum_{j=1}^{2} \omega_j z_j \middle| \boldsymbol{z}_S = \boldsymbol{x}_S\right] \\ &= \omega_0 + \sum_{j\in S} \omega_j x_j + \sum_{j\notin S} \omega_j \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[z_j|\boldsymbol{z}_S = \boldsymbol{x}_S].\end{aligned} \tag{A.15}$$

Note that $\nu^{\text{obs}}_{h,\boldsymbol{x},\mathcal{B}}(\emptyset) = \omega_0 + \sum_{j=1}^{2} \omega_j \mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[z_j] = \omega_0$ and $\nu^{\text{obs}}_{h,\boldsymbol{x},\mathcal{B}}(\{1,2\}) = \omega_0 + \sum_{j=1}^{2} \omega_j x_j$. More-

over,

$$\begin{aligned}\nu^{\text{obs}}_{h,\boldsymbol{x},\mathcal{B}}(\{1\}) &= \omega_0 + \omega_1 x_1 + \omega_2 \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[z_2|z_1 = x_1] \\ &= \omega_0 + \omega_1 x_1 + \omega_2 \rho x_1\end{aligned} \tag{A.16}$$

and

$$\begin{aligned}\nu^{\text{obs}}_{h,\boldsymbol{x},\mathcal{B}}(\{2\}) &= \omega_0 + \omega_2 x_2 + \omega_1 \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[z_1|z_2 = x_2] \\ &= \omega_0 + \omega_2 x_2 + \omega_1 \rho x_2\end{aligned} \tag{A.17}$$

Putting it all together, the Observational Shapley Values are

$$\begin{aligned}\phi_1^{\text{SHAP-obs}}(h, \boldsymbol{x}, \mathcal{B}) :=& \frac{1}{2}\Big(\big(\nu^{\text{obs}}_{h,\boldsymbol{x},\mathcal{B}}(\{1,2\}) - \nu^{\text{obs}}_{h,\boldsymbol{x},\mathcal{B}}(\{2\}) \big) + \big(\nu^{\text{obs}}_{h,\boldsymbol{x},\mathcal{B}}(\{1\}) - \nu^{\text{obs}}_{h,\boldsymbol{x},\mathcal{B}}\emptyset) \big) \Big) \\ =& \frac{1}{2}\Big((\omega_1(x_1 - \rho x_2) + x_1(\omega_1 + \rho\omega_2) \Big) = \omega_1 x_1 + \frac{\rho}{2}(\omega_2 x_1 - \omega_1 x_2).\end{aligned} \tag{A.18}$$

$$\begin{aligned}\phi_2^{\text{SHAP-obs}}(h, \boldsymbol{x}, \mathcal{B}) :=& \frac{1}{2}\Big(\big(\nu^{\text{obs}}_{h,\boldsymbol{x},\mathcal{B}}(\{1,2\}) - \nu^{\text{obs}}_{h,\boldsymbol{x},\mathcal{B}}(\{1\}) \big) + \big(\nu^{\text{obs}}_{h,\boldsymbol{x},\mathcal{B}}(\{2\}) - \nu^{\text{obs}}_{h,\boldsymbol{x},\mathcal{B}}\emptyset) \big) \Big) \\ =& \frac{1}{2}\Big((\omega_2(x_2 - \rho x_1) + x_2(\omega_2 + \rho\omega_1) \Big) = \omega_2 x_2 + \frac{\rho}{2}(\omega_1 x_2 - \omega_2 x_1).\end{aligned} \tag{A.19}$$

□

A.3 Insertion and Deletion

Proposition A.3.1 (**Proposition 2.4.1**). *Let $\boldsymbol{\phi}$ be a post-hoc additive explainer that falls back to the ante-hoc explanation $\phi_j(h^{add}, \boldsymbol{x}, \mathcal{B}) = h_j(x_j) - \mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[h_j(z_j)]$ when the model is additive (e.g. SHAP). Also, let $\boldsymbol{\phi}'$ be any alternative local feature attribution that does not fall back to ante-hoc definitions.*

Then, if h^{add} is additive, the Insertion and Deletion metrics are equivalent

$$\begin{aligned} \mathit{Insertion}(h^{add}, \boldsymbol{\phi}, \boldsymbol{x}, \mathcal{B}) &= \mathit{Deletion}(h^{add}, \boldsymbol{\phi}, \boldsymbol{x}, \mathcal{B}) \\ \mathit{Insertion}(h^{add}, \boldsymbol{\phi}', \boldsymbol{x}, \mathcal{B}) &= \mathit{Deletion}(h^{add}, \boldsymbol{\phi}', \boldsymbol{x}, \mathcal{B}). \end{aligned} \tag{A.20}$$

Moreover, when all local feature attributions $\{\phi_j(h^{add}, \boldsymbol{x}, \mathcal{B})\}_{j=1}^d$ have the same sign, we have

$$\mathit{Insertion}(h^{add}, \boldsymbol{\phi}, \boldsymbol{x}, \mathcal{B}) \leq \mathit{Insertion}(h^{add}, \boldsymbol{\phi}', \boldsymbol{x}, \mathcal{B}). \tag{A.21}$$

Insertion/Deletion metrics rightfully claim that $\boldsymbol{\phi}$ is more faithful to h^{add} than $\boldsymbol{\phi}'$. However, if we allow local feature attributions $\{\phi_j(h^{add}, \boldsymbol{x}, \mathcal{B})\}_{j=1}^d$ to have both positive and negative signs, then there exists a functional $\boldsymbol{\phi}'$, model h^{add}, input $\boldsymbol{x}$ and reference $\mathcal{B}$ where

$$\mathit{Insertion}(h^{add}, \boldsymbol{\phi}', \boldsymbol{x}, \mathcal{B}) < \mathit{Insertion}(h^{add}, \boldsymbol{\phi}, \boldsymbol{x}, \mathcal{B}). \tag{A.22}$$

Insertion and Deletion fail as unfaithfulness metrics.

Proof. We first reformulate the Insertion and Deletion metrics in terms of a permutation $\pi : [d] \to [d]$ such that $\pi(i) < \pi(j) \Rightarrow |\phi_i(h, \boldsymbol{x}, \mathcal{B})| \geq |\phi_j(h, \boldsymbol{x}, \mathcal{B})|$. Said permutation encodes a ranking of feature importance. We also define the sets $\pi_{:j} := \{i \in [d] : \pi(i) < \pi(j)\}$ containing features that are more important than j according to $\boldsymbol{\phi}$. Insertion/Deletion can be written

$$\begin{aligned} \text{Insertion}(h, \boldsymbol{\phi}, \boldsymbol{x}, \mathcal{B}) &= \sum_{j=1}^{d} |h(\boldsymbol{x}) - \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_{\pi_{:j}}, \boldsymbol{z}_{\overline{\pi_{:j}}})]| - |h(\boldsymbol{x}) - \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{z})]| \\ \text{Deletion}(h, \boldsymbol{\phi}, \boldsymbol{x}, \mathcal{B}) &= \sum_{j=1}^{d} |\mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{z})] - \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_{\overline{\pi_{:j}}}, \boldsymbol{z}_{\pi_{:j}})]| - |h(\boldsymbol{x}) - \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{z})]| \end{aligned} \tag{A.23}$$

If h^{add} is additive, $h^{\text{add}}(\boldsymbol{x}) - \mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[h^{\text{add}}(\boldsymbol{x}_{\pi_{:j}}, \boldsymbol{z}_{\overline{\pi_{:j}}})]$ and $\mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[h^{\text{add}}(\boldsymbol{z})] - \mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[h^{\text{add}}(\boldsymbol{x}_{\overline{\pi_{:j}}}, \boldsymbol{z}_{\pi_{:j}})]$

are both equal to $\sum_{i \notin \pi_{:j}} h_i(x_i) - \mathbb{E}_{\boldsymbol{z} \sim \mathcal{B}}[h_i(z_i)]$. Consequently

$$\begin{aligned} \text{Insertion}(h^{\text{add}}, \boldsymbol{\phi}, \boldsymbol{x}, \mathcal{B}) &= \text{Deletion}(h^{\text{add}}, \boldsymbol{\phi}, \boldsymbol{x}, \mathcal{B}) \\ &= \sum_{j=1}^{d} \Big| \sum_{i \notin \pi_{:j}} h_i(x_i) - \mathop{\mathbb{E}}_{\boldsymbol{z} \sim \mathcal{B}}[h_i(z_i)] \Big| - |h(\boldsymbol{x}) - \mathop{\mathbb{E}}_{\boldsymbol{z} \sim \mathcal{B}}[h(\boldsymbol{z})]|. \end{aligned} \tag{A.24}$$

Letting $\boldsymbol{\phi}'$ be an alternative post-hoc explainer and π' be its feature importance ranking such that $\pi'(i) < \pi'(j) \Rightarrow |\phi'_i(h, \boldsymbol{x}, \mathcal{B})| \geq |\phi'_j(h, \boldsymbol{x}, \mathcal{B})|$, the same reasoning applies

$$\begin{aligned} \text{Insertion}(h^{\text{add}}, \boldsymbol{\phi}', \boldsymbol{x}, \mathcal{B}) &= \text{Deletion}(h^{\text{add}}, \boldsymbol{\phi}', \boldsymbol{x}, \mathcal{B}) \\ &= \sum_{j=1}^{d} \Big| \sum_{i \notin \pi'_{:j}} h_i(x_i) - \mathop{\mathbb{E}}_{\boldsymbol{z} \sim \mathcal{B}}[h_i(z_i)] \Big| - |h(\boldsymbol{x}) - \mathop{\mathbb{E}}_{\boldsymbol{z} \sim \mathcal{B}}[h(\boldsymbol{z})]|. \end{aligned} \tag{A.25}$$

We have proven Equation A.20.

The remainder of the proof distinguishes two cases 1) $\{\phi_j(h^{\text{add}}, \boldsymbol{x}, \mathcal{B})\}_{j=1}^{d}$ all have the same sign or 2) $\{\phi_j(h^{\text{add}}, \boldsymbol{x}, \mathcal{B})\}_{j=1}^{d}$ have both positive and negative signs.

Same Signs When $\{\phi_j(h^{\text{add}}, \boldsymbol{x}, \mathcal{B})\}_{j=1}^{d}$ all have the same signs

$$\begin{aligned} \text{Insertion}(h^{\text{add}}, \boldsymbol{\phi}, \boldsymbol{x}, \mathcal{B}) &= \sum_{j=1}^{d} | \sum_{i \notin \pi_{:j}} h_i(x_i) - \mathop{\mathbb{E}}_{\boldsymbol{z} \sim \mathcal{B}}[h_i(z_i)]| - |h(\boldsymbol{x}) - \mathop{\mathbb{E}}_{\boldsymbol{z} \sim \mathcal{B}}[h(\boldsymbol{z})]| \\ &= \sum_{j=1}^{d} \sum_{i \notin \pi_{:j}} |h_i(x_i) - \mathop{\mathbb{E}}_{\boldsymbol{z} \sim \mathcal{B}}[h_i(z_i)]| - \sum_{j=1}^{d} |h_j(x_j) - \mathop{\mathbb{E}}_{\boldsymbol{z} \sim \mathcal{B}}[h_j(z_j)]| \\ &= \sum_{j=1}^{d} (\pi(j) - 1) \times |h_j(x_j) - \mathop{\mathbb{E}}_{\boldsymbol{z} \sim \mathcal{B}}[h_j(z_j)]|. \end{aligned} \tag{A.26}$$

Insertion scales the ground-truth importance $|h_j(x_j) - \mathbb{E}_{\boldsymbol{z} \sim \mathcal{B}}[h_j(z_j)]|$ by a factor $\pi(j) - 1$ *i.e.* the number of features ranked higher in the explanation. Thus, having a low Insertion score requires ranking the truly important features first to avoid multiplying them by a large $\pi(j) - 1$.

Because the ranks π of explainer $\boldsymbol{\phi}$ already order features from largest ground-truth importance $|h_j(x_j) - \mathbb{E}_{\boldsymbol{z} \sim \mathcal{B}}[h_j(z_j)]|$ to lowest, it must be the ranking that minimizes the Insertion score. To see it, note that any alternative ranking $\pi' \neq \pi$ must have a pair $i, j \in [d]$ such that

$$\pi'(i) < \pi'(j) \quad \text{and} \quad |h_i(x_i) - \mathop{\mathbb{E}}_{\boldsymbol{z} \sim \mathcal{B}}[h_i(z_i)]| \leq |h_j(x_i) - \mathop{\mathbb{E}}_{\boldsymbol{z} \sim \mathcal{B}}[h_j(z_j)]|. \tag{A.27}$$

That is, the ranking of $\boldsymbol{\phi}'$ contradict the ground-truth orderings of i and j. By defining a

new permutation π'' where $\pi''(k) = \pi'(k)\ \forall k \in [d] \setminus \{i, j\}$ and $\pi''(i) = \pi'(j)$ and $\pi''(j) = \pi'(i)$, the Insertion score decreases

$$\begin{aligned}
&\text{Insertion}(h^{\text{add}}, \boldsymbol{\phi}', \boldsymbol{x}, \mathcal{B}) \ - \ \text{Insertion}(h^{\text{add}}, \boldsymbol{\phi}'', \boldsymbol{x}, \mathcal{B}) \\
&\quad = (\pi'(j) - \pi''(j)) \times |h_j(x_j) - \mathop{\mathbb{E}}_{z \sim \mathcal{B}}[h_j(z_j)] + (\pi'(i) - \pi''(i)) \times |h_i(x_i) - \mathop{\mathbb{E}}_{z \sim \mathcal{B}}[h_i(z_i)]| \\
&\quad = (\pi'(j) - \pi'(i)) \times |h_j(x_j) - \mathop{\mathbb{E}}_{z \sim \mathcal{B}}[h_j(z_j)] + (\pi'(i) - \pi'(j)) \times |h_i(x_i) - \mathop{\mathbb{E}}_{z \sim \mathcal{B}}[h_i(z_i)]| \\
&\quad = (\pi'(j) - \pi'(i)) \times \Big(|h_j(x_j) - \mathop{\mathbb{E}}_{z \sim \mathcal{B}}[h_j(z_j)]| - |h_i(x_i) - \mathop{\mathbb{E}}_{z \sim \mathcal{B}}[h_i(z_i)]|\Big) \\
&\quad \geq 0. \qquad \text{(cf. Equation A.27)}
\end{aligned}$$

Simply put, whenever a feature importance ranking π' contradicts the ground-truth ordering of a pair $i, j \in [d]$, switching the ranks of i and j leads to a new ordering π'' with decreased Insertion. As a result, the ground-truth ranks π yielded by $\boldsymbol{\phi}$ minimize the Insertion score

$$\text{Insertion}(h^{\text{add}}, \boldsymbol{\phi}, \boldsymbol{x}, \mathcal{B}) \leq \text{Insertion}(h^{\text{add}}, \boldsymbol{\phi}', \boldsymbol{x}, \mathcal{B}), \tag{A.28}$$

and so Equation A.21 is proven.

Different Signs When feature attributions $\{\phi_j(h^{\text{add}}, \boldsymbol{x}, \mathcal{B})\}_{j=1}^{d}$ are allowed to have different signs, the ranking π is not guaranteed to be the one minimizing Insertion. Here is a counter-example with $d = 3$: let $\boldsymbol{x} = [1, 1, 1]^T$, $\mathcal{B} = \delta(\mathbf{0})$, and $h^{\text{lin}}_{\boldsymbol{\omega}} = -x_1 + \frac{2}{3}(x_2 + x_3)$. The additive explanation that would be returned by $\boldsymbol{\phi}$ is $\boldsymbol{\phi}(h^{\text{lin}}_{\boldsymbol{\omega}}, \boldsymbol{x}, \mathcal{B}) = [-1, 2/3, 2/3]^T$. The associated feature importance rankings are $\pi(1) = 1$, $\pi(2) = 2$, $\pi(3) = 3$ leading to the Insertion score

$$\text{Insertion}(h^{\text{lin}}_{\boldsymbol{\omega}}, \boldsymbol{\phi}, \boldsymbol{x}, \mathcal{B}) = |\omega_2 + \omega_3| + |\omega_3| = |2/3 + 2/3| + |2/3| = 2. \tag{A.29}$$

However, the alternative ranking $\pi'(1) = 2$, $\pi'(2) = 1$, $\pi'(3) = 3$, which could be provided by an arbitrary explainer $\boldsymbol{\phi}'$, leads to a smaller Insertion

$$\text{Insertion}(h^{\text{lin}}_{\boldsymbol{\omega}}, \boldsymbol{\phi}', \boldsymbol{x}, \mathcal{B}) = |\omega_1 + \omega_3| + |\omega_3| = |\text{-}1 + 2/3| + |2/3| = 1. \tag{A.30}$$

□

APPENDIX B SUPPLEMENTARY ON FUNCTIONAL DECOMPOSITION

B.1 Unification

Proposition B.1.1 (**Proposition 3.2.1**). *Let $\mathcal{B}_{ind} = \prod_{j=1}^{d} \mathcal{B}_j$ be a background distribution of independent features over $\mathcal{X} = \{0,1\}^d$ such that $\mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}}[z_j] = p$. The classical LIME approach advocates for fitting a linear model*

$$(\omega_0, \omega_1, \ldots, \omega_d) = \underset{\boldsymbol{\omega}\in\mathbb{R}^{d+1}}{\operatorname{argmin}} \mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}_{ind}}\left[\Big(h(\boldsymbol{z}) - \sum_{j=1}^{d} \omega_j z_j - \omega_0\Big)^2\right] \tag{B.1}$$

and reporting the coefficients $(w_1, w_2, \ldots, w_d)$ as the local feature importance for input $\boldsymbol{x} = \mathbf{1}$ (i.e. the full text/image). These weights are proportional to the functional decomposition

$$h_{j,\mathcal{B}_{ind}}(\mathbf{1}) = \omega_j \times (1-p). \tag{B.2}$$

Proof. Let, $z'_j := z_j - p$ for $j = 1, \ldots, d$ be a translation of $\boldsymbol{z}$ to the hypercube $\{-p, 1-p\}^d$. We can then define the translated function $h'(\boldsymbol{z}') = h(\boldsymbol{z}' + p\mathbf{1})$ and distribution $\mathcal{B}'_{\text{ind}}(A) := \mathcal{B}_{\text{ind}}(A + p\mathbf{1})$ over the domain $\{-p, 1-p\}^d$. These translated inputs are still independent but they are now zero-mean $\mathbb{E}_{\boldsymbol{z}'\sim\mathcal{B}'_{\text{ind}}}[z'_i] = 0$.

The key behind the proof is to realize that the optimal weights $(\omega_j)_{j=1}^d$ of Equation B.1 are invariant to translation of the inputs

$$(\omega'_0, \omega_1, \ldots, \omega_d) = \underset{\boldsymbol{\omega}'\in\mathbb{R}^{d+1}}{\operatorname{argmin}} \mathbb{E}_{\boldsymbol{z}'\sim\mathcal{B}'_{\text{ind}}}\left[\Big(h'(\boldsymbol{z}') - \sum_{j=1}^{d} \omega'_j z'_j - \omega'_0\Big)^2\right]. \tag{B.3}$$

This is because the new intercept ω'_0 can account for said translations. Let $N := 2^d$ and $\{\boldsymbol{z}'^{(i)}\}_{i=1}^N$ be the collection of all vertices of the $\{-p, 1-p\}^d$ hypercube. This data is stored in a $N \times (d+1)$ matrix

$$\boldsymbol{Z}' := \begin{bmatrix} 1 & z_1'^{(1)} & \ldots & z_d'^{(1)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & z_1'^{(N)} & \ldots & z_d'^{(N)} \end{bmatrix} \tag{B.4}$$

and their corresponding model predictions are stored in the vector $\boldsymbol{y}' := [h'(\boldsymbol{z}'^{(1)}), \ldots, h'(\boldsymbol{z}'^{(N)})]^T$. The solution to Equation B.3 has a closed form

$$\boldsymbol{\omega}' = (\boldsymbol{Z}'^T \boldsymbol{Z}')^{-1} \boldsymbol{Z}'^T \boldsymbol{y}. \tag{B.5}$$

Expressing this solution in terms of the model h and probability p requires computing various expectations over $\boldsymbol{z}' \sim \mathcal{B}'_{\text{ind}}$. To inverse the matrix $\boldsymbol{Z}'^T\boldsymbol{Z}'$, we must first evaluate

- $\mathbb{E}_{\boldsymbol{z}'\sim\mathcal{B}'_{\text{ind}}}[z'_i\, z'_j] = \mathbb{E}_{\boldsymbol{z}'\sim\mathcal{B}'_{\text{ind}}}[z'_i] \times \mathbb{E}_{\boldsymbol{z}'\sim\mathcal{B}'_{\text{ind}}}[z'_j] = 0$
- $\mathbb{E}_{\boldsymbol{z}'\sim\mathcal{B}'_{\text{ind}}}[1 \times z'_i] = 0$
- $\mathbb{E}_{\boldsymbol{z}'\sim\mathcal{B}'_{\text{ind}}}[(z'_i)^2] = \mathbb{V}_{\boldsymbol{z}'\sim\mathcal{B}'_{\text{ind}}}[z'_i] = \mathbb{V}_{\boldsymbol{z}\sim\mathcal{B}_{\text{ind}}}[z_i] = p(1-p)$.

This leads to the inversion

$$(\boldsymbol{Z}'^T\boldsymbol{Z}')^{-1} = \left(N \begin{bmatrix} 1 & \mathbf{0}^T \\ \mathbf{0} & p(1-p)\boldsymbol{I}_d \end{bmatrix}\right)^{-1} = \frac{1}{N} \begin{bmatrix} 1 & \mathbf{0}^T \\ \mathbf{0} & 1/[p(1-p)]\boldsymbol{I}_d. \end{bmatrix} \tag{B.6}$$

To evaluate $\boldsymbol{Z}'^T\boldsymbol{y}$, let $\mu := \mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}_{\text{ind}}}[h(\boldsymbol{z})]$ and note that $\mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}_{\text{ind}}}[z_j h(\boldsymbol{z})] = p\,\mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}_{\text{ind}}}[h(\boldsymbol{z})|z_j = 1]$.

$$\begin{aligned}
\boldsymbol{Z}'^T\boldsymbol{y} &= N\Big[\mathop{\mathbb{E}}_{\boldsymbol{z}'\sim\mathcal{B}'_{\text{ind}}}[h'(\boldsymbol{z}')], \mathop{\mathbb{E}}_{\boldsymbol{z}'\sim\mathcal{B}'_{\text{ind}}}[z'_1 h'(\boldsymbol{z}')], \ldots, \mathop{\mathbb{E}}_{\boldsymbol{z}'\sim\mathcal{B}'_{\text{ind}}}[z'_d h'(\boldsymbol{z}')]\Big]^T \\
&= N\Big[\mu, \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}_{\text{ind}}}[(z_1 - p)h(\boldsymbol{z})], \ldots, \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}_{\text{ind}}}[(z_d - p)h(\boldsymbol{z})]\Big]^T \\
&= N\Big[\mu, \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}_{\text{ind}}}[z_1 h(\boldsymbol{z})] - p\mu, \ldots, \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}_{\text{ind}}}[z_d h(\boldsymbol{z})] - p\mu\Big]^T \\
&= N\Big[\mu, p\mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}_{\text{ind}}}[h(\boldsymbol{z})|z_1 = 1] - p\mu, \ldots, p\mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}_{\text{ind}}}[h(\boldsymbol{z})|z_d = 1] - p\mu\Big]^T.
\end{aligned} \tag{B.7}$$

Plugging Equations B.6 & B.7 into Equation B.5 yields

$$\omega_j = \Big(\mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}_{\text{ind}}}[h(\boldsymbol{z})|z_j = 1] - \mu\Big)/(1-p). \tag{B.8}$$

Looking back at the definition of Interventional Decompositions (cf. Definition 3.1.4), the term $(\mathbb{E}_{\boldsymbol{z}\sim\mathcal{B}_{\text{ind}}}[h(\boldsymbol{z})|z_j = 1] - \mu)$ is non-other than the main effect $h_{j,\mathcal{B}_{\text{ind}}}(\mathbf{1})$ and so

$$\omega_j = h_{j,\mathcal{B}_{\text{ind}}}(\mathbf{1})/(1-p). \tag{B.9}$$

□

Proposition B.1.2 (**Proposition 3.2.2**). *Assume that $y = h(\boldsymbol{x}) + \epsilon$ where ϵ is a random variable that is independent of $\boldsymbol{x}$, $\mathbb{E}[\epsilon] = 0$, and $\mathbb{E}[\epsilon^2] = \sigma^2$. Then,*

$$\begin{aligned}\Phi_j^{PFI\text{-}O}(h, \mathcal{B}) &= \mathop{\mathbb{E}}_{\substack{\boldsymbol{x}\sim\mathcal{B}\\ \boldsymbol{z}\sim\mathcal{B}}}\Big[\Big(h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j) - h(\boldsymbol{x})\Big)^2\Big] \\ &= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}_j\times\mathcal{B}_{-j}}\Big[\Big(\sum_{u\subseteq[d]:j\in u} h_{u,\mathcal{B}_j\times\mathcal{B}_{-j}}(\boldsymbol{x})\Big)^2\Big] + \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\Big[\Big(\sum_{u\subseteq[d]:j\in u} h_{u,\mathcal{B}}(\boldsymbol{x})\Big)^2\Big].\end{aligned} \tag{B.10}$$

Proof. Remember that

$$\Phi_j^{\text{PFI-O}}(h, \mathcal{D}) := \mathop{\mathbb{E}}_{\substack{(\boldsymbol{x},y)\sim\mathcal{D}\\ (\boldsymbol{z},y')\sim\mathcal{D}}}\Big[\Big(h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j) - y\Big)^2\Big] - \mathop{\mathbb{E}}_{(\boldsymbol{x},y)\sim\mathcal{D}}\Big[\Big(h(\boldsymbol{x}) - y\Big)^2\Big], \tag{B.11}$$

where $\mathcal{D}$ is the data generating distribution over $\mathcal{X}\times\mathcal{Y}$. We let the background $\mathcal{B}$ be the marginal of $\mathcal{D}$ over $\mathcal{X}$. Now, to remove any y from the PFI-O explainer, we leverage the assumptions of symmetric noise ϵ that is independent of $\boldsymbol{x}$. Given these assumptions

$$\mathop{\mathbb{E}}_{(\boldsymbol{x},y)\sim\mathcal{D}}\Big[\Big(h(\boldsymbol{x}) - y\Big)^2\Big] = \mathbb{E}[\epsilon^2] = \sigma^2.$$

The left term in the PFI-O can be expressed

$$\begin{aligned}\mathop{\mathbb{E}}_{\substack{(\boldsymbol{x},y)\sim\mathcal{D}\\ (\boldsymbol{z},y')\sim\mathcal{D}}}\Big[\Big(h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j) - y\Big)^2\Big] &= \mathop{\mathbb{E}}_{\substack{(\boldsymbol{x},\epsilon)\sim\mathcal{D}\\ \boldsymbol{z}\sim\mathcal{B}}}\Big[\Big(h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j) - h(\boldsymbol{x}) - \epsilon\Big)^2\Big] \\ &= \mathop{\mathbb{E}}_{\substack{\boldsymbol{x}\sim\mathcal{B}\\ \boldsymbol{z}\sim\mathcal{B}}}\Big[\Big(h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j) - h(\boldsymbol{x})\Big)^2\Big] - 2\,\mathbb{E}[\epsilon]\mathop{\mathbb{E}}_{\substack{\boldsymbol{x}\sim\mathcal{B}\\ \boldsymbol{z}\sim\mathcal{B}}}[h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j) - h(\boldsymbol{x})] + \sigma^2 \\ &= \mathop{\mathbb{E}}_{\substack{\boldsymbol{x}\sim\mathcal{B}\\ \boldsymbol{z}\sim\mathcal{B}}}\Big[\Big(h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j) - h(\boldsymbol{x})\Big)^2\Big] + \sigma^2\end{aligned}$$

and so the noise σ^2 computed previously cancels out. We are left with

$$\Phi_j^{\text{PFI-O}}(h, \mathcal{B}) = \mathop{\mathbb{E}}_{\substack{\boldsymbol{x}\sim\mathcal{B}\\ \boldsymbol{z}\sim\mathcal{B}}}\Big[\Big(h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j) - h(\boldsymbol{x})\Big)^2\Big], \tag{B.12}$$

which only depends on the model h and marginal $\mathcal{B}$ over $\mathcal{X}$. We now wish to express the

Original PFI in a manner that involves Interventional Decompositions.

$$
\begin{aligned}
\Phi_j^{\text{PFI-O}}(h, \mathcal{B}) &= \mathop{\mathbb{E}}_{\substack{\boldsymbol{x}\sim\mathcal{B}\\ \boldsymbol{z}\sim\mathcal{B}}}\Big[\Big(h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j) - h(\boldsymbol{x})\Big)^2\Big] \\
&= \mathop{\mathbb{E}}_{\substack{\boldsymbol{x}\sim\mathcal{B}\\ \boldsymbol{z}\sim\mathcal{B}}}\Big[h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j)^2 - 2\,h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j)\,h(\boldsymbol{x}) + h(\boldsymbol{x})^2\Big] \\
&= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\Big[\mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j)^2] - 2\,h(\boldsymbol{x})\mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j)] + h(\boldsymbol{x})^2\Big] \\
&= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\Big[\mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j)^2] - 2\,h(\boldsymbol{x})\mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j)] + h(\boldsymbol{x})^2 \\
&\qquad + \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j)]^2 - \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j)]^2\Big] \\
&= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\Big[\mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j)^2] - \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j)]^2 + \Big(h(\boldsymbol{x}) - \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j)]\Big)^2\Big] \\
&= \underbrace{\mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\Big[\mathop{\mathbb{V}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j)]\Big]}_{A} + \underbrace{\mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\Big[\Big(h(\boldsymbol{x}) - \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j)]\Big)^2\Big]}_{B}
\end{aligned}
$$

The A term can be rewritten in terms of the Interventional Decomposition $h_{u,\mathcal{B}_j\times\mathcal{B}_{-j}}(\boldsymbol{x})$

$$
\begin{aligned}
\mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\Big[\mathop{\mathbb{V}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j)]\Big] &= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\Big[\mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}\Big[\Big(h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j) - \mathop{\mathbb{E}}_{\boldsymbol{v}\sim\mathcal{B}}[h(\boldsymbol{x}_{-j}, \boldsymbol{v}_j)]\Big)^2\Big]\Big] \\
&= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}_j\times\mathcal{B}_{-j}}\Big[\Big(h(\boldsymbol{x}) - \mathop{\mathbb{E}}_{\boldsymbol{v}\sim\mathcal{B}_j\times\mathcal{B}_{-j}}[h(\boldsymbol{x}_{-j}, \boldsymbol{v}_j)]\Big)^2\Big] \\
&= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}_j\times\mathcal{B}_{-j}}\Big[\Big(\sum_{u\subseteq[d]} h_{u,\mathcal{B}_j\times\mathcal{B}_{-j}}(\boldsymbol{x}) - \sum_{u\subseteq[d]\setminus\{j\}} h_{u,\mathcal{B}_j\times\mathcal{B}_{-j}}(\boldsymbol{x})\Big)^2\Big] \\
&= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}_j\times\mathcal{B}_{-j}}\Big[\Big(\sum_{u\subseteq[d]:j\in u} h_{u,\mathcal{B}_j\times\mathcal{B}_{-j}}(\boldsymbol{x})\Big)^2\Big].
\end{aligned}
$$

The term B can be expressed as a function of the Interventional Decomposition $h_{u,\mathcal{B}}(\boldsymbol{x})$

$$
\begin{aligned}
\mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\Big[\Big(h(\boldsymbol{x}) - \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j)]\Big)^2\Big] &= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\Big[\Big(\sum_{u\subseteq[d]} h_{u,\mathcal{B}}(\boldsymbol{x}) - \sum_{u\subseteq[d]\setminus\{j\}} h_{u,\mathcal{B}}(\boldsymbol{x})\Big)^2\Big] \\
&= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\Big[\Big(\sum_{u\subseteq[d]:j\in u} h_{u,\mathcal{B}}(\boldsymbol{x})\Big)^2\Big].
\end{aligned}
$$

Combining A and B yields the desired result. □

Proposition B.1.3 (Proposition 3.2.3). *Let* $(\Omega_{-j}^{[1]}, \dots, \Omega_{-j}^{[M]})$ *be a partition of* $\mathcal{X}_{-j}$ *and define the corresponding regions* $\Omega^{[t]} := \Omega_{-j}^{[t]} \times \mathcal{X}_j$. *If* $x_j \perp \boldsymbol{x}_{-j} | \{\boldsymbol{x}_{-j} \in \Omega_{-j}^{[t]}\}$ *for all* $t = 1, 2, \dots M$, *it holds that*

$$\Phi_j^{\textit{Total-Sobol}}(h, \mathcal{B}) = \frac{1}{2}\Phi_j^{\textit{CPFI}}(h, \mathcal{B}) = \sum_{t=1}^{M} \mathcal{B}(\Omega^{[t]})\, \Phi_j^{\textit{Marginal-Sobol}}(h, \mathcal{B}_{\Omega^{[t]}}) = \sum_{t=1}^{M} \mathcal{B}(\Omega^{[t]})\, \Phi_j^{\textit{PFI}}(h, \mathcal{B}_{\Omega^{[t]}}). \tag{B.13}$$

Proof. Let $f : \mathcal{X}_{-j} \to \mathbb{R}$ be an arbitrary function. Since $(\Omega_{-j}^{[1]}, \dots, \Omega_{-j}^{[M]})$ forms a partition of $\mathcal{X}_{-j}$, the law of total expectation implies

$$\mathop{\mathbb{E}}_{\boldsymbol{x}_{-j} \sim \mathcal{B}_{-j}}[f(\boldsymbol{x}_{-j})] = \sum_{t=1}^{M} \mathop{\mathbb{P}}_{\boldsymbol{x}_{-j} \sim \mathcal{B}_{-j}}[\boldsymbol{x}_{-j} \in \Omega_{-j}^{[t]}] \times \mathop{\mathbb{E}}_{\boldsymbol{x}_{-j} \sim \mathcal{B}_{-j}}[f(\boldsymbol{x}_{-j}) | \boldsymbol{x}_{-j} \in \Omega_{-j}^{[t]}]. \tag{B.14}$$

Equation B.14 can be written in an equivalent form that involves the regional backgrounds $\mathcal{B}_{\Omega^{[t]}}$

$$\mathop{\mathbb{E}}_{\boldsymbol{x}_{-j} \sim \mathcal{B}_{-j}}[f(\boldsymbol{x}_{-j})] = \sum_{t=1}^{M} \mathcal{B}(\Omega^{[t]}) \times \mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mathcal{B}_{\Omega^{[t]}}}[f(\boldsymbol{x}_{-j})]. \tag{B.15}$$

Now, if we define f as the function $f(\boldsymbol{x}_{-j}) := \mathbb{V}_{x_j}[h(\boldsymbol{x}) | \boldsymbol{x}_{-j}]$ that returns the variance conditioned on $\boldsymbol{x}_{-j}$, the Total Sobol Index can be reformulated

$$\begin{aligned}
\Phi_j^{\text{Total-Sobol}}(h, \mathcal{B}) &:= \mathop{\mathbb{E}}_{\boldsymbol{x}_{-j}} \Big[\mathop{\mathbb{V}}_{x_j}[h(\boldsymbol{x}) | \boldsymbol{x}_{-j}] \Big] \\
&= \sum_{t=1}^{M} \mathcal{B}(\Omega^{[t]})\, \Phi_j^{\text{Total-Sobol}}(h, \mathcal{B}_{\Omega^{[t]}}) && \text{(cf. Equation B.15)} \\
&= \sum_{t=1}^{M} \mathcal{B}(\Omega^{[t]}) \mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mathcal{B}_{\Omega^{[t]}}} \Big[\mathop{\mathbb{V}}_{\boldsymbol{z} \sim \mathcal{B}_{\Omega^{[t]}}} [h(\boldsymbol{x}_{-j}, \boldsymbol{z}_j)] \Big] && \text{(Since } x_j \perp \boldsymbol{x}_{-j} \text{ in } \mathcal{B}_{\Omega^{[t]}}) \\
&= \sum_{t=1}^{M} \mathcal{B}(\Omega^{[t]})\, \Phi_j^{\text{Marginal-Sobol}}(h, \mathcal{B}_{\Omega^{[t]}}) && \text{(By Def. of Marginal-Sobol)} \\
&= \sum_{t=1}^{M} \mathcal{B}(\Omega^{[t]})\, \Phi_j^{\text{PFI}}(h, \mathcal{B}_{\Omega^{[t]}}). && \text{(Since } x_j \perp \boldsymbol{x}_{-j} \text{ in } \mathcal{B}_{\Omega^{[t]}})
\end{aligned}$$

□

B.2 Model-Agnostic

Proposition B.2.1 (**Proposition 4.1.1**). *Let $\{\boldsymbol{x}^{(i)}\}_{i=1}^N \sim \mathcal{F}^N$ be a sequence of N iid foreground observations, then the following holds*

$$\frac{(-1)^{|u|}}{N}\sum_{i=1}^{N} H_{ij}^{u} \xrightarrow{p} \sum_{v\subseteq[d]:u\subseteq v} h_{v,\mathcal{F}}(\boldsymbol{z}^{(j)}). \tag{B.16}$$

Proof. We have

$$\begin{aligned}
\frac{(-1)^{|u|}}{N}\sum_{i=1}^{N} H_{ij}^{u} &= \frac{1}{N}\sum_{i=1}^{N}(-1)^{|u|}h_{u,\boldsymbol{z}^{(j)}}(\boldsymbol{x}^{(i)}) \\
&= \frac{1}{N}\sum_{i=1}^{N}(-1)^{|u|}\sum_{v\subseteq u}(-1)^{|u|-|v|}h(\boldsymbol{x}_v^{(i)}, \boldsymbol{z}_{-v}^{(j)}) \\
&= \frac{1}{N}\sum_{i=1}^{N}\sum_{v\subseteq u}(-1)^{|v|}h(\boldsymbol{x}_v^{(i)}, \boldsymbol{z}_{-v}^{(j)}) \\
&= h(\boldsymbol{z}^{(j)}) + \sum_{v\subseteq u:|v|\geq 1}(-1)^{|v|}\frac{1}{N}\sum_{i=1}^{N} h(\boldsymbol{x}_v^{(i)}, \boldsymbol{z}_{-v}^{(j)}) \\
&\xrightarrow{p} h(\boldsymbol{z}^{(j)}) + \sum_{v\subseteq u:|v|\geq 1}(-1)^{|v|}\mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{F}}[h(\boldsymbol{x}_v, \boldsymbol{z}_{-v}^{(j)})] \\
&= h(\boldsymbol{z}^{(j)}) + \sum_{v\subseteq u:|v|\geq 1}(-1)^{|v|}\sum_{w\subseteq[d]\setminus v} h_{w,\mathcal{F}}(\boldsymbol{z}^{(j)}).
\end{aligned}$$

Before simplifying the term $\sum_{v\subseteq u:|v|\geq 1}(-1)^{|v|}\sum_{w\subseteq[d]\setminus v} h_{w,\mathcal{F}}(\boldsymbol{z}^{(j)})$ we must introduce a few definitions. Let

$$E_i := \{w \subseteq [d] : i \notin w\} \tag{B.17}$$

be the set of all subsets that exclude a certain element i. These sets can be combined to create the following

$$\{w \subseteq [d] \setminus v\} := \cap_{i\in v} E_i \tag{B.18}$$

and

$$\{v \subseteq [d] : u \nsubseteq v\} := \cup_{i\in u} E_i. \tag{B.19}$$

Additionally, recall the inclusion-exclusion principle

$$|\cup_{i\in u} E_i| = \sum_{v\subseteq u:|v|\geq 1}(-1)^{|v|+1}\,|\cap_{i\in v} E_i|, \tag{B.20}$$

that generalizes the formula $|A\cup B\cup C| = |A|+|B|+|C|-|A\cap B|-|A\cap C|-|B\cap C|+|A\cap B\cap C|$.

From the point of view of Functional Decomposition, this principle implies

$$\sum_{v \in \cup_{i \in u} E_i} h_{v,\mathcal{F}}(\boldsymbol{z}^{(j)}) = \sum_{v \subseteq u : |v| \geq 1} (-1)^{|v|+1} \sum_{w \in \cap_{i \in v} E_i} h_{w,\mathcal{F}}(\boldsymbol{z}^{(j)}). \tag{B.21}$$

Going back to the derivation

$$\begin{aligned}
\frac{(-1)^{|u|}}{N} \sum_{i=1}^{N} H_{ij}^{u} &\xrightarrow{p} h(\boldsymbol{z}^{(j)}) + \sum_{v \subseteq u : |v| \geq 1} (-1)^{|v|} \sum_{w \subseteq [d] \setminus v} h_{w,\mathcal{F}}(\boldsymbol{z}^{(j)}) \\
&= h(\boldsymbol{z}^{(j)}) + \sum_{v \subseteq u : |v| \geq 1} (-1)^{|v|} \sum_{w \in \cap_{i \in v} E_i} h_{w,\mathcal{F}}(\boldsymbol{z}^{(j)}) && \text{(Equation B.18)} \\
&= h(\boldsymbol{z}^{(j)}) - \sum_{v \in \cup_{i \in u} E_i} h_{v,\mathcal{F}}(\boldsymbol{z}^{(j)}) && \text{(Equation B.21)} \\
&= h(\boldsymbol{z}^{(j)}) - \sum_{v \subseteq [d] : u \not\subseteq v} h_{v,\mathcal{F}}(\boldsymbol{z}^{(j)}) && \text{(Equation B.19)} \\
&= \sum_{v \subseteq [d]} h_{v,\mathcal{F}}(\boldsymbol{z}^{(j)}) - \sum_{v \subseteq [d] : u \not\subseteq v} h_{v,\mathcal{F}}(\boldsymbol{z}^{(j)}) = \sum_{v \subseteq [d] : u \subseteq v} h_{v,\mathcal{F}}(\boldsymbol{z}^{(j)}).
\end{aligned}$$

Concluding the proof. □

B.3 TreeSHAP

Lemma B.3.1 (Lemma 5.4.4). *If P contains no type B edges and the sets $\mathcal{I}(S_X)$ and $\mathcal{I}(S_Z)$ are disjoint, then for any $u \subseteq \mathcal{I}(S_{XZ})$ we have*

$$(h^P \circ \boldsymbol{\xi})(\, \boldsymbol{r}_u^{\boldsymbol{z}}(\boldsymbol{x})\,) = \begin{cases} v_l & \text{if } u = \mathcal{I}(S_X) \\ 0 & \text{otherwise.} \end{cases} \tag{B.22}$$

where $l \in L$ is the leaf node at the end of P.

Proof. Since P does not contain a type B edge, we can use Equation 5.23, which, as a reminder, states that

$$(h_P \circ \boldsymbol{\xi})(\boldsymbol{r}_u^{\boldsymbol{z}}(\boldsymbol{x})) = v_l \prod_{\substack{e \in P \\ e \text{ type X}}} \mathbb{1}(\mathcal{I}(i_e) \in u) \prod_{\substack{e \in P \\ e \text{ type Z}}} \mathbb{1}(\mathcal{I}(i_e) \notin u). \tag{B.23}$$

We need to prove the following two statements.

1. $(h_P \circ \boldsymbol{\xi})(\boldsymbol{r}^{\boldsymbol{z}}_{\mathcal{I}(S_X)}(\boldsymbol{x})) = v_l$. On the one hand, for every edge e of type X in P, by definition $\mathcal{I}(i_e) \in \mathcal{I}(S_X)$. On the other hand, for every edge e of type Z in P, by definition $\mathcal{I}(i_e) \in \mathcal{I}(S_Z)$, so $\mathcal{I}(i_e) \notin \mathcal{I}(S_X)$ since we assume that $\mathcal{I}(S_X) \cap \mathcal{I}(S_Z) = \emptyset$. Therefore, it follows that $(h_P \circ \boldsymbol{\xi})(\boldsymbol{r}^{\boldsymbol{z}}_{\mathcal{I}(S_X)}(\boldsymbol{x})) = v_l$ using Equation B.23.

2. If $u \subseteq \mathcal{I}(S_{XZ})$ and $u \neq \mathcal{I}(S_X)$, then $(h_P \circ \boldsymbol{\xi})(\boldsymbol{r}_u^{\boldsymbol{z}}(\boldsymbol{x})) = 0$. Assume that $u \subseteq \mathcal{I}(S_{XZ})$ and $u \neq \mathcal{I}(S_X)$. Since, $u \neq \mathcal{I}(S_X)$, at least one of the two following cases must occur, (a) there exists $j \in \mathcal{I}(S_X) \setminus u$ or (b) there exists $j \in u \setminus \mathcal{I}(S_X)$.

   (a) Let $j \in \mathcal{I}(S_X) \setminus u$. We can find $e \in P$ of type X such that $\mathcal{I}(i_e) = j$. We then have $\mathbb{1}(\mathcal{I}(i_e) \in u) = 0$.

   (b) Let $j \in u \setminus \mathcal{I}(S_X)$. Since $u \subseteq \mathcal{I}(S_{XZ})$, it follows that $j \in \mathcal{I}(S_Z)$. We can find $e \in P$ of type Z such that $\mathcal{I}(i_e) = j$. Then as $j \in u$, we have $\mathbb{1}(\mathcal{I}(i_e) \notin u) = 0$.

   Given that either case (a) or case (b) must occur, using Equation B.23 we see that $(h_P \circ \boldsymbol{\xi})(\boldsymbol{r}_u^{\boldsymbol{z}}(\boldsymbol{x})) = 0$, as desired.

□

Theorem B.3.1 (**Theorem 5.4.1**). *If P contains no type B edges and the sets $\mathcal{I}(S_X)$ and $\mathcal{I}(S_Z)$ are disjoint, then*

$$(h^P \circ \boldsymbol{\xi})_{u,\boldsymbol{z}}(\boldsymbol{x}) = \begin{cases} (-1)^{|u|-|\mathcal{I}(S_X)|}\, v_l & \text{if } \mathcal{I}(S_X) \subseteq u \subseteq \mathcal{I}(S_{XZ}) \\ 0 & \text{otherwise.} \end{cases} \tag{B.24}$$

Proof. Let $u \nsubseteq \mathcal{I}(S_{XZ})$, then by Lemma 5.4.3 there exists a dummy feature $j \in u$. By the dummy property of the $\boldsymbol{z}$-Anchored Decomposition (cf. Equation 3.19) the component $(h^P \circ \boldsymbol{\xi})_{u,\boldsymbol{z}}(\boldsymbol{x}) = 0$.

Alternatively, let $u \subseteq \mathcal{I}(S_{XZ})$, then the $\boldsymbol{z}$-Anchored Decomposition is calculated via

$$(h^P \circ \boldsymbol{\xi})_{u,\boldsymbol{z}}(\boldsymbol{x}) = \sum_{v \subseteq u} (-1)^{|u|-|v|} (h^P \circ \boldsymbol{\xi})(\boldsymbol{r}_v^{\boldsymbol{z}}(\boldsymbol{x})). \tag{B.25}$$

Since we are assuming that P contains no type B edges and the sets $\mathcal{I}(S_X)$ and $\mathcal{I}(S_Z)$ are disjoint, Lemma 5.4.4 is applicable. The Lemma states that the only subset $v \subseteq u$ that has $(h^P \circ \boldsymbol{\xi})(\boldsymbol{r}_v^{\boldsymbol{z}}(\boldsymbol{x})) \neq 0$ is $v = \mathcal{I}(S_X)$, which is only attainable when $\mathcal{I}(S_X) \subseteq u$ in the first place. Consequently, if $\mathcal{I}(S_X) \subseteq u \subseteq \mathcal{I}(S_{XZ})$, the functional component is given by

$$(h^P \circ \boldsymbol{\xi})_{u,\boldsymbol{z}}(\boldsymbol{x}) = \sum_{v \subseteq u} (-1)^{|u|-|v|} (h^P \circ \boldsymbol{\xi})(\boldsymbol{r}_v^{\boldsymbol{z}}(\boldsymbol{x})) = (-1)^{|u|-|\mathcal{I}(S_X)|} v_l, \tag{B.26}$$

concluding the proof. □

Theorem B.3.2 (**Theorem 5.4.2**). *Let P be a maximal path that contains no type B edges and let the sets $\mathcal{I}(S_X)$ and $\mathcal{I}(S_Z)$ be disjoint. Moreover, define the interventional game $\nu^{int}_{h,\boldsymbol{x},\boldsymbol{z}}(u) := (h^P \circ \boldsymbol{\xi})(\, \boldsymbol{r}_u^{\boldsymbol{z}}(\boldsymbol{x})\,)$. Then all features that are not in $\mathcal{I}(S_{XZ})$ are dummies and get a zero Shapley value. Features $k \in \mathcal{I}(S_{XZ})$ get a Shapley value of*

$$\phi_k^{\text{SHAP-int}}(h^P \circ \boldsymbol{\xi}, \boldsymbol{x}, \boldsymbol{z}) = \sum_{u \subseteq \mathcal{I}(S_{XZ}) \setminus \{k\}} W(|u|, |\mathcal{I}(S_{XZ})|) \Big(\nu^{int}_{h,\boldsymbol{x},\boldsymbol{z}}(u \cup \{k\}) - \nu^{int}_{h,\boldsymbol{x},\boldsymbol{z}}(u) \Big) \tag{B.27}$$

The exponential cost $\mathcal{O}(2^{|\mathcal{I}(S_{XZ})|})$ of computing these terms reduces to $\mathcal{O}(1)$ following

$$k \in \mathcal{I}(S_X) \Rightarrow \phi_k^{\text{SHAP-int}}(h^P \circ \boldsymbol{\xi}, \boldsymbol{x}, \boldsymbol{z}) = W(|\mathcal{I}(S_X)| - 1, |\mathcal{I}(S_{XZ})|) v_l \tag{B.28}$$

$$k \in \mathcal{I}(S_Z) \Rightarrow \phi_k^{\text{SHAP-int}}(h^P \circ \boldsymbol{\xi}, \boldsymbol{x}, \boldsymbol{z}) = -W(|\mathcal{I}(S_X)|, |\mathcal{I}(S_{XZ})|) v_l. \tag{B.29}$$

Proof. By Lemma 5.4.3, all features not in $\mathcal{I}(S_{XZ})$ are dummies. By the dummy axiom of the Shapley values (cf. Equation 2.73), the corresponding Shapley values are null. Now, the

expression of the Shapley values can be simplified knowing that some features are dummies. For instance, let $i \notin \mathcal{I}(S_{XZ})$ be a dummy and $k \in \mathcal{I}(S_{XZ})$ be a non-dummy. The Shapley values can be rewritten

$$\begin{aligned}
\phi_k^{\text{SHAP-int}}(h^P \circ \boldsymbol{\xi}, \boldsymbol{x}, \boldsymbol{z}) &= \sum_{u \subseteq [d] \setminus \{i,k\}} W(|u|, d)\Big(\nu^{\text{int}}_{h,\boldsymbol{x},\boldsymbol{z}}(u \cup \{k\}) - \nu^{\text{int}}_{h,\boldsymbol{x},\boldsymbol{z}}(u)\Big) + \\
&\qquad W(|S|+1, d)\Big(\nu^{\text{int}}_{h,\boldsymbol{x},\boldsymbol{z}}(u \cup \{i,k\}) - \nu^{\text{int}}_{h,\boldsymbol{x},\boldsymbol{z}}(u \cup \{i\})\Big) \\
&= \sum_{u \subseteq [d] \setminus \{i,k\}} \Big(W(|u|, d) + W(|u|+1, d)\Big)\Big(\nu^{\text{int}}_{h,\boldsymbol{x},\boldsymbol{z}}(u \cup \{k\}) - \nu^{\text{int}}_{h,\boldsymbol{x},\boldsymbol{z}}(u)\Big) \\
&= \sum_{u \subseteq [d] \setminus \{i,k\}} W(|u|, d-1)\Big(\nu^{\text{int}}_{h,\boldsymbol{x},\boldsymbol{z}}(u \cup \{k\}) - \nu^{\text{int}}_{h,\boldsymbol{x},\boldsymbol{z}}(u)\Big).
\end{aligned}$$

Repeating this procedure for all dummies $i \notin \mathcal{I}(S_{XZ})$ yields Equation B.27.

We now prove Equations B.28 and B.29 separately. First, suppose that $k \in \mathcal{I}(S_X)$. Then according to Lemma 5.4.4, the only coalition $u \subseteq \mathcal{I}(S_{XZ}) \setminus \{k\}$ for which $\nu^{\text{int}}_{h,\boldsymbol{x},\boldsymbol{z}}(u \cup \{k\}) - \nu^{\text{int}}_{h,\boldsymbol{x},\boldsymbol{z}}(u) \neq 0$ is when $u = \mathcal{I}(S_X) \setminus \{k\}$. Indeed, when $u = \mathcal{I}(S_X) \setminus \{k\}$

$$\nu^{\text{int}}_{h,\boldsymbol{x},\boldsymbol{z}}(u \cup \{k\}) - \nu^{\text{int}}_{h,\boldsymbol{x},\boldsymbol{z}}(u) = \nu^{\text{int}}_{h,\boldsymbol{x},\boldsymbol{z}}(\mathcal{I}(S_X)) - \nu^{\text{int}}_{h,\boldsymbol{x},\boldsymbol{z}}(\mathcal{I}(S_X) \setminus \{k\}) = v_l - 0 = v_l. \tag{B.30}$$

The corresponding Shapley value is $W(|u|, |\mathcal{I}(S_{XZ})|)\, v_l = W(|\mathcal{I}(S_X) \setminus \{k\}|, |\mathcal{I}(S_{XZ})|)\, v_l = W(|\mathcal{I}(S_X)| - 1, |\mathcal{I}(S_{XZ})|)\, v_l$.

Second, suppose that $k \in \mathcal{I}(S_Z)$. According to Lemma 5.4.4, the only coalition $u \subseteq \mathcal{I}(S_{XZ}) \setminus \{k\}$ for which $\nu^{\text{int}}_{h,\boldsymbol{x},\boldsymbol{z}}(u \cup \{k\}) - \nu^{\text{int}}_{h,\boldsymbol{x},\boldsymbol{z}}(u) \neq 0$ is when $u = \mathcal{I}(S_X)$ Indeed, when $u = \mathcal{I}(S_X)$

$$\nu^{\text{int}}_{h,\boldsymbol{x},\boldsymbol{z}}(u \cup \{k\}) - \nu^{\text{int}}_{h,\boldsymbol{x},\boldsymbol{z}}(u) = \nu^{\text{int}}_{h,\boldsymbol{x},\boldsymbol{z}}(\mathcal{I}(S_X) \cup \{k\}) - \nu^{\text{int}}_{h,\boldsymbol{x},\boldsymbol{z}}(\mathcal{I}(S_X)) = 0 - v_l = \text{-}v_l. \tag{B.31}$$

We therefore get a Shapley value $-W(|u|, |\mathcal{I}(S_{XZ})|)\, v_l = -W(|\mathcal{I}(S_X)|, |\mathcal{I}(S_{XZ})|)\, v_l$. □

APPENDIX C SUPPLEMENTARY OF FDTREES

C.1 Proofs

Lemma C.1.1. *If h is additive in feature j then*

$$j \notin u \Rightarrow h_{u,\mathcal{B}_j \times \mathcal{B}_{-j}} = h_{u,\mathcal{B}'_j \times \mathcal{B}_{-j}} \tag{C.1}$$

for any two distributions $\mathcal{B}_j$ and $\mathcal{B}'_j$ on x_j.

Proof. According to Equation 3.18, the $h_{u,\mathcal{B}}$ component of the Interventional Decomposition can be written

$$h_{u,\mathcal{B}}(\boldsymbol{x}) = \sum_{v \subseteq u} (-1)^{|u \setminus v|} \mathop{\mathbb{E}}_{\boldsymbol{z} \sim \mathcal{B}} [h(\boldsymbol{x}_v, \boldsymbol{z}_{-v})]. \tag{C.2}$$

By our assumptions, $h(\boldsymbol{x}) = g_j(x_j) + g_{-j}(\boldsymbol{x}_{-j})$ for some g_j and g_{-j}, and the the probability density under $\mathcal{B}$ can be expressed $\rho(\boldsymbol{x}) = \rho_j(x_j)\rho_{-j}(\boldsymbol{x}_{-j})$. Now, for any subset u excluding j

we have

$$
\begin{aligned}
h_{u,\mathcal{B}}(\boldsymbol{x}) &= \sum_{v\subseteq u}(-1)^{|u\setminus v|}\mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[h(\boldsymbol{x}_v,\boldsymbol{z}_{-v})] \\
&= \sum_{v\subseteq u}(-1)^{|u\setminus v|}\left(\mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[g_j(\boldsymbol{x}_v,\boldsymbol{z}_{-v})] + \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[g_{-j}(\boldsymbol{x}_v,\boldsymbol{z}_{-v})]\right) \\
&= \sum_{v\subseteq u}(-1)^{|u\setminus v|}\left(\mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[g_j(z_j)] + \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[g_{-j}(\boldsymbol{x}_v,\boldsymbol{z}_{-v})]\right) \qquad \text{(Since } j\notin v\text{)} \\
&= \mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[g_j(z_j)]\sum_{v\subseteq u}(-1)^{|u\setminus v|} + \sum_{v\subseteq u}(-1)^{|u\setminus v|}\mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[g_{-j}(\boldsymbol{x}_v,\boldsymbol{z}_{-v})] \\
&\qquad\qquad \text{(Note that } \textstyle\sum_{v\subseteq u}(-1)^{|u\setminus v|}=0\text{)} \\
&= \sum_{v\subseteq u}(-1)^{|u\setminus v|}\mathop{\mathbb{E}}_{\boldsymbol{z}\sim\mathcal{B}}[g_{-j}(\boldsymbol{x}_v,\boldsymbol{z}_{-v})] \\
&= \sum_{v\subseteq u}(-1)^{|u\setminus v|}\int_{\mathcal{X}} g_{-j}(\boldsymbol{x}_v,\boldsymbol{z}_{-v})\rho(\boldsymbol{z})d\boldsymbol{z} \\
&= \sum_{v\subseteq u}(-1)^{|u\setminus v|}\int_{\mathcal{X}} \underbrace{g_{-j}(\boldsymbol{x}_v,\boldsymbol{z}_{-v})}_{\text{does not depend on } z_j}\ \rho_{-j}(\boldsymbol{z}_{-j})\rho_j(z_j)d\boldsymbol{z}_{-j}dz_j \\
&= \sum_{v\subseteq u}(-1)^{|u\setminus v|}\int_{\mathcal{X}_{-j}} g_{-j}(\boldsymbol{x}_v,\boldsymbol{z}_{-v})\rho_{-j}(\boldsymbol{z}_{-j})d\boldsymbol{z}_{-j}\int\rho_j(z_j)dz_j \\
&= \sum_{v\subseteq u}(-1)^{|u\setminus v|}\int_{\mathcal{X}_{-j}} g_{-j}(\boldsymbol{x}_v,\boldsymbol{z}_{-v})\rho_{-j}(\boldsymbol{z}_{-j})d\boldsymbol{z}_{-j}.
\end{aligned}
$$

This expression is independent of the choice of $\rho_j(z_j)$. □

Theorem C.1.1 (**Theorem 6.2.1**). *Any function*

$$
L_h(\mathcal{B}) = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\left[\sum_{\substack{u,v\subseteq[d]\\|u|\geq 2,|v|\geq 2}} a(u,v)h_{u,\mathcal{B}}(\boldsymbol{x})h_{v,\mathcal{B}}(\boldsymbol{x})\right] \tag{C.3}
$$

for some $a : 2^{[d]}\times 2^{[d]}\to\mathbb{R}$ *is a LoA.*

Proof. We demonstrate that the function L_h from Equation C.3 respects the three properties of Definition 6.2.1.

Property 1 By the minimality of the Interventional Decomposition (cf. Corollary 3.1.1), if h is additive over a rectangle $R\supseteq\text{supp}(\mathcal{B})$, then $|u|\geq 2\Rightarrow\forall\boldsymbol{x}\in R\ \ h_{u,\mathcal{B}}(\boldsymbol{x})=0$. So we have

$$
L_h(\mathcal{B}) = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\left[\sum_{\substack{u,v\subseteq[d]\\|u|\geq 2,|v|\geq 2}} a(u,v)h_{u,\mathcal{B}}(\boldsymbol{x})h_{v,\mathcal{B}}(\boldsymbol{x})\right] = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}[0] = 0.
$$

Property 2 By feature independence, we have [Owen, 2013, Appendix A]

$$u \neq v \Rightarrow \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}[h_{u,\mathcal{B}_{\text{ind}}}(\boldsymbol{x})\, h_{v,\mathcal{B}_{\text{ind}}}(\boldsymbol{x})] = 0. \tag{C.4}$$

Letting $\sigma_u^2 := \mathbb{E}_{\boldsymbol{x}\sim\mathcal{B}_{\text{ind}}}[h_{u,\mathcal{B}_{\text{ind}}}(\boldsymbol{x})^2]$, we therefore get

$$L_h(\mathcal{B}_{\text{ind}}) = \sum_{u\subseteq[d]:|u|\geq 2} a(u,u)\,\sigma_u^2, \tag{C.5}$$

and by identification, the interaction penalization weights are $w(u) = a(u,u)$.

Property 3 Let h be additive in feature j meaning that $h(\boldsymbol{x}) = g_j(x_j) + g_{-j}(\boldsymbol{x}_{-j})$. Since the Interventional Decomposition is minimal, there will not be any interaction h_u with $j \in u$. Moreover, let the background factorize as $\mathcal{B} := \mathcal{B}_j \times \mathcal{B}_{-j}$ implying that the corresponding probability density is $\rho(\boldsymbol{x}) = \rho_j(x_j)\rho_{-j}(\boldsymbol{x}_{-j})$.

$$\begin{aligned}
L_h(\mathcal{B}_j \times \mathcal{B}_{-j}) &:= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\left[\sum_{\substack{u,v\subseteq[d]\\|u|\geq 2,|v|\geq 2}} a(u,v) h_{u,\mathcal{B}}(\boldsymbol{x}) h_{v,\mathcal{B}}(\boldsymbol{x})\right] = \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\left[\sum_{\substack{u,v\subseteq[d]\setminus\{j\}\\|u|\geq 2,|v|\geq 2}} a(u,v) h_{u,\mathcal{B}}(\boldsymbol{x}) h_{v,\mathcal{B}}(\boldsymbol{x})\right] \\
&= \int_{\mathcal{X}} \sum_{\substack{u,v\subseteq[d]\setminus\{j\}\\|u|\geq 2,|v|\geq 2}} a(u,v) h_{u,\mathcal{B}}(\boldsymbol{x}) h_{v,\mathcal{B}}(\boldsymbol{x})\, \rho(\boldsymbol{x}) d\boldsymbol{x} \\
&= \int_{\mathcal{X}} \sum_{\substack{u,v\subseteq[d]\setminus\{j\}\\|u|\geq 2,|v|\geq 2}} a(u,v) h_{u,\mathcal{B}}(\boldsymbol{x}) h_{v,\mathcal{B}}(\boldsymbol{x})\, \rho_{-j}(\boldsymbol{x}_{-j})\rho_j(x_j)\, dx_j \\
&= \int_{\mathcal{X}_{-j}} \sum_{\substack{u,v\subseteq[d]\setminus\{j\}\\|u|\geq 2,|v|\geq 2}} a(u,v) h_{u,\mathcal{B}}(\boldsymbol{x}) h_{v,\mathcal{B}}(\boldsymbol{x})\, \rho_{-j}(\boldsymbol{x}_{-j}) d\boldsymbol{x}_{-j} \int \rho_j(x_j) dx_j \\
&= \int_{\mathcal{X}_{-j}} \sum_{\substack{u,v\subseteq[d]\setminus\{j\}\\|u|\geq 2,|v|\geq 2}} a(u,v) h_{u,\mathcal{B}_j\times\mathcal{B}_{-j}}(\boldsymbol{x}) h_{v,\mathcal{B}_j\times\mathcal{B}_{-j}}(\boldsymbol{x})\, \rho_{-j}(\boldsymbol{x}_{-j}) d\boldsymbol{x}_{-j}.
\end{aligned}$$

By Lemma C.1.1, for any u not containing j, the subfunction $h_{u,\mathcal{B}_j\times\mathcal{B}_{-j}}$ does not depend on the choice of $\mathcal{B}_j$. Thus, we have proven $L_h(\mathcal{B}_j \times \mathcal{B}_{-j}) = L_h(\mathcal{B}'_j \times \mathcal{B}_{-j})$ for any alternative distribution $\mathcal{B}'_j$ on feature j. □

Corollary C.1.1 (**Corollary 6.2.1**). *The distances $D(\boldsymbol{\phi}, \boldsymbol{\phi}')$ between local PDP/SHAP/PFI explainers, as well as the CoE are all LoA functions.*

Proof. We prove that these various functions take the form of Equation C.3. First comparing

local PDP and SHAP, we have

$$\begin{aligned}
D(\boldsymbol{\phi}^{\mathrm{PDP}}, \boldsymbol{\phi}^{\mathrm{SHAP}}) &= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\left[\sum_{k=1}^{d}\left(h_k(\boldsymbol{x}) - \sum_{u\subseteq[d]:k\in u}\frac{h_u(\boldsymbol{x})}{|u|}\right)^2\right] \\
&= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\left[\sum_{k=1}^{d}\left(\sum_{u\subseteq[d]:k\in u,|u|\geq 2}\frac{h_u(\boldsymbol{x})}{|u|}\right)^2\right] \\
&= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\left[\sum_{k=1}^{d}\sum_{\substack{u,v\subseteq[d]\\ k\in u,k\in v\\ |u|\geq 2,|v|\geq 2}}\frac{h_u(\boldsymbol{x})\,h_v(\boldsymbol{x})}{|u|\,|v|}\right] \\
&= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\left[\sum_{u,v\subseteq[d]:|u|\geq 2,|v|\geq 2}\frac{|u\cap v|}{|u|\,|v|}\,h_u(\boldsymbol{x})\,h_v(\boldsymbol{x})\right].
\end{aligned}$$

The corresponding interaction penalization is $w(u) = a(u,u) = \frac{1}{|u|}$. Now comparing local PDP and PFI, we get

$$\begin{aligned}
D(\boldsymbol{\phi}^{\mathrm{PDP}}, \boldsymbol{\phi}^{\mathrm{PFI}}) &= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\left[\sum_{k=1}^{d}\left(h_k(\boldsymbol{x}) - \sum_{u\subseteq[d]:k\in u}h_u(\boldsymbol{x})\right)^2\right] \\
&= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\left[\sum_{k=1}^{d}\left(\sum_{u\subseteq[d]:k\in u,|u|\geq 2}h_u(\boldsymbol{x})\right)^2\right] \\
&= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\left[\sum_{k=1}^{d}\sum_{\substack{u,v\subseteq[d]\\ k\in u,k\in v\\ |u|\geq 2,|v|\geq 2}}h_u(\boldsymbol{x})\,h_v(\boldsymbol{x})\right] \\
&= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\left[\sum_{u,v\subseteq[d]:|u|\geq 2,|v|\geq 2}|u\cap v|\,h_u(\boldsymbol{x})\,h_v(\boldsymbol{x})\right].
\end{aligned}$$

The corresponding interaction penalization is $w(u) = a(u,u) = |u|$. The disagreement be-

tween local SHAP and PFI yields

$$
\begin{aligned}
D(\boldsymbol{\phi}^{\mathrm{SHAP}}, \boldsymbol{\phi}^{\mathrm{PFI}}) &= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\left[\sum_{k=1}^{d}\left(\sum_{u\subseteq[d]:k\in u}\frac{h_u(\boldsymbol{x})}{|u|} - h_u(\boldsymbol{x})\right)^2\right] \\
&= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\left[\sum_{k=1}^{d}\left(\sum_{u\subseteq[d]:k\in u,|u|\geq 2}\frac{(1-|u|)\,h_u(\boldsymbol{x})}{|u|}\right)^2\right] \\
&= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\left[\sum_{k=1}^{d}\sum_{\substack{u,v\subseteq[d]\\ k\in u,k\in v\\ |u|\geq 2,|v|\geq 2}}\frac{(1-|u|)\,(1-|v|)\,h_u(\boldsymbol{x})\,h_v(\boldsymbol{x})}{|u|\,|v|}\right] \\
&= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\left[\sum_{u,v\subseteq[d]:|u|\geq 2,|v|\geq 2}\frac{|u\cap v|\,(|u|-1)\,(|v|-1)}{|u|\,|v|}\,h_u(\boldsymbol{x})\,h_v(\boldsymbol{x})\right].
\end{aligned}
$$

The corresponding interaction penalization is $w(u) = a(u,u) = (|u|-1)^2/|u|$.

The L_2 Cost of Exclusion (CoE) is also a LoA

$$
\begin{aligned}
L_h^{\mathrm{CoE}}(\mathcal{B}) &:= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\left[\left(h(\boldsymbol{x}) - \sum_{u\subseteq[d]:|u|\leq 1} h_u(\boldsymbol{x})\right)^2\right] \\
&= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\left[\left(\sum_{u\subseteq[d]:|u|\geq 2} h_u(\boldsymbol{x})\right)^2\right] \\
&= \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathcal{B}}\left[\sum_{\substack{u,v\subseteq[d]\\ |u|\geq 2,|v|\geq 2}} h_u(\boldsymbol{x})\,h_v(\boldsymbol{x})\right].
\end{aligned}
$$

The corresponding interaction penalization is $w(u) = a(u,u) = 1$. □

Figure C.1 presents the various penalization functions $w(u)$ implicit to each LoA when features are independent. The weights w are normalized so that $w(u) = 1$ when $|u| = 2$. Note that any LoA that involves a disagreement with the PFI explainer will penalize the higher-order interactions to a greater extent. This is because the PFI counts the interaction h_u several times. In opposition, the disagreements between PDP and SHAP put more weight on low-order interactions.

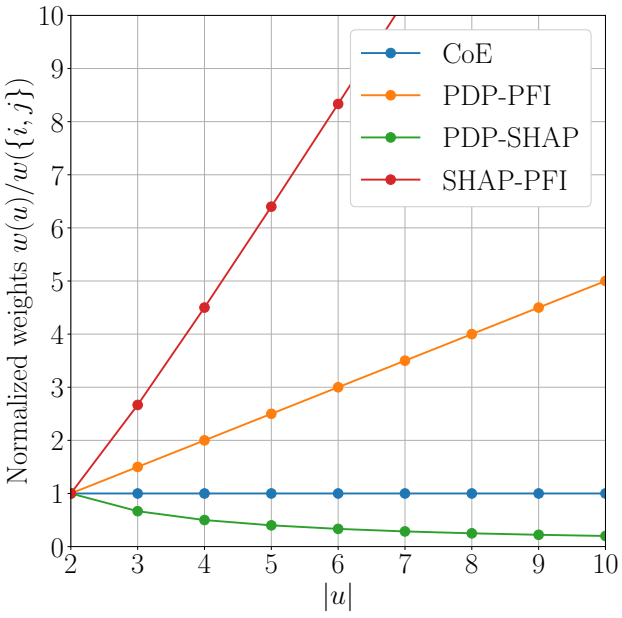


Figure C.1 How various LoA penalize interaction orders differently.

Theorem C.1.2 (**Theorem 6.2.2**). *GADGET-PDP is a LoA.*

Proof. The function $L_h^{\text{GADGET-PDP}}(\mathcal{B})$ (cf. Equation 6.11) respects the three properties of Definition 6.2.1.

Property 1 When the model is additive, all ICE curves of a given feature are parallel, which implies that the GAGDET-PDP loss is zero.

Property 2 We first express the centered ICE curves in the Interventional Decomposition with $\mathcal{B}_{\text{ind}}$. Our derivation employs the following "annihilation" property [Kuo et al., 2010]

$$v \cap u \neq \emptyset \Rightarrow \mathop{\mathbb{E}}_{\boldsymbol{x}_v \sim \mathcal{B}_{\text{ind},v}} \left[h_{u,\mathcal{B}_{\text{ind}}}(\boldsymbol{x}) \right] = 0. \tag{C.6}$$

We have

$$\begin{aligned}
\phi_k^{\text{ICE-c}}(h, x_k, \boldsymbol{z}_{-k}) &:= h(x_k, \boldsymbol{z}_{-k}) - \mathop{\mathbb{E}}_{x_k \sim \mathcal{B}_{\text{ind},k}} \Big[h(x_k, \boldsymbol{z}_{-k}) \Big] \\
&= \sum_{u \subseteq [d]} h_{u,\mathcal{B}_{\text{ind}}}(x_k, \boldsymbol{z}_{-k}) - \mathop{\mathbb{E}}_{x_k \sim \mathcal{B}_{\text{ind},k}} \left[\sum_{u \subseteq [d]} h_{u,\mathcal{B}_{\text{ind}}}(x_k, \boldsymbol{z}_{-k}) \right] \\
&= \sum_{u \subseteq [d]} h_{u,\mathcal{B}_{\text{ind}}}(x_k, \boldsymbol{z}_{-k}) - \sum_{u \subseteq [d]} \mathop{\mathbb{E}}_{x_k \sim \mathcal{B}_{\text{ind},k}} [h_{u,\mathcal{B}_{\text{ind}}}(x_k, \boldsymbol{z}_{-k})] \\
&= \sum_{u \subseteq [d]} h_{u,\mathcal{B}_{\text{ind}}}(x_k, \boldsymbol{z}_{-k}) - \sum_{u \subseteq [d] \setminus \{k\}} \mathop{\mathbb{E}}_{x_k \sim \mathcal{B}_{\text{ind},k}} [h_{u,\mathcal{B}_{\text{ind}}}(x_k, \boldsymbol{z}_{-k})] \\
&\qquad\qquad \text{(Annihilation Property)} \\
&= \sum_{u \subseteq [d]} h_{u,\mathcal{B}_{\text{ind}}}(x_k, \boldsymbol{z}_{-k}) - \sum_{u \subseteq [d] \setminus \{k\}} h_{u,\mathcal{B}_{\text{ind}}}(x_k, \boldsymbol{z}_{-k}) \\
&= \sum_{u \subseteq [d] : k \in u} h_{u,\mathcal{B}_{\text{ind}}}(x_k, \boldsymbol{z}_{-k}).
\end{aligned}$$

The centered-PDP can also be expressed in terms of the FD

$$\begin{aligned}
\phi_k^{\text{PDP-c}}(h, x_k) &:= \mathop{\mathbb{E}}_{\boldsymbol{z}_{-k} \sim \mathcal{B}_{\text{ind},-k}} \Big[\phi_k^{\text{ICE-c}}(h, x_k, \boldsymbol{z}_{-k}) \Big] \\
&= \mathop{\mathbb{E}}_{\boldsymbol{z}_{-k} \sim \mathcal{B}_{\text{ind},-k}} \left[\sum_{u \subseteq [d] : k \in u} h_{u,\mathcal{B}_{\text{ind}}}(x_k, \boldsymbol{z}_{-k}) \right] \\
&= \sum_{u \subseteq [d] : k \in u} \mathop{\mathbb{E}}_{\boldsymbol{z}_{-k} \sim \mathcal{B}_{\text{ind},-k}} [h_{u,\mathcal{B}_{\text{ind}}}(x_k, \boldsymbol{z}_{-k})] \\
&= \mathop{\mathbb{E}}_{\boldsymbol{z}_{-k} \sim \mathcal{B}_{\text{ind},-k}} [h_{k,\mathcal{B}_{\text{ind}}}(x_k, \boldsymbol{z}_{-k})] \qquad \text{(Annihilation property)} \\
&= h_{k,\mathcal{B}_{\text{ind}}}(x_k, \boldsymbol{z}_{-k}).
\end{aligned}$$

Thus the GADGET-PDP loss simplifies

$$
\begin{aligned}
L_h^{\text{GADGET-PDP}}(\mathcal{B}) &:= \sum_{k=1}^{d} \mathop{\mathbb{E}}_{\substack{x_k \sim \mathcal{B}_{\text{ind},k} \\ \boldsymbol{z}_{-k} \sim \mathcal{B}_{\text{ind},-k}}} \left[\left(\phi_k^{\text{ICE-c}}(h, x_k, \boldsymbol{z}_{-k}) - \phi_k^{\text{PDP-c}}(h, x_k) \right)^2 \right] \\
&= \sum_{k=1}^{d} \mathop{\mathbb{E}}_{\substack{x_k \sim \mathcal{B}_{\text{ind},k} \\ \boldsymbol{z}_{-k} \sim \mathcal{B}_{\text{ind},-k}}} \left[\left(\sum_{u \subseteq [d]: k \in u} h_{u,\mathcal{B}_{\text{ind}}}(x_k, \boldsymbol{z}_{-k}) - h_{k,\mathcal{B}_{\text{ind}}}(x_k, \boldsymbol{z}_{-k}) \right)^2 \right] \\
&= \sum_{k=1}^{d} \mathop{\mathbb{E}}_{\substack{x_k \sim \mathcal{B}_{\text{ind},k} \\ \boldsymbol{z}_{-k} \sim \mathcal{B}_{\text{ind},-k}}} \left[\left(\sum_{\substack{u \subseteq [d]: k \in u \\ |u| \geq 2}} h_{u,\mathcal{B}_{\text{ind}}}(x_k, \boldsymbol{z}_{-k}) \right)^2 \right] \\
&= \sum_{k=1}^{d} \mathop{\mathbb{E}}_{\boldsymbol{x} \sim \mathcal{B}_{\text{ind}}} \left[\left(\sum_{\substack{u \subseteq [d]: k \in u \\ |u| \geq 2}} h_{u,\mathcal{B}_{\text{ind}}}(\boldsymbol{x}) \right)^2 \right] \qquad \text{(By feature independence)} \\
&= \sum_{k=1}^{d} \sum_{\substack{u \subseteq [d]: k \in u \\ |u| \geq 2}} \sigma_u^2 = \sum_{\substack{u \subseteq [d] \\ |u| \geq 2}} |u| \sigma_u^2.
\end{aligned}
$$

(By feature independence and Equation C.4)

Thus, when features are independent, GADGET-PDP penalizes $|u|$-way interactions with a weight $w(u) = |u|$. Like the LoA $D(\boldsymbol{\phi}^{\text{PDP}}, \boldsymbol{\phi}^{\text{PFI}})$, higher order interactions are penalized more than low order ones.

Property 3 We assume that h is additive in feature j and must prove that $L_h^{\text{GADGET-PDP}}(\mathcal{B}_j \times \mathcal{B}_{-j}) = L_h^{\text{GADGET-PDP}}(\mathcal{B}'_j \times \mathcal{B}_{-j})$ for any two distributions $\mathcal{B}_j$ and $\mathcal{B}'_j$ on feature j. We will study the ICE curves of feature j and features $k \neq j$ separately.

Feature j : Because the model is additive in j, the ICE curves and the PDP of feature j will be parallel. Hence the contribution of feature j to the GADGET-PDP loss is null :

$$\mathop{\mathbb{E}}_{\substack{x_j\sim\mathcal{B}_j\\ \boldsymbol{z}_{-j}\sim\mathcal{B}_{-j}}}\left[\left(\phi_j^{\text{ICE-c}}(h,x_j,\boldsymbol{z}_{-j})-\phi_j^{\text{PDP-c}}(h,x_j)\right)^2\right]=0. \tag{C.7}$$

Feature $k\neq j$: Since the Interventional Decomposition is minimal, there will not be any interaction terms h_u with $j\in u$. In that case, the Centered ICE curves of feature $k\neq j$ are

$$\begin{aligned}
\phi_k^{\text{ICE-c}}(h,x_k,\boldsymbol{z}_{-k}) &:= h(x_k,\boldsymbol{z}_{-k})-\mathop{\mathbb{E}}_{x_k\sim\mathcal{B}_k}[h(x_k,\boldsymbol{z}_{-k})]\\
&=\sum_{u\subseteq[d]}h_{u,\mathcal{B}}(x_k,\boldsymbol{z}_{-k})-\mathop{\mathbb{E}}_{x_k\sim\mathcal{B}_k}\left[h_{u,\mathcal{B}}(x_k,\boldsymbol{z}_{-k})\right]\\
&=h_{j,\mathcal{B}}(x_k,\boldsymbol{z}_{-k})-\mathop{\mathbb{E}}_{x_k\sim\mathcal{B}_k}\left[h_{j,\mathcal{B}}(x_k,\boldsymbol{z}_{-k})\right]+\sum_{u\subseteq[d]\setminus\{j\}}h_{u,\mathcal{B}}(x_k,\boldsymbol{z}_{-k})-\mathop{\mathbb{E}}_{x_k\sim\mathcal{B}_k}\left[h_{u,\mathcal{B}}(x_k,\boldsymbol{z}_{-k})\right]\\
&=h_{j,\mathcal{B}}(z_j)-h_{j,\mathcal{B}}(z_j)+\sum_{u\subseteq[d]\setminus\{j\}}h_{u,\mathcal{B}}(x_k,\boldsymbol{z}_{-k})-\mathop{\mathbb{E}}_{x_k\sim\mathcal{B}_k}\left[h_{u,\mathcal{B}}(x_k,\boldsymbol{z}_{-k})\right]\\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\text{(Since } k\neq j\text{)}\\
&=\sum_{u\subseteq[d]\setminus\{j\}}h_{u,\mathcal{B}}(x_k,\boldsymbol{z}_{-k})-\mathop{\mathbb{E}}_{x_k\sim\mathcal{B}_k}\left[h_{u,\mathcal{B}}(x_k,\boldsymbol{z}_{-k})\right].
\end{aligned}$$

We have just proven that

$$\phi_k^{\text{ICE-c}}(h,x_k,\boldsymbol{z}_{-k})\ \text{ does } \textbf{not} \text{ depend on } z_j \tag{C.8}$$

since we sum over all interactions h_u that exclude feature j. Finally, the total GADGET-PDP loss is

$$\begin{aligned}
L_h^{\text{GADGET-PDP}}(\mathcal{B}) &:= \sum_{k\in[d]}\mathop{\mathbb{E}}_{x_k\sim\mathcal{B}_k}\left[\ \mathbb{V}_{\boldsymbol{z}_{-k}\sim\mathcal{B}_{-k}}\left[\phi_k^{\text{ICE-c}}(h,x_k,\boldsymbol{z}_{-k})\right]\ \right]\\
&=\sum_{k\in[d]\setminus\{j\}}\mathop{\mathbb{E}}_{x_k\sim\mathcal{B}_k}\left[\ \mathbb{V}_{\boldsymbol{z}_{-k}\sim\mathcal{B}_{-k}}\left[\phi_k^{\text{ICE-c}}(h,x_k,\boldsymbol{z}_{-k})\right]\ \right] &&\text{(Equation C.7)}\\
&=\sum_{k\in[d]\setminus\{j\}}\mathop{\mathbb{E}}_{x_k\sim\mathcal{B}_k}\left[\ \mathbb{V}_{\boldsymbol{z}_{-\{j,k\}}\sim\mathcal{B}_{-\{j,k\}}}\left[\phi_k^{\text{ICE-c}}(h,x_k,\boldsymbol{z}_{-k})\right]\ \right]. &&\text{(Equation C.8)}
\end{aligned}$$

This expression does not involve $\mathcal{B}_j$ so it is independent of how feature j is distributed. The direct implication is that $L_h^{\text{GADGET-PDP}}(\mathcal{B}_j\times\mathcal{B}_{-j})=L_h^{\text{GADGET-PDP}}(\mathcal{B}'_j\times\mathcal{B}_{-j})$. □

APPENDIX D SUPPLEMENTARY OF FOOL SHAP

D.1 Proofs

D.1.1 Statistical Result

Proposition D.1.1 (**Proposition 7.2.1**). *Let S_0' be **fixed**, and let $\xrightarrow{p}$ represent convergence in probability as the size M of the set $S_1' \sim \mathcal{B}_{\boldsymbol{\omega}}^M$ increases, we have*

$$\begin{aligned}\widehat{\Phi}_s^{Fair}(h, S_0', S_1') &\xrightarrow{p} \sum_{\boldsymbol{z}^{(j)} \in D_1} \omega_j \left(\frac{1}{M} \sum_{\boldsymbol{x}^{(i)} \in S_0'} \phi_s^{SHAP\text{-}int}(h, \boldsymbol{x}^{(i)}, \boldsymbol{z}^{(j)}) \right) \\ &:= \sum_{\boldsymbol{z}^{(j)} \in D_1} \omega_j a_j \end{aligned} \tag{D.1}$$

Proof. Since S_0' is assumed to be fixed, the only random variable in $\widehat{\Phi}_s^{\text{Fair}}(f, S_0', S_1')$ is the subset $S_1' \sim \mathcal{B}_{\boldsymbol{\omega}}^M$ of background instances. Therefore, we get

$$\begin{aligned}\widehat{\Phi}_s^{\text{Fair}}(f, S_0', S_1') &:= \frac{1}{M^2} \sum_{\boldsymbol{x}^{(i)} \in S_0} \sum_{\boldsymbol{z}^{(j)} \in S_1} \phi_s^{\text{SHAP-int}}(h, \boldsymbol{x}^{(i)}, \boldsymbol{z}^{(j)}) \\ &= \frac{1}{M} \sum_{\boldsymbol{z}^{(j)} \in S_1'} \left(\frac{1}{M} \sum_{\boldsymbol{x}^{(i)} \in S_0'} \phi_s^{\text{SHAP-int}}(h, \boldsymbol{x}^{(i)}, \boldsymbol{z}^{(j)}) \right) \\ &= \frac{1}{M} \sum_{\boldsymbol{z}^{(j)} \in S_1'} a(\boldsymbol{z}^{(j)}), \end{aligned} \tag{D.2}$$

where we have introduced the function

$$a(\boldsymbol{z}) := \frac{1}{M} \sum_{\boldsymbol{x}^{(i)} \in S_0'} \phi_s^{\text{SHAP-int}}(h, \boldsymbol{x}^{(i)}, \boldsymbol{z}). \tag{D.3}$$

By the weak law of large number, the following holds as M goes to infinity

$$\frac{1}{M} \sum_{\boldsymbol{z}^{(j)} \in S_1'} a(\boldsymbol{z}^{(j)}) \xrightarrow{p} \mathop{\mathbb{E}}_{\boldsymbol{z} \sim \mathcal{B}_{\boldsymbol{\omega}}} [a(\boldsymbol{z})] = \sum_{j=1}^{N_1} \omega_j a(\boldsymbol{z}^{(j)}). \tag{D.4}$$

Relabelling $a(\boldsymbol{z}^{(j)}) \equiv a_j$ conclude the proof. □

D.1.2 Optimization

Technical Lemmas We provide some technical lemmas that will be essential when proving Theorem D.1.1. These are presented for completeness and are not intended as contributions by the authors. Let us first write the formal definition of the minimum of a function.

Definition D.1.1 (Minimum)**.** *Given some function $f : D \to \mathbb{R}$, the minimum of f over D (denoted $f^\star$) is defined as follows:*

$$f^\star = \min_{x \in D} f(x) \iff \exists x^\star \in D \text{ s.t. } f^\star = f(x^\star) \leq f(x) \quad \forall x \in D.$$

Basically, the notion of minimum coincides with the infimum $\inf f(D)$ (highest lower bound) when this lower bound is attained for some $x^\star \in D$. By the Extreme Values Theorem, the minimum always exists when D is compact and f is continuous. For the rest of this appendix, we shall only study optimization problems where points on the domain set $D = \{(x, y) : x \in \mathcal{X}, y \in \mathcal{Y}_x \subset \mathcal{Y}\}$ can be *selected* by the following procedure

1. Choose some $x \in \mathcal{X}$
2. Given the selected x, choose some $y \in \mathcal{Y}_x \subset \mathcal{Y}$, where the set $\mathcal{Y}_x$ is non-empty and depends on the value of x.

When optimizing functions over these domains, one can optimize in two steps, as highlighted in the following lemma.

Lemma D.1.1. *Given a compact domain D of the form described above and a continuous objective function $f : D \to \mathbb{R}$, the minimum $f^\star$ is attained for some $(x^\star, y^\star)$ and the following holds*

$$\min_{(x,y) \in D} f(x, y) = \min_{x \in \mathcal{X}} \min_{y \in \mathcal{Y}_x} f(x, y).$$

Proof. Let $\widetilde{f}(x) := \inf_{y \in \mathcal{Y}_x} f(x, y)$, which is a well-defined function on $\mathcal{X}$. We can then take its infimum $f^\star = \inf_{x \in \mathcal{X}} \widetilde{f}(x)$. But is $f^\star$ an infimum of $f(D)$? By the definition of infimum

$$\begin{aligned} f^\star &\leq \widetilde{f}(x) && \forall\, x \in \mathcal{X} \\ &= \inf_{y \in \mathcal{Y}_x} f(x, y) \\ &\leq f(x, y) && \forall\, y \in \mathcal{Y}_x, \end{aligned}$$

so that $f^\star$ is a lower bound of $f(D)$. In fact, it is the highest lower bound possible so

$$\inf_{(x,y)\in D} f(x,y) = \inf_{x\in\mathcal{X}} \inf_{y\in\mathcal{Y}_x} f(x,y). \tag{D.5}$$

By the Extreme Value Theorem, since D is compact and f is continuous, there exists $(x^\star, y^\star) \in D$ s.t. $f^\star = \inf_{(x,y)\in D} f(x,y) = \max_{(x,y)\in D} f(x,y) = f(x^\star, y^\star)$. Since the infimum is attained on the left-hand-side of Equation D.5, then it must also be attained on the right-hand-side and therefore we can replace all inf with min in Equation D.5, leading to the desired result. □

Lemma D.1.2. *Given a compact domain D of the form described above and two continuous functions $h : \mathcal{X} \to \mathbb{R}$ and $g : \mathcal{Y} \to \mathbb{R}$, then*

$$\min_{(x,y)\in D} \Big(h(x) + g(y) \Big) = \min_{x\in\mathcal{X}} \Big(h(x) + \min_{y\in\mathcal{Y}_x} g(y) \Big)$$

Proof. Applying Lemma D.1.1 with the function $f(x,y) := h(x)+g(y)$ proves the Lemma. □

Minimum Cost Flows Let $\mathbb{G} = (\mathcal{V}, \mathcal{E})$ be a graph with vertices $v \in \mathcal{V}$ with directed edges $e \in \mathcal{E} \subset \mathcal{V} \times \mathcal{V}$, $c : \mathcal{E} \to \mathbb{R}^+$ be a capacity and $a : \mathcal{E} \to \mathbb{R}$ be a cost. Moreover, let $s, t \in \mathcal{E}$ be two special vertices called the source and the sink respectively, and $d \in \mathbb{R}^+$ be a total flow. The Minimum-Cost Flow (MCF) problem of $\mathbb{G}$ consists of finding the flow function $f : \mathcal{E} \to \mathbb{R}^+$ that minimizes the total cost

$$\begin{aligned} \min_f \quad & \sum_{e\in\mathcal{E}} a(e) f(e) \\ \text{s.t.} \quad & 0 \le f(e) \le c(e) \ \forall e \in \mathcal{E} \\ & \sum_{e\in u^+} f(e) - \sum_{e\in u^-} f(e) = \begin{cases} 0 & u \in \mathcal{V} \setminus \{s,t\} \\ d & u = s \\ -d & u = t \end{cases} \end{aligned} \tag{D.6}$$

where $u^+ := \{(u,v) \in \mathcal{E}\}$ and $u^- := \{(v,u) \in \mathcal{E}\}$ are the outgoing and incoming edges from u. The terminology of *flow* arises from the constraint that, for vertices that are not the source nor the sink, the outgoing flow must equal the incoming one, which is reminiscent of conservation laws in fluidic. We shall refer to $f((u,v))$ as the flow from u to v.

Now that we have introduced minimum cost flows, let us specify the graph that will be

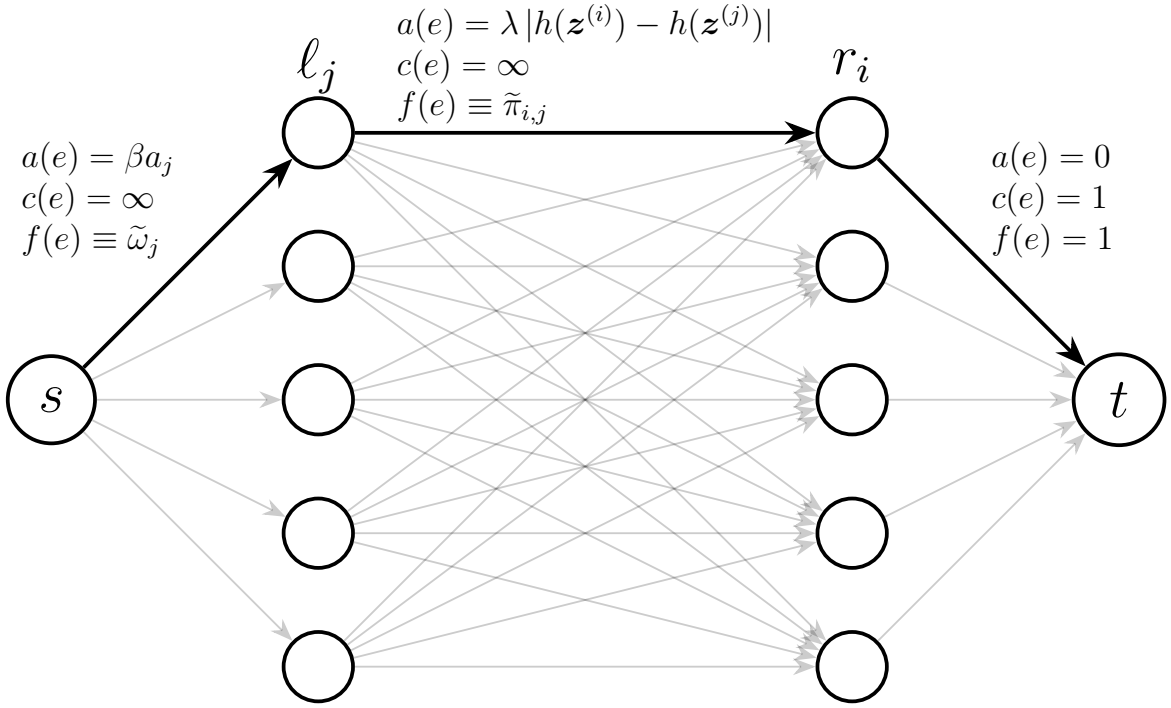


Figure D.1 Graph $\mathbb{G}$ on which we solve the MCF. Note that the total amount of flow is $d = N_1$ and there are N_1 left and right nodes ℓ_j, r_i.

employed in FoolSHAP, see Figure D.1. We label the flow going from the sink s to one of the left vertices as $\widetilde{\omega}_j \equiv \omega_j \times N_1$, and the flow going from ℓ_j to r_i as $\widetilde{\pi}_{i,j} \equiv \pi_{i,j} \times N_1$. The required flow is fixed at $d = N_1$.

Theorem D.1.1. *Solving the MCF of Figure D.1 leads to a solution of the linear program in Algorithm 6.*

Proof. We begin by showing that the flow conservation constraints in the MCF imply that π is a coupling measure (*i.e.* $\pi \in \Delta(\mathcal{B}, \mathcal{B}_{\boldsymbol{\omega}})$), and $\boldsymbol{\omega}$ is constrained to the probability simplex $\Delta(N_1)$. Applying the conservation law on the left-side of the graph leads to the conclusion that the flows entering vertices ℓ_j must sum up to N_1

$$\sum_{j=1}^{N_1} \widetilde{\omega}_j = N_1.$$

This implies that $\boldsymbol{\omega}$ is must be part of the probability simplex. By conservation, the amount of flow that leaves a specific vertex ℓ_j must also be $\widetilde{\omega}_j$, hence

$$\sum_i \widetilde{\pi}_{ij} = \widetilde{\omega}_j.$$

For any edge outgoing from r_i to the sink t, the flow must be exactly 1. This is because we have N_1 edges with capacity $c(e) = 1$ going into the sink and the sink must receive an incoming flow of N_1. As a consequence of the conservation law on a specific vertex r_i, the amount of flow that goes into each r_i is also 1

$$\sum_j \widetilde{\pi}_{ij} = 1.$$

Putting everything together, from the conservation laws on $\mathbb{G}$, we have that $\boldsymbol{\omega} \in \Delta(N_1)$, and $\pi \in \Delta(\mathcal{B}, \mathcal{B}_{\boldsymbol{\omega}})$. Now, to make the parallel between the MCF and Algorithm 6, we must use Lemma D.1.2. Note that $\boldsymbol{\omega}$ is restricted to the probability simplex, while π is restricted to be a coupling measure. Importantly, the set of all possible coupling measures $\Delta(\mathcal{B}, \mathcal{B}_{\boldsymbol{\omega}})$ is different for each $\boldsymbol{\omega}$ because $\mathcal{B}_{\boldsymbol{\omega}}$ depends on $\boldsymbol{\omega}$. Hence, the domain has the same structure as the ones tackled in Lemma D.1.2 (where $x \in \mathcal{X}$ becomes $\boldsymbol{\omega} \in \Delta(N_1)$) and $y \in \mathcal{Y}_x$ becomes $\pi \in \Delta(\mathcal{B}, \mathcal{B}_{\boldsymbol{\omega}})$). Now importantly, the set of possible $\boldsymbol{\omega}$ and π are bounded simplexes and so are compact. Also, the objective function of the MCF is linear, thus continuous. We can therefore apply Lemma D.1.2 to the MCF.

$$\begin{aligned}
\min_f \sum_{e\in\mathcal{E}} f(e)a(e) &= \min_{\widetilde{\omega},\widetilde{\pi}} \sum_{j=1}^{N_1} \beta\widetilde{\omega}_j a_j + \lambda \sum_{i,j} \widetilde{\pi}_{ij} |h(\boldsymbol{z}^{(i)}) - h(\boldsymbol{z}^{(j)})| \\
&= \min_{\widetilde{\omega},\widetilde{\pi}} \frac{N_1}{N_1} \left(\beta \sum_{j=1}^{N_1} \widetilde{\omega}_j a_j + \lambda \sum_{i,j} \widetilde{\pi}_{ij} |h(\boldsymbol{z}^{(i)}) - h(\boldsymbol{z}^{(j)})| \right) \\
&= N_1 \min_{\widetilde{\omega},\widetilde{\pi}} \left(\beta \sum_{j=1}^{N_1} \frac{\widetilde{\omega}_j}{N_1} a_j + \lambda \sum_{i,j} \frac{\widetilde{\pi}_{ij}}{N_1} |h(\boldsymbol{z}^{(i)}) - h(\boldsymbol{z}^{(j)})| \right) \\
&= N_1 \min_{\omega\in\Delta(N_1),\pi\in\Delta(\mathcal{B},\mathcal{B}_{\boldsymbol{\omega}})} \left(\beta \sum_{j=1}^{N_1} \omega_j a_j + \lambda \sum_{i,j} \pi_{i,j} |h(\boldsymbol{z}^{(i)}) - h(\boldsymbol{z}^{(j)})| \right) \\
&= N_1 \min_{\omega\in\Delta(N_1),\pi\in\Delta(\mathcal{B},\mathcal{B}_{\boldsymbol{\omega}})} \Big(h(\omega) + g(\pi) \Big) \\
&= N_1 \min_{\omega\in\Delta(N_1)} \Big(h(\omega) + \min_{\pi\in\Delta(\mathcal{B},\mathcal{B}')} g(\pi) \Big) \qquad \text{(cf. Lemma D.1.2)} \\
&= N_1 \min_{\omega\in\Delta(N_1)} \left(\beta \sum_{j=1}^{N_1} \omega_j a_j + \lambda \min_{\pi\in\Delta(\mathcal{B},\mathcal{B}_{\boldsymbol{\omega}})} \sum_{i,j} \pi_{i,j} |h(\boldsymbol{z}^{(i)}) - h(\boldsymbol{z}^{(j)})| \right) \\
&= N_1 \min_{\omega\in\Delta(N_1)} \Big(\beta \sum_{j=1}^{N_1} \omega_j a_j + \lambda\, \mathcal{W}(h(\mathcal{B}), h(\mathcal{B}_{\boldsymbol{\omega}})) \Big)
\end{aligned}$$

which (up to a multiplicative constant N_1) is a solution of the linear program of Algorithm 6. □

D.2 Genetic Algorithm

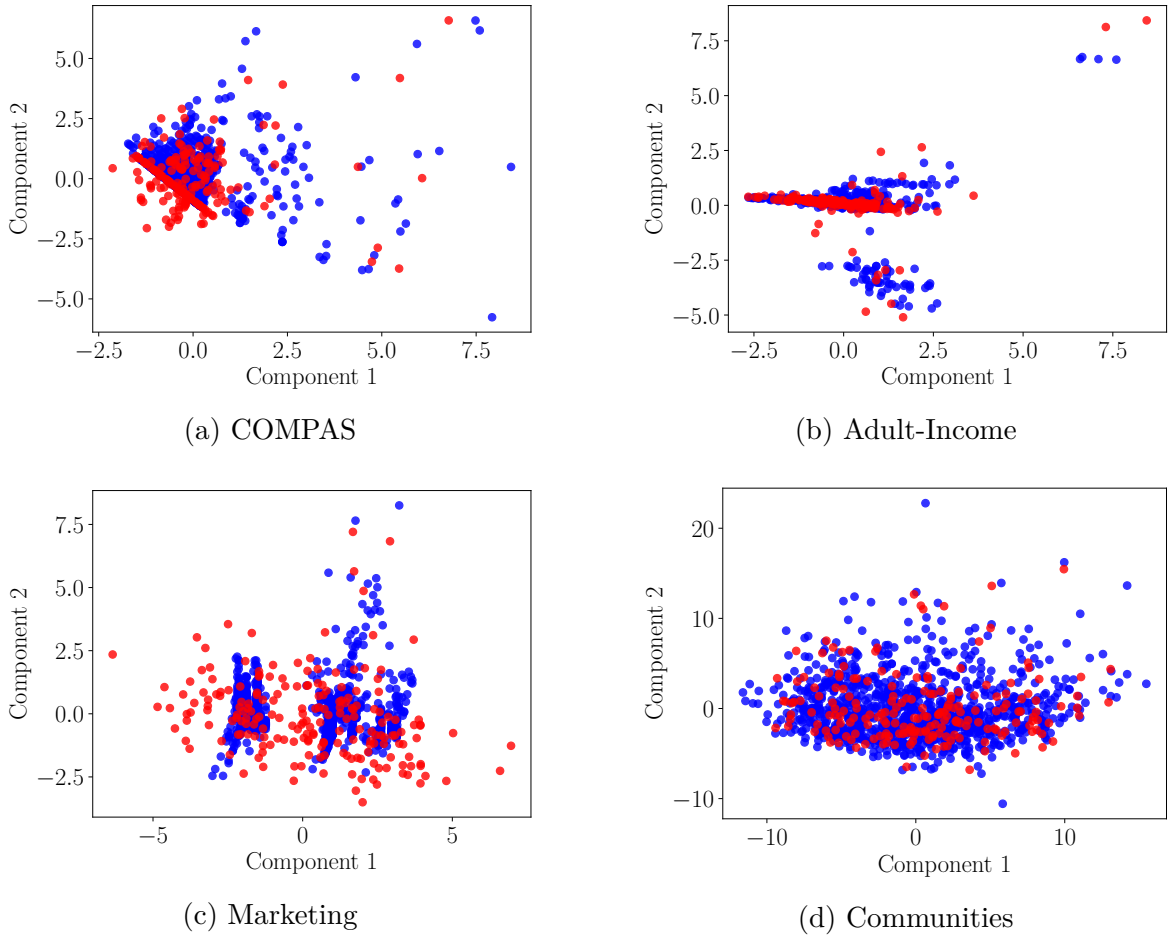
This section motivates the use of stealthily biased sampling to perturb Shapley Values in place of the method of [Baniecki and Biecek, 2022], which fools `SHAP` by perturbing the background dataset S_1' via a genetic algorithm. In said genetic algorithm, a population of P fake background datasets $\{S_1'^{(k)}\}_{k=1}^{P}$ evolves iteratively following three biological mechanisms

- **Cross-Over:** Two parents produce two children by switching some of their feature values.
- **Mutation:** Some individuals are perturbed with small Gaussian noise.
- **Selection:** The individuals $S_1'^{(k)}$ with the smallest amplitudes $|\widehat{\Phi}_s^{\text{Fair}}(h, S_0', S_1'^{(k)})|$ are selected for the next generation.

Although the use of a genetic algorithm makes the method of [Baniecki and Biecek, 2022] very versatile, its main drawback is that there is no constraint on the similarity between the perturbed background and the original one. Moreover, the mutation and cross-over operations ignore the correlations between features and hence the perturbed dataset can contain unrealistic instances. To highlight these issues, Figure D.2 presents the first two principal components of D_1 and S_1' for the XGB models used in Section 7.3. On COMPAS and Marketing especially, we see that the fake samples S_1' lie in regions outside the data manifold. For Adult-Income and Marketing, the fake data overlaps more with the original one, but this could be an artifact of only visualizing 2 dimensions.

For a more rigorous analysis of similarity between S_1' and D_1, we must study the detection rate of the audit detector. To this end, Figures D.3 and D.4, present the amplitude reduction and the detection rate after a given number of iterations of the genetic algorithm. These curves show the average and standard deviation across the 5 train/test splits employed in Section 7.3. Moreover, window 20 convolutions were used to smooth the curves and make them more readable. On the Marketing and Communities datasets, we see that for both XGB and Random Forests models, the detector is quickly able to assert that the data was manipulated. We suspect the genetic algorithm cannot fool the detector on these two datasets because they contain numerous features (Marketing has 20, Communities has 98). Such numerous features could make it harder to perturb samples while staying close to the data manifold. Since the model behavior is unpredictable outside the data manifold, it is impossible for the genetic algorithm to guarantee that the CDF of $h(S_1')$ will be close to the CDF of $h(D_1)$. For adult-income, the detection rate appears to be lower but still, the largest reductions in amplitude of the sensitive feature were about 10%, even after 2.5 hours of run-time.

Contrary to the genetic algorithm, our method FoolSHAP addresses both constraints of making the fake data realistic and keeping it close to the original dataset. Indeed, the optimization objective to penalizes large Wasserstein distances between the true and fake data. Moreover, no unrealistic samples are generated. Rather, FoolSHAP applies non-uniform weights to pre-existing datum.



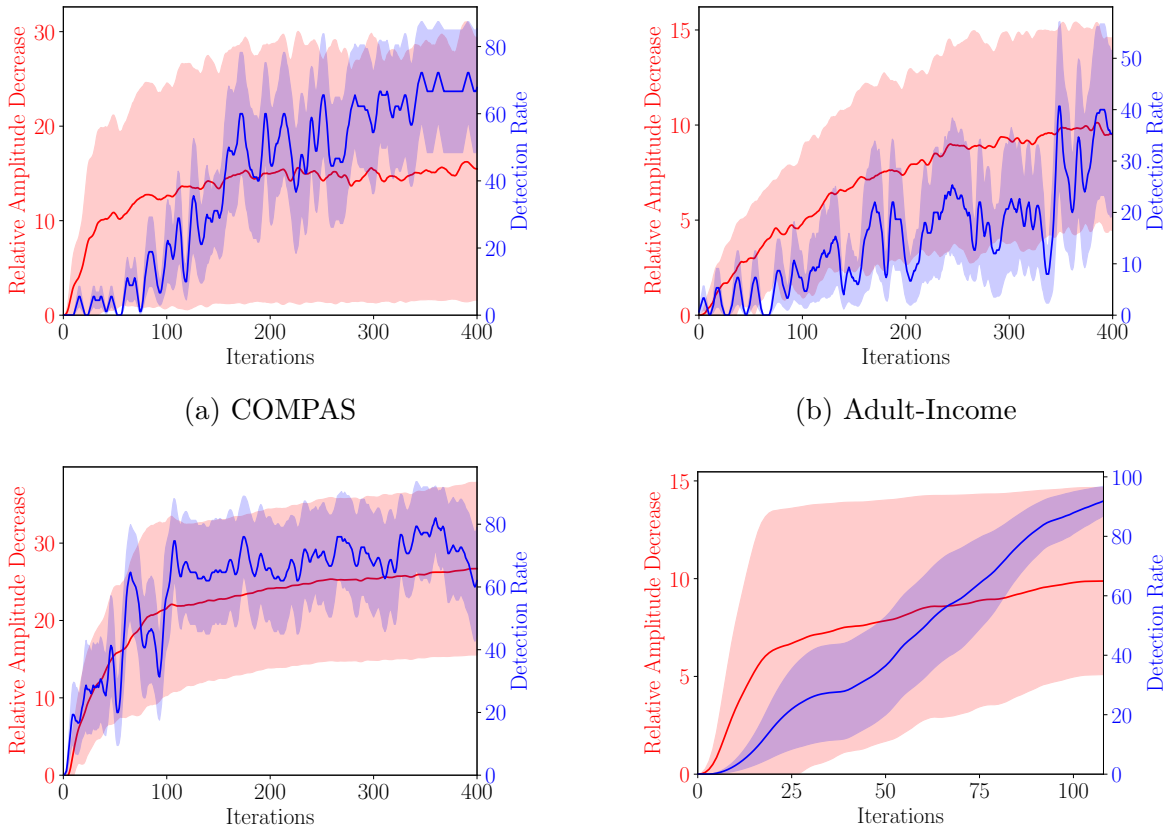
(a) COMPAS

(b) Adult-Income

(c) Marketing

(d) Communities

Figure D.2 First two principal components of D_1 (Blue) and S_1' (Red) returned by the genetic algorithm on XGB models.



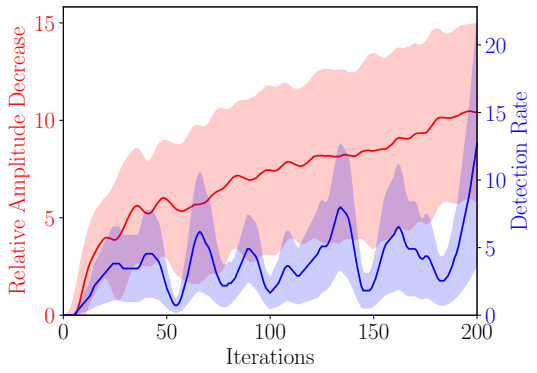
(a) COMPAS

(b) Adult-Income

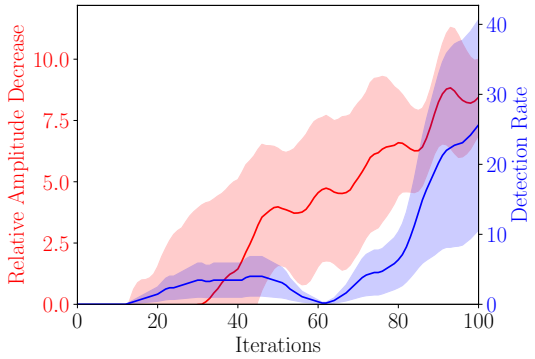
(c) Marketing

(d) Communities

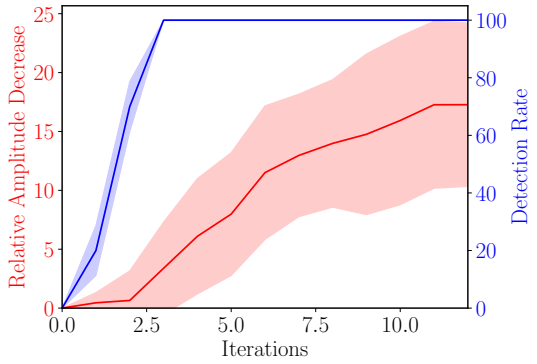
Figure D.3 Iterations of the genetic algorithm applied to 5 XGB models per dataset.



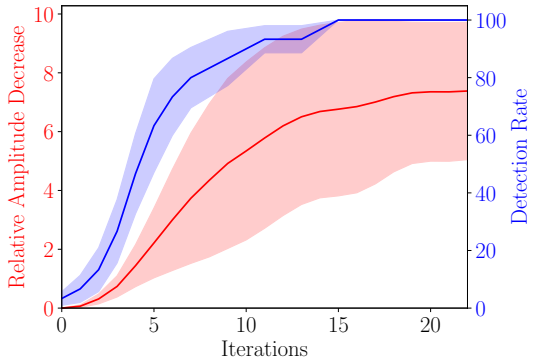
(a) COMPAS



(b) Adult-Income



(c) Marketing



(d) Communities

Figure D.4 Iterations of the genetic algorithm applied to 5 RF models per dataset.

APPENDIX E SUPPLEMENTARY ON UNDERSPECIFICATION

E.1 Proofs

E.1.1 Statistical Bounds

Proposition E.1.1 (**Proposition 8.2.2**). *Under the assumption that the data was generated by the optimal model $h^\star$ plus zero-mean Gaussian noise*

$$y = h^\star(\boldsymbol{x}) + \Delta, \quad \textit{where} \quad \Delta \sim \mathcal{N}(0, \sigma^2), \tag{E.1}$$

and using the squared loss $\ell(y', y) = (y' - y)^2$, we have that

$$\mathop{\mathbb{P}}_{S\sim\mathcal{D}^N}[\widehat{\mathcal{L}}_S(h^\star) > \epsilon_{max}] = 1 - F_{\chi^2_N}\left(\frac{N}{\sigma^2}\epsilon_{max}\right), \tag{E.2}$$

where $F_{\chi^2_N}$ is the CDF of a chi-2 random variable with N degrees of freedom.

Proof. Under the assumption that Equation E.1 is valid, we have that

$$\widehat{\mathcal{L}}_S(h^\star) = \frac{1}{N}\sum_{i=1}^{N}(h^\star(\boldsymbol{x}) - y^{(i)})^2 = \frac{1}{N}\sum_{i=1}^{N}(\Delta^{(i)})^2,$$

where each $\Delta^{(i)}$ is sampled iid from a $\mathcal{N}(0, \sigma^2)$ Gaussian. Now we have

$$\begin{aligned}
\mathop{\mathbb{P}}_{S\sim\mathcal{D}^N}[\widehat{\mathcal{L}}_S(h^\star) > \epsilon_{\max}] &= \mathop{\mathbb{P}}_{\boldsymbol{\Delta}\sim\mathcal{N}(0,\sigma^2)^N}\left[\frac{1}{N}\sum_{i=1}^{N}(\Delta^{(i)})^2 > \epsilon_{\max}\right] \\
&= \mathop{\mathbb{P}}_{\boldsymbol{\Delta}\sim\mathcal{N}(0,\sigma^2)^N}\left[\sum_{i=1}^{N}\left(\frac{\Delta^{(i)}}{\sigma}\right)^2 > \frac{N}{\sigma^2}\epsilon_{\max}\right] \\
&= \mathop{\mathbb{P}}_{\boldsymbol{\Delta}\sim\mathcal{N}(0,1)^N}\left[\sum_{i=1}^{N}(\Delta^{(i)})^2 > \frac{N}{\sigma^2}\epsilon_{\max}\right] \\
&= \mathop{\mathbb{P}}_{c\sim\chi^2_N}\left[c > \frac{N}{\sigma^2}\epsilon_{\max}\right] \\
&= 1 - F_{\chi^2_N}\left(\frac{N}{\sigma^2}\epsilon_{\max}\right).
\end{aligned} \tag{E.3}$$

□

Proposition E.1.2 (Proposition 8.2.3). *Let $\ell(\widehat{y}, y) = \mathbb{1}(\widehat{y} \neq y)$ be the 0–1 loss, $S \sim \mathcal{D}^N$ be a dataset, $h_{ref} \in \mathcal{H}$ be a reference model that is independent of S, and $h^\star$ be a best in-class hypothesis, for any $\epsilon' \in \mathbb{R}^+$, we have*

$$\mathop{\mathbb{P}}_{S\sim\mathcal{D}^N}[\widehat{\mathcal{L}}_S(h^\star) \geq \epsilon' + \widehat{\mathcal{L}}_S(h_{ref})] \leq \exp\left\{-\frac{N\epsilon'^2}{2}\right\}. \tag{E.4}$$

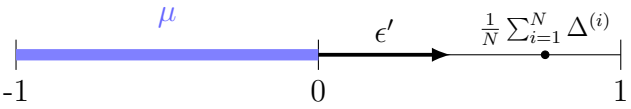
Proof. We assume that $\widehat{\mathcal{L}}_S(h^\star) \geq \epsilon' + \widehat{\mathcal{L}}_S(h_{\text{ref}})$ and show that this implies the occurrence of an unlikely event. We first have

$$\widehat{\mathcal{L}}_S(h^\star) - \widehat{\mathcal{L}}_S(h_{\text{ref}}) = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}[h^\star(\boldsymbol{x}^{(i)}) \neq y^{(i)}] - \mathbb{1}[h_{\text{ref}}(\boldsymbol{x}^{(i)}) \neq y^{(i)}] = \frac{1}{N}\sum_{i=1}^{N} \Delta^{(i)}, \tag{E.5}$$

where the N random variables $\Delta^{(i)} := \mathbb{1}[h^\star(\boldsymbol{x}^{(i)}) \neq y^{(i)}] - \mathbb{1}[h_{\text{ref}}(\boldsymbol{x}^{(i)}) \neq y^{(i)}]$ are iid, take values between -1 and 1, and have the expectancy

$$\mu = \mathop{\mathbb{E}}_{S\sim\mathcal{D}^N}[\Delta^{(i)}] = \mathop{\mathbb{E}}_{(\boldsymbol{x}^{(i)},y^{(i)})\sim\mathcal{D}}[\Delta^{(i)}] = \mathcal{L}_{\mathcal{D}}(h^\star) - \mathcal{L}_{\mathcal{D}}(h_{\text{ref}}). \tag{E.6}$$

We accentuate that Equation E.6 only holds if the reference model h_{ref} is independent on the dataset S used to assess model performance. Now, by definition of $h^\star$, we have $\mu \leq 0$. However, under our assumption that $\widehat{\mathcal{L}}_S(h^\star) \geq \epsilon' + \widehat{\mathcal{L}}_S(h_{\text{ref}})$, we have that $\frac{1}{N}\sum_{i=1}^{N} \Delta^{(i)} \geq \epsilon' \geq 0$. Hence, we have a bounded random variable Δ whose true mean is negative but whose empirical mean is large and positive. This event becomes highly improbable as ϵ' increases or the sample size N increases, see the following picture.



Formally, using Hoeffding's inequality yields the bound

$$\begin{aligned}
\mathop{\mathbb{P}}_{S\sim\mathcal{D}^N}[\widehat{\mathcal{L}}_S(h^\star) \geq \epsilon' + \widehat{\mathcal{L}}_S(h_{\text{ref}})] &= \mathop{\mathbb{P}}_{S\sim\mathcal{D}^N}\left[\frac{1}{N}\sum_{i=1}^{N}\Delta^{(i)} \geq \epsilon'\right] && \text{(E.7)}\\
&\leq \mathop{\mathbb{P}}_{S\sim\mathcal{D}^N}\left[\frac{1}{N}\sum_{i=1}^{N}\Delta^{(i)} - \mu \geq \epsilon'\right] && \text{(Since } \mu \leq 0\text{)}\\
&\leq \exp\left\{-\frac{N\epsilon'^2}{2}\right\}. && \text{(With Hoeffding's inequality)}
\end{aligned}$$

□

E.1.2 Relation to Prior Work

Proposition E.1.3 (Proposition 8.2.1). *Let $\boldsymbol{\phi}(\cdot, \boldsymbol{x}, \mathcal{B})$ be a linear local feature attribution functional, and $E = \{h_k\}_{k=1}^M$ be an ensemble of M models from $\mathcal{H}$ trained with the same stochastic learning algorithm $h_k \sim \mathcal{A}(S)$. Said local feature attribution and ensemble will be employed in the methods of [Schulz et al., 2021, Shaikhina et al., 2021]. Moreover, let $\epsilon \geq \max\{\widehat{\mathcal{L}}_S(h_k)\}_{k=1}^M$ be an error tolerance, and let $\preceq_{\epsilon,\boldsymbol{x},\mathcal{B}}$ be the consensus order relation on $SA(\epsilon, \boldsymbol{x}, \mathcal{B})$ (cf. Equation 8.5). If the relation $i \preceq_{\epsilon,\boldsymbol{x},\mathcal{B}} j$ holds, we have that i is locally less important than j in the two total orders of prior work [Schulz et al., 2021, Shaikhina et al., 2021].*

Proof. We first note that, since $i, j \in \mathrm{SA}(\epsilon, \boldsymbol{x}, \mathcal{B})$, there is a consensus across the Rashomon Set that these features attributions have sign s_i and s_j respectively. As a reminder, this simplifies the expression of the feature importance : $\forall\, h \in \mathcal{R}(\mathcal{H}, \epsilon) \quad |\phi_i(h, \boldsymbol{x}, \mathcal{B})| = s_i \phi_i(h, \boldsymbol{x}, \mathcal{B})$. Additionally, our assumption that $\epsilon \geq \max\{\widehat{\mathcal{L}}_S(h_k)\}_{k=1}^M$, guarantees that $E \subseteq \mathcal{R}(\mathcal{H}, \epsilon)$. We now prove that the order relation $i \preceq_{\epsilon,\boldsymbol{x},\mathcal{B}} j$ is present in the two rankings from the literature.

[Shaikhina et al., 2021] compute the average model $\overline{h} = \frac{1}{M}\sum_{k=1}^M h_k$ and rank features according to their importance for this model $|\boldsymbol{\phi}(\overline{h}, \boldsymbol{x}, \mathcal{B})|$. For any $i, j \in \mathrm{SA}(\epsilon, \boldsymbol{x}, \mathcal{B})$, we deduce

$$
\begin{aligned}
i \preceq_{\epsilon,\boldsymbol{x},\mathcal{B}} j &\Rightarrow \forall\, h \in \mathcal{R}(\mathcal{H}, \epsilon) \quad |\phi_i(h, \boldsymbol{x}, \mathcal{B})| \leq |\phi_j(h, \boldsymbol{x}, \mathcal{B})| \\
&\Rightarrow \forall\, h \in \mathcal{R}(\mathcal{H}, \epsilon) \quad s_i\phi_i(h, \boldsymbol{x}, \mathcal{B}) \leq s_j\phi_j(h, \boldsymbol{x}, \mathcal{B}) \\
&\Rightarrow \forall\, h \in E \qquad\quad s_i\phi_i(h, \boldsymbol{x}, \mathcal{B}) \leq s_j\phi_j(h, \boldsymbol{x}, \mathcal{B}) \\
&\Rightarrow \frac{1}{M}\sum_{k=1}^M s_i\phi_i(h_k, \boldsymbol{x}, \mathcal{B}) \leq \frac{1}{M}\sum_{k=1}^M s_j\phi_j(h_k, \boldsymbol{x}, \mathcal{B}) \\
&\Rightarrow s_i\phi_i(\overline{h}, \boldsymbol{x}, \mathcal{B}) \leq s_j\phi_j(\overline{h}, \boldsymbol{x}, \mathcal{B}) \qquad \text{(By Linearity of } \boldsymbol{\phi}) \\
&\Rightarrow |\phi_i(\overline{h}, \boldsymbol{x}, \mathcal{B})| \leq |\phi_j(\overline{h}, \boldsymbol{x}, \mathcal{B})|, \qquad \text{(By Linearity of } \boldsymbol{\phi},\ s_i = \mathrm{sign}[\,\phi_i(\overline{h}, \boldsymbol{x}), \mathcal{B}\,])
\end{aligned}
$$

thus proving that the order relation is also present when explaining the average model.

[Schulz et al., 2021] order features using the mean rank $\frac{1}{M}\sum_{k=1}^{M}\mathbf{r}[\,|\boldsymbol{\phi}(h_k,\boldsymbol{x},\mathcal{B})|\,]$, where $\mathbf{r}:\mathbb{R}_+^d\to[d]$ is the rank function. By the definition, for any model h, we have $|\phi_i(h,\boldsymbol{x},\mathcal{B})|\leq|\phi_j(h,\boldsymbol{x},\mathcal{B})| \iff r_i[\,|\boldsymbol{\phi}(h,\boldsymbol{x},\mathcal{B})|\,]\leq r_j[\,|\boldsymbol{\phi}(h,\boldsymbol{x},\mathcal{B})|\,]$. Therefore,

$$\begin{aligned} i\preceq_{\epsilon,\boldsymbol{x},\mathcal{B}} j &\Rightarrow \forall\, h\in\mathcal{R}(\mathcal{H},\epsilon) \quad |\phi_i(h,\boldsymbol{x},\mathcal{B})|\leq|\phi_j(h,\boldsymbol{x},\mathcal{B})| \\ &\Rightarrow \forall\, h\in E \qquad\quad |\phi_i(h,\boldsymbol{x},\mathcal{B})|\leq|\phi_j(h,\boldsymbol{x},\mathcal{B})| \\ &\Rightarrow \forall\, h\in E \qquad r_i[\,|\boldsymbol{\phi}(h,\boldsymbol{x},\mathcal{B})|\,]\leq r_j[\,|\boldsymbol{\phi}(h,\boldsymbol{x},\mathcal{B})|\,] \\ &\Rightarrow \frac{1}{M}\sum_{k=1}^{M} r_i[\,|\boldsymbol{\phi}(h_k,\boldsymbol{x},\mathcal{B})|\,]\leq\frac{1}{M}\sum_{k=1}^{M} r_j[\,|\boldsymbol{\phi}(h_k,\boldsymbol{x},\mathcal{B})|\,], \end{aligned}$$

which implies that the order relation is also supported by the mean ranks. □

E.1.3 Random Forests

Proposition E.1.4 (Proposition 8.5.1). *Let $\mathcal{T}:=\{h^{tree,[r]}\}_{r=1}^{M}$ be a set of M trees, $\mathcal{H}_{m:}$ be the set of all subsets of at least m trees from $\mathcal{T}$, and $\phi:\mathcal{H}_{m:}\to\mathbb{R}$ be a linear functional, then $\min_{h\in\mathcal{H}_{m:}}\phi(h)$ amounts to averaging the m smallest values of $\phi(h^{tree,[r]})$ for $r=1,2,\dots M$.*

Proof. We can compute the linear functional on every tree $\{\phi(h^{\text{tree},[r]})\}_{r=1}^{M}$ and store the indices of the m smallest ones in a set C_m s.t. $|C_m|=m$ and

$$r\in C_m \text{ and } r'\notin C_m \Rightarrow \phi(h^{\text{tree},[r]})\leq\phi(h^{\text{tree},[r']}). \tag{E.8}$$

Now, to prove to proposition, we must show that $\phi(\frac{1}{m}\sum_{r\in C_m}h^{\text{tree},[r]})\leq\phi(h)\ \forall h\in\mathcal{H}_{m:}$. Since $\min_{h\in\mathcal{H}_{m:}}\phi(h)=\min_{k=m,\dots,M}\min_{h\in\mathcal{H}_k}\phi(h)$, the proof can be done in two parts: first for a fixed k we prove that $\phi(\frac{1}{k}\sum_{r\in C_k}h^{\text{tree},[r]})\leq\phi(h)\ \forall h\in\mathcal{H}_k$ and secondly prove that $\operatorname{argmin}_{k=m,\dots,M}\phi(\frac{1}{k}\sum_{r\in C_k}h^{\text{tree},[r]})=m$.

Part 1 By linearity $\phi(\frac{1}{k}\sum_{r\in C_k}h^{\text{tree},[r]})=\frac{1}{k}\sum_{r\in C_k}\phi(h^{\text{tree},[]}r)$. Also, remember that any model $h\in\mathcal{H}_k$ is associated to a subset C'_k of k seeds *i.e.* $h=\frac{1}{k}\sum_{r\in C'_k}h^{\text{tree},[r]}$. Importantly, since C_k and C'_k have the same size, the two sets $C_k\setminus C'_k$ and $C'_k\setminus C_k$ have a one-to-one

correspondence. We get

$$\begin{aligned}
\frac{1}{k}\sum_{r\in C_k}\phi(h^{\text{tree},[r]}) &= \frac{1}{k}\Bigg(\sum_{r\in C_k\cap C'_k}\phi(h^{\text{tree},[r]}) + \sum_{r\in C_k\setminus C'_k}\phi(h^{\text{tree},[r]})\Bigg) \\
&\leq \frac{1}{k}\Bigg(\sum_{r\in C_k\cap C'_k}\phi(h^{\text{tree},[r]}) + \sum_{r'\in C'_k\setminus C_k}\phi(h^{\text{tree},[r']})\Bigg) \qquad \text{(cf. Equation E.8)} \\
&= \frac{1}{k}\sum_{r\in C'_k}\phi(h^{\text{tree},[r]}) = \phi\Bigg(\frac{1}{k}\sum_{r\in C'_k}h^{\text{tree},[r]}\Bigg) = \phi(h).
\end{aligned}$$

Part 2 We now prove that $\operatorname{argmin}_{k=m,\ldots,M}\phi(\frac{1}{k}\sum_{r\in C_k}h^{\text{tree},[r]}) = m$. The key insight is that given $m' > m$, the set C_m contains the m smallest elements of $C_{m'}$. We get

$$\begin{aligned}
\frac{1}{m'}\sum_{r\in C_{m'}}\phi(h^{\text{tree},[r]}) &= \frac{1}{m'}\Bigg(\sum_{r\in C_m}\phi(h^{\text{tree},[r]}) + \sum_{r'\in C_{m'}\setminus C_m}\phi(h^{\text{tree},[r']})\Bigg) \\
&\geq \frac{1}{m'}\Bigg(\sum_{r\in C_m}\phi(h^{\text{tree},[r]}) + \sum_{r'\in C_{m'}\setminus C_m}\Bigg[\frac{1}{m}\sum_{r\in C_m}\phi(h^{\text{tree},[r]})\Bigg]\Bigg) \\
&= \frac{1}{m'}\Bigg(\sum_{r\in C_m}\phi(h^{\text{tree},[r]}) + \frac{m'-m}{m}\sum_{r\in C_m}\phi(h^{\text{tree},[r]})\Bigg) \\
&= \frac{1}{m'}\frac{m'}{m}\sum_{r\in C_m}\phi(h^{\text{tree},[r]}) = \frac{1}{m}\sum_{r\in C_m}\phi(h^{\text{tree},[r]}),
\end{aligned}$$

which ends the proof. □

E.2 Optimization

E.2.1 Optimization over a Ellipsoid

Linear Objective

We study the optimization of a linear function over an ellipsoid

$$\begin{aligned} \max_{\boldsymbol{\omega}} \quad & \boldsymbol{a}^T \boldsymbol{\omega} \\ \text{s.t.} \quad & (\boldsymbol{\omega} - \boldsymbol{\omega}_S)^T \boldsymbol{A} (\boldsymbol{\omega} - \boldsymbol{\omega}_S) \leq \epsilon - \widehat{\mathcal{L}}_S(\boldsymbol{\omega}_S), \end{aligned} \tag{E.9}$$

which is necessary to compute the local feature attribution consensus on the Rashomon Set of Additive Regression and Kernel Ridge Regression. To lighten the notation, we will introduce the variable $\epsilon' := \epsilon - \widehat{\mathcal{L}}_S(\boldsymbol{\omega}_S)$. Solving Equation E.9 can be done efficiently with a Cholesky decomposition of $\boldsymbol{A} = \boldsymbol{C}\boldsymbol{C}^T$, which we know exists since $\boldsymbol{A}$ is symmetric positive definite. We also have $\boldsymbol{A}^{-1} = (\boldsymbol{C}^{-1})^T \boldsymbol{C}^{-1}$. Now, it is always possible to map an ellipsoid back to a sphere by defining a new variable

$$\boldsymbol{z} := \boldsymbol{C}^T (\boldsymbol{\omega} - \boldsymbol{\omega}_S), \tag{E.10}$$

see Figure E.1. Applying the inverse change of variable to $\boldsymbol{\omega}$ in Equation E.9, we get

$$\begin{aligned} \boldsymbol{a}^T \boldsymbol{\omega} &= \boldsymbol{a}^T \Big((\boldsymbol{C}^{-1})^T \boldsymbol{z} + \boldsymbol{\omega}_S \Big) \\ &= \underbrace{\boldsymbol{a}^T (\boldsymbol{C}^{-1})^T}_{\boldsymbol{a}'^T} \boldsymbol{z} + \boldsymbol{a}^T \boldsymbol{\omega}_S, \end{aligned} \tag{E.11}$$

leading to the optimization problem

$$\begin{aligned} \max_{\boldsymbol{z}} \quad & \boldsymbol{a}'^T \boldsymbol{z} + \boldsymbol{a}^T \boldsymbol{\omega}_S \\ \text{s.t.} \quad & \boldsymbol{z}^T \boldsymbol{z} \leq \epsilon. \end{aligned} \tag{E.12}$$

Importantly, the optimization problems of Equations E.9 and E.12 both reach the same optimal values. Since the objective $\boldsymbol{a}'^T \boldsymbol{z}$ is a scalar product, it reaches its maximum objective value $\sqrt{\epsilon'} \|\boldsymbol{a}'\|$ when the vector $\boldsymbol{z}$ points in the same direction as $\boldsymbol{a}'$. The minimum and maximum values of the objective are therefore $\pm\sqrt{\epsilon - \widehat{\mathcal{L}}_S(\boldsymbol{\omega}_S)} \|\boldsymbol{a}'\| + \boldsymbol{a}^T \boldsymbol{\omega}_S$.

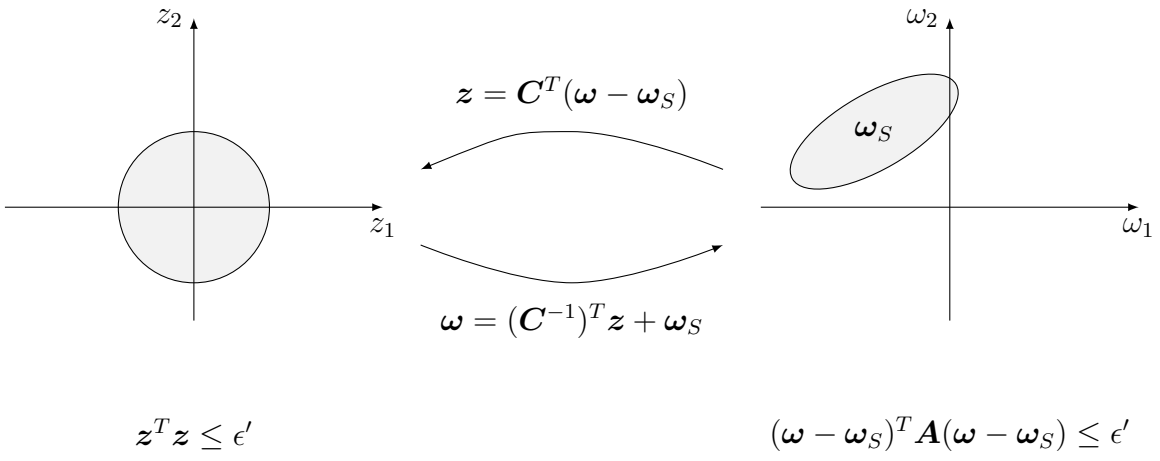


Figure E.1 Mapping an ellipsoid to the unit sphere.

Quadratic Objective

We now investigate the optimization of a quadratic form over an ellipsoid

$$\begin{aligned} \min_{\boldsymbol{\omega}} \quad & \boldsymbol{\omega}_i^T\boldsymbol{B}_i\boldsymbol{\omega}_i - \boldsymbol{\omega}_j^T\boldsymbol{B}_j\boldsymbol{\omega}_j \\ \text{s.t.} \quad & (\boldsymbol{\omega} - \boldsymbol{\omega}_S)^T\boldsymbol{A}(\boldsymbol{\omega} - \boldsymbol{\omega}_S) \leq \epsilon'. \end{aligned} \tag{E.13}$$

Letting $\boldsymbol{\omega}_{ij} \in \mathbb{R}^{M_i+M_j}$ be the concatenation of $\boldsymbol{\omega}_i$ and $\boldsymbol{\omega}_j$, and relabelling the least-square $\widehat{\boldsymbol{\omega}} := \boldsymbol{\omega}_S$, we express the optimization problem as

$$\begin{aligned} \min_{\boldsymbol{\omega}_{ij}} \quad & \boldsymbol{\omega}_{ij}^T\,\boldsymbol{B}_{ij}\,\boldsymbol{\omega}_{ij} \\ \text{s.t.} \quad & (\boldsymbol{\omega}_{ij} - \widehat{\boldsymbol{\omega}}_{ij})^T\boldsymbol{A}_{ij}(\boldsymbol{\omega}_{ij} - \widehat{\boldsymbol{\omega}}_{ij}) \leq \epsilon', \end{aligned} \tag{E.14}$$

where $\boldsymbol{B}_{ij}$ is a block-diagonal matrix containing $\boldsymbol{B}_i$ and $-\boldsymbol{B}_j$, and $\boldsymbol{A}_{ij}$ is the Schur complement of $\boldsymbol{A}$. The Schur complement is computed because we must project the Rashomon Set (which is an ellipsoid in $\mathbb{R}^{1+\sum_j M_j}$) onto the subspace $\mathbb{R}^{M_i+M_j}$ in which $\boldsymbol{\omega}_{ij}$ resides. Importantly, the projection of an ellipsoid on a subspace is still an ellipsoid whose covariance matrix is the Schur complement. Taking the Cholesky decomposition $\boldsymbol{A}_{ij} = \boldsymbol{C}\boldsymbol{C}^T$ and using the change of variable in Equation E.10, we get

$$\boldsymbol{\omega}_{ij}^T\,\boldsymbol{B}_{ij}\,\boldsymbol{\omega}_{ij} = (\boldsymbol{z}_{ij} - \widehat{\boldsymbol{z}}_{ij})^T\boldsymbol{B}'_{ij}(\boldsymbol{z}_{ij} - \widehat{\boldsymbol{z}}_{ij}), \tag{E.15}$$

with $\boldsymbol{B}'_{ij} = \boldsymbol{C}^{-1}\boldsymbol{B}_{ij}(\boldsymbol{C}^{-1})^T$ and $\widehat{\boldsymbol{z}}_{ij} := -\boldsymbol{C}^T\widehat{\boldsymbol{\omega}}_{ij}$. Thus, we can express the optimization in standard TRS form

$$\begin{aligned} \min_{\boldsymbol{z}_{ij}} \quad & (\boldsymbol{z}_{ij} - \widehat{\boldsymbol{z}}_{ij})^T \boldsymbol{B}'_{ij}(\boldsymbol{z}_{ij} - \widehat{\boldsymbol{z}}_{ij}) \\ \text{s.t.} \quad & \boldsymbol{z}_{ij}^T \boldsymbol{z}_{ij} \leq \epsilon' \end{aligned} \tag{E.16}$$

and solve the following necessary optimality conditions adapted from Corollary 7.2.2 in [Conn et al., 2000, Section 7.2].

Corollary E.2.1 (TRS Necessary Optimality Condition)**.** *Letting $\{\sigma_k\}_k$ be the eigenvalues of the matrix $\boldsymbol{B}'_{ij}$, any global minimizer $\boldsymbol{z}_{ij}$ of the TRS (Equation E.16) must satisfy*

$$\boldsymbol{B}'_{ij}(\boldsymbol{z}_{ij} - \widehat{\boldsymbol{z}}_{ij}) = \lambda \boldsymbol{z}_{ij} \tag{E.17}$$

$$\lambda(\boldsymbol{z}_{ij}^T \boldsymbol{z}_{ij} - \epsilon') = 0, \tag{E.18}$$

*for some $\lambda \geq \max\{0\} \cup \{\text{-}\sigma_k\}_k$. If $\lambda > \max\{\text{-}\sigma_k\}_k$ then $\boldsymbol{z}_{ij}$ is the **unique** global minimizer.*

To solve these conditions, we diagonalize $\boldsymbol{B}'_{ij} = \boldsymbol{V}\boldsymbol{D}\boldsymbol{V}^T$, define $\boldsymbol{t} = \boldsymbol{V}^T\boldsymbol{z}_{ij}$ and $\widehat{\boldsymbol{t}} = \boldsymbol{V}^T\widehat{\boldsymbol{z}}_{ij}$. Then, assuming $\lambda > \max\{-\sigma_k\}_k$, we rewrite Equation E.17 as

$$\boldsymbol{t} = (\boldsymbol{D} + \lambda \boldsymbol{I})^{-1}\boldsymbol{D}\widehat{\boldsymbol{t}}. \tag{E.19}$$

Also assuming $\lambda > 0$, Equation E.18 becomes $\boldsymbol{t}^T\boldsymbol{t} = \epsilon'$, which combined with Equation E.19 yields

$$q(\lambda) := \sum_k \frac{\sigma_k^2}{(\sigma_k + \lambda)^2}\widehat{t}_k = \epsilon'. \tag{E.20}$$

We finally solve the non-linear Equation $q(\lambda) = \epsilon'$ for $\lambda > \max\{0\} \cup \{\text{-}\sigma_k\}_k$ with the bisection algorithm. From the resulting λ we can determine the TRS solution $\boldsymbol{z}_{ij}$.

If we do not assume $\lambda > \max\{0\} \cup \{\text{-}\sigma_k\}_k$, there are two additional cases to consider:

1. The solution is inside the ball ($\lambda = 0$).
2. The so-called "Hard Case" where $\lambda = \max\{\text{-}\sigma_k\}_k$ and $(\boldsymbol{D} + \lambda\boldsymbol{I})$ becomes singular.

For simplicity, we do not address them in this Appendix. We instead refer to [Conn et al., 2000, Section 7.3] for discussion on these technicalities.

E.2.2 Combinatorial Optimization and Relaxations

Min/Max of Global Importance

In this section we discuss the combinatorial optimization problems that occur when computing the global feature importance over the Rashomon Set of Random Forests. As a reminder, we have defined

$$\mathcal{H}_m := \left\{ \frac{1}{m} \sum_{t \in T} t \ : \ T \subseteq \mathcal{T} \ \text{ and } \ |T| = m \right\}, \tag{E.21}$$

as the set of RFs containing m trees. An alternative way to represent such a set is to introduce binary variables $\boldsymbol{z} \in \{0,1\}^M$ with $\sum_{r=1}^M z_r = m$ and view all RFs from $\mathcal{H}_m$ as $\frac{1}{m} \sum_{r=1}^M z_r t_r$ for some $\boldsymbol{z}$.

Now letting $\phi_j^{\text{SHAP-int}}$ be the Interventional Shapley Value feature j, we want to maximize/minimize the GFI $\widehat{\Phi}_j^{\text{SHAP},[1]}(h) := \frac{1}{N} \sum_{i=1}^N | \phi_j^{\text{SHAP-int}}(h, \boldsymbol{x}^{(i)}) |$ across all RFs with m trees

$$\begin{aligned} \min_{\boldsymbol{z}}/\max_{\boldsymbol{z}} \quad & \frac{1}{Nm} \sum_{i=1}^N \left| \sum_{r=1}^M z_r \, \phi_j^{\text{SHAP-int}}(t_r, \boldsymbol{x}^{(i)}) \right| \\ \text{s.t.} \quad & \boldsymbol{z} \in \{0,1\}^M \text{ and } \sum_{r=1}^M z_r = m. \end{aligned} \tag{E.22}$$

These are non-linear combinatorial problems that are extremely hard to solve. For that reason, we will provide quick approximate solutions based on a Linear relaxation of Equation E.22. The first step of the relaxation is to enlarge the domain of $\boldsymbol{z}$ to allow fractional values.

$$\begin{aligned} \min_{\boldsymbol{z}}/\max_{\boldsymbol{z}} \quad & \sum_{i=1}^N \left| \sum_{r=1}^M z_r \, \phi_j^{\text{SHAP-int}}(t_r, \boldsymbol{x}^{(i)}) \right| \\ \text{s.t.} \quad & \boldsymbol{z} \in [0,1]^M \text{ and } \sum_{r=1}^M z_r = m. \end{aligned} \tag{E.23}$$

The corresponding domain is a polytope and so it is compatible with Linear Programs. The second step of the Linear relaxation is to rephrase the absolute value function $|\cdot|$ as a Linear Program

$$\begin{aligned} |t| = \min_{\beta} \quad & \beta \\ \text{s.t.} \quad & t \leq \beta \\ & -t \leq \beta \end{aligned} \tag{E.24}$$

$$\begin{aligned} |t| = \max_{\beta} \quad & \beta t \\ \text{s.t.} \quad & -1 \leq \beta \\ & \beta \leq 1 \end{aligned} \tag{E.25}$$

After we get a solution to the relaxation of Equation E.23, we project $\boldsymbol{z}$ back on $\{0,1\}^M$

using the following heuristic: if there are o components with $z = 1$, we select $M - o$ fractional values in decreasing order and set them to one. The other fractional values are set to zero. For example, if we have $M = 3$ and find a solution $\boldsymbol{z} = [0, 1, 1, 0.75, 0.25]$ to the relaxation, we would discretize the solution to get $\boldsymbol{z} = [0, 1, 1, 1, 0]$. This heuristic may be sub-optimal, but our goal is to provide quick approximate solutions.

Maximize By leveraging Equation E.25, we can reformulate Equation E.23 as

$$\begin{aligned}\max_{\boldsymbol{z}} \sum_{i=1}^{N} \left| \sum_{r=1}^{M} z_r \, \phi_j^{\text{SHAP-int}}(t_r, \boldsymbol{x}^{(i)}) \right| &= \max_{\boldsymbol{z}} \sum_{i=1}^{N} \max_{\beta_i \in [-1,1]} \beta_i \sum_{r=1}^{M} z_r \phi_j^{\text{SHAP-int}}(t_r, \boldsymbol{x}^{(i)}) \\ &= \max_{\boldsymbol{z},\boldsymbol{\beta}} \sum_{i=1}^{N} \sum_{r=1}^{M} z_r \, \beta_i \phi_j^{\text{SHAP-int}}(t_r, \boldsymbol{x}^{(i)}) \\ &= \max_{\boldsymbol{z},\boldsymbol{\beta}} \boldsymbol{\beta}^T \boldsymbol{B} \boldsymbol{z},\end{aligned} \tag{E.26}$$

where $\boldsymbol{z}$ and $\boldsymbol{\beta}$ are each restricted to a separate polytope and $B_{ir} \equiv \phi_j^{\text{SHAP-int}}(t_r, \boldsymbol{x}^{(i)})$. Equation E.26 is known as a Bilinear Program which is a non-convex optimization problem that can be solved to local optima via the coordinate ascent algorithm [Nahapetyan, 2009]. In our setting, the output of the coordinate ascent algorithm will already respect $\boldsymbol{z} \in \{0, 1\}^M$ since $\max_{\boldsymbol{z}} \boldsymbol{\beta}^T \boldsymbol{B} \boldsymbol{z}$ under the constraints on $\boldsymbol{z}$ yields $z_r = 1$ for the m largest values of $\sum_{i=1}^{N} \beta_i \phi_j^{\text{SHAP-int}}(t_r, \boldsymbol{x}^{(i)})$ and $z_r = 0$ for the others.

Minimize By leveraging Equation E.24, we can reformulate Equation E.23 as

$$\begin{aligned}\min_{\boldsymbol{z},\boldsymbol{\beta}} \quad & \sum_{i=1}^{N} \beta_i \\ \text{s.t.} \quad & \boldsymbol{z} \in [0, 1]^M \text{ and } \sum_{r=1}^{M} z_r = m \\ & \sum_{r=1}^{M} z_r \, \phi_j^{\text{SHAP-int}}(t_r, \boldsymbol{x}^{(i)}) \leq \beta_i \\ & -\sum_{r=1}^{M} z_r \, \phi_j^{\text{SHAP-int}}(t_r, \boldsymbol{x}^{(i)}) \leq \beta_i,\end{aligned} \tag{E.27}$$

which is a Linear Program with $N + M$ variables and $2(N + M) + 1$ constraints that we can solve efficiently if N and M are not too large. However, the solution of this LP can have fractional values so we must use the discretization heuristic to get the final solution $\boldsymbol{z} \in \{0, 1\}^M$.

Now that we have discussed approximate schemes to get the min/max global feature impor-

tance across $\mathcal{H}_m$, we are left with addressing *relative* importance relations between features.

Global Relative Importance

To assert a consensus on global relative importance (cf. Definitions 8.2.6 & 8.2.7) we must solve $\min/\max_h \Phi_j^{\text{SHAP},[1]}(h) - \Phi_k^{\text{SHAP},[1]}(h)$. However, as previously discussed, we cannot guarantee to minimize/maximize $\Phi_j^{\text{SHAP},[1]}(h)$ to optimality for Random Forests. Consequently, we cannot guarantee to solve $\min/\max_h \Phi_j^{\text{SHAP},[1]}(h) - \Phi_k^{\text{SHAP},[1]}(h)$ to optimality either. This is a critical issue because the resulting partial order may not be transitive. Our solution is to create an ensemble E containing

1. Approximates of $\text{argmin}/\max_h \Phi_j^{\text{SHAP},[1]}(h)$ for $1 \leq j \leq d$.
2. Approximates of $\text{argmin}/\max_h \Phi_j^{\text{SHAP},[1]}(h) - \Phi_k^{\text{SHAP},[1]}(h)$ for $1 \leq j < k \leq d$.

After, we assert a consensus among all models in $E \subset \mathcal{H}_{m(\epsilon)}$ leading to the partial order

$$j \mathrel{\widehat{\preceq}_\epsilon} k \iff \forall\, h \in E \quad \Phi_j(h) \leq \Phi_k(h). \tag{E.28}$$

We underestimate the diversity of our models but the resulting partial order of global importance is guaranteed to be transitive. To approximate $\text{argmin}/\max_h \Phi_j^{\text{SHAP},[1]}(h) - \Phi_k^{\text{SHAP},[1]}(h)$, we define the sets

$$S_j := \{i \in [N] : \forall h \in \mathcal{H}_m,\ \text{sign}[\phi_j(h, \boldsymbol{x}^{(i)})] = s_{ij}\} \tag{E.29}$$

storing instances whose local attributions for features j has a consistent sign across all RFs in $\mathcal{H}_m$. Then we solve

$$\begin{aligned}
&\underset{\boldsymbol{z}}{\text{argmin}/\max} \sum_{i \in S_j \cap S_k} \left| \sum_{r=1}^{M} z_r\, \phi_j^{\text{SHAP-int}}(t_r, \boldsymbol{x}^{(i)}) \right| - \left| \sum_{r=1}^{M} z_r\, \phi_k^{\text{SHAP-int}}(t_r, \boldsymbol{x}^{(i)}) \right| \\
&= \underset{\boldsymbol{z}}{\text{argmin}/\max} \sum_{i \in S_j \cap S_k} s_{ij} \sum_{r=1}^{M} z_r\, \phi_j^{\text{SHAP-int}}(t_r, \boldsymbol{x}^{(i)}) - s_{ik} \sum_{r=1}^{M} z_r\, \phi_k^{\text{SHAP-int}}(t_r, \boldsymbol{x}^{(i)}) \\
&= \underset{\boldsymbol{z}}{\text{argmin}/\max} \sum_{r=1}^{M} z_r \left(\sum_{i \in S_j \cap S_k} s_{ij} \phi_j^{\text{SHAP-int}}(t_r, \boldsymbol{x}^{(i)}) - s_{ik}\, \phi_k^{\text{SHAP-int}}(t_r, \boldsymbol{x}^{(i)}) \right) \\
&= \underset{\boldsymbol{z}}{\text{argmin}/\max} \sum_{r=1}^{M} z_r a_{jkr}
\end{aligned} \tag{E.30}$$

which is a linear function of $\boldsymbol{z}$ thus we can leverage Proposition 8.5.1.