POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Scaling Logical Analysis of Data for large volume and streaming data in Industry 4.0 applications

OSAMA ELFAR
Département de mathématiques et de génie industriel

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

Mai 2022

Scaling Logical Analysis of Data for large volume and streaming data in Industry 4.0 applications

DEDICATION

To my parents…

For their endless love, support and encouragement…

ACKNOWLEDGEMENTS

First of all, I would like to thank my supervisor Professor Soumaya Yacout, for her enthusiasm for the research, for her support, encouragement and patience. Professor Soumaya continuously provided encouragement and was always willing and enthusiastic to assist in any way she could throughout the research project. I would like to thank my co-supervisor Dr. Hany Osman for his consistent support and guidance during the running of this project. I would like to extend my thanks to the jury members Professor Luc Adjengue, Professor Hanane Dagdougui and Dr. Tamer Gyed to take the time to evaluate this work. Furthermore I would like to thank Adham Mohammed for his collaborative effort, teamwork, and his productive discussions.

I am really grateful and thankful to my whole family, to my mother who always pray for me and push me hard to pursue my goals; to my father who is ever an exceptional advisor throughout my life, and always very happy for my success, there are no words can describe my gratitude to them and their endless support; to my uncle, Abdul Elfar who always supports and encourages me in Canada. I am really thankful to my brothers and sister who are always there for me no matter what and whenever needed – I just want you know that every success of you was an encourage to me to be a better version of myself, and every unsuccessful attempt of you was a big experience for me.

The utmost grateful and thankful to my beloved wife for her endless support, comprehension and patience throughout the years I took to finish this work. I am really thankful to my mother-in-law for her deep comprehension in the critical situations, and for her big support for me and my family. Last but not least, my endless thanks to my sons, Abdelrahman and Malek, for their understanding and patience for the long time I spend away to finish this work – you are the reasons to pursue my goals.

RÉSUMÉ

Les avancées significatives dans la collecte et le stockage des données ont contribué au développement et à l'émergence des techniques d'apprentissage automatique (ML) en tant qu'outils puissants pour le traitement des données et la prise de décision dans les systèmes industriels. Ces systèmes ont essayé de s'adapter aux évolutions des progrès qui ont conduit à de nouveaux concepts tels que l'industrie 4.0 et la qualité 4.0. Dans de tels systèmes, les données sont stockées dans de grands volumes et se présentent sous la forme de flux continus de grandes quantités, au lieu des petits ensembles de données statiques traditionnels qui ont longtemps été utilisés pour créer des modèles statistiques et pour former des modèles ML. Ces systèmes industriels nécessitent des modèles ML à fort pouvoir d'interprétabilité. Dans ce contexte, l'analyse logique des données (LAD), en tant que technique de classification hautement interprétable, devient un candidat de l'approche ML qui a la capacité de traiter un grand volume de données ainsi que des données en continu tout en préservant son pouvoir distinct d'interprétabilité. De telles versions de LAD n'ont pas encore été développées, et c'est la proposition de cette thèse. En tant que tel, l'objectif de cette thèse est de développer une technique ML basée sur le LAD classique, mais avec la capacité de traiter un grand volume de données en continu, qui présente un phénomène de dérive de concept. Ce phénomène est très courant en milieu industriel car tous les actifs physiques subissent un vieillissement et une détérioration.

Premièrement, une technique basée sur les ensembles est développée pour accélérer LAD et le rendre capable de traiter un grand volume de données. Un nouveau mécanisme est proposé pour développer un système d'ensemble pour LAD (LAD-ENS) afin d'améliorer son efficacité de calcul, tout en préservant son interprétabilité et sa précision. Ce nouveau mécanisme vise à maintenir le pouvoir explicatif de la LAD classique en combinant les classificateurs individuels au niveau des motifs. À l'aide d'ensembles de données obtenus à partir du référentiel d'apprentissage automatique de l'UCI, des expériences informatiques sont menées pour démontrer les performances de LAD-ENS en termes de temps de calcul, de précision de la classification et d'interprétabilité. En plus d'obtenir une réduction statistiquement significative du temps de calcul, le LAD-ENS développé atteint des précisions de classification compétitives par rapport à deux approches LAD classiques et à cinq algorithmes d'apprentissage automatique courants.

Deuxièmement, une adaptation de la technique LAD est fournie pour devenir dynamique et adaptative afin de pouvoir gérer des flux de données continus qui incluent des dérives de concept - c’est ce qu’on appelle l’analyse logique dynamique et adaptative des données en continu, ou DA-LASD. De plus, DA-LASD est conçu pour avoir la capacité de gérer des données de flux déséquilibrées. Le cadre proposé est construit de différents modules qui modifient dynamiquement les caractéristiques des modèles LAD ; éliminer les modèles pourris et inefficaces ; et en générer de nouveaux si nécessaire. Cela met à jour en permanence le classificateur LAD pour s’adapter aux changements dans les flux de données. Le DA-LASD est testé sur plusieurs jeux de données synthétiques couvrant différents types de dérives de concept. Les résultats montrent comment le cadre proposé adapte dynamiquement le modèle et améliore avec succès toutes les mesures de performance qui commencent à décliner. De plus, il s’avère assez compétitif en termes de précision de classification, par rapport à d’autres techniques d’apprentissage automatique qui traitent des données en continu. En plus de sa puissance d’interprétabilité distinctive, le DA-LASD est un cadre prometteur pour une gamme variée d’applications où la haute précision et l’interprétabilité sont toutes deux essentielles.

Le système de contrôle de la qualité des processus industriels est considéré comme l’un des systèmes qui devraient être adaptés pour être plus intelligents et automatisés sous le label de l’Industrie 4.0. Cela peut être réalisé en utilisant une technique d’apprentissage automatique interprétable afin de prendre des actions correctives automatiques pour les états hors de contrôle et de ramener le processus à l’état sain. De plus, cette technique doit traiter et s’adapter dynamiquement à la nature des données de flux collectées à partir de ces systèmes avancés et complexes. Par conséquent, DA-LASD est renforcé par un processus d’ingénierie des fonctionnalités et utilisé dans ce contexte à travers une étude de cas de l’industrie aérospatiale. Le modèle proposé montre à quel point il est résilient et durable, et comment il s’adapte dynamiquement aux dérives du concept et améliore avec succès les mesures de performance qui commencent à se détériorer après les dérives du concept. De plus, le modèle proposé surpasse statistiquement les autres techniques d’apprentissage automatique en termes de sensibilité de classification, ce qui est important pour mesurer la capacité à détecter les défauts et les états hors de contrôle, tout en fournissant de puissants modèles interprétables qui aident à prendre des mesures de contrôle automatiques.

Avec leurs hautes précisions et leur pouvoir d’interprétabilité distinctif, le LAD-ENS et le DA-LASD montrent une performance prometteuse dans les applications où un grand volume de données ou de flux de données existe et où l’interprétabilité est requise, à savoir dans les applications industrielles et de machines.

ABSTRACT

The significant advancements in data collection and storage have helped the development and the emergence of Machine Learning (ML) techniques as powerful tools for data processing and decision making in industrial systems. These systems have been trying to adapt to the advancements changes led to new concepts such as Industry 4.0 and Quality 4.0. In such systems, data is stored in large volumes and comes in the form of continuous streams of large amounts, instead of the traditional small and static datasets that have long been used to build statistical models, and to train ML models. These industrial systems require ML models with high-interpretability power. In this context, Logical Analysis of Data (LAD) as a highly interpretable classification technique, becomes a candidate of ML approach that has the ability to process large volume of data as well as streaming data while preserving its distinct power of interpretability. Such versions of LAD have not yet been developed, and it is the proposal in this thesis. As such, the objective of this thesis is to develop a ML technique that is based on the classical LAD, but with the ability of processing large volume of streaming data, which exhibits concept-drift phenomenon. This phenomenon is very common in industrial setting because all physical assets experience aging and deterioration.

Firstly, an ensemble-based technique is developed to accelerate LAD and make it able to process a large volume of data. A novel mechanism is proposed for developing an ensemble system for LAD (LAD-ENS) to improve its computational efficiency, while preserving its interpretability and accuracy. This new mechanism aims to maintain the explanatory power of classical LAD by combining the individual classifiers at the level of patterns. Using datasets obtained from the UCI Machine Learning Repository, computational experiments are conducted to demonstrate the performance of LAD-ENS in terms of computational time, classification accuracy, and interpretability. In addition to achieving a statistically significant reduction in computational time, the developed LAD-ENS achieves competitive classification accuracies compared to two classical LAD approaches and five common machine learning algorithms.

Secondly, an adaptation of LAD technique is provided to become dynamic and adaptive to be able to handle continuous data streams that include concept drifts – it is called Dynamic and Adaptive Logical Analysis of Streaming Data, or DA-LASD. Moreover, DA-LASD is devised in order to have the ability for handling imbalanced streaming data. The proposed framework is built of different modules that dynamically modify the characteristics of the LAD patterns; eliminate decayed and inefficient patterns; and generate new ones if needed. This continuously updates the LAD classifier to adapt to the changes in the data streams. The DA-LASD is tested on several synthetic datasets covering different types of concept drifts. The results show how the proposed framework dynamically adapts the model, and successfully improves any performance measures that start to decline. Moreover, it proves quite competitive in terms of the classification accuracy, compared to other machine learning techniques that handle streaming data. In addition to its distinctive interpretability power, the DA-LASD is a promising framework for a diverse range of applications where high accuracy and interpretability are both essential.

The industrial process quality control system is considered one of the systems that should be adapted to be more intelligent and automated under the label of Industry 4.0. That can be achieved by using an interpretable machine learning technique in order to take automatic corrective actions for out-of-control states and bring the process back to the healthy state. Moreover, this technique has to deal and adapt dynamically with the nature of the streaming data collected from such advanced and complex systems. Therefore, DA-LASD is reinforced with a feature engineering process and used in this context through a case study from the aerospace industry. The proposed model shows how resilient and sustainable it is, and how it is dynamically adaptive to the concept drifts and successfully improves performance measures that start to deteriorate after the concept drifts. Moreover, the proposed model outperforms statistically other machine learning techniques in terms of classification sensitivity which is important to measure the ability to detect faults and out of control states, while providing powerful interpretable patterns that help automatic control actions to be taken.

With their high accuracies and their distinctive interpretability power, the LAD-ENS and the DA-LASD show a promising performance in applications where large volume of data or data stream exists, and interpretability is required, namely, in industrial and machinery applications.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1 INTRODUCTION

1.1 Industry 4.0

The combination of future-oriented technologies in intelligent machines and products, and advanced network technologies pushes to a new paradigm shift in industrial production underlying digital transformation. This new industrial shift has been considered the fourth industrial revolution and known as Industry 4.0 [3, 4]. In this new paradigm, products will control their own manufacturing process by further increase in mechanization and adopting automatic solutions which depend on analytical components. That will lead to fully smart factories which will completely devised with sensors, actors, and autonomous controlling systems [3].

Accordingly, that leads to industrial significant advancements in data collection and storage which help the development and emergence of machine learning (ML) techniques as powerful tools in autonomous controlling systems in Industry 4.0 applications such as process control, quality management and maintenance systems, to name a few. Most of these applications now produce data in the form of streams. Data streams are defined by algorithmic community to support real-time analytics. They are sequences of data observations, possibly infinite, each has a timestamp and a temporal order. Observations arrive one by one, and should be processed in real time. The data distribution may be shifted over time which is called concept drift [5].

1.2 Machine learning in industry 4.0

Due to the demand of autonomous controlling systems in different applications under the label of Industry 4.0, ML techniques have increasingly been investigated to deliver optimal models in this context of Industry 4.0. In order to achieve an optimal model in this context, it should consider four important performance criteria: robustness, resilience, sustainability, and interpretability.

Robustness. A robust machine learning model should have a testing error consistent with its training error, i.e., the model accuracy doesn't deteriorate too much when testing with slightly different data after adding some noise. Therefore, the model will have low bias and low variance errors. This is the first basic problem that is faced during building the model.

Resilience. Resilient machine learning model is the one maintains optimal performance after deployment and on variety of test sets. It should not be overfitted to one test data set. Resilience is the robustness against changes overtime. This problem occurs in industrial application when there is lack of training data at the modeling phase and data arrives in different time in the future. Therefore, a resilient machine learning model is demanded so that it solve this practical problem.

Sustainability. A sustainable machine learning model is the one which can capture the distribution drifts in the new data, and hence automatically adapt itself to the new concept. In the real and practical data after deployment the model in industrial applications, there are shifts that occur overtime, and hence the distribution of data used to train the model is different from the newly available data. Therefore, a sustainable machine learning model is needed to overcome this practical problem which is very common in industrial setting because all physical assets experience aging and deterioration, and hence the distribution of new data changes overtime.

Interpretability. An interpretable machine learning model is the one that have an explanatory power able to provide root-cause analysis of certain phenomena to facilitate the decision-making process. This criteria is very important in the demanded industrial autonomous controlling systems in order to provide feedback so that autonomous corrective decisions can be made.

1.3 Logical Analysis of Data

Most of the well-known ML techniques, despite they provide robust models and some of resilient models, still do not provide the sustainability and the interpretability needed for the real and practical autonomous systems such as demanded ones in Industry 4.0. On the other hand, Logical Analysis of Data (LAD) is a robust classification technique that provides the needed interpretability through generating patterns containing structural knowledge that explain the hidden phenomena under study. As such, the patterns generated are the most useful results that indicate the candidate root causes behind the observed physical phenomena, and consequently, the best way to provide autonomous feedback to them. The patterns it generates are considered unique components, and hence, they can be flexibly modified or eliminated solely without the need to change the entire model. This flexibility makes the LAD quite promising for delivering a resilient and sustainable models for Industry 4.0 applications.

Limitations of Logical Analysis of Data. Despite its robust models and its provided interpretability, LAD suffers from relatively long computational time that makes it unsuitable for manipulating data with large volumes. In addition to that, LAD does not has a version that has the ability to adapt the model dynamically to concept drifts occur in the data while preserving its distinct power of interpretability, and hence increases the resilience and sustainability.

1.4 Research objectives

1.4.1 General objectives

In this doctoral research, the ultimate objective is to provide robust, resilient, sustainable, and interpretable classification model that able to be utilized in the demanded autonomous controlling systems in the different applications in Industry 4.0. The model that is based on LAD, should be able to perform self-monitoring for its performance while it is deployed in the application and be able to detect concept drifts in data, and hence adapt itself to the new concepts in the data. Maintaining the interpretability of LAD is an important target in order to be useful for feedback and autonomous corrective actions in Industry 4.0 applications.

1.4.2 Specific research objectives

The above general objective is achieved fulfilling the following specific objectives:

1. Develop an accelerated version of LAD based on an ensemble technique in order to overcome its limitation in processing large volume of data. The developed version should maintain the robustness and interpretability of LAD.

2. Develop a dynamic and adaptive version of LAD that able to process and learn from streaming data. Such version is able to monitor its performance while it is on the service and able to detect any drift in the concept through the streaming data, and hence adapt itself to the new data. As such model has high resilience and sustainability over time.

3. Apply the new dynamic and adaptive version of LAD on one of the applications of Industry 4.0, specifically the industrial process control, in order to evaluate the developed model on practical, high-dimensional, and dynamic data.

1.5 Originality

The originality and novelty of this research are as follows:

1. A new ensemble Logical Analysis of Data (LAD-ENS) is developed by introducing a novel combining mechanism that integrates all patterns that are provided by the individual LAD classifiers. This mechanism preserves the explanatory power of LAD patterns and does not reduce their interpretability. it is the first ensemble model developed for LAD.

2. A comprehensibility index is introduced in order to study the change in the explanatory power of LAD.

3. A novel dynamic and adaptive Logical Analysis of Data is developed to process streaming data online and adapt LAD to change in data distribution over time–concept drifts. The proposed methodology is called Dynamic and Adaptive Logical Analysis of Data. To the best of our knowledge, it is the first sustainable LAD for streaming data.

4. A novel characteristics for LAD patterns are introduced to evaluate the pattern efficiency in the case of streaming data; the covering index and the protected homogeneity.

5. A methodology is developed to build a DA-LASD model reinforced with a feature engineering mechanism for an intelligent and interpretable monitoring system of turbofan engine. The model gives accurate detection of faults and provide high resilience and sustainability, specifically after a concept drift.

1.6 Literature review

1.6.1 Accelerating LAD

Some techniques have been presented in the literature to implement LAD with the aim of enhancing computational efficiency. In [6], a polynomial algorithm was used to enumerate all LAD patterns with a selected degree to limit the number of features in the generated patterns. The degree of a pattern is the number of features the pattern is constructed with. However, with datasets that have large amounts of features, this technique becomes computationally expensive as it still generates a high number of patterns. In [7], the column generation technique is used to generate patterns with specific characteristics instead of generating all possible patterns. In this framework, the master problem is developed with the objective of building a LAD model with a maximum separation margin between the classes. This is achieved by generating patterns to enlarge this margin in subproblems. This column generation framework generates only one pattern in each iteration, which affects its computational efficiency. Therefore in [8], a multi-pattern generation framework is developed to improve the efficiency of LAD column generation models by generating more than one pattern in each iteration. However, none of these techniques is aimed at achieving a scalable LAD that handles a large volume of data.

Developing scalable ML algorithms that can handle large datasets has attracted researchers due to continuous increase in the volume of data [9]. In [10], a taxonomy was proposed for ML techniques that can process a large volume of data through either designing more efficient algorithms or relying on parallelism. The parallelism-based category reflects most of the state of the art [11]. Specifically, the taxonomy classified the parallelized methods into two sub-categories: (i) parallelized model/parameters: developing parallelized versions of learning algorithms by first dividing the learning model/parameters and then performing computations on each division concurrently, and (ii) parallelized data: partitioning input data vertically, horizontally, or even arbitrarily into manageable subsets, and hence computing all subsets simultaneously [10], [12]. The parallelized-data techniques are mainly ensemble based, such as random forests [13] and XGBoost [14]. There is no research study yet to develop ensemble technique for LAD in a way that preserve its interpretability and explanatory power. And this is one of the objectives of this doctoral research.

1.6.2 Processing data streams by ML techniques

One of the first ML techniques that was modified to process streaming data is Decision Trees in Hoeffding Tree algorithm [15]. This algorithm is built based on splitting the leaves of the constructed tree by scanning each observation to extract the new information into the model. In order to speed up the process, the same authors then proposed the Very Fast Decision Tree (VFDT) as an improved version of the algorithm of incorporating the new information. This is achieved by computing the best leaf splitting when a specific number of observations arrive. In addition, the least promising nodes are deactivated to decrease the computational memory. By using the sliding window technique, the VFDT was extended to the Concept-Adapting Very Fast Decision Tree (CVFDT) in order to consider concept drifts in the data stream [16].

Other adaptations to ML techniques include the Lazy Learning algorithm [17] applied to the k-nearest neighbors method. This algorithm uses a sliding window as the search space used in determining the k-nearest neighbors to a new, unclassified observation, and as the window slides, the method naturally takes into account any data concept drift that may exist.

The reservoir sampling [18] and the ensemble methods [19, 20] are more generic frameworks that can be applied to different ML techniques to handle streaming data. Reservoir sampling is based on maintaining a sampling reservoir from the data stream, and frequently updating this reservoir, and accordingly the model. On the other hand, ensemble methods are based on building an ensemble model of different classifiers trained on different batches from the data stream. As the data stream flows and more classifiers are built, the ensemble model is updated based on the accuracy of its classifiers. This ensures that the model is tuned dynamically to optimize the accuracy according to the recent part of the data stream.

All these techniques, despite being quite helpful in many applications, still do not provide the interpretability needed for Industry 4.0 applications – or at least do not maintain this interpretability as their models grow in size. Therefore one the main objectives in this research is to develop a framework based on LAD able to process data streams.

1.6.3 ML techniques in industrial process control

Industrial process control monitors the process performance and provides the required feedback for corrective actions [21]. Lately, this has been achieved through applying data-driven approaches and machine learning (ML) techniques to detect anomalies and predict the out-of-control states.

Machine learning techniques, such as support vector data description (SVDD), neural networks, decision trees, k-means clustering, linear discriminant analysis, principal component analysis (PCA) and Logical Analysis of Data (LAD), have been used with control charts to detect out-of-control states and anomalities [22, 23]. In [24], a pattern recognition-based fault detection method was proposed using SVDD as a one-class classification algorithm. This method was applied for chillers monitoring and faults detection in [25]. To improve the fault detection performance, a PCA-R-SVDD based method is proposed in [26] by developing a SVDD model in the residual subspace (Rs) using the PCA modeling residual data. In [27], neural network is used for real-time detection of faults. A neural network model for fault detection is defined by using dynamic neurons in [28] and applied in sugar evaporation process. In [29] and [30], deep neural networks were used for fault detection and monitoring non-linear processes. PCA is used as a data-driven method for industrial process monitoring [31]. Kernel PCA (KPCA) was proposed in [32] in order to extend PCA to nonlinear systems as most practical and real industrial systems are nonlinear. In order to deal with the streaming data, online KPCA was proposed in [33]. However, this method suffers from a high computational time. Therefore, reduced rank optimized KPCA (RR-KPCA) was introduced in [34] with reduced complexity to overcome this issue and was applied for monitoring an air quality monitoring network.

Some of these techniques are however not learning online and not dynamically adapting for streaming data, i.e., their models are trained on static datasets and then put in action without accounting for dynamically changing data concepts other than those existing in the training datasets. Although the others have been extended to learn online from streaming data, these techniques lack the interpretability needed to provide feedback for automatic corrective actions. Therefore, LAD was proposed for a first time as a fault detection and diagnosis tool for industrial systems in [35]. In [36], LAD was applied for the detection of faults in rotating machinery using vibration signals. LAD is applied in [37] to detect and diagnose faults in industrial chemical processes and provide patterns to build a decision model that diagnoses faults and explains the potential causes of these faults. A tool wear multiclass detection method based on LAD is proposed in [38] by deriving the information from machining process variables. In spite of its distinct interpretability, classical LAD doesn't process streaming data, and hence its resilience and sustainability are low when deploy it on the production. Therefore, one another main objective of this research is to build a sustainable model based on LAD to process data stream from real and practical industrial process.

CHAPTER 2 THESIS ORGANIZATION

This thesis is consisted of five chapters. Chapter 1 explain the problem being study. It also states the objective of this doctoral research and states its originality. It also includes a literature review to describe the state of art in the different problems under the study.

Chapters 3, 4 and 5 presents the three articles that are accomplished in this research.

Chapter 3 proposes a novel LAD ensemble technique (LAD-ENS) to accelerate LAD and make it able to process large volume of data. The performance of the technique is illustrated on different type of datasets. The proposed LAD-ENS demonstrates better performance in terms of computational time and quality of patterns over classical LAD models.

Chapter 4 presents a novel and innovative framework for LAD to process streaming data with concept drifts. The methodology that contains of three different modules is called Dynamic and Adaptive Logical Analysis of Streaming Data (DA-LASD). The online performance of DA-LASD is illustrated on different synthetic data streams which have different type of concept drifts. DA-LASD shows high sustainability through the different type of concept drifts.

Chapter 5 proposes a methodology to build DA-LASD model reinforced by feature engineering mechanism (DA-LASD-FE). DA-LASD-FE is proposed to be used in autonomous industrial process control by detecting faults and out-of-control states, and providing interpretable patterns which are responsible to guide the controlling systems to make automatic corrective actions. The proposed methodology is demonstrated through an aerospace case study of turbojet engine failure detection. The proposed DA-LASD-FE is able to accurately detect the faults and dynamically adapt to the concept drift in the data stream.

Chapter 7 presents a general discussion on the three articles in this thesis, followed by conclusion and future work directions to extend this research.

CHAPTER 3 ARTICLE 1: ACCELERATING LOGICAL ANALYSIS OF DATA USING AN ENSEMBLE-BASED TECHNIQUE

Authors: Osama Elfar, Soumaya Yacout, Hany Osman.

Journal: Engineering Letters (published) [39].

Abstract: Logical Analysis of Data (LAD) is a well-known classification technique that generates interpretable patterns with competitive accuracy. The challenge encountered in applying LAD comes from its long computational time, which makes it unsuitable for handling a large volume of data. In this paper, we propose a novel mechanism for developing an ensemble system for LAD (LAD-ENS) to improve its computational efficiency, while preserving its interpretability and promising accuracy. This new mechanism aims to maintain the explanatory power of classical LAD by combining the individual classifiers at the level of patterns. The developed ensemble system enables LAD to be run in parallel computing environments. Using datasets obtained from the UCI Machine Learning Repository, computational experiments are conducted to demonstrate the performance of LAD-ENS in terms of computational time, classification accuracy, and interpretability. Furthermore, we introduce the concept of the comprehensibility index in order to study the change in the explanatory power of LAD. In addition to achieving a statistically significant reduction in computational time, the developed LAD-ENS achieves competitive classification accuracies compared to two classical LAD approaches and five common machine learning algorithms.

3.1 Introduction

As an Artificial Intelligence (AI) application, a machine learning (ML) algorithm enhances the decision-making capabilities of various manufacturing and business systems. Therefore, achieving good performance of an ML algorithm is essential in ensuring efficient operation throughout these systems. Nevertheless, this goal becomes difficult to attain when dealing with a large volume of data, especially with the emergence of the Internet of Things (IoT) and Industry 4.0. IoT is the network of devices, buildings, machines, products, and other objects that are connected with sensors, software, and network connectivity, allowing these things to gather and interchange data [40]. Industry 4.0 aims to utilize advanced technologies in connectivity to gather all of the important data in manufacturing processes and products to develop analytical models through the means of ML techniques. One of the most important objectives of these models is predicting manufacturing performance and providing feedback so that corrective decisions can be made during the manufacturing processes. Therefore, the ML techniques that will be used in this context should have good explanatory power and interpretable results in order to provide root-cause analysis of certain phenomena to facilitate the decision-making process. However, well-known ML techniques, such as ensemble decision trees, support vector machines and neural network [41–43], show high accuracy in the literature, but do not have enough explanatory power and interpretable results. On the other hand, Logical Analysis of Data (LAD) is a classification approach that generates patterns containing structural knowledge that explain the hidden phenomena under study. As such, the patterns generated are the most useful results that indicate the candidate root causes behind the observed physical phenomena, and consequently, the best way to respond to them [44–46]. Additionally, LAD is used to develop regression models [23]. Because of these abilities, LAD is used in various fields such as medical, services, business, and manufacturing [46], [36–38, 47–52]. However, a relatively long computational time makes LAD unsuitable for manipulating data with large volumes.

Some techniques have been presented in the literature to implement LAD with the aim of enhancing computational efficiency. A polynomial algorithm was used in [6] to enumerate all LAD patterns with a selected degree to limit the number of features in the generated patterns. The degree of a pattern is the number of features it is constructed with. However, this technique still generates a high number of patterns, which is computationally expensive to handle when there are datasets with large amounts of features. In [7], instead of generating all possible patterns with specific characteristics, the column generation technique is used. In this framework, the master problem has the objective of building a LAD model with a maximum separation margin between the classes by generating patterns to enlarge this margin in subproblems. This column generation framework generates only one pattern in each iteration, which affects its computational efficiency. Therefore, a multi-pattern generation framework is developed to improve the efficiency of LAD column generation models by generating more than one pattern in each iteration [8]. However, none of these techniques is aimed at achieving a scalable LAD that handles a large volume of data.

With a continuous increase in the volume of data, the need to develop scalable ML algorithms that can handle large datasets has attracted researchers [9]. A taxonomy was proposed in [10] for ML techniques that can handle a large volume of data by improving the computational efficiency through either designing more efficient algorithms or relying on parallelism. The parallelism-based category reflects most of the state of the art in scalable ML algorithms [11]. Specifically, parallelized methods that make ML algorithms more scalable are classified into two sub-categories: (i) parallelized model/parameters: developing parallelized versions of learning algorithms by first dividing the learning model/parameters and then performing computations on each division concurrently, and (ii) parallelized data: partitioning input data vertically, horizontally, or even arbitrarily into manageable pieces, and then computing all data subsets simultaneously [10], [12]. The parallelized-data techniques are mainly ensemble based, such as random forests [13] and XGBoost [14]. In this paper, we introduce a novel technique that belongs to this sub-category: an ensemble LAD (LAD-ENS).

While building an ensemble LAD may seem like an intuitive way to improve both accuracy and computational time, the challenge is actually to build an ensemble system that preserves the explanatory power of LAD. Such explanatory power can guide the decision maker on keeping a current process under control, or directing that process to reach its maximum yield. For example, specifying the cutting conditions at which a machining process provides the desired surface roughness and the required output. The objective of the proposed technique is establishing a mechanism to combine the knowledge of individual LAD classifiers while preserving such explanatory power. In order to achieve this, LAD-ENS deals with the individual classifiers at the level of patterns, and the explanatory power is based on these patterns. Different ensemble systems in other ML techniques use voting mechanisms that significantly decrease the interpretability of the results, such as Random Forest and XGboost. Since the explanatory power of LAD is affected by the total number of generated patterns and their average degree [6], an index is introduced in this paper as a measure of interpretability.

The remainder of this chapter is organized as follows: Section 3.2 provides essential background on the LAD classification technique. Section 3.3 introduces the LAD ensemble system (LAD-ENS). Computational experiments and a comparison with classical LAD and other machine learning algorithms are conducted in Section 3.4. Section 3.5 concludes the work and discusses the future directions.

3.2 Logical Analysis of Data

LAD, originated in [53], is a classification technique that is characterized by extracting interpretable patterns from two–class or multi-class datasets [54]. The generated patterns are utilized as decision rules, used to classify unlabeled data into distinct classes [55]. LAD can also be used to develop regression models [23]. However, in this paper we only address the classification models. The LAD classification technique consists of three main steps: data binarization, pattern generation, and theory formation, as shown in Figure 3.1.

The data binarization step converts numerical and nominal data to binary data. Pattern generation is an essential step that extracts structural information in the form of patterns that characterize each different class in the binarized dataset. Many approaches are used for pattern generation, mostly based on enumeration, heuristics, or mixed integer linear programming (MILP) algorithms. In this research, we use cbmLAD software [56], in which patterns are generated by using the ant colony optimization technique. Theory formation is the final step that uses the generated patterns to create a discriminant function that is used as a classifier for new data [45].

In the case of a two-class dataset, the training set is $\Omega = \Omega^+ + \Omega^-$, which is formed from positive and negative subsets with n features. After the binarization and pattern generation steps, LAD forms the pattern set $\Pi = \{P_1, \dots, P_r\}$, where r is the number of generated patterns from the training set Ω. The pattern set has positive and negative patterns. Each P_i is a conjunction of d features, where $d \leq n$ is the pattern degree. Each P_i covers at least one observation from one of the subsets Ω^+ (or Ω^-) and no observations from the other set Ω^- (or Ω^+). Each P_i has characteristics that have been formed by the observations it covers.

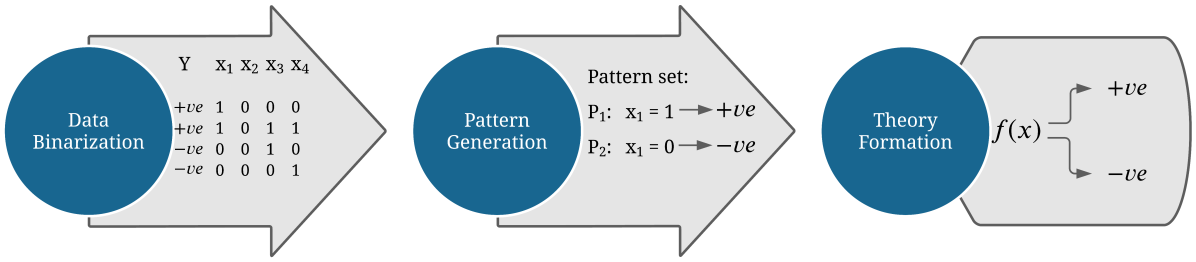


Figure 3.1 Framework of LAD.

These characteristics are illustrated as follows:

1. C_i is the class (the sign positive or negative) which is assigned to pattern P_i,
2. Q_i is the number of observations covered by pattern P_i,
3. δ_i is the degree which is the number of features constructing the pattern P_i,
4. π_i is the prevalence, i.e., $\pi_i = Q_i/|\Omega^+|$ in case of positive C_i, or $\pi_i = Q_i/|\Omega^-|$ in case of negative C_i,
5. ω_i is the weight of the pattern P_i, i.e., $\omega_i = Q_i / \sum^{r}_{\substack{j=1 \\ C_j = C_i}} Q_j$.

In the theory formation step, LAD uses the weights of patterns to formulate the discriminant function as follows:

$$f(x) = \sum_{i=1}^{r} \omega_i y_i(x) \tag{3.1}$$

,where

$$y_i(x) = \begin{cases} 1 & \text{if } C_i \text{ is } +ve \text{ AND } P_i \text{ covers } x \\ -1 & \text{if } C_i \text{ is } -ve \text{ AND } P_i \text{ covers } x \\ 0 & \text{if } P_i \text{ doesn't cover } x \end{cases} \tag{3.2}$$

It is obvious that $f(x) = [-1, 1]$. The new observation x is classified as a positive observation if $f(x) > 0$, negative observation if $f(x) < 0$, and unclassified if $f(x) = 0$.

In order to use LAD in multi-class classification problems, a decomposition approach is used to divide the multi-class problem into many two-class problems. This approach is applied in two different methods: One-Versus-All (OvA) and One-Versus-One (OvO) [57]. The OvA method divides the multi-class problem into k different two–class classification problems, where k is the number of classes. Each problem considers one class $i \in [1:k]$ as the positive class and all the remaining $(k-1)$ classes as the negative class. The OvO method divides the multi-class problem into $\binom{k}{2}$ two-class problems by considering each possible class pair as an individual two-class classification problem. Each problem considers class $i \in [1:k]$ as a positive class and $j \in [1:k]$ as a negative class, where $i \neq j$. In this research, we use the multi-class OvO method for handling multi-class datasets in our computational experiments.

3.3 Ensemble Logical Analysis of Data

Building an ensemble system is based on three pillars: (i) sampling the training data for individual basic classifiers, (ii) the training procedures of individual classifiers, (iii) the combination mechanism for merging individual classifiers and obtaining an ensemble model [58].

In this work, we use stratified sampling without a replacement to generate m different data subsets for training individual LAD classifiers, as shown in Figure 3.2. In the case of a two-class dataset, the training set $\Omega = \Omega^+ + \Omega^-$, with n features, is partitioned into m subsets, indexed by i. Each subset $\Omega_i = \Omega_i^+ + \Omega_i^-$, has the same n features of Ω. LAD is applied on each data subset in a parallel manner to generate m individual LAD classifiers. Each classifier i has an independent pattern set, $\Pi_i = \{P_{i1}, \dots, P_{ir_i}\}$, where r_i is the number of patterns generated from applying LAD on Ω_i. Each P_{ij}, where $j = 1, \dots,\ r_i$, is a positive (or negative) pattern that covers at least one observation from the set Ω_i^+ (or Ω_i^-) and does not cover any observation from the other set Ω_i^- (or Ω_i^+).

The characteristics of any pattern P_{lk}, $l \in [1 : m]$ and $k \in [1 : r_l]$, are based on the data subset Ω_l, and are as follows:

1. C_{lk} is the class positive (or negative) of pattern k generated from data subset l.,
2. Q_{lk} is the number of observations from Ω_l covered by pattern P_{lk}, ,
3. ω_{lk} is the weight of the pattern P_{lk}, i.e., $\omega_{lk} = Q_{lk} / \sum^{r_l}_{\substack{j=1 \\ C_lj = C_{lk}}} Q_{lj}$.

In order to build a combination mechanism that merges the knowledge of individual classifiers and preserves the explanatory power of LAD, we introduced a mechanism at the level of patterns before formulating the discriminant function of LAD. This mechanism updates the weight of each pattern to take into consideration the other patterns from other individual classifiers. The weight ω_{lk} of pattern P_{lk} is adjusted to a combined weight ω_{lk}^c, which is the ratio of the coverage of P_{lk} to the total coverage of all classifiers' patterns with the same class as P_{lk}.

$$\omega_{lk}^c = \frac{Q_{lk}}{\sum_{i=1}^{m} \sum^{r_i}_{\substack{j=1: \\ C_ij = C_{lk}}} Q_{ij}} \tag{3.3}$$

After updating the weights of the patterns, the mechanism formulates the LAD-ENS discriminant function using all patterns from all individual classifiers in a single function as follows:

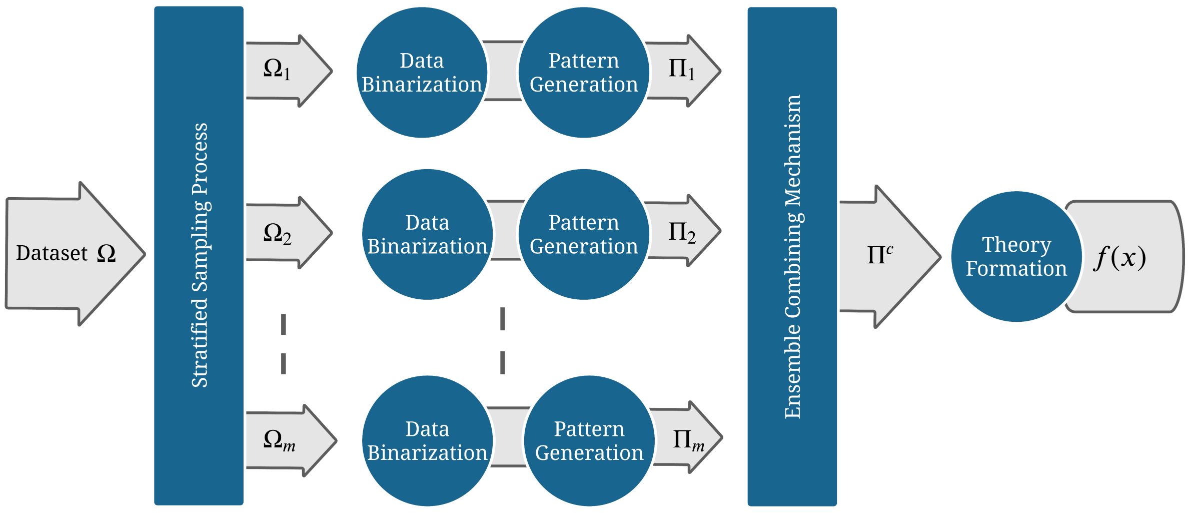


Figure 3.2 Framework of LAD-ENS.

$$f_{ensemble}(x) = \sum_{i=1}^{m} \sum_{j=1}^{r_i} \omega_{ij}^{c} y_{ij}(x) \tag{3.4}$$

,where

$$y_{ij}(x) = \begin{cases} 1 & \text{if } C_{ij} \text{ is } +ve \text{ AND } P_{ij} \text{ covers } x \\ -1 & \text{if } C_{ij} \text{ is } -ve \text{ AND } P_{ij} \text{ covers } x \\ 0 & \text{if } P_{ij} \text{ doesn't cover } x \end{cases} \tag{3.5}$$

Even though the voting mechanism provides good accuracy and results in the literature with other ML techniques, such as Random Forest and XGboost, this proposed mechanism preserves the explanatory power of LAD by using the patterns to directly predict the class instead of getting a vote from each individual classifier, which significantly reduces the interpretability.

To illustrate how a LAD-ENS model is developed compared to classical LAD, we consider, for example, a two–class dataset $\Omega = \Omega^+ + \Omega^-$ with two features (X and Y). The data is partitioned by using a stratified sampling approach without replacement into two data subsets $\Omega_1 = \Omega_1^+ + \Omega_1^-$ and $\Omega_2 = \Omega_2^+ + \Omega_2^-$. The dataset Ω, and the two data subsets Ω_1 and Ω_2 are illustrated in Figure 3.3. Running classical LAD on Ω generates a pattern set Π, as shown in Figure 3.4.c. Each data subset Ω_1 and Ω_2 was processed by LAD separately, providing two basic pattern sets $\Pi_1 = \{P_{11}, P_{12}\}$, and $\Pi_2 = \{P_{21}, P_{22}\}$, from Ω_1 and Ω_2, respectively, as shown in Figure 3.4.a and Figure 3.4.b. The characteristics of the patterns are determined and illustrated in Table 3.1.

In order to prepare these two sets of patterns Π_1 and Π_2 for a LAD-ENS discriminant function, the weights of the patterns are adjusted according to 3.3. For example, $\omega_{11} = 1$ is adjusted to $\omega^c_{11} = 0.3$. Other patterns' weights are illustrated in Table 3.1. In this table, the patterns of Π_2 have relatively higher combined weights than Π_1 because they are generated using a relatively larger sized data subset.

$$\omega^c_{11} = \frac{Q_{11}}{\sum_{i=1}^{2} \sum^{r_i}_{\substack{j=1: \\ C_ij=C_{11}}} Q_{ij}} = \frac{Q_{11}}{Q_{11} + Q_{21}} = \frac{30}{30+70} = 0.3$$

Afterwards, 3.4 and 3.5 will be used to formulate the ensemble discriminant functions $f_{ensemble}$ to classify new unlabeled observations.

Table 3.1 Characteristics of the patterns in Π_1 and Π_2 sets.

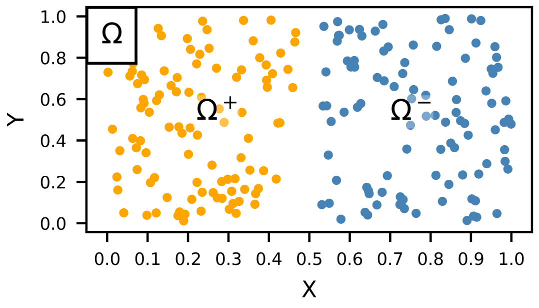
Pattern	Class	Q	ω	ω^c
P_{11}	Positive	30	1	0.3
P_{12}	Negative	30	1	0.3
P_{21}	Positive	70	1	0.7
P_{22}	Negative	70	1	0.7

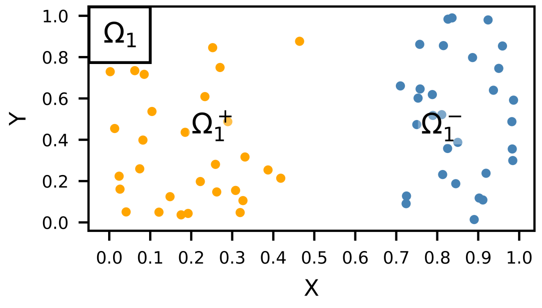
3.4 Computational experiments

In this section, the computational performance and the classification accuracy of the developed LAD-ENS are demonstrated using twenty datasets obtained from the UCI machine learning repository. Table 3.2 shows the descriptions of these datasets. The k–fold cross validation approach is used with k = 5 to average the results obtained from five different training data subsets and five corresponding testing data subsets. As our main objective is to reduce the computational time of LAD and solve classification problems with huge datasets, we compare between our proposed LAD-ENS and the classical LAD models in terms of computational time and classification accuracy. Additionally, the qualities of the patterns are compared in terms of the total number of patterns and their degree and prevalence. Moreover, a comprehensibility index (CI) is introduced in order to study the effect of LAD-ENS on the explanatory power of the generated patterns. Additionally, to evaluate the competitivity of LAD-ENS compared to other classifiers, LAD-ENS is compared to five ML techniques in terms of the classification accuracy.

The mechanism of the developed LAD-ENS allows the pattern generation step to be per-

Feature		Value
# of classes		2
# of attributes		2
# of observations of Ω	Ω^+	100
	Ω^+	100
# of observations of Ω_1	Ω_1^+	30
	Ω_1^-	30
# of observations of Ω_2	Ω_2^+	70
	Ω_2^-	70





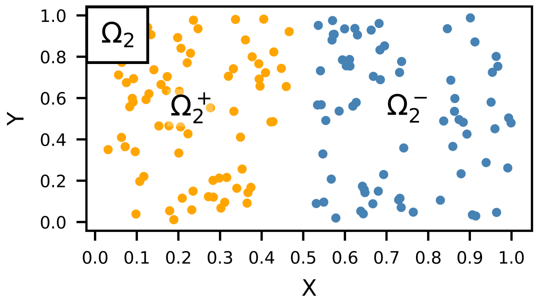


Figure 3.3 The generated dataset Ω, and two data subsets Ω_1 and Ω_2.

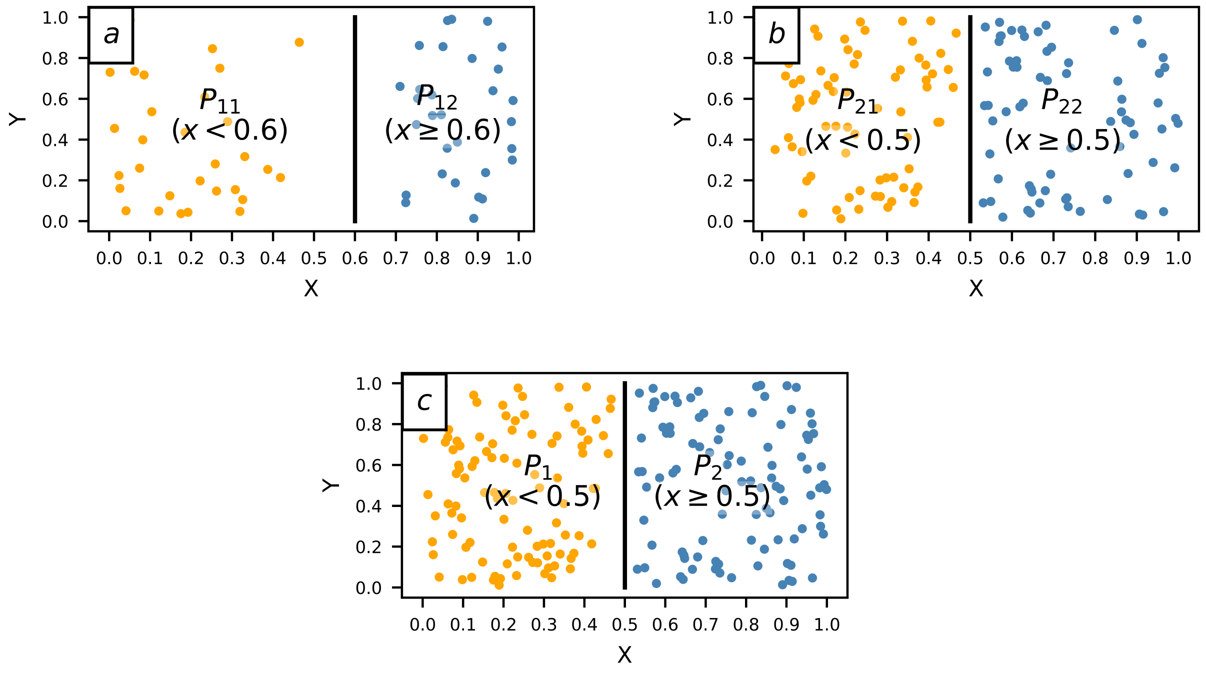


Figure 3.4 a: pattern set Pi_1, b: pattern set Pi_2, and c: the patterns generated by classical LAD.

Table 3.2 Characteristics of the patterns in Π_1 and Π_2 sets.

ID	Name	# of classes	# of observations	# of positive observations	# of negative observations	# of features	# of subsets
wbc	Wisconsin Breast Cancer	2	699	458	241	9	10
wpbc	Wisconsin Prognostic Breast Cancer	2	198	47	151	33	2
wdbc	Wisconsin Diagnostic Breast Cancer	2	569	212	357	30	4
hrt-c	Heart disease diagnosis - Cleveland	2	303	139	164	13	4
hrt-h	Heart disease diagnosis - Hungarian	2	294	106	188	10	4
hrt-s	Heart disease diagnosis – Switzerland	2	122	114	8	10	2
hrt-lb	Heart disease diagnosis - Long Beach VA	2	200	149	51	8	2
hpts	Hepatitis Domain	2	142	28	114	18	2
bld	BUPA liver disorders	2	325	200	125	6	2
pid	Pima Indians Diabetes	2	768	500	268	8	10
SPECTF	SPECTF Heart Data	2	267	212	55	44	2
SPECT	SPECT Heart Data	2	267	212	55	22	2
prks	Parkinsons Disease	2	195	147	48	22	2
SB	Spambase	2	4,601	2788	1813	56	10
WFR	Wall-Following Robot Navigation	4	5,456	–	–	24	10
LR	Letter Recognition	26	20,000	–	–	16	10
MG	MAGIC Gamma Telescope	2	19,020	12332	6688	10	65
MBP	MiniBooNE particle identification	2	130,065	93565	36499	50	500
SS	Skin Segmentation	2	245,057	194198	50859	3	90
CT[a]	Covertype	7	581,012	–	–	7	50

All datasets are available on the UCI machine learning repository: https://archive.ics.uci.edu/ml/index.php
[a] A sample of only 50,000 observations from Covertype dataset was used in the experiments to fit the available memory in compute Canada clusters.

formed in a parallel manner. In these computational experiments, we run LAD-ENS in parallel on Cedar cloud computing clusters provided by Compute Canada using 20 cores. Each core has a 2.1 GHz CPU. Utilizing cloud-computing systems allows LAD to handle large volumes of data. The number of subsets in LAD-ENS is different for each dataset, as shown in the last column of Table 3.2. This number is chosen empirically based on the size and the separability of the classes of the datasets to generate patterns from each subset in a reasonable computational time. We increase the number of individual classifiers if the data size is big, or if the separability is low.

Table 3.3 provides a comparison between LAD-ENS and two classical LAD models: cbmLAD [56] and the multi-pattern generation framework of LAD (MPG-LAD) [8]. The results of MPG-LAD are gathered from [8]. The reduction in processing time by using LAD-ENS is more than 75% in most of the datasets compared to the cbmLAD model, and more than 99% in all datasets compared to the MPG-LAD model. The n/a means that the classical LAD models were not able to solve the problems within 24 hours of running time. However, LAD-ENS was able to solve these problems in fewer than 10 minutes. Sampling datasets that have complex boundaries and low separability between the classes into different subsets enables LAD-ENS to handle them with fewer binary attributes and in less computational time. In terms of classification accuracy, LAD-ENS performs competitively for almost all datasets except wpbc, hrt-h, hrt-s, hrt-lb, and prks.

Table 3.3 Comparing the performance of LAD-ENS with classical LAD models.

Dataset	cbmLAD		MPG-LAD		LAD-ENS			
	Accur (%)	Time (sec.)	Accur (%)	Time (sec.)	Accur (%)	Time (sec.)	Time reduction % wrt:	
							cbmLAD	MPG-LAD
wbc	95.13	6.07	95	2283	94.44	0.98	83.86	99.96
wpbc	68.86	21.42	98	534	60.56	4.36	79.65	99.18
wdbc	95.81	41.9	99	1798	94.96	1.97	95.3	99.89
hrt-c	82.11	8.28	81	1438	83.11	1.54	81.4	99.89
hrt-h	68.61	5.08	78	2816	61.56	1.14	77.56	99.96
hrt-s	47.88	0.7	78	106	50	0.6	14.29	99.43
hrt-lb	56.68	1.71	68	3505	55.85	1.41	17.54	99.96
hpts	77.8	0.81	71	794	79.86	0.74	8.64	99.91
bld	70.52	11.24	63	3843	66.9	2.08	81.49	99.95
pid	75.88	140.21	70	16354	72.87	4.74	96.62	99.97
SPECTF	72.81	29.55	73	1595	64.81	2.12	92.83	99.87
SPECT	67.92	1.21	70	1809	74.81	0.93	23.14	99.95
prks	70.45	4.61	100	205	68.07	1.38	70.07	100
SB	91.99	16191	n/a	n/a	93.73	61.06	99.62	100
WFR	99.74	1582	n/a	n/a	98.83	16.01	98.99	100
LR	n/a	n/a	n/a	n/a	83.17	181.49	100	100
MG	n/a	n/a	n/a	n/a	84.42	35.37	100	100
MBP	n/a	n/a	n/a	n/a	88.88	276.83	100	100
SS	n/a	n/a	n/a	n/a	84.52	145.33	100	100
CT	n/a	n/a	n/a	n/a	71.52	612.64	100	100

In order to statistically evaluate the computational time and accuracy of LAD-ENS, the Friedman test is used to compare the results of the three LAD models. The Friedman test is a non-parametric statistical test used to detect differences between the values of various populations' means [59]. The test is applied to the computational time and the accuracy for the three LAD models, cbmLAD, MPG-LAD and LAD-ENS, in two phases. We do not include the last 7 datasets in the test, since cbmLAD and MPG-LAD failed to handle them. The null hypothesis of phase 1 states that all of those models have the same means of computational time (or accuracy), as follows:

$$H_0 = \mu_{cbmLAD} = \mu_{MPG-LAD} = \mu_{LAD-ENS}$$

The alternative hypothesis is formulated as follows:

$$H_a = \text{Not all means are equal}$$

To reject or accept H_0, the test statistic F_r and the calculated significance level p are calculated as given in [59] and compared to the significant values; $F_{critical} = 6.0$ and $p_{critical} = 0.05$. Tables 3.4 and 3.5 show phase 1 of the Friedman tests on computational time and accuracy, respectively. For computational time, H_0 is rejected with a high value of $F_r = 26.0$ which means that there are very significant differences in terms of the computational time between the three LAD models. Meanwhile, H_0 is accepted in the test for accuracy, which indicates that the three LAD models are similar in terms of accuracy.

Phase 2 of the Friedman test is aimed to distinguish the best model that will lead to a significantly lower computational time or significantly higher accuracy. Accordingly, pairwise comparisons were performed between LAD-ENS and each model j of the two other LAD models. The null and the alternative hypothesis are formulated as follows:

$$H_0 : \mu_{LAD-ENS} = \mu_j$$
$$H_a : \mu_{LAD-ENS} \neq \mu_j$$

Table 3.4 Freidman test on computational time performance.

Model	Phase 1		Phase 2		
	Sum of ranks (R)	Mean of ranks	*AD*	*AD > post-hoc*?	Significant?
cbmLAD	26	2	13	Yes	Yes
MPG-LAD	39	3	26	Yes	Yes
LAD-ENS	13	1			
($F_r = 26.0 > 6.0$, $p = 0.0 < 0.05$) $\rightarrow$ Rejecting H_0			$\alpha_f = 0.025$, $d_{\alpha_f} = 1.96$, *post-hoc value* $= 9.994$		

Table 3.5 Freidman test on accuracy performance.

Model	Phase 1		Phase 2		
	Sum of ranks (R)	Mean of ranks	*AD*	*AD > post-hoc*?	Significant?
cbmLAD	25	1.92	5	No	No
MPG-LAD	23	1.77	7	No	No
LAD-ENS	30	2.3			
($F_r = 2.0 < 6.0$, $p = 0.368 > 0.05$) $\rightarrow$ fail to reject H_0			$\alpha_f = 0.025$, $d_{\alpha_f} = 1.96$, *post-hoc value* $= 9.994$		

The absolute difference (AD) between the rank sums, $|R_{LAD-ENS} - R_j|$, is computed, where R_j is the rank summation for the model j. The null hypothesis is rejected if the AD exceeds the post-hoc value $d_{\alpha_f}\sqrt{Nk(k+1)/6}$. d_{α_f} is the $100(1-\alpha_f)^{th}$ of the standard normal distribution, α_f is the family-wise significant level, N is the number of datasets, and k is the number of models [59]. As shown in Table 3.4, the results of phase 2 declared that the LAD-ENS model significantly outperforms both cbmLAD and MPG-LAD models in terms of computational time. However, phase 2 declared that the accuracies are not significantly different between LAD-ENS and the other models, as shown in Table 3.5. Moreover, LAD-ENS was able to handle the last seven datasets, while cbmLAD only handled two and MPG-LAD was not able to handle any.

In order to study how LAD-ENS affects the interpretability power of LAD, we introduce the complexity measure shown in equation 3.6. This measure is computed for LAD-ENS and cbmLAD models for each dataset to give a quantitative measure of the model's complexity. Furthermore, a comprehensibility index (CI) is introduced in equation 3.7. Values of CI near 0 indicate a low explanatory power, whereas values near 1 indicate a high explanatory power.

$$Complexity = \frac{No.\ of\ Patterns \times Avg.\ degree\ of\ Pattern}{No.\ of\ Classes} \tag{3.6}$$

$$CI = \frac{1}{Complexity} \tag{3.7}$$

Table 3.6 shows a comparison between LAD-ENS and the cbmLAD model in terms of the number of patterns, average degree of the patterns, and the CI over the datasets that cbmLAD was able to solve.

LAD-ENS generated a number of patterns, up to 25% more in some datasets, compared to the number of patterns in cbmLAD. In other datasets, LAD-ENS produced a number of patterns that was less than that of cbmLAD by 6.6% to 37%. On the other hand, the average degree of patterns was reduced in most datasets, which led to an increase in the CI of LAD-ENS over classical LAD. The sampling process that is used to extract the data subsets with better separability between the classes explains this decrease in the average degree of the patterns. Therefore, a small number of patterns are generated from each data subset with low degrees. This resulted in developing a LAD-ENS model with a reasonable number of patterns and with low degrees compared to the classical LAD model. For a WFR dataset, this reduction in the average degree of patterns did not prevent an increase in the complexity of the LAD-ENS model. This is due the number of patterns that increases significantly, 3.8 times, compared to the number of patterns generated by the classical cbmLAD model. Nevertheless, LAD-ENS has an efficient computational time of 16 seconds compared to 1582 seconds of cbmLAD. However, by eliminating LAD-ENS patterns that have homogeneity less than the 75th percentile, the number of patterns is considered only 5% more than the number of patterns in cbmLAD. Moreover, the average degree is reduced to 2.72, resulting in a CI of 40.84×10^{-3} which is better than the CI of cbmLAD. Removing these low homogeneity patterns does not affect the accuracy of LAD-ENS on a WFR dataset, as will be discussed further in the paper.

The main limitation of LAD-ENS is that the homogeneity of some patterns can change when

Table 3.6 Comparing the patterns generated by LAD-ENS and cbmLAD.

Dataset	LAD-ENS			cbmLAD		
	# of patterns	Avg. Degree	CI(E-3)	# of patterns	Avg. Degree	CI(E-3)
wbc	35	2.24	25.52	28	4.95	14.44
wpbc	24	7.5	11.12	23	10.41	8.36
wdbc	25	5.16	15.5	22	9.25	9.82
hrt-c	46	4.13	10.52	54	5.36	6.9
hrt-h	40	3.83	13.06	49	4.72	8.64
hrt-s	10	3.96	50.5	11	4.75	38.28
hrt-lb	44	3.29	13.82	44	3.41	13.32
hpts	10	3.96	50.5	16	4.99	25.06
bld	70	4.46	6.4	75	4.36	6.12
pid	127	4.38	3.6	158	5.71	2.22
SPECTF	19	12.1	8.7	28	15.86	4.5
SPECT	35	4.8	11.9	33	5.42	11.18
prks	13	3.16	48.68	17	6.48	18.16
SB	307	12.63	0.52	283	15.26	0.46
WFR	129	4.89	6.36	34	3.26	36.08
WFRa	36	2.72	40.84	34	3.26	36.08

scanning the entire dataset. As each pattern was generated on one chunk of data, a pattern might cover observations that belong to opposite classes from other chunks. In the case of a positive (negative) pattern, homogeneity is the proportion of the covered positive (negative) observations over all of the covered observations from positive and negative classes. The homogeneity of a pattern is an important characteristic, since it refers to the confidence of a positive/negative pattern belonging to a positive/negative class.

In order to analyze the quality of patterns in terms of homogeneity and prevalence, all observations are scanned by each pattern for every dataset. Homogeneity and prevalence are calculated for each pattern. Empirical cumulative distribution functions and box plots of the homogeneity are shown in Figure 3.5 and Figure 3.6, respectively. The means of homogeneity are illustrated with red triangles in Figure 3.6. These figures illustrate that 25% or fewer patterns have homogeneity that is lower than 70% over most datasets. The figures also show that most of the generated patterns have a very high homogeneity/confidence of 0.8 or more.

The results obtained with the proposed LAD-ENS mechanism are encouraging, since they demonstrate that the patterns generated by LAD-ENS could be used to guide a decision maker when monitoring a business or industrial process of interest. For example, controlling parameters of a manufacturing process such as the operating conditions and the measurements that keep the process under control, and predicting future events such as failure, alarm,

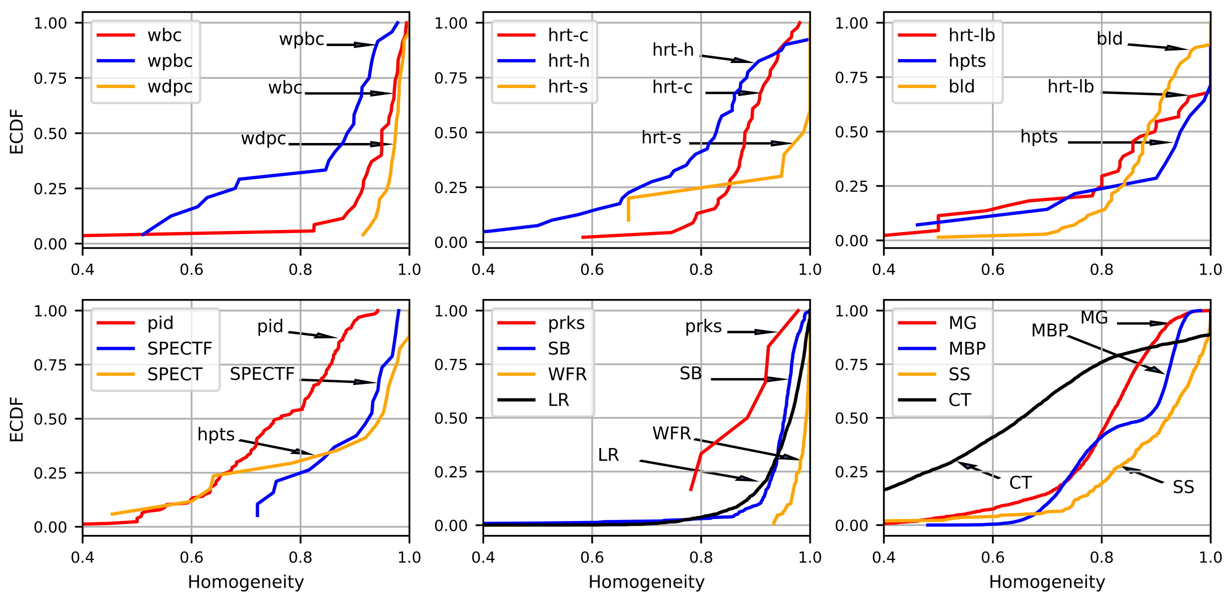


Figure 3.5 Empirical cumulative distribution functions for the homogeneity of LAD-ENS patterns.

and fraudulent cases. In addition, patterns with a high confidence are also useful in monitoring the status of a patient regarding a specific disease, and in detecting network intruders and spam emails.

In order to investigate whether patterns with low homogeneity negatively affect classification accuracy, the patterns with homogeneity lower than the 25th, 50th, 75th and 85th percentiles were eliminated when forming the discriminant function. After each elimination, the testing accuracy is computed using a new set of patterns. The accuracy of the results is shown in Figure 3.7. The results demonstrate that accuracy is not negatively affected by keeping low homogeneity patterns. In most of the datasets, testing accuracy decreases by discarding patterns that have a homogeneity lower than the 75th percentiles. Therefore, low homogeneity patterns could be eliminated to enhance the CI of the model, as we did earlier for the WFR dataset.

Additionally, the prevalence of patterns is illustrated in Table 3.7 and compared with cbmLAD for the datasets that cbmLAD was able to solve. The patterns of LAD-ENS show a high average prevalence compared with the patterns of cbmLAD. Overall, while the purity of the patterns might be lost in LAD-ENS, it provides better prevalence and explanatory power.

Finally, Table 3.8 provides a comparison between LAD-ENS with accuracy results of other

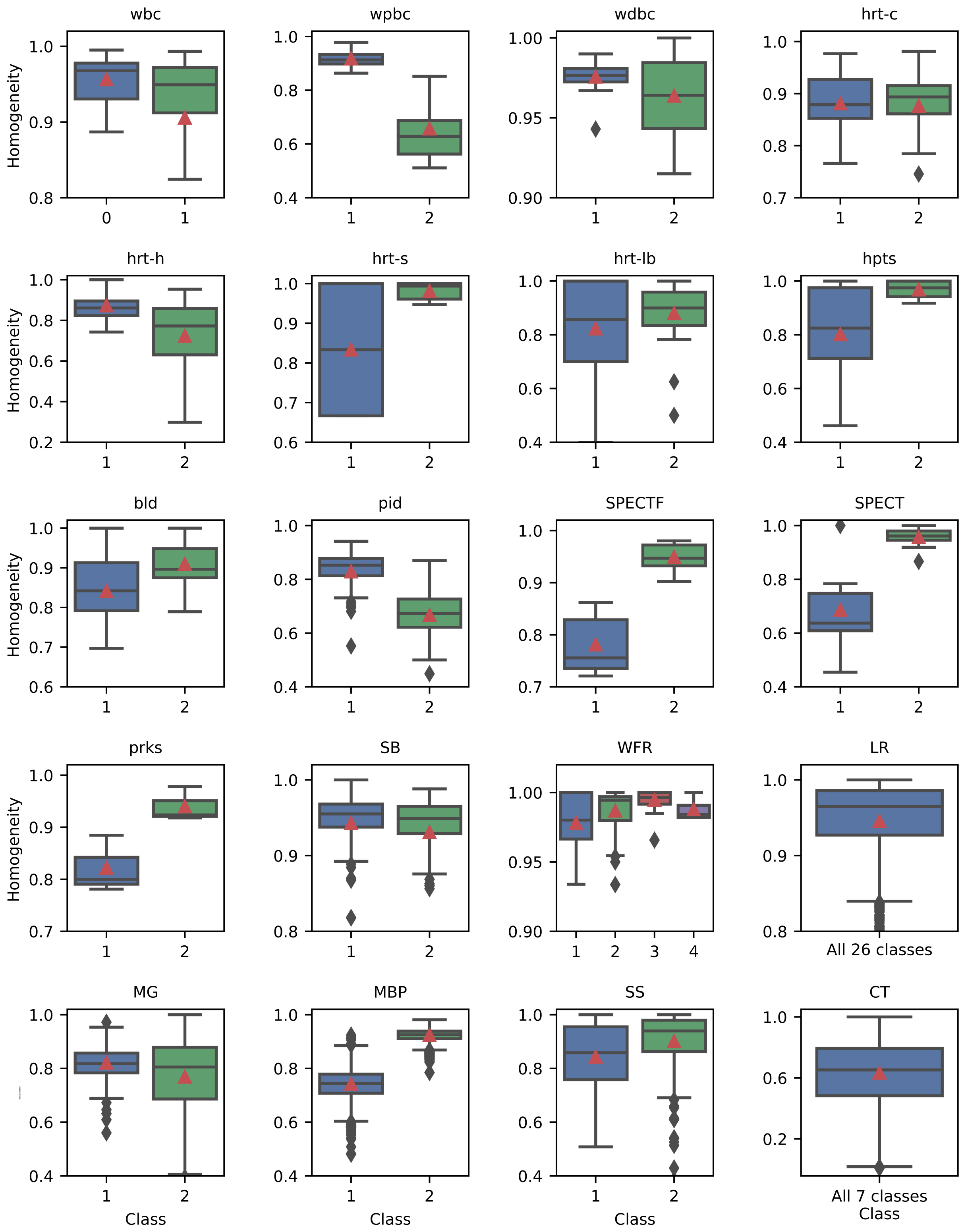


Figure 3.6 Box plots of the homogeneity of LAD-ENS patterns.

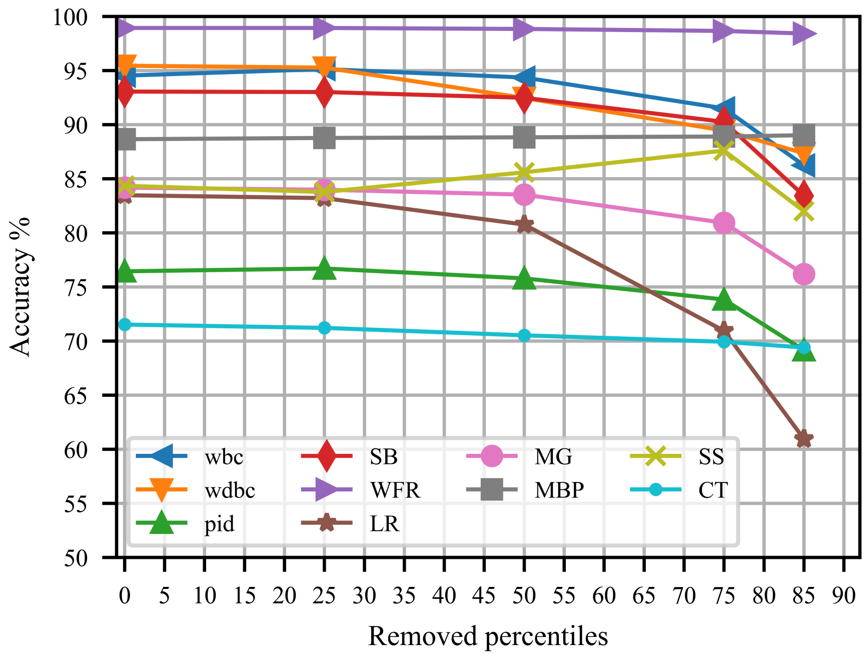


Figure 3.7 Testing accuracy of LAD-ENS after discarding patterns with homogeneity lower than different percentiles.

Table 3.7 The prevalence of patterns generated by LAD-ENS and cbmLAD.

Dataset	Average prevalence	
	LAD-ENS	cbmLAD
wbc	0.79	0.42
wpbc	0.44	0.33
wdbc	0.82	0.75
hrt-c	0.39	0.22
hrt-h	0.29	0.15
hrt-s	0.49	0.45
hrt-lb	0.12	0.09
hpts	0.58	0.477
bld	0.13	0.08
pid	0.29	0.07
SPECTF	0.56	0.51
SPECT	0.25	0.11
prks	0.6	0.5
SB	0.37	0.15
WFR	0.75	0.68

well-known machine learning techniques, as summarized in [8], such as support vector machines (SVM) [42], decision tree (J48) [60], random forest (RF) [13], multilayer perceptron (NN) [43] and logistic regression (LR). It can be observed that LAD-ENS provides competitive classification performance. This is obvious for the wbc, wdbc, hrt-c, bld, and pid datasets. Although SVM outperforms LAD-ENS in some datasets, SVM lacks the capability of providing interpretable results. As explained before, LAD-ENS provides the interpretability power that guides a decision maker when monitoring a business or industrial process of interest.

Table 3.8 Comparing the accuracy of LAD-ENS with other machine learning techniques.

Dataset	LAD-ENS	SVM	J48	RF	NN	LR
wbc	94±4	97±2	94±2	97±1	96±2	96±2
wpbc	60±7	77±2	75±5	80±4	77±5	80±5
wdbc	95±3	97±1	93±2	96±2	97±1	97±2
hrt-c	83±4	84±5	78±5	83±5	79±5	83±5
hrt-h	61±2	81±4	79±4	80±4	78±5	83±5
hrt-s	50±4	94±2	93±2	93±3	89±6	92±4
hrt-lb	55±5	75±1	72±5	75±4	69±7	74±4
hpts	79±4	87±5	82±6	87±5	81±6	85±6
bld	66±5	58±0	62±5	73±6	68±6	69±5
pid	72±6	77±3	74±3	76±3	75±3	77±3
SPECTF	74±5	79±0	78±5	81±3	77±5	79±4
SPECT	64±4	83±4	80±3	82±4	80±4	82±5
prks	68±8	87±4	83±7	91±5	92±5	85±6

3.5 Conclusion and future work

In this paper, we have developed an ensemble LAD system called LAD-ENS to enhance computation time and to train, test and classify large volumes of data. This ensemble system was built based on stratified sampling without a replacement technique, in addition to a proposed combining mechanism that integrates all patterns that are provided by the individual LAD classifiers. This mechanism combines the knowledge at the level of patterns and then formulates a new ensemble discriminant function. This mechanism preserves the explanatory power of LAD patterns and does not reduce their interpretability like the voting mechanism does. By means of this system, we successfully ran LAD-ENS on cloud computing clusters. The LAD-ENS system was evaluated in terms of computational time, accuracy, pattern quality and comprehensibility. The statistical Friedman tests revealed that LAD-ENS significantly outperforms classical LAD models in terms of computational performance. Moreover, Friedman tests revealed that very competitive accuracy is obtained. Although the patterns may lose their purity, the patterns of LAD-ENS have lower degrees than those of classical LAD, which enhanced the comprehensibility index. The computational experiments show that if the sampling process is not able to relax complex boundaries of a dataset enough, the number of generated patterns in LAD-ENS will increase significantly and reduce the explanatory power of the patterns. However, by eliminating low homogeneity patterns, the explanatory power will improve significantly, and accuracy performance will not be affected. In general, the proposed LAD-ENS demonstrates better performance in terms of computational time and quality of patterns over classical LAD models.

The novel research on ensemble LAD systems introduced in this paper could be extended into many different directions. One of these directions would be to use various sampling methods to enhance the quality of the data subsets provided to the pattern generation processes. This feature level could be another direction, focusing mainly on the features of original data in order to select appropriate subsets, or sample the features in subsets and provide them to the pattern generation processes with an aim to enhance accuracy. The combining mechanism level is another direction, which would focus mainly on enhancing the combining process and selecting the most appropriate patterns. This research direction could enhance the explanatory power by reducing the number of patterns.

CHAPTER 4 ARTICLE 2: DYNAMIC AND ADAPTIVE LOGICAL ANALYSIS OF STREAMING DATA WITH CONCEPT DRIFTS

Authors: Osama Elfar, Hany Osman, Soumaya Yacout, Adham Mohamed.

Journal: Knowledge and information systems (submitted).

Abstract: The nature of the data collected in the recent intelligent systems is evolving from the classical static datasets to continuous flow of data streams that may include concept drifts over time. On the other hand, since such intelligent systems rely mainly on machine learning in order to detect, characterize and solve problems, machine learning techniques have to adapt to this change in the nature of data they deal with. With this motivation, we aim in this research to provide an adaptation of the Logical Analysis of Data technique to become dynamic and adaptive to be able to handle continuous data streams that include concept drifts – we call it Dynamic and Adaptive Logical Analysis of Streaming Data, or DA-LASD. Moreover, DA-LASD is devised in order to have the ability for handling imbalanced streaming data. The proposed framework is built of different modules that dynamically modify the characteristics of the LAD patterns; eliminate decayed and inefficient patterns; and generate new ones if needed. This continuously updates the LAD classifier to adapt to the changes in the data streams. The DA-LASD is tested on several synthetic datasets covering different types of concept drifts. The results show how the proposed framework dynamically adapts the model, and successfully improves any performance measures that start to decline. Moreover, it proves quite competitive in terms of the classification accuracy, compared to other machine learning techniques that handle streaming data. In addition to its distinctive interpretability power, the DA-LASD is a promising framework for a diverse range of applications where high accuracy and interpretability are both essential.

4.1 Nomenclature

P	is a set of patterns of Logical Analysis of Data model
p_i	is a pattern in P. $p_i \in P$, where $i = [1, \ldots, \lvert P \rvert]$
S	is a true-labeled data stream
x_j	is an observation in S. $x_j \in S$, where $j = [1, \ldots, \lvert S \rvert]$
l_j^t	is the true label of observation x_j
C_j	is a subset of P, which is covering observation x_j. $C_j \subset P$
l_j^p	is the predicted label of observation x_j by the model
q_{ij}	is the coverage of pattern p_i up to observation x_j
$\bar{q_{ij}}$	is the opposite coverage of pattern p_i up to observation x_j
h_{ij}	is the homogeneity of pattern p_i up to observation x_j
δ_{ij}	is the coverage index of pattern p_i up to observation x_j
α_p	is the decay factor of the coverage index δ of the patterns
$\grave{q}_{ij}$	is the normalized coverage of pattern p_i up to observation x_j
$\bar{\grave{q}}_{ij}$	is the normalized coverage of pattern p_i up to observation x_j
$\grave{h}_{ij}$	is the normalized homogeneity of pattern p_i up to observation x_j
$\hat{h}_{ij}$	is the protected homogeneity of pattern p_i up to observation x_j
A_j	is the model accuracy up to observation x_j
AA_j	is the model arithmetic accuracy up to observation x_j
GA_j	is the model geometric accuracy up to observation x_j
DA_j	is the model decayed accuracy up to observation x_j
DAA_j	is the model decayed arithmetic accuracy up to observation x_j
DGA_j	is the model decayed geometric accuracy up to observation x_j
α_a	is the decay factor of the decayed accuracies

4.2 Introduction

The significant advancements in data collection and storage have helped the development and emergence of Machine Learning (ML) techniques as powerful tools in making decisions and better improving our systems; and has even given birth to a large variety of new systems. Yet, the development and evolution does never stop; systems continue to evolve, and so does the nature of the data collected from such systems [61]. Social media platforms, cloud systems, and the interconnections of Internet of Things (IoT) in many different applications are some examples of this evolution. In such systems, data now comes in the form of continuous streams of large amounts, instead of the traditional historical and static datasets that have long been used to build and train ML models. Such continuous steams may have concept drifts over time due to changes in the observed phenomenon which require adaptation for the trained ML models. Therefore, ML techniques have to cope with such evolution by developing the ability to dynamically handle streaming data as illustrated in Figure 4.1, and many have already done, as it will be reviewed later in this section.

However, while the vast majority of ML techniques technically try to provide the same kinds of answers to decision makers, they still differ among each other in certain features that some may provide while others do not. These subtle differences can be quite crucial in applications where certain features are essential. To make this statement clearer with the case relevant to this research, the Logical Analysis of Data (LAD) [45] features the powerful ability of interpretability in classification problems [44–46] – whether they are two-class [45] or multi-class [48] problems. Such powerful ability has made the LAD quite useful in applications where interpretability is highly important. Examples of these applications include financial [62], medical [46], aviation [63], and industrial [37], to name a few. More specifically, applications in industrial systems where interpretability is essential to better understand, and hence improve, such systems and their processes include manufacturing and quality systems [37], and condition based maintenance systems [56].

In the light of the forementioned evolution of systems, industrial and quality systems have also experienced such changes. With concepts such as Industry 4.0 [64] and Quality 4.0 [65], such systems will be collecting continuous streams of large amounts of data. And while these systems require high-interpretability, and hence their traditional versions have benefited from such technique as the LAD, Industry 4.0 and Quality 4.0, among other types of systems, would also significantly benefit from a version of LAD that has the ability to handle streaming data while preserving its distinct power of interpretability. Such version of LAD has not yet been developed, and it is our proposal in this paper.

One of the first ML techniques to be modified to handle streaming data is Decision Trees in Hoeffding Tree algorithm [15]. This algorithm is based on splitting the leaves of the constructed tree by scanning the streaming observations in order to incorporate the new information into the model. The authors then, in the same publication, proposed the Very Fast Decision Tree (VFDT) as an improved version of the algorithm in order to speed up the process of incorporating the new information. This is achieved by computing the best leaf splitting when a minimum number of observations arrive, in addition to deactivating the least promising nodes to decrease the computational memory and processing time. As an extension to the VFDT, the Concept-Adapting Very Fast Decision Tree (CVFDT) accounts for data concept drift by using the sliding window technique [16].

Other adaptations to ML techniques include the Lazy Learning algorithm [17] applied to

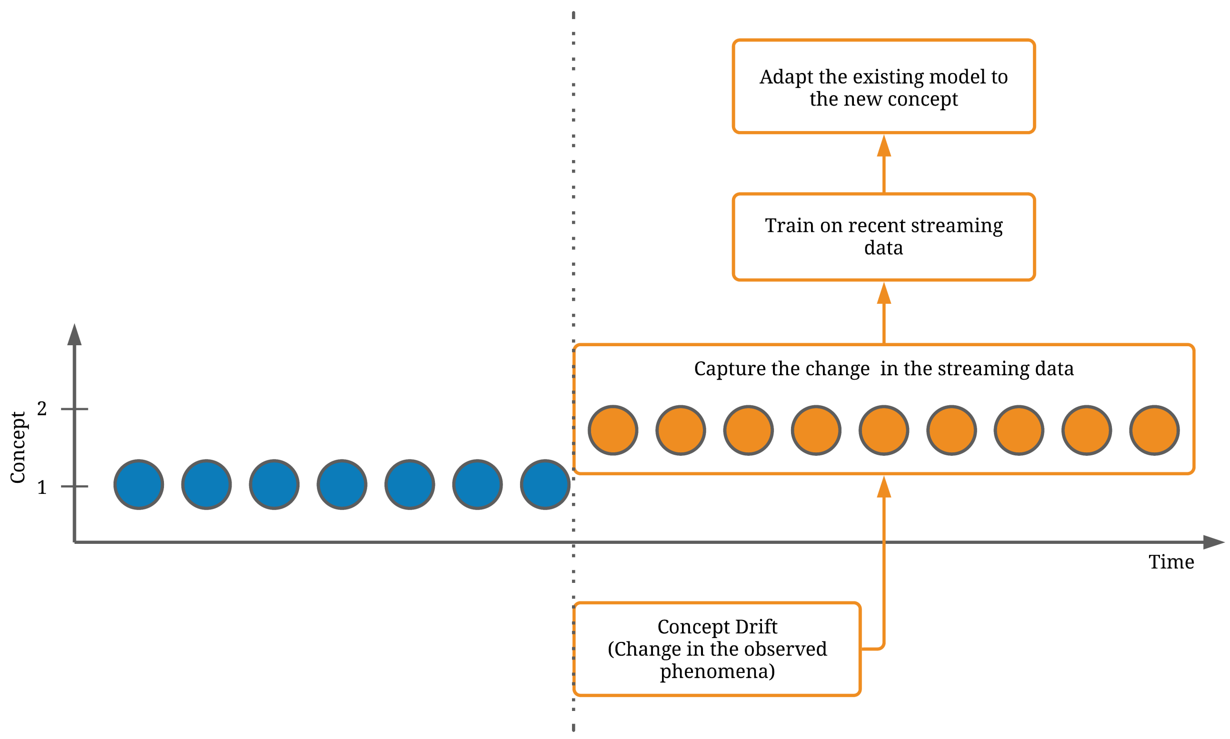

Adapt the existing model to the new concept
Train on recent streaming data
Capture the change in the streaming data
Concept
2
1
Time
Concept Drift (Change in the observed phenomena)

Figure 4.1 Adapting ML model after detecting a concept drift in a streaming data.

the k-nearest neighbors method. This algorithm uses a sliding window as the search space used in determining the k-nearest neighbors to a new, unclassified observation, and as the window slides, the method naturally takes into account any data concept drift that may exist. More generic frameworks that can be applied to different ML techniques, to handle streaming data, include reservoir sampling [18] and ensemble methods [19, 20]. Reservoir sampling is based on maintaining a sampling reservoir from the data stream, and frequently updating this reservoir, and accordingly the model. On the other hand, ensemble methods are based on building an ensemble model of different classifiers trained on different chunks from the data stream. As the data stream flows and more classifiers are built, the ensemble model is updated based on the accuracy of its classifiers. This ensures that the model is tuned dynamically to optimize the accuracy according to the recent part of the data stream.

All these techniques, despite being quite helpful in many applications, still do not provide the interpretability needed for many other systems – or at least do not maintain this interpretability as their models grow in size. The LAD, on the other hand, has this ability of providing and maintaining its interpretability for real world problem sizes. This is because the patterns it generates are considered unique components, and hence, they can be flexibly modified or eliminated solely without the need to change the entire model. This flexibility makes the LAD quite promising for handling streaming data. This ability to handle streaming data, along with the high interpretability the LAD already has, will make it a powerful tool for systems such as Industry 4.0 and Quality 4.0 that require both.

This paper proposes a Dynamic and Adaptive Logical Analysis of Streaming Data (DA-LASD) framework for handling streaming data while maintaining all the interpretability powers of the LAD. Through advanced modification of patterns characteristics, while combining concepts such as reservoir sampling and ensemble models, the proposed method builds a dynamic classification model for streaming data, considering concept drifts that may be present. It dynamically adapts to the most recent information present in the data stream by continuously updating the model through modifying the characteristics of existing patterns, generating new patterns, and eliminating decayed and/or inefficient patterns. This allows DA-LASD model to adapt dynamically, autonomously, and continuously through infinite data streams. In addition, the advanced modifications of patterns characteristics allowed DA-LASD model to have the ability for handling imbalanced streaming data which is a difficult challenge in classification problems of data streams [66].

This paper is organized as follows. Section 4.3 presents the framework of DA-LASD model and the algorithm of the adaptive and dynamic mechanism. Section 4.4 presents how DA-LASD handles imbalanced data streams. Section 4.5 demonstrates the proposed framework through experimentation. Finally, section 4.6 concludes the paper.

4.3 The framework of DA-LASD

4.3.1 Classical Logical Analysis of Data

LAD is a data mining and classification approach that generates prescriptive patterns containing structural knowledge that explains the hidden phenomena under study. It consists of three main steps: (1) data binarization, which converts numerical and nominal data to binary data; (2) pattern generation, which extracts the structural information in the form of patterns characterizing every class, and hence, every generated pattern has a class label; and (3) theory formation, which creates a discriminant function, based on the patterns' characteristics, which is then used to classify new observations.

The main characteristics of every generated pattern are illustrated as follows [8]: (1) c is the pattern class; (2) q is the coverage which is the number of observations covered by the pattern and have the same class label as the pattern class; (3) $\bar{q}$ is the opposite coverage which is the number of observations covered by the pattern but have a class label opposite to the pattern's class; (4) h is the homogeneity which is the ratio of the pattern's coverage with respect to the summation of its coverage and opposite coverage; (5) *prevalence*, which is the ratio of the pattern's coverage with respect to the total number of observations having the class label as the pattern's class; and (6) *degree*, which is the number of features defining the pattern.

The challenge in the case of streaming data is that the values of many of these characteristics mentioned above are not constant values. This is because they are mainly based on how many observations a pattern covers, and these observations are being streamed in over time. Therefore, these values should be continuously updated in order to reflect the characteristics of the patterns at every part of the data stream. To account for this, various modifications are introduced to the existing characteristics as it will be further discussed along with the proposed framework.

4.3.2 Devising DA-LASD framework

DA-LASD is defined as an online classification model that is constructed by extracting hidden patterns in a data stream by implementing the theory of LAD. In most data streams, the data evolves and has concept drifts over time, which means that various statistical properties, such as correlations between features and between features and class labels, may change over time. The goal is to keep the model dynamically and automatically updated to adapt to any concept drift in the data.

As Figure 4.2 shows, there are two independent input streams of data: a stream of unlabeled data and a stream of true-labeled data. The idea is to use the stream of true-labeled data to improve the classification accuracy of the LAD Classification Model (LAD-CM) for the flowing stream of unlabeled data. This stream of true-labeled data could be there intentionally to assist the model, e.g., having a human agent to manually label a certain number of observations such that these true-labeled observations are used to update the model, and hence, improve its accuracy. However, it could also be there as newly available historical data, e.g., the today unknown status of a certain stock in the stock market – which is the goal of many ML algorithms to predict today – will tomorrow be a true label that can be used as feedback to improve the classification accuracy of the model.

DA-LASD is a framework to automatically and dynamically do the following: updating the LAD classification model by incorporating the available stream of true-labeled data, with all the recent information and concept drift it contains, in order to improve the classification accuracy of this model for the flowing stream of unlabeled data. Updating the model is done through three main actions: (1) Updating, dynamically and continuously, the characteristics of the existing patterns; (2) Eliminating decayed and/or inefficient patterns that do not describe the current concept in the data stream; and (3) Extracting new patterns to describe any new information or concepts in the data stream. Therefore, in addition to the Module I, LAD Classification Model (LAD-CM), these three actions are done in the two other modules of the framework. Module II, the Dynamic Upgrading of Existing Patterns (DUEP), is responsible for the first and second actions; updating the characteristics of the existing patterns and eliminating the decayed and/or inefficient ones. Module III, Extracting New Patterns (ENP), as the name implies, is responsible for the third action. These three modules are illustrated in Figure 4.2.

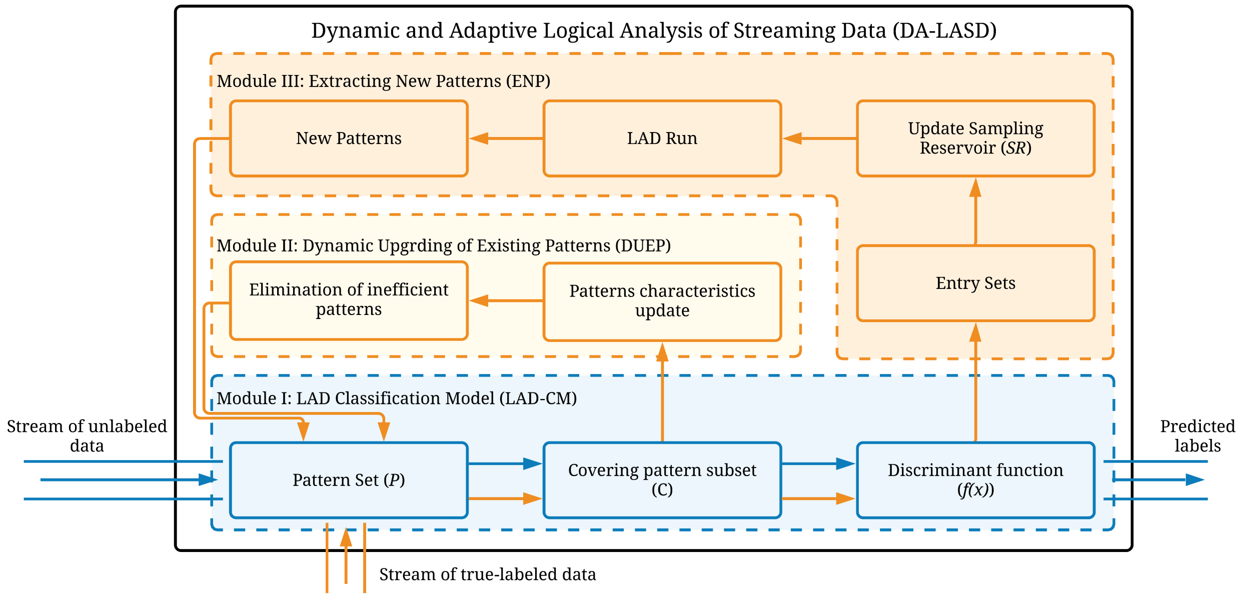

Dynamic and Adaptive Logical Analysis of Streaming Data (DA-LASD)
Module III: Extracting New Patterns (ENP)
New Patterns
LAD Run
Update Sampling Reservoir (SR)
Entry Sets
Module II: Dynamic Upgrding of Existing Patterns (DUEP)
Elimination of inefficient patterns
Patterns characteristics update
Module I: LAD Classification Model (LAD-CM)
Stream of unlabeled data
Pattern Set (P)
Covering pattern subset (C)
Discriminant function (f(x))
Predicted labels
Stream of true-labeled data

Figure 4.2 Schematic of DA-LASD.

Module I: LAD Classification Model (LAD-CM) module

The LAD-CM is the classifier module of the DA-LASD and is responsible for predicting classification labels for input observations. As a classical LAD classifier, it uses a set of patterns (P) to establish the LAD discriminant function, which then generates the predicted labels. The P set can be initialized using either historical pre-labeled data or the first chunk of the true-labeled data stream. This pattern set is then dynamically updated using the other two modules of the DA-LASD.

As Figure 4.3 illustrates, the LAD-CM module receives two independent input streams: a stream of unlabeled data and a stream of true-labeled data. While the model is primarily running to classify the stream of unlabeled data, it is simultaneously used to process the stream of true-labeled data in order to use the model parameters during such processing as input to the other two modules, which in return updates the model for continuous improvement.

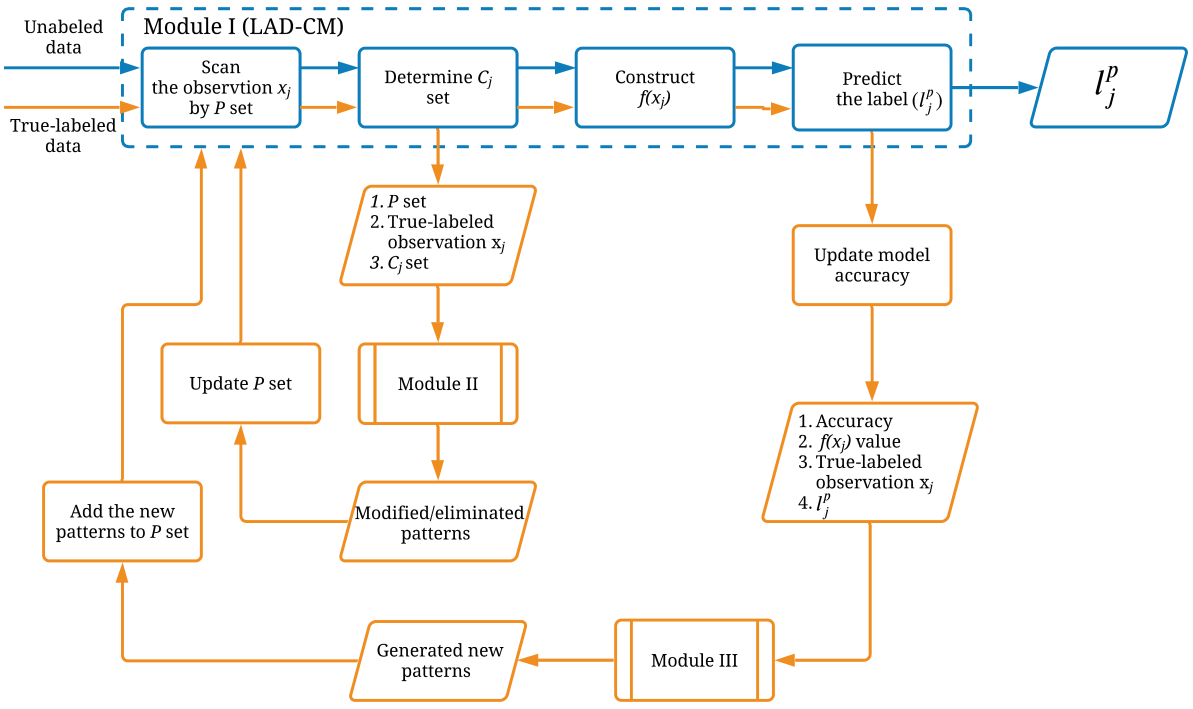

Module I (LAD-CM)
Unabeled data
True-labeled data
Scan the observtion x_j by P set
Determine C_j set
Construct $f(x_j)$
Predict the label (l_j^p)
l_j^p
1. P set 2. True-labeled observation x_j 3. C_j set
Update model accuracy
Update P set
Module II
1. Accuracy 2. $f(x_j)$ value 3. True-labeled observation x_j 4. l_j^p
Add the new patterns to P set
Modified/eliminated patterns
Generated new patterns
Module III

Figure 4.3 Flowchart of Module I: LAD Classification Model (LAD-CM).

Every observation in each of these two streams is scanned by P set, and the subset of covering patterns (C) is obtained for each. A discriminant function ($f(x)$) is then constructed, and the class label (l^p) is predicted. Specific model parameters are collected during the processing of the true-labeled stream (S) to be then sent to the other two modules. After processing an observation $x_j \in S$, Module II, DUEP, is sent the P, the C_j, and the true-labeled observation x_j with its true label (l_j^t), in order to update the characteristics of the patterns in P and eliminate decayed and/or inefficient ones, if any. Module III, ENP, is sent the current model accuracy, the $f(x_j)$ value, the true-labeled observation x_j with its l_j^t, and its predicted label l_j^p, as inputs to its mechanism of generating new patterns. These new patterns will then be added to the P.

Module II: Dynamic Updating of Existing Patterns (DUEP) module

The DUEP module is responsible for updating the characteristics of the patterns in P and eliminating any decayed and/or inefficient ones.

Out of the forementioned pattern characteristics, two main qualities are quite crucial when it comes to handling streaming data. The first one is the ability of a pattern to maintain correct coverage, i.e., covering observations that are truly belongs to its own class, and not the other class(es). This can be indicated by the pattern's homogeneity h based on its coverage q and opposite coverage $\hat{q}$ characteristics. At the time of processing an observation $x_j \in S$, the coverage q_ij, opposite coverage $\hat{q}_{ij}$, and homogeneity h_{ij} of a pattern $p_i \in P$, where $i = [1, \dots, |P|]$, can be calculated as follows:

$$q_{ij} = \begin{cases} q_{i(j-1)} + 1 & , c_i = l^t_j \text{ and } p_i \in C_j \\ q_{i(j-1)} & , \text{otherwise} \end{cases} \tag{4.1}$$

$$\bar{q}_{ij} = \begin{cases} \bar{q}_{i(j-1)} + 1 & , c_i \neq l^t_j \text{ and } p_i \in C_j \\ \bar{q}_{i(j-1)} & , \text{otherwise} \end{cases} \tag{4.2}$$

$$h_{ij} = \frac{q_{ij}}{q_{ij} + \bar{q}_{ij}} \tag{4.3}$$

The second crucial quality when handling streaming data is the ability of the pattern to remain useful, i.e., contributing to the classification decision. This quality decreases when a pattern is increasingly unable to cover observations of its own class. This could be that it incorrectly covers observations of other classes but not its own, which is then taken care of by the homogeneity characteristic; or it could be that it frequently does not appear in the covering subset while observations of its own class are flowing in. To measure this quality, a coverage index (δ) is introduced. δ is an index between 0 and 1 indicating how covering a pattern is in the recent part of the data stream. Its value that is close to 0 indicates low covering, and a value close to 1 indicates high covering.

Each pattern is initially generated with $\delta = 1$, and continuously updated using a decay factor α_p, where $0 < \alpha < 1$. At the time of processing new observation $x_j \in S$, the δ_{ij} of a pattern $p_i \in P$ is calculated as follows:

$$\delta_{ij} = \begin{cases} \alpha_p \times \delta_{i(j-1)} & , \ c_i = l^t_j \text{ and } p_i \notin C_j \\ \delta_{i(j-1)}/\alpha_p & , \ c_i = l^t_j \text{ and } p_i \in C_j \text{ and } \alpha_p < 1 \\ \delta_{i(j-1)} & , \ \text{otherwise} \end{cases} \tag{4.4}$$

As the equation 4.4 shows, the covering indicator δ is updated in only two cases: (1) if the pattern fails to cover an observation of its own class (decay), or (2) if the pattern successfully covers an observation of its own class while its $\delta < 1$ (improvement). Otherwise, the δ is not updated as it is either already at the maximum value of 1 while the pattern has successfully covered an observation of its own class, or the pattern class is different from the observation true label.

As Figure 4.4 illustrates, module II receives the true-labeled observation x_j, the P set, and the C_j subset. It then checks for every pattern in P, and hence the h and the δ of this pattern are modified based on whether it is a covering pattern or not, and whether the covering is correct or not. If the homogeneity falls below a certain threshold (h-*Threshold*), the pattern is considered inefficient, and hence, eliminated. Similarly, if the δ falls below a decay threshold (δ-*Threshold*), it is considered decayed, and hence, eliminated as well. These thresholds are model hyperparameters that can be tuned using hyperparameter search methods, e.g., random search and grid search.

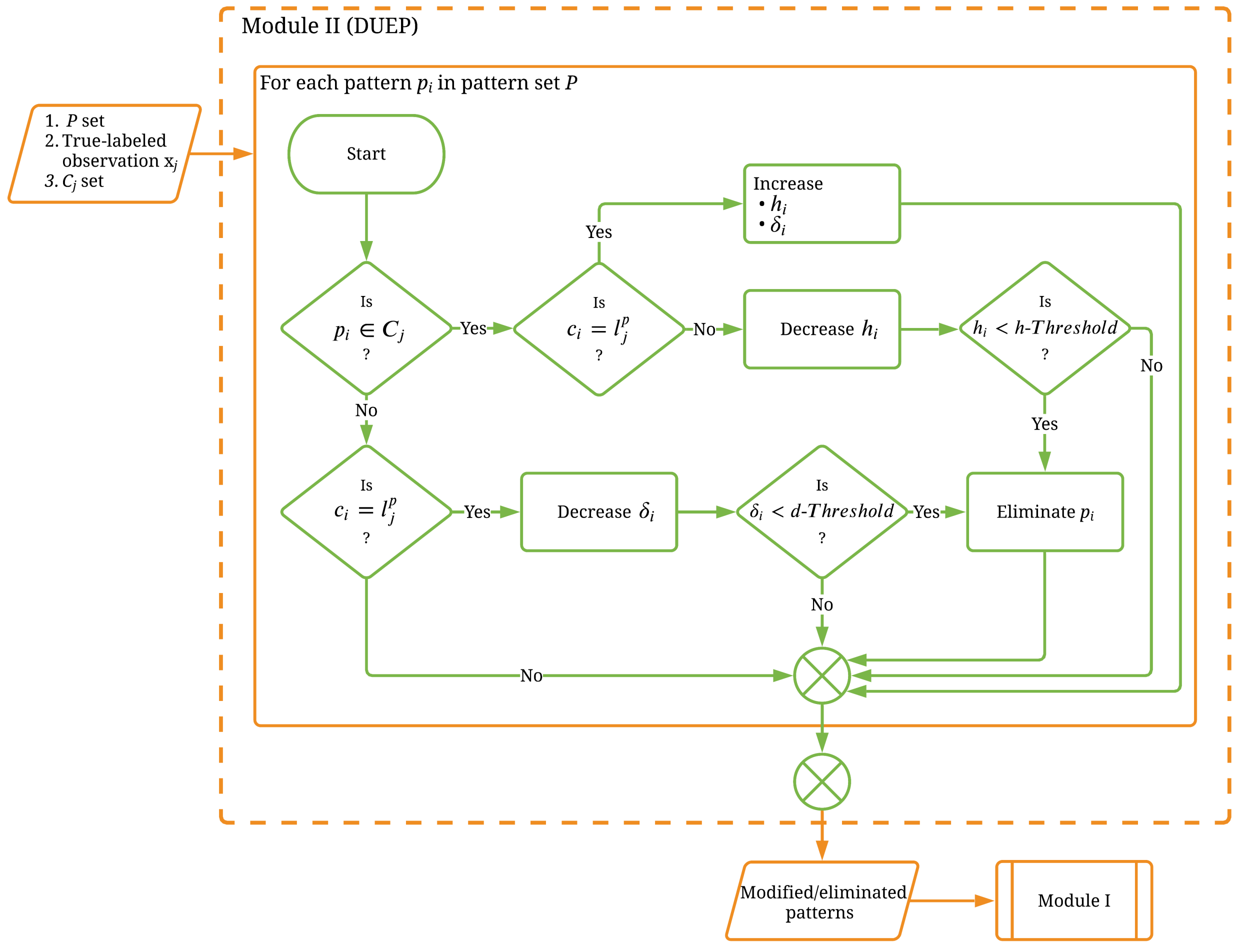

Module II (DUEP)
For each pattern p_i in pattern set P
1. P set
2. True-labeled observation x_j
3. C_j set
Start
Is $p_i \in C_j$?
Yes
No
Is $c_i = l_j^p$?
Yes
No
Increase • h_i • δ_i
Decrease h_i
Is h_i < h-Threshold ?
Yes
No
Is $c_i = l_j^p$?
Yes
No
Decrease δ_i
Is δ_i < d-Threshold ?
Yes
No
Eliminate p_i
Modified/eliminated patterns
Module I

Figure 4.4 Flowchart of Module II: Dynamic Upgrading Existing Pattern (DUEP)

The output of the module II, namely the modified characteristics of the patterns and whether any patterns should be eliminated, is then sent to update the LAD-CM module. Furthermore, new patterns are generated by module III, ENP, as it is described in the following section.

Module III: Extracting New Patterns (ENP) module

The ENP module, as its name implies, is the module responsible for generating new patterns to be added to the P set in order to reflect the recent concepts in the data stream. To do so, it maintains a reservoir of observations, called the sampling reservoir, based on which the new patterns are generated. This reservoir is continuously updated, to be always representative of the most recent concepts of the data streams, through two parallel updating mechanisms.

Each of the two updating mechanisms updates the sampling reservoir by constructing two sets of observations: an entering set, which is the set to be added to the reservoir; and an exiting set, which is the set to be removed from the reservoir – note that the reservoir is fixed in size and balanced between the observations of each class. The difference between the two types of the updating mechanism is in how each mechanism constructs the entering and exiting sets based on its main purpose.

The first updating mechanism, Type I update, aims at capturing both the incorrectly classified observations and the observations classified with low confidence, i.e., with low $f(x)$ value. Then based on the current model accuracy and the observation's true label, it is determined whether each specific observation is to be added to the entering set of Type I update, En-SET I, as illustrated in Figure 4.5. Two accuracy thresholds, $a\text{-}Threshold\text{-}1$ and $a\text{-}Threshold\text{-}2$ are used to control the trade-off between improving the model accuracy, by adding the observation, and overloading the model, and hence, increasing the running time. $a\text{-}Threshold\text{-}1$ is the higher one, over which the model accuracy is considered to be good enough, and hence, the new observation is not added, in favor of not overloading the model. Below $a\text{-}Threshold\text{-}1$, all incorrectly predicted observations are added, but not only that. If the model accuracy drops below $a\text{-}Threshold\text{-}2$, the observations predicted correctly but with low confidence, i.e., low $f(x)$ value, are also added in favor of improving the model accuracy. As for the exiting set of Type I update, Ex-SET I, it is constructed by selecting the number of observations from the reservoir, equal to the cardinality of En-SET I, with the highest $f(x)$ values.

The second updating mechanism, Type II update, aims at keeping the reservoir always representative of the most recent part of the data stream by continuously adding sampled observations from the most recent part of the stream.

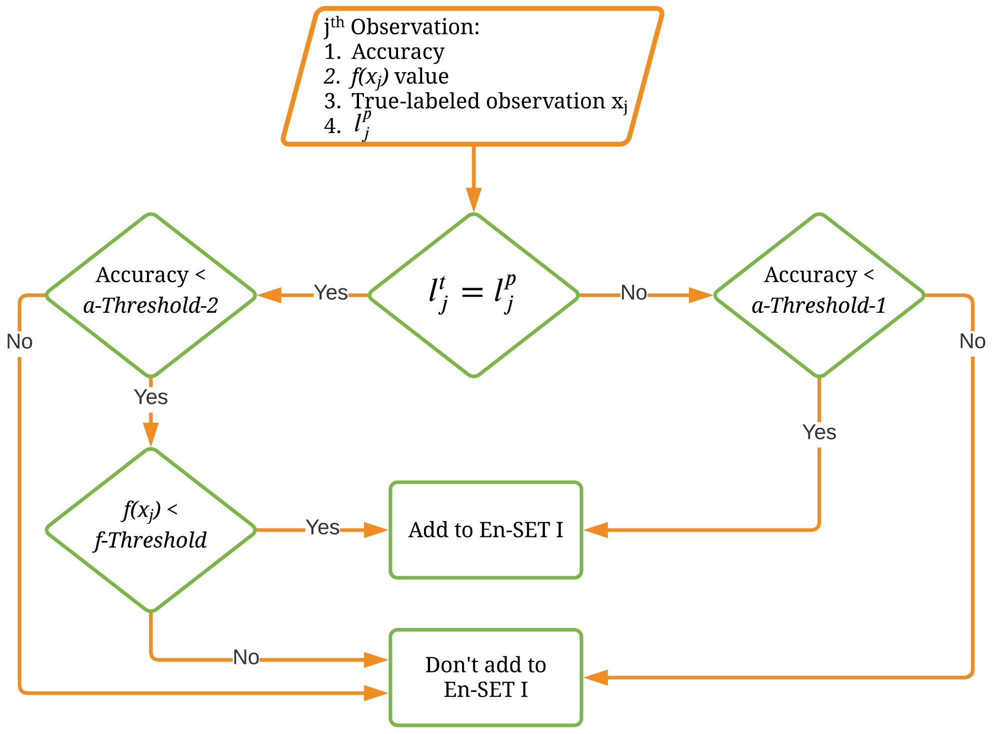


Figure 4.5 Flowchart of constructing En-SET I.

This is done by considering the most recent observations of a specific size as the sampling population, out of which a smaller number of observations are sampled as the entering set, En-SET II. As for the exiting set, it is constructed from the oldest observations at the time of the update, i.e., the observations that have been in the reservoir for the longest amount of time.

An important point to be noted while updating the reservoir is that the reservoir has to be maintained balanced. This is achieved by making sure that the number of observations of every class in an entering set replaces the same number of observations of the same class in the exiting set. This is a governing rule for constructing exiting sets, under which all the other considerations are implied, e.g., sorting the observation based on their $f(x)$ values, and how long they have been in the reservoir, is done for each class present in the reservoir.

Figure 4.6 illustrates how Module III works with Type I and Type II updating mechanism being performed in parallel. The LAD is run, and new patterns are generated, every time a Type I update is done. This is because Type I update is responsible for capturing the observations that need new patterns to be better represented. This is also supported by Type II update that keeps the sampling reservoir always representative of the recent part of the data stream.

To conclude this section, the main goal of the modifications and eliminations done in Module II, and the additions done in Module III, is to keep the model up to date with the most recent changes and information in the data stream, in order to maintain high classification accuracy for future observations.

4.4 Handling imbalanced data streams

As imbalanced static datasets exist in several applications, the concept of data imbalance also exists in data streams. This happens when observations of a certain class or more are appearing in the data stream in significantly small or significantly large amounts, compared to the observations of other classes, such that they are barely noticeable or mostly dominating the data stream, respectively. This phenomenon can happen either temporarily or in the entirety of the data stream. This, however, does not imply that the most occurring class(es) is the most important to be detected while the least occurring class is negligible. Contrarily, it could be the case that a least-appearing class is the most important to detect. For instance, a process failure or a machine breakdown, although rarely occurring, are quite important to be detected, and hence preserved in the information extracted from the data stream, i.e., the patterns generated in the proposed DA-LASD framework. This is the motivation behind all the modifications presented in this section in order to preserve the information about the minority class(es) present in the data stream.

Since the DA-LASD framework employs modules that continuously modify, eliminate, and generate patterns. It is crucial to make sure that the patterns characterizing minority classes do not get eliminated while they are still representative of these minority classes. This could happen when the homogeneity of these patterns is reduced over time as observations of other classes are flowing in significantly larger amounts. Thus homogeneity, with its classical definition, could be deceiving in this case, and hence it needs to be redefined. This indeed extends to the coverage and opposite coverage characteristics, based on which homogeneity is defined.

Normalized pattern characteristics are instead proposed to account for this data imbalance. These characteristics take into account not only the number of observations of a certain class that are covered by a pattern, but also the total number of observations of this class that are present in the entire data stream S. At the time of processing the most recent observation $x_j \in S$, these normalized characteristics of a pattern $p_i \in P$ are calculated as follows:

$$\grave{q}_{ij} = \frac{q_{ij}}{N_{ij}} \tag{4.5}$$

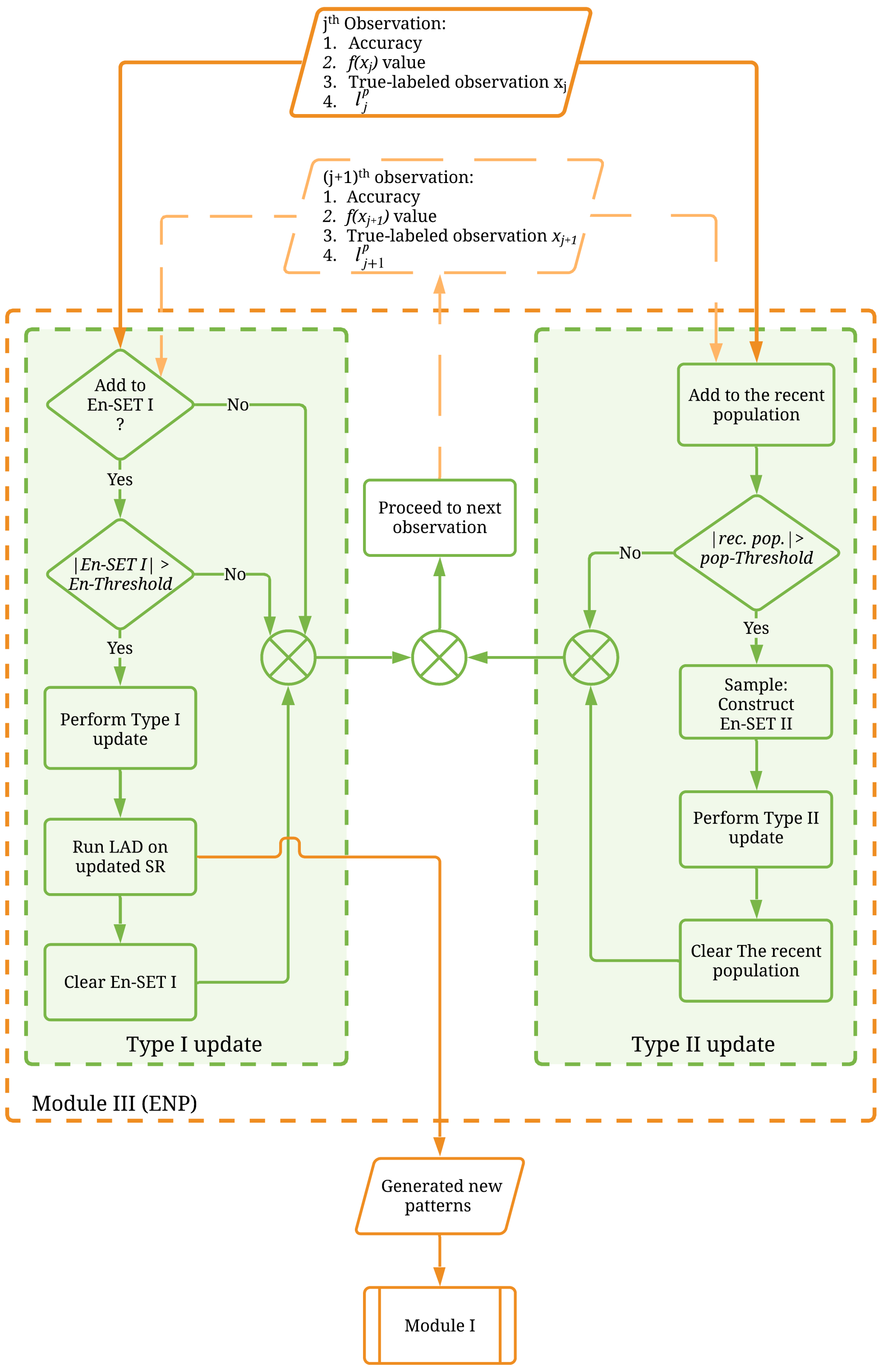


Figure 4.6 Flowchart of Module III: Extracting New Patterns (ENP).

$$\grave{\bar{q}}_{ij} = \frac{\bar{q}_{ij}}{\bar{N}_{ij}} \tag{4.6}$$

$$\grave{h}_{ij} = \frac{\grave{q}_{ij}}{\grave{q}_{ij} + \grave{\bar{q}}_{ij}} \tag{4.7}$$

Where N_{ij} is the total number of observations of the same class as pattern p_i in the entire data stream S up to observation x_j, and N_{ij} is the total number of observations of the opposite class of pattern p_i in the entire data stream S up to observation x_j.

Furthermore, even the normalized definition of a pattern's homogeneity may in some cases not help protecting a pattern that is representative of a minority class from being deleted by Module II. These cases may include the imbalance being severe in the entire data stream, or the minority class(es) not being present in a significant part of the data stream — which does not provide a chance to test the pattern's ability to cover observations of its own class while they are present. Therefore, a more elaborate definition for the homogeneity characteristic is proposed that takes into account the previously considered measures in addition to the ability of the pattern to continue covering observations of its own class as they flow in. This latter measure is simply the previously defined covering index, δ, based on which and the normalized homogeneity, the protected homogeneity characteristic is defined as follows:

$$\hat{h}_{ij} = \frac{(\grave{q}_{ij})^{1-\delta_{ij}}}{(\grave{q}_{ij})^{1-\delta_{ij}} + \grave{\bar{q}}_{ij}} \tag{4.8}$$

According to equation 4.8, the protection of a pattern's homogeneity is proportional to its δ, i.e., the homogeneity is not significantly decreased as long as the pattern's δ is high. However, if the pattern's δ starts to drop, this indicates that the pattern is becoming less able to cover observations of its own class as they flow in, reflected by its δ, and hence the protection is accordingly reduced allowing the $\hat{h}$ to start to drop as well. This makes the protected homogeneity characteristic, $\hat{h}$, as representative as possible of how a pattern is performing, taking into account the possibility that it may be a minority class pattern.

It is important though to note that the protected homogeneity, which accounts for any type of imbalance, is a still a perfectly valid measure of homogeneity even if no imbalance exists. Hence, it is the main measure of homogeneity used in the DA-LASD framework. In addition, it is used as a weight for each pattern in the LAD discriminant function.

4.5 Experimentation

In this section, the DA-LASD algorithm is tested in several classification problems. First, the evaluation of the model performance, in the case of data streams, is explained. Then the datasets used for experimentation are described, and the results are presented along with relevant discussion.

4.5.1 Model performance evaluation

The first and most straightforward measure of model performance is its classification accuracy, which is simply the ratio of the correctly classified observations to the total number of observations. In the case of data streams, this accuracy can be recomputed every time a new observation is added. At the time of processing of the most recent observation x_j, this can be written as follows:

$$A_j = \frac{E_j}{B_j} \tag{4.9}$$

where

$$E_j = L_j + E_{j-1}, \quad B_j = 1 + B_{j-1} \tag{4.10}$$

L_j is the loss function for each observation x_j, and is equal to 1 if x_j is correctly classified, and 0 otherwise. Therefore, E_j and B_j are the total number of observations that are correctly classified, and the total number of observations that are received, respectively up to observation x_j.

However, in order to account for any possible imbalance in the data stream, other measures that take into account the individual accuracy of each class are considered, namely arithmetic and geometric accuracies. For an n number of classes, where each class is indexed by k, the arithmetic and geometric accuracies are calculated as follows, respectively [5]:

$$AA_j = \frac{\sum_{k=1}^{n} A_{kj}}{n} \tag{4.11}$$

$$GA_j = (\prod_{k=1}^{n} A_{kj})^{\frac{1}{n}} \tag{4.12}$$

4.5.2 Model performance evaluation

Since these measures of accuracy consider all the observations in the entire data stream, they may not accurately reflect the most recent changes in the model performance in the recent part of the data stream. Therefore, as it is more important how the model performance has been recently, than how it was long ago at the beginning of the stream, the concept of decayed accuracy is introduced to these measures.

In order to measure the classification accuracy of the model for only the most recent data, a forgetting mechanism is used with fading factors that weigh data using a decay factor α_a to calculate a Decayed Accuracy DA_j as follows [5]:

$$DA_j = \frac{DE_j}{DB_j} \tag{4.13}$$

where

$$DE_j = L_j + \alpha_a \times DE_{j-1}, \quad DB_j = 1 + \alpha_a \times DB_{j-1} \tag{4.14}$$

Similarly, this concept can be extended to both arithmetic and geometric accuracies as follows:

$$DAA_j = \frac{\sum_{k=1}^{n} DA_{kj}}{n} \tag{4.15}$$

$$DGA_j = (\prod_{k=1}^{n} DA_{kj})^{\frac{1}{n}} \tag{4.16}$$

4.5.3 Datasets and Implementation details

The experiments presented in this paper are for ten different synthetic datasets, each representing a data stream, generated using MOA, an open-source software [67]. The ten datasets, named D1 to D10, are generated using different generating functions, namely random tree generator which is introduced in [15], SEA generator which is introduced in [68], and rotating hyperplane generator which is introduced in [16]. In addition, the datasets contain different types and numbers of concept drifts. Table 4.1 presents the details of these datasets, while Figure 4.7 illustrates the different types of concept drifts mentioned in the fourth column in this table. The last column illustrates the sequence of concepts in the data stream, e.g., D1 starts with concept 1, then a drift happens into concept 2, and finally into concept 3. Datasets D9 and D10 have an incremental and continuous drifts which means the concept is changing in a continuous way.

Table 4.1 Datasets details.

Dataset	# of Classes	Generator Function	Drift type	Concepts
D1	2	Random Tree	abrupt	$1 \rightarrow 2 \rightarrow 3$
D2	2	Random Tree	Gradual	$1 \rightarrow 2 \rightarrow 3$
D3	2	Random Tree	Recurring - Abrupt	$1 \rightarrow 2 \rightarrow 1$
D4	2	Random Tree	Recurring - Gradual	$1 \rightarrow 2 \rightarrow 1$
D5	2	SEA	abrupt	$1 \rightarrow 2 \rightarrow 3$
D6	2	SEA	Gradual	$1 \rightarrow 2 \rightarrow 3$
D7	2	SEA	Recurring - abrupt	$1 \rightarrow 2 \rightarrow 1$
D8	2	SEA	Recurring - Gradual	$1 \rightarrow 2 \rightarrow 1$
D9	2	Rotating Hyperplane	Incremental - high	Continuous
D10	2	Rotating Hyperplane	Incremental - low	Continuous

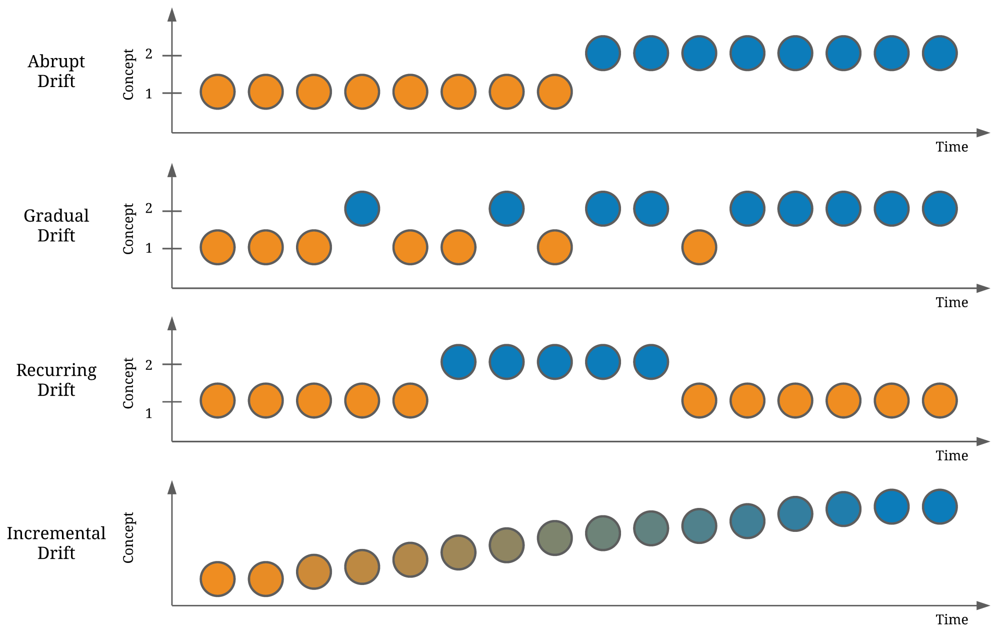


Figure 4.7 Different types of concept drifts.

The DA-LASD is implemented in python using Faust package and Apache Kafka to build the streaming environment. For the LAD part that generates new patterns, the cbmLAD software [56] is used. For simplicity, it is assumed that the model immediately receives the actual label for every observation after the model predicts the label and before receiving the next observation [5].

A random search technique is applied to select best combination of hyper-parameters values that maximizes the accuracy for each data stream. Table 4.2 shows the results of random search technique for D1 data stream, as an example. Table 4.3 summarizes the model parameters used in the experimentation for all data streams.

4.5.4 Results and Discussion

The ten datasets represent five pairs containing the same types of concept drifts. While the results of all datasets are summarized in Table 4.4, one dataset of each pair is discussed in detail.

For more detailed analysis, the trends of the different accuracy measures along the data stream, as well as the number of generated patterns and number of observations for every class, are displayed for the selected datasets. These datasets are D1, D2, D3, D4, D9 and D10.

D1, similar to D5, has an abrupt concept drift with three different concepts, each lasts for 20k observations, for a total of 60k stream. Figure 4.8 shows the different performance measures for D1. As expected, sudden drops in decayed accuracy measures occurred at the abrupt drift points, namely at 20k and 40k observations, which is then followed by an uptrend as the DA-LASD adapts to this change. Temporary imbalance is also noticed during the second and third concepts, where class 2 and then class 1 become majority classes, respectively. This is when pattern protection proves useful in preserving the patterns of the minority class until their usefulness has a chance to be tested as the minority class starts to reappear in the data stream.

D2, similar to D6, has a gradual concept drift of three different concepts. These gradual drifts happened over a span of 5k observations, each centered at the points of 20k and 40k observations, respectively, as shown in Figure 4.9. As the graphs in Figure 4.9 show, the DA-LASD has adapted to these changes, and succeeded in improving the different accuracy measures again after their decrease. It is also noticed that due to the longer time span of the gradual drifts, which includes observations from both concepts, the number of generated patterns is higher than the number of those generated in the case of abrupt drifts. However,

Table 4.2 Datasets details.

Run	α_a	*a-Threshold*-1	*En-Threshold*	α_p *h-Threshold*	δ-*Threshold*	A	AA	GA	
1	0.9999	0.85	400	0.9995	0.85	0.2	0.835	0.834	0.834
2	0.9997	0.85	400	0.9995	0.85	0.2	0.846	0.847	0.847
3	0.9995	0.85	400	0.9995	0.85	0.2	0.848	0.846	0.846
4	0.9995	0.85	300	0.9995	0.85	0.2	0.854	0.855	0.855
5	0.9995	0.85	200	0.9995	0.85	0.2	0.874	0.873	0.873
6	0.9995	0.85	200	0.99975	0.85	0.2	0.869	0.868	0.868
7	0.9995	0.85	200	0.9999	0.85	0.2	0.88	0.879	0.879
8	0.9995	0.85	200	0.9999	0.85	0.1	0.873	0.872	0.872
9	0.9995	0.85	200	0.9999	0.85	0.1	0.896	0.896	0.896
10	0.9995	0.90	200	0.9999	0.85	0.1	0.906	0.903	0.903
11	0.9995	0.90	200	0.9999	0.8	0.1	0.907	0.905	0.905
12	0.9995	0.90	200	0.9999	0.8	0.05	0.892	0.895	0.894
13	0.9995	0.90	200	0.99995	0.8	0.05	0.907	0.906	0.906
14	0.9995	0.90	200	0.999925	0.8	0.05	0.899	0.899	0.899
15	0.9995	0.90	200	0.999975	0.8	0.05	0.883	0.887	0.887
16	0.999125	0.90	200	0.99995	0.8	0.05	0.896	0.898	0.898

Table 4.3 Values of DA-LASD parameters.

Dataset	α_a	*a-Threshold*-1	α_p	*h-Threshold*	δ-*Threshold*
D1	0.9995	0.9	0.9999	0.8	0.05
D2	0.999125	0.9	0.99995	0.8	0.05
D3	0.9995	0.9	0.9999	0.8	0.05
D4	0.999125	0.9	0.99995	0.8	0.05
D5	0.999125	0.87	0.99995	0.8	0.05
D6	0.999	0.87	0.99995	0.8	0.05
D7	0.999125	0.87	0.99995	0.8	0.05
D8	0.999	0.87	0.99995	0.8	0.05
D9	0.999	0.9	0.99995	0.85	0.1
D10	0.999	0.925	0.99995	0.85	0.1

Table 4.4 Summary of DA-LASD results.

Data	Acc.	Arithmetic Acc.	Geometric Acc.	Mean of Decayed Acc.	Mean of Decayed Arithmetic Acc.	Mean of Decayed Geometric Acc.
D1	0.9073	0.9060	0.9059	0.9053	0.8954	0.8934
D2	0.8781	0.8811	0.8808	0.8781	0.8789	0.8783
D3	0.8972	0.8948	0.8940	0.8955	0.8846	0.8823
D4	0.8948	0.8953	0.8952	0.8944	0.8876	0.8868
D5	0.8903	0.8893	0.8893	0.8887	0.8878	0.8874
D6	0.8976	0.8971	0.8969	0.8970	0.8963	0.8960
D7	0.8919	0.8917	0.8916	0.8916	0.8912	0.8908
D8	0.9000	0.8974	0.8974	0.8989	0.8964	0.8961
D9	0.8858	0.8858	0.8857	0.8839	0.8814	0.8811
D10	0.8866	0.867	0.8866	0.8844	0.8832	0.8829

the algorithm adapts by eliminating inefficient patterns as the new concept stabilizes.

Recurring drifts occurred in an abrupt manner in D3 and D7 and in a gradual manner in D4 and D8. Figures 4.10 and 4.11 show the results for D3 and D4, respectively. Again, it is clear how the algorithm adapted to these two different concept drifts, and successfully improved the accuracy measures. It can also be noticed how the algorithm responded to the drop of class 1 accuracy by generating more patterns to characterize it, which in return helped improving the accuracy of this class.

In D9 and D10, the drifts happened incrementally and continuously with a relatively high and low change rates, respectively. As Figures 4.12 and 4.13 show, the algorithm was able to maintain high accuracy measures along the entire data streams. They also show how the number of patterns is smoothly changing in the case of incremental drift, compared to other concept drifts, which is expected as there are no major points to mark the change in the concept, but rather an incremental change. When it comes to the difference between D9 and D10, the high change rate in D9 has led to a larger number of patterns being generated compared to D10.

This detailed analysis shows how the proposed DA-LASD is quite successful in dynamically adapting to all the different changes in the data stream, and consequently maintaining high accuracy measures across the entirety of the data streams in all the discussed datasets.

Finally, Table 4.5 provides a comparison between the proposed DA-LASD and other machine learning techniques capable of handling streaming data, namely Hoeffding Tree, Hoeffding Option Tree (HO-Tree), Hoeffding Adaptive Tree (HA-Tee), Naive Bayes incremental learner, and Self Adjusting Memory K nearest neighbors (SAMKNN).

Table 4.5 Comparison between DA-LASD and different ML techniques in terms of the mean accuracy over all the stream.

Data	DA-LASD	Hoeffding Tree	HO-Tree	HA-Tee	Naive Byes	SAMKNN
D1	0.907	0.733	0.759	0.889	0.603	0.904
D2	0.878	0.74	0.747	0.839	0.604	0.871
D3	0.897	0.774	0.781	0.874	0.622	0.885
D4	0.894	0.773	0.78	0.829	0.623	0.853
D5	0.89	0.899	0.9	0.925	0.897	0.933
D6	0.897	0.899	0.901	0.921	0.897	0.929
D7	0.891	0.919	0.92	0.925	0.906	0.934
D8	0.9	0.918	0.91	0.922	0.907	0.931
D9	0.885	0.876	0.889	0.911	0.824	0.921
D10	0.886	0.894	0.891	0.908	0.861	0.921

It is obvious that the proposed framework provides competitive classification performance,

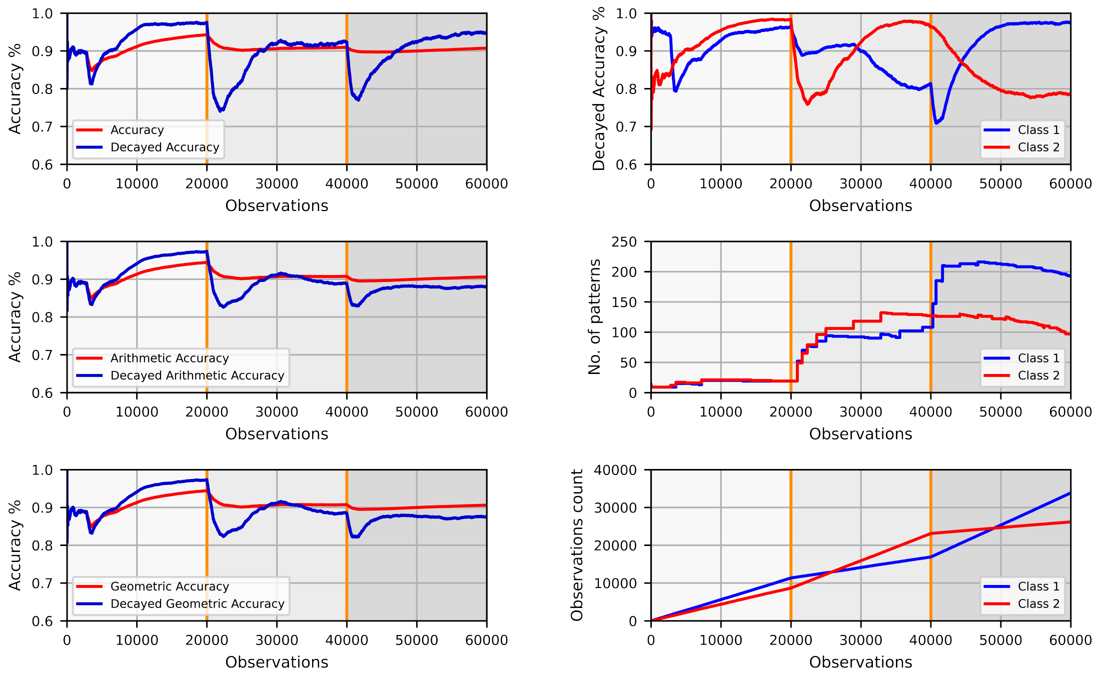


Figure 4.8 Performance of DA-LASD on D1 stream with abrupt drifts between 3 concepts.

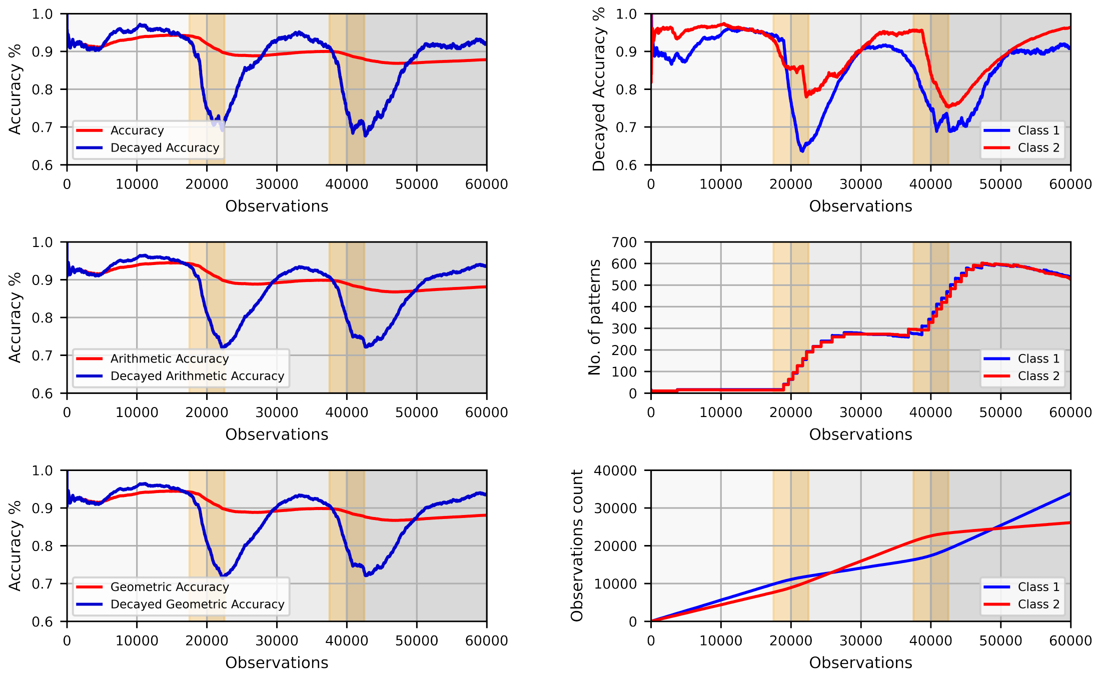


Figure 4.9 Performance of DA-LASD on D2 stream with gradual drifts between 3 concepts.

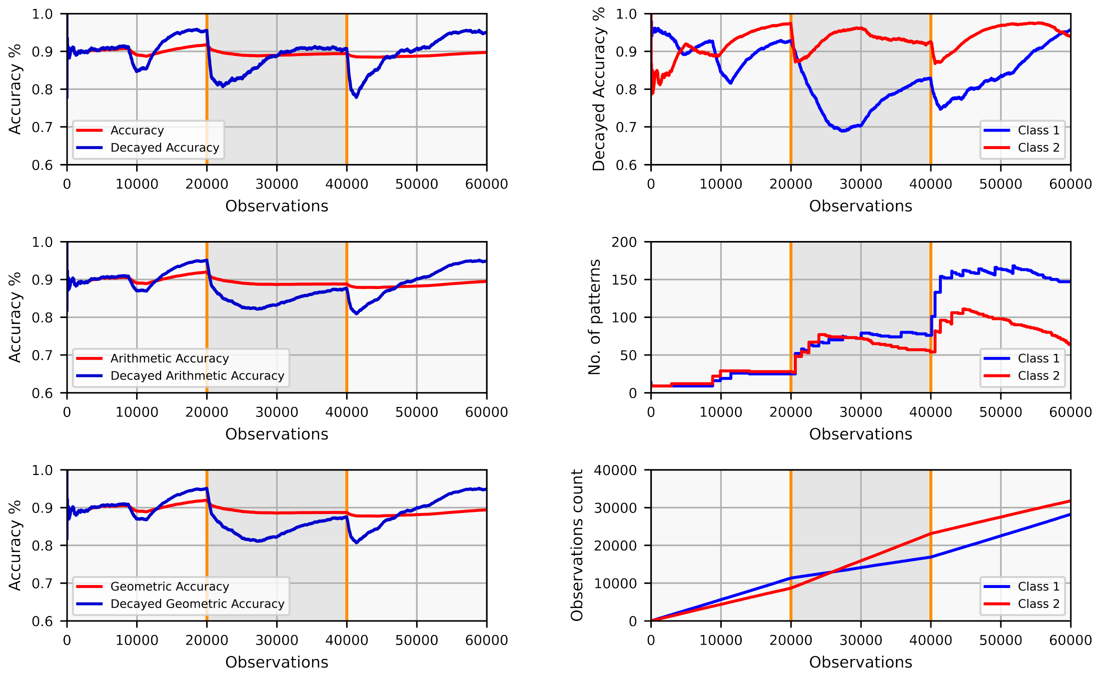


Figure 4.10 Performance of DA-LASD on D3 stream with recurring drifts between 2 concepts in an abrupt manner.

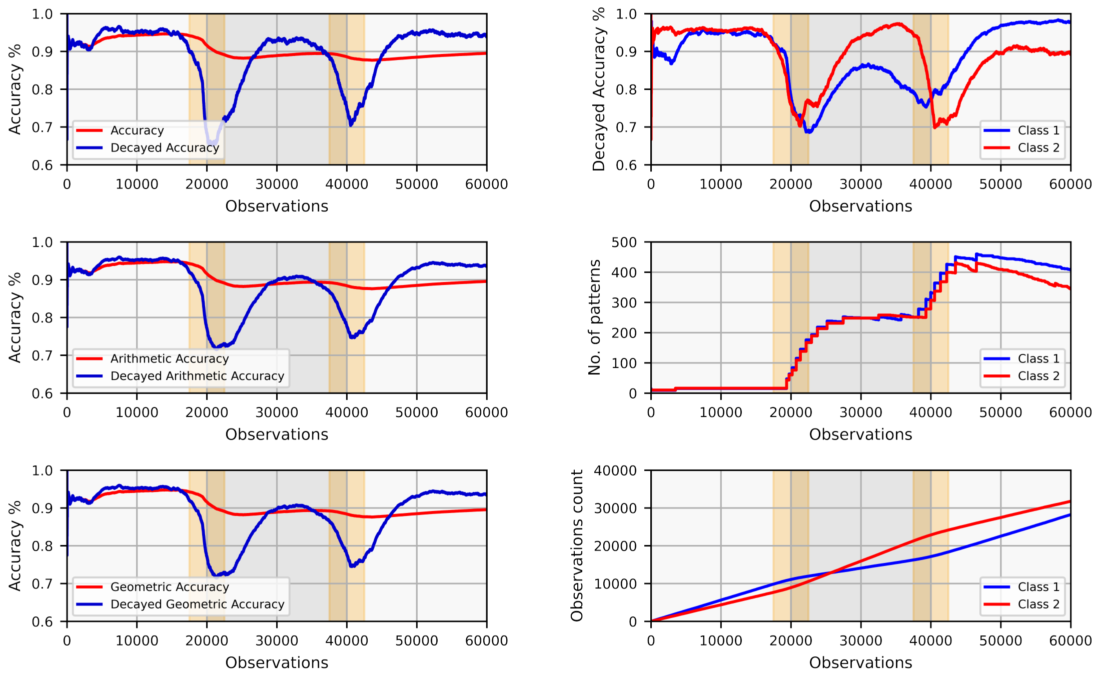


Figure 4.11 Performance of DA-LASD on D4 stream with recurring drifts between 2 concepts in a gradual manner.

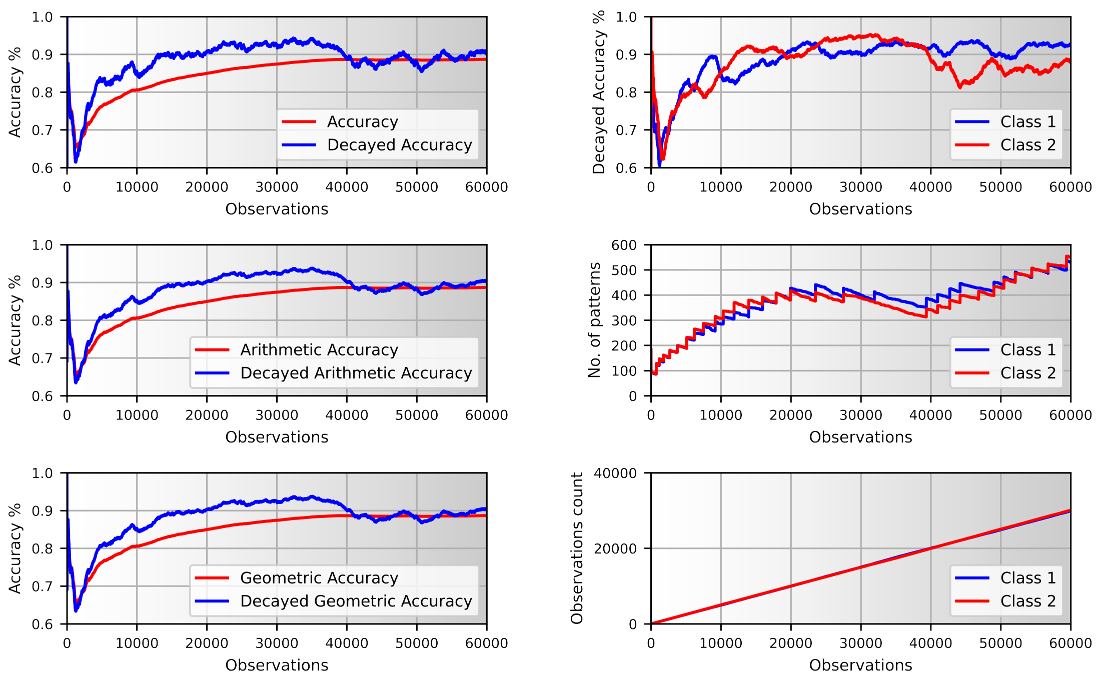


Figure 4.12 Performance of DA-LASD on D9 stream containing incremental drift with high change rate.

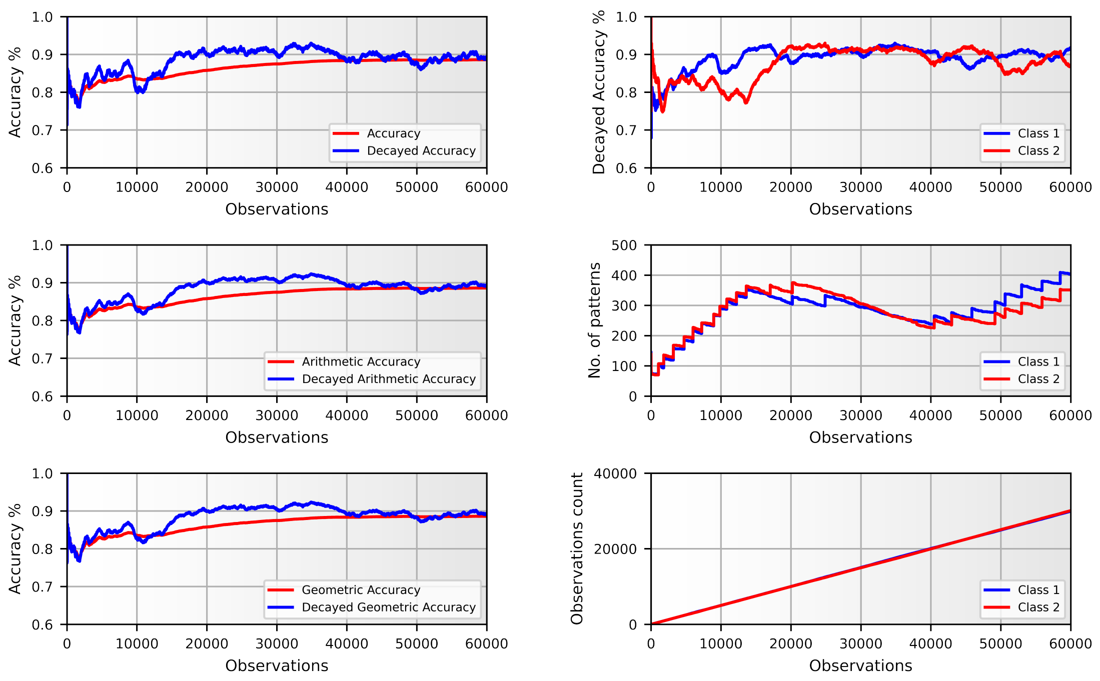


Figure 4.13 Performance of DA-LASD on D10 stream containing incremental drift with low change rate.

even though not entirely dominant. What makes the proposed DA-LASD quite promising is how it combines this competitive classification performance with its distinctive interpretability power, which is essential in providing physical meanings and describing hidden phenomena. This is particularly important in business and industrial applications, especially in Industry 4.0 and Quality 4.0 paradigms, in addition to a diverse number of domains such as medical, financial and aviation sectors, to name a few.

4.6 Conclusion

In this research, we propose a Dynamic and Adaptive Logical Analysis of Streaming Data framework, DA-LASD, which is a LAD-based classifier capable of handling data streams. The DA-LASD consists of three main modules. LAD-CM is the LAD classifier model, which classifies observations based on the set of patterns it contains. However, this set of patterns has to be continuously updated to adapt to the changes in the data stream as it flows. The update is done by modifying the characteristics of existing patterns, eliminating decayed and inefficient patterns, and added new patterns. The first two steps are done in the Module II: Dynamic Update of Existing Patterns, DUEP. The third step is taken care of in the Module III: Extracting New Patterns, ENP.

In order to correctly detect decayed and inefficient patterns, several modifications on the classical pattern characteristics are proposed. Particularly, the covering and opposite covering characteristics are normalized, and the homogeneity is normalized and protected. This enables the framework to take into account any data imbalance that may exist. Moreover, a new characteristic is proposed, namely the covering index, δ, which is an updated measure of the pattern's ability to cover observations of its own class as they continue to flow in.

The proposed DA-LASD is then tested on several datasets of streams generated by different generating functions and containing different types and numbers of concept drifts. The detailed analysis of the experimentation results shows how the proposed framework dynamically adapts the model, and successfully improves any performance measures that start to decline, i.e., successfully maintains high levels of classification accuracy. This accuracy of the DA-LASD is found to be competitive when compared with other machine learning techniques that handle streaming data. With high accuracy and its distinctive interpretability power, the DA-LASD shows a lot of promise in applications where high accuracy and interpretability are required, e.g., industrial, financial, and medical applications, to name a few.

CHAPTER 5 ARTICLE 3: SUSTAINABLE MODEL OF DYNAMIC AND ADAPTIVE LOGICAL ANALYSIS OF STREAMING DATA WITH INTERPRETABLE PATTERNS FOR DEGRADING TURBOFAN ENGINE

Authors: Osama Elfar, Adham Mohamed, Soumaya Yacout, Hany Osman.

Journal: Computers & Industrial Engineering (submitted).

Abstract: Industrial process control is adapting to become more intelligent and automated using artificial intelligent and machine learning techniques under the label of Industry 4.0. Automated and intelligent process control have to be able to handle and adapt dynamically to the streaming data collected from such advanced industrial processes. These machine learning techniques should be able to interprets their outputs and decide what action should be taken to bring out-of-control process back to the normal state. In this paper, a Dynamic and Adaptive Logical Analysis of Streaming Data (DA-LASD) model reinforced with feature engineering mechanism is proposed to develop an intelligent and interpretable monitoring system for turbofan engine. The feature engineering mechanism comprises different steps of feature selection and feature extraction to enhance the model performance. Through a case study of turbofan jet engine failure detection, the proposed model shows how it is able to detect out-of-control states and provides interpretable patterns for them. Also, it shows a high sustainability through a dynamic adapting to concept drifts in the streaming data. Moreover, it statistically outperforms other machine learning techniques in terms of classification sensitivity which is important to measure the ability to detect faults and out-of-control states.

5.1 Introduction

Industrial processes have been rapidly evolving in the recent decades due to significant advancements in sensor technology, networks, intelligent robotics, automation and big data analytics, under the label of Industry 4.0 [69]. Such an evolution provides significant, positive impact on operational efficiency, agility and flexibility; productivity growth; as well as consumer experience and cost and revenue management [70]. Industrial process control monitors the process performance and provides the required feedback for corrective actions [21].

Running a jet engine is considered a process which is controlled through an advanced controlling system in order to operate a given jet engine at maximum efficiency and deliver optimal performance. Jet engines have many controllable variables, such as fan speed, core speed, pressure in bypass-duct and fuel-air ratio, which are controlled through different actuators directly or indirectly by advanced controlling systems [71, 72]. However, an intelligent monitoring system should be incorporated to detect out-of-control states and faults that cause low efficiency and bad performance, and hence give the autonomous corrective action to the controller to reset the state to the optimal without human interaction. Such intelligent monitoring system could be achieved through applying data-driven approaches and machine learning (ML) techniques to detect anomalies and predict the out-of-control. However, in such advanced process, data arrives in the form of streams. Data streams are defined by algorithmic community to support real-time analytics. They are sequences of data observations, possibily infinite, each has a timestamp and a temporal order. Observations arrive one by one, and should be processed in real time. [5]. In this context ML techniques must be able to process data streams and be featured by high robustness, resilience, sustainability, and interpretability.

Robustness A robust machine learning model is the one have a testing error consistent with its training error, i.e., the model accuracy doesn't deteriorate too much when testing with slightly different data after adding some noise. Therefore, the model will have low bias and low variance errors. This is the first basic problem that is faced during building an ML model.

Resilience A resilient machine learning model is the one maintains optimal performance after deployment and on variety of test sets. It should not be overfitted to only one test data set. Resilience is the robustness against time. This problem occurs in industrial application when there are lack of training data at the modeling phase and data arrives in different time in the future.

Sustainability A sustainable machine learning model is the one which can capture the distribution drifts in the new data, and hence automatically adapt itself to the new concept. In the real and practical data after deploying the model in industrial applications, there are shifts that occur overtime, and hence the distribution of data used to train the model is different from the newly available data which is called a concept drift. Therefore, a sustainable machine learning model is needed to overcome this practical problem which is very common in industrial settings because all physical assets experience aging and deterioration, and hence the distribution of new data changes overtime.

Interpretability An interpretable machine learning model is the one that have an explanatory power able to provide root-cause analysis of certain phenomena to facilitate the decision-making process. That will guide automatically the advanced controlling systems to demand corrective actions to maintain the engine at the optimal performance and healthy state [73].

Using an ML technique featured with these mentioned abilities is essential to develop an intelligent monitoring system that is resilient, sustainable, and interpretable. Such system will be able to guide automatically the advanced controlling system in a jet engine to demand corrective actions for maintaining the engine at the optimal performance and continuously in-control state or giving an alarm for a fetal fault in the engine.

In this paper, a model based on dynamic and logical analysis of streaming data (DA-LASD) is proposed for intelligent and interpretable monitoring of the operational process of turbofan engine. The proposed model relies on dynamically and automatically updated interpretable patterns to determine process state and provide interpretation, and hence feedback corrective actions. The model is reinforced by feature engineering steps in order to determine the set of most relevant/significant features for the process control, and hence improve the model resilience and sustainability which make the model to correctly classify the observations and adapt to concept drifts in data stream. Therefore, the process control will be able automatically to detect out-of-control states, provide interpretation for them, give corrective feedback, and update the model to any concept drifts without any human interactions.

The paper is organized as follows. In section 5.2 related works and the objective are presented. In section 5.3, the proposed methodology is presented, including feature engineering and DA-LASD methodology. In section 5.4, a case study is presented, including problem generalization as well as evaluation criteria. Section 5.5 walks the reader through the key results and comparative analysis with other techniques. Finally, section 5.6 concludes the presented work and discusses areas of further research.

5.2 Related Works

In this section, we begin by reviewing the most relevant works proposed for using machine learning techniques in industrial process monitoring and fault detection. In [74] Different ML techniques, such linear regression, decision tree, random forest, gradient boosting, and k-nearest neighbors, are used to monitor turbofan engine and and predict the remaining useful life for the engine. for the same purpose, deep convolutional neural network is used in [75]. In addition, ML techniques, such as support vector data description (SVDD), neural networks, decision trees, k-means clustering, linear discriminant analysis, principal component analysis (PCA) and Logical Analysis of Data (LAD), have been used with control charts to detect out-of-control states and anomalities [22, 23]. In [24], a pattern recognition-based fault detection method was proposed using SVDD as a one-class classification algorithm. This method was applied for chillers monitoring and faults detection in [25]. To improve the fault detection performance, a PCA-R-SVDD based method is proposed in [26] by developing a SVDD model in the residual subspace (Rs) using the PCA modeling residual data. In [27], neural network is used for real-time detection of faults. A neural network model for fault detection is defined by using dynamic neurons in [28] and applied in sugar evaporation process. In [29] and [30], deep neural networks were used for fault detection and monitoring non-linear processes. PCA is used as a data-driven method for industrial process monitoring [31]. Kernel PCA (KPCA) was proposed in [32] in order to extend PCA to nonlinear systems as most practical and real industrial systems are nonlinear. In order to deal with the data streams, online KPCA was proposed in [33]. However, this method suffers from a high computational time. Therefore, reduced rank optimized KPCA (RR-KPCA) was introduced in [34] with reduced complexity to overcome this issue and was applied for monitoring an air quality monitoring network.

Some of these techniques are however not learning online and not dynamically adapting for streaming data, i.e., their models are trained on static dataset and then put in action without accounting for the dynamical change in data concepts other than those existing in the training dataset. Accordingly, these techniques will suffer from low sustainability. Although the others have been extended to learn online from streaming data, these techniques lack the interpretability needed to provide feedback for automatic corrective actions. LAD is a classification method that generates interpretable patterns [44–46]. This characteristic made it useful option in a wide variety of industrial settings where interpretability is required. LAD was proposed for a first time as a fault detection and diagnosis tool for industrial systems in [35]. In [36], LAD was applied for the detection of faults in rotating machinery using vibration signals. LAD is applied in [37] to detect and diagnose faults in industrial chemical processes and provide patterns to build a decision model that diagnoses faults and explains the potential causes of these faults. A tool wear multiclass detection method based on LAD is proposed in [38] by deriving the information from machining process variables. In spite of its distinct interpretability and competitive accuracy, classical LAD suffers from a high computational time. Therefore, the authors in [39] proposed LAD ensemble (LAD-ENS) technique to accelerate the processing and preserve its high accuracy and interpretability by combining different LAD models in the pattern level. Although, process control, under Indusrty 4.0 label, is required to be dynamic and adaptive to streaming data, which exhibits concept-drift phenomenon. This phenomenon is very common in industrial settings because all physical assets experience aging and deterioration. In order to overcome this issue, the authors in [2] proposed a framework that addresses a methodology for handling streaming data with high interpretability. The introduced methodology is dynamic and adaptive logical analysis of streaming data (DA-LASD). DA-LASD is built with LAD as its core classifier module, and provides modules for dynamically updating and adapting the model according to the changes in the data streams.

Based on these related works, this paper presents a DA-LASD model for intelligent and interpretable monitoring of the the operational process of turbofan engine. This model can be extended to other industrial process control systems. The goal is to replace the traditional statistical process control (SPC) that is shown in Figure 5.1, and which relies on detecting out-of-control states and non-random variations in the process quality metrics and then performing root-cause analysis by humans, with the DA-LASD that relies on dynamically and automatically updated interpretable patterns to determine process state and provide interpretation, and hence feedback corrective actions, as illustrated in Figure 5.2. The proposed DA-LASD is reinforced by feature engineering steps in order to determine the set of most relevant/significant features for the process control, and hence improve the model resilience and sustainability. That means enhancing the model ability to correctly classify the observations and adapt to concept drifts in data stream. Therefore, the process control will be able automatically to detect out-of-control states, provide interpretation for them, give corrective feedback, and update the model to any concept drifts without any human interactions.

5.3 Methodology

The proposed methodology consists of two main parts. The first part is feature engineering, which consists of feature selection and extraction. These steps prepare the data in such a way that enhances the performance of machine learning techniques in extracting knowledge from the data. The second part is applying the DA-LASD, which is a LAD-based methodology that is dynamically updated according to the newly received observations to maintain high

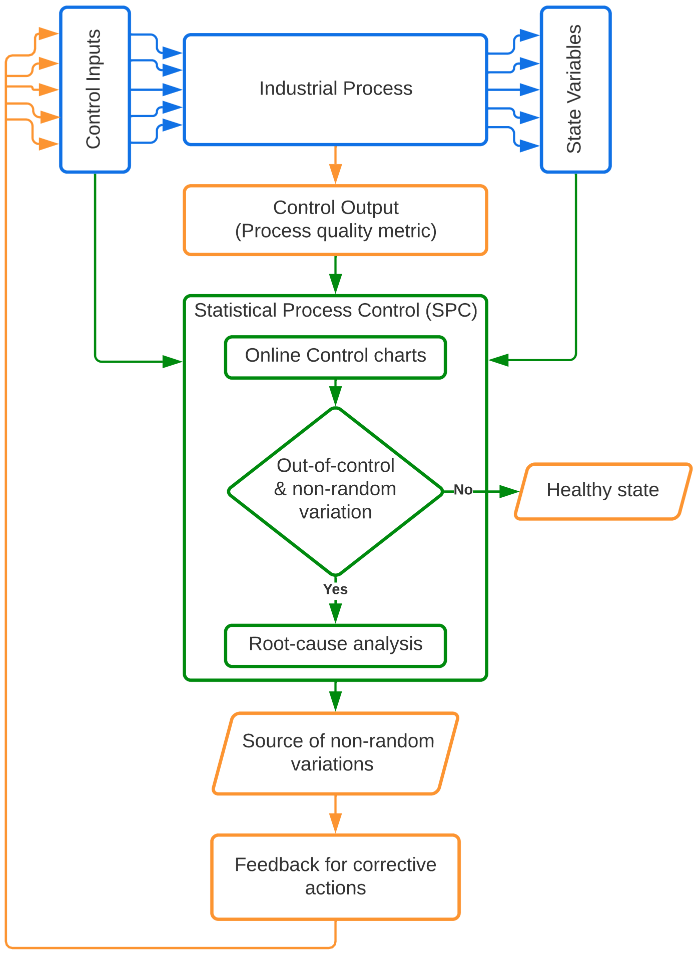


Figure 5.1 Traditional Statistical Process Control procedure

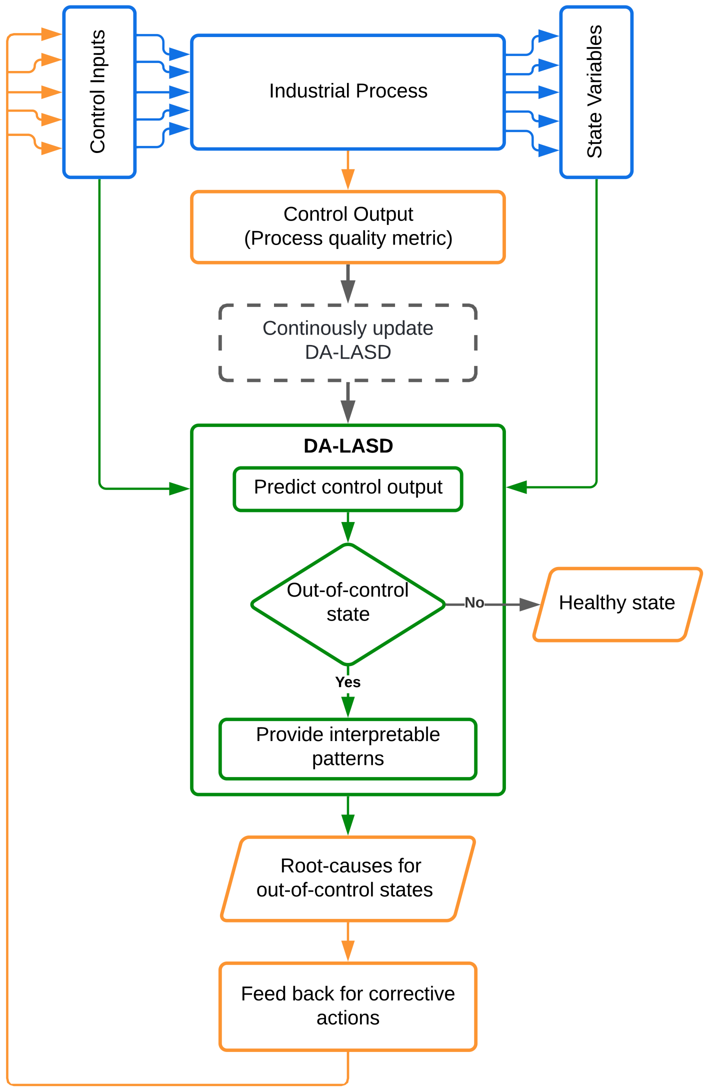


Figure 5.2 DA-LASD Process Control procedure

level of learning accuracy along with the distinct interpretability ability of the LAD.

The feature engineering and the initialization of the DA-LASD model are done using an initial pool of observations. This pool is built by collecting an accumulated number of observations as the data stream starts to flow, and it is then used to do the feature engineering steps and to extract initial patterns for the LAD classifier module of the DA-LASD, as further presented in this section.

5.3.1 Feature Engineering

In machine learning terminology, the word "feature" is a generic word that in the case of industrial process refers to the control inputs and state variables of such control systems. It is therefore further used here with this significance.

The purpose of the feature engineering is to determine the set of most relevant/significant features to be considered in the machine learning model to enhance the model's performance and its ability to extract knowledge from the data. This is done through two main steps: i) feature selection, and ii) time series feature extraction, as illustrated in Figure 5.3.

Feature selection The first sub-step of feature selection is which features with low variation are excluded–other criteria may also be used to further reduce the data if needed. These features are those with a variance below a certain threshold $\sigma^2 < \sigma_{th}^2$, and can hence be considered constants. In order to determine a proper variance threshold, scaling step is performed firstly for all features, and then all variances are determined, i.e., the variance threshold will be determined on the scaled values. The aim of this sub-step is to avoid unnecessarily overloading the model, and consequently enhance its efficiency. The second sub-step of feature selection is which then focuses on selecting the most significant features. This is achieved by using a voting-based feature selection mechanism that comprises three different classifiers, namely random forests, gradient boosting classifier and LASSO (least absolute shrinkage and selection operator). It then combines the individual outputs of these classifiers, based on their respective selection votes, in order to determine the set of significant features. Random forests and gradient boosting techniques provide a ranked feature importance based on mean of accumulation of impurity decrease within each tree in the forest by using the feature to split nodes [76]. That is employed to select the set of most relevant/significant features. In our methodology, these two feature selection techniques are performed with recursive feature elimination (RFE) technique [77] to optimize selecting features to be eliminated and their number. LASSO technique is based on a regression estimator that shrinks the coefficients of irrelevant features. It also chooses only one among highly correlated features by shrinking

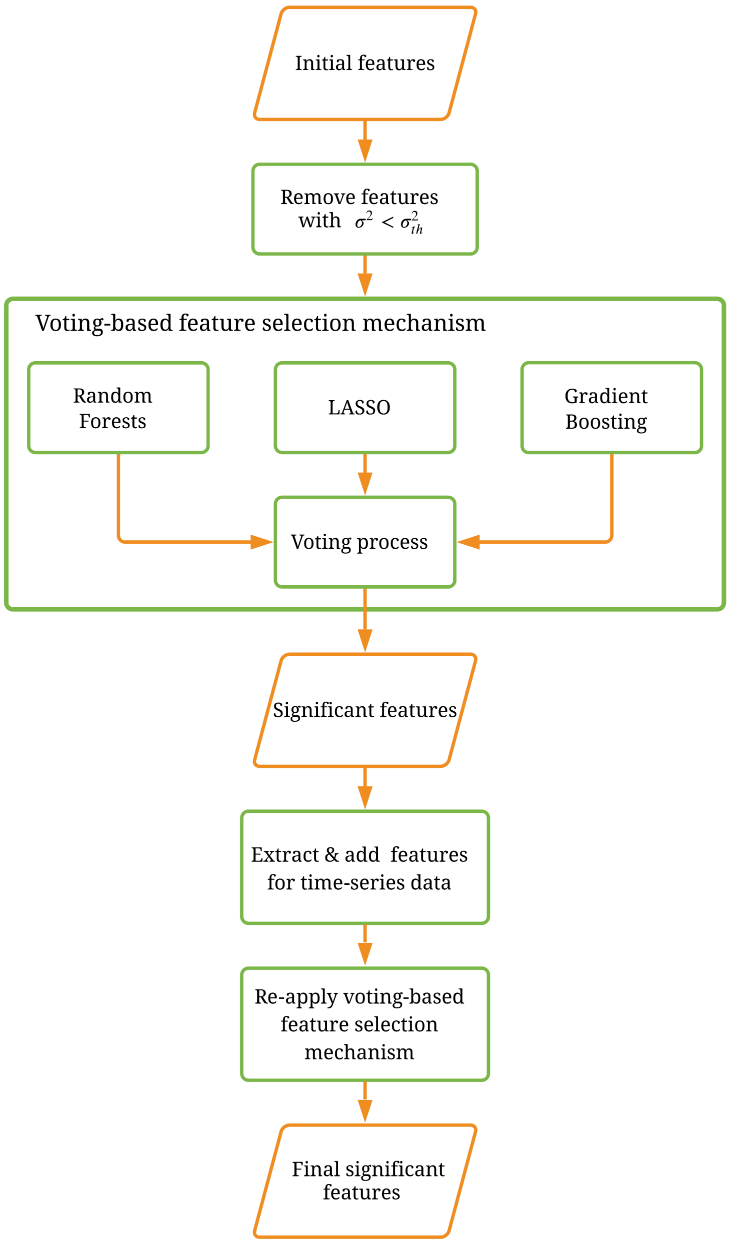

Initial features
Remove features with $\sigma^2 < \sigma^2_{th}$
Voting-based feature selection mechanism
Random Forests
LASSO
Gradient Boosting
Voting process
Significant features
Extract & add features for time-series data
Re-apply voting-based feature selection mechanism
Final significant features

Figure 5.3 Feature Engineering

the coefficients of others to zero. That is performed by defining the LASSO model to decrease the residual sum of squares subject to the sum of the absolute value of the coefficients being less than a constant [78].

Time series feature extraction Data streams collected from most practical and real industrial process are stationary time series data in the healthy state (in-control state). That means that such data streams have statistical properties, e.g., mean and variance, that don't change through time. However, these statistical properties can change over time in the case of the process is out-of-control. As such changes should be captured by the model. Therefore, time series feature extraction is used as a step to generate new features that capture, for every observation, specific change information over time from a certain number of preceding observations [79]. For instance, this could be a moving average of some features or the difference in value of a specific feature from a previous observation to the current one. Extracted features are added to the previously selected set of significant features, then the voting-based feature selection mechanism is performed again to determine the final set of significant features.

5.3.2 DA-LASD Framework

DA-LASD is a machine learning classification framework that not only possesses strong interpretability powers, by using the concept of Logical Analysis of Data (LAD), but also adapts dynamically to the changes in the data as it flows in. This gives the DA-LASD its distinct ability of maintaining high levels of accuracy and interpretability along dynamically changing data streams [2].

As Figure 5.4 illustrates, the DA-LASD framework consists of three main modules. Module I is the LAD classification model which determines for every observation a set of patterns covering the observation, constructs the discriminant function, and accordingly predicts a label. As the data stream flows in, carrying new knowledge, Modules II and III take care of updating the set of all patterns which is used by Module I for classification. Specifically, Module II updates the pattern's characteristics and eliminates inefficient and decayed patterns, while Module III generates new patterns.

Inefficient pattern is the pattern whose homogeneity below a certain threshold. The authors introduced in [2] a novel protected homogeneity characteristic $\hat{h}$ in order to make the model able to handle imbalanced data streams and measure the pattern efficiency effectively. In order to define decayed patterns, the authors also introduced in [2] a coverage index δ, which is a measure of the pattern's ability to remain useful by correctly contributing to the

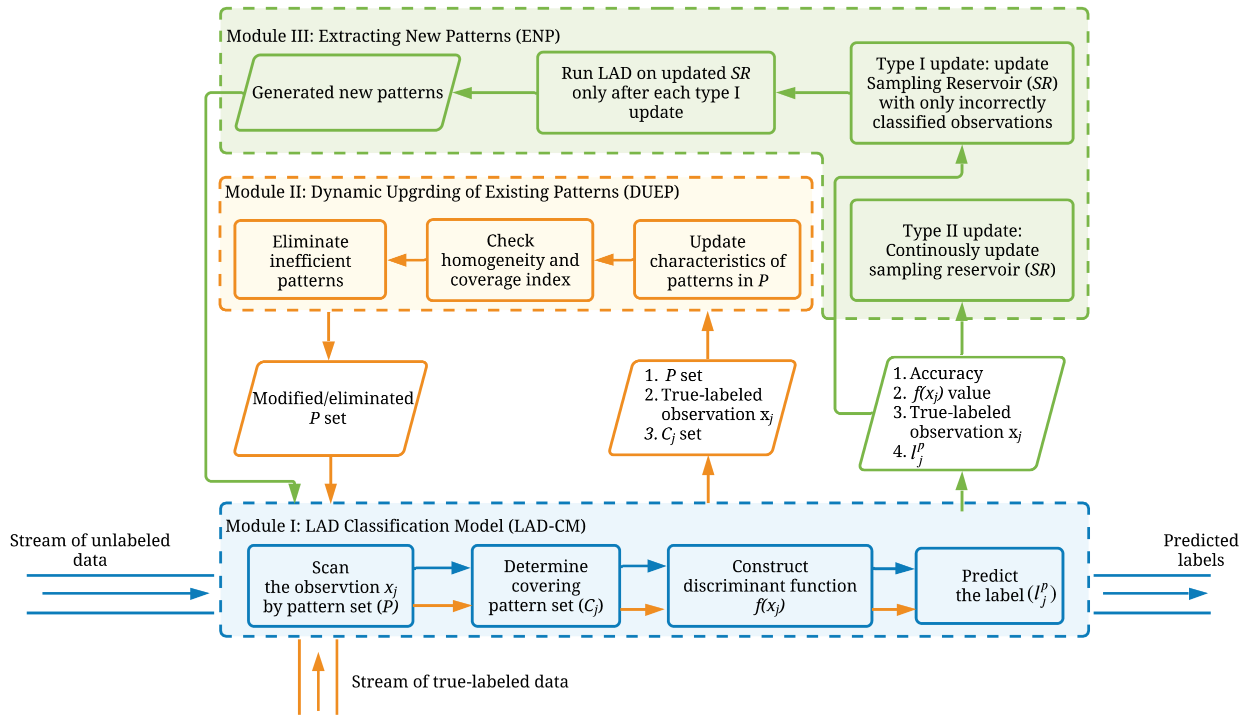

Module III: Extracting New Patterns (ENP)
Generated new patterns
Run LAD on updated *SR* only after each type I update
Type I update: update Sampling Reservoir (*SR*) with only incorrectly classified observations
Module II: Dynamic Upgrding of Existing Patterns (DUEP)
Eliminate inefficient patterns
Check homogeneity and coverage index
Update characteristics of patterns in *P*
Type II update: Continously update sampling reservoir (*SR*)
Modified/eliminated *P* set
1. *P* set 2. True-labeled observation x_j 3. C_j set
1. Accuracy 2. $f(x_j)$ value 3. True-labeled observation x_j 4. l_j^p
Module I: LAD Classification Model (LAD-CM)
Stream of unlabeled data
Scan the observtion x_j by pattern set (*P*)
Determine covering pattern set (C_j)
Construct discriminant function $f(x_j)$
Predict the label (l_j^p)
Predicted labels
Stream of true-labeled data

Figure 5.4 Dynamic and Adaptive Logical Analysis of Streaming Data (DA-LASD) Framework [2]

classification decision, i.e., continuing to cover observations of its own class as they flow in. Patterns with a coverage index below a certain threshold are considered decayed $\delta < \delta_{th}$.

In addition to model performance measures, such as accuracy, sensitivity and specificity, decayed versions of them were introduced to account to a larger extent for the model's performance in the recent part of the data stream than its performance earlier at the beginning of the stream in addition to measure its resilience and sustainability. All performance measures developed as evaluation criteria are discussed in detail in the next section.

5.4 Case Study

5.4.1 Dataset

In this paper, the dataset comes from a jet engine failure detection simulation using Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) [80]. The simulation results are for 100 jet engines. Every engine runs for a different number of cycles up to failure, ranging from 127 to 361 cycles, with a total of 20631 cycles for the 100 engines. For every cycle, the readings of 21 sensors measuring engine health, as well as 3 operational settings, are measured. The descriptions of the sensors are illustrated in Table 5.1 [1].

The simulation begins for every engine with different values of initial wear and manufacturing variations [81]. Then a fault starts to develop at an unknown point along the engine's lifetime. This fault results in degradation in engine health measures, continuing up to its failure.

5.4.2 Problem Generalization

In general, running a jet engine is considered a process, and engine health measures are considered process behavior measures. Since the cycle at which a fault is developed is unknown, the out-of-control states is assumed to be after a specific cycles threshold before the engine failure. At such threshold cycle, the process is considered out-of-control and corrective actions are needed to be taken. The data set will flow to the model as a stream of data, i.e., data observations will stream from one engine after another. In order to create a concept drift in the data stream, two out-of-control thresholds are used, namely 15 and 30 cycles before failure, for first 50 engines and last 50 engines, respectively. This threshold could be more restricted or more released according to the practical situations.

Therefore, we consider streaming data for the 100 engine life cycles of a total 20,631 observations flowing in as a data stream. For every observation, the features are the previously mentioned 24 measures, and the label is either in control (negative - before the threshold)

Table 5.1 Description of Dataset Features [1]

Symbol	Type	Description	Units
f1	Setting 1	Operational regime setting	
f2	Setting 2	Operational regime setting	
f3	Setting 3	Operational regime setting	
f4	Sensor 1	Total temperature at fan inlet	$^\circ R$
f5	Sensor 2	Total temperature at LPC outlet	$^\circ R$
f6	Sensor 3	Total temperature at HPC outlet	$^\circ R$
f7	Sensor 4	Total temperature at LPT outlet	$^\circ R$
f8	Sensor 5	Pressure at fan inlet	psia
f9	Sensor 6	Total pressure in bypass-duct	psia
f10	Sensor 7	Total pressure at HPC outlet	psia
f11	Sensor 8	Physical fan speed	rpm
f12	Sensor 9	Physical core speed	rpm
f13	Sensor 10	Engine pressure ratio	
f14	Sensor 11	Static pressure at HPC outlet	psia
f15	Sensor 12	Ratio of fuel flow to Ps30	pps/psi
f16	Sensor 13	Corrected fan speed	rpm
f17	Sensor 14	Corrected core speed	rpm
f18	Sensor 15	Bypass ratio	
f19	Sensor 16	Burner fuel-air ratio	
f20	Sensor 17	Bleed Enthalpy	
f21	Sensor 18	Demanded fan speed	rpm
f22	Sensor 19	Demanded corrected fan speed	rpm
f23	Sensor 20	HPT coolant bleed	lbm/s
f24	Sensor 21	LPT coolant bleed	lbm/s

or out-of-control (positive - after the threshold). This leads to a generalized classification problem for streaming data–the DA-LASD is therefore used for this problem.

5.4.3 Evaluation Criteria

The prequential interleaved test-then-train technique is used for the evaluation through the data stream [5]. In which, each individual labeled observation is used to test the model before it is used for training process. Therefore, evaluation measures; such as accuracy, sensitivity, and specificity can be incrementally computed.

Accuracy The first evaluation criterion is the accuracy which is the ratio of the number of correctly classified observations to the total number of observations. It is a measure of the quality of learning. In the case of streaming data, this accuracy can be recomputed every time a new observation x_j is received, and this can be computed as follow:

$$A_j = \frac{E_j}{B_j} \tag{5.1}$$

where

$$E_j = L_j + E_{j-1}, \quad B_j = 1 + B_{j-1} \tag{5.2}$$

where L_j is the loss function for each observation x_j and is equal to 1 if x_j is correctly classified and 0 otherwise. E_j and B_j are the number of observations correctly classified, and the total number of observations that are received, respectively up to observation x_j. $x_j \in S$, where $j = [1, ..., |S|]$. S is the streaming data [5].

Sensitivity It is the proportion of positive class observations get correctly classified. It is the most important measure for industrial process control application, which reflects how correctly the model detect out-of-control states. In the case of streaming data, this sensitivity can be recomputed only every time a new positive observation is received, and this can be computed as follow:

$$Sensitivity_j = \frac{TP_j}{P_j} \tag{5.3}$$

where

$$TP_j = L_j + TP_{j-1}, \quad P_j = 1 + P_{j-1} \tag{5.4}$$

L_j is the loss function for each observation x_j. TP_j and P_j are the number of faulty observations correctly classified, and the total number of faulty observations that are received, respectively up to observation x_j.

Specificity Which is the proportion of the negative class observations got correctly classified, i.e., the proportion of healthy observations correctly classified to the total number of healthy observations. This is criteria measures the capability of the model to correctly distinguish the healthy observations and not give a false alarm which could interrupt the process and increase the total cost. In the case of streaming data, the specificity can be also recomputed only every time a new negative observation is received and this can be computed as follow:

$$Specificity_j = \frac{TF_j}{F_j} \tag{5.5}$$

where

$$TF_j = L_j + TF_{j-1}, \quad F_j = 1 + F_{j-1} \tag{5.6}$$

L_j is the loss function for each observation x_j. FP_j and F_j are the number of healthy observations correctly classified, and the total number of healthy observations that are received, respectively up to observation x_j.

Since these evaluation measures consider all the observations in the entire data stream, they may not accurately reflect how the model resilience and sustainability are in the recent part of the data stream. Therefore, as it is more important how the model performance has been recently, than how it was long ago at the beginning of the stream, the concept of decayed evaluation measures is introduced to have an indicator for the model resilience and sustainability.

The decayed accuracy, decayed sensitivity and decayed specificity of the model for only the most recent data are computed using a forgetting mechanism with fading factors that weigh data using a decay factor α. Decayed accuracy DA_j is computed as follows [5]:

$$DA_j = \frac{DE_j}{DB_j} \tag{5.7}$$

where

$$DE_j = L_j + \alpha \times DE_{j-1}, \quad DB_j = 1 + \alpha \times DB_{j-1} \tag{5.8}$$

Similarly, both sensitivity and specificity are computed as follows:

$$D.\ Sensitivity_j = \frac{DTP_j}{DP_j} \tag{5.9}$$

where

$$DTP_j = L_j + \alpha \times DTP_{j-1}, \quad DP_j = 1 + \alpha \times DP_{j-1} \tag{5.10}$$

$$D.\ Specificity_j = \frac{DTF_j}{DF_j} \tag{5.11}$$

where

$$DTF_j = L_j + \alpha \times DTF_{j-1}, \quad DF_j = 1 + \alpha \times DF_{j-1} \tag{5.12}$$

5.5 Results

5.5.1 Feature Engineering

Based on the proposed modeling framework, the first portion of data enters different feature engineering steps to determine the set of most relevant/significant features. In this section, the results of each feature engineering step is presented.

Feature Selection This step starts by eliminating lowly-variable features that can be considered constants. Robust scaler technique was used firstly to transform the features into a similar scale. Robust scaler is performed by subtracting the median and then dividing by the interquartile range. Box plot of each scaled feature is shown in Figure 5.5. The variances of scaled features was calculated and is shown in 5.6. The variance threshold $\sigma^2_{th} = 0.2$ has been selected to eliminate features with variance below it. Eight features were excluded in this step; f3, f4, f8, f9, f13, f19, f21 and f22.

Then, the voting-based feature selection mechanism, with the three forementioned classifiers, random forests, gradient boosting and LASSO, was used in this step to select the most significant features with the majority vote. Random forests and gradient boosting classifiers were performed with recursive feature eliminations (RFE) technique [77] to optimize selecting features to be eliminated and their number. The final selected feature must have the majority vote by the classifiers. Table 5.2 shows the selected features by each classifiers and the voting outputs. At the end of voting process, 10 features were selected by the mechanism.

Feature Extraction In this step, three new features were extracted out of each previously selected feature using different statistics, namely the 10-cycle moving average, 10-cycle

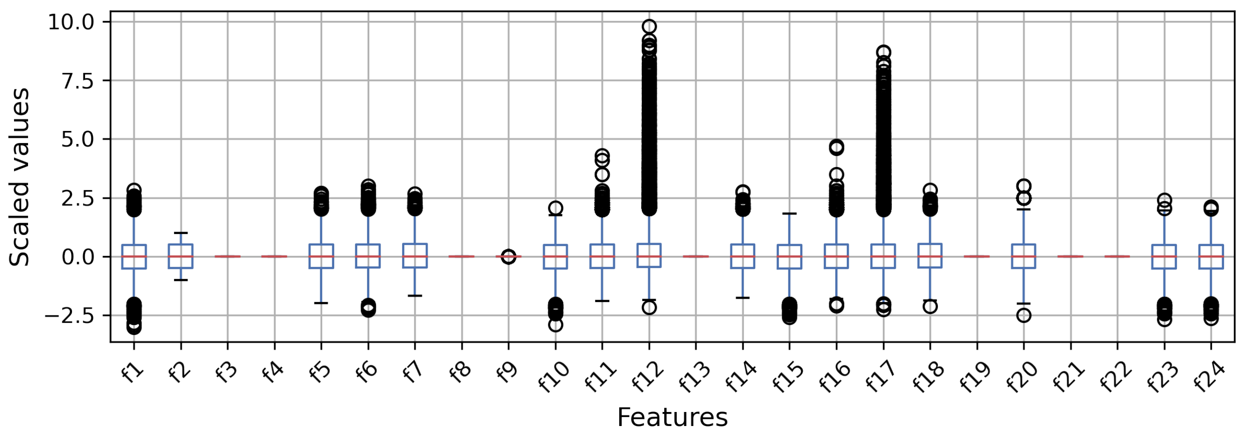


Figure 5.5 Box plots of the scaled values of data features

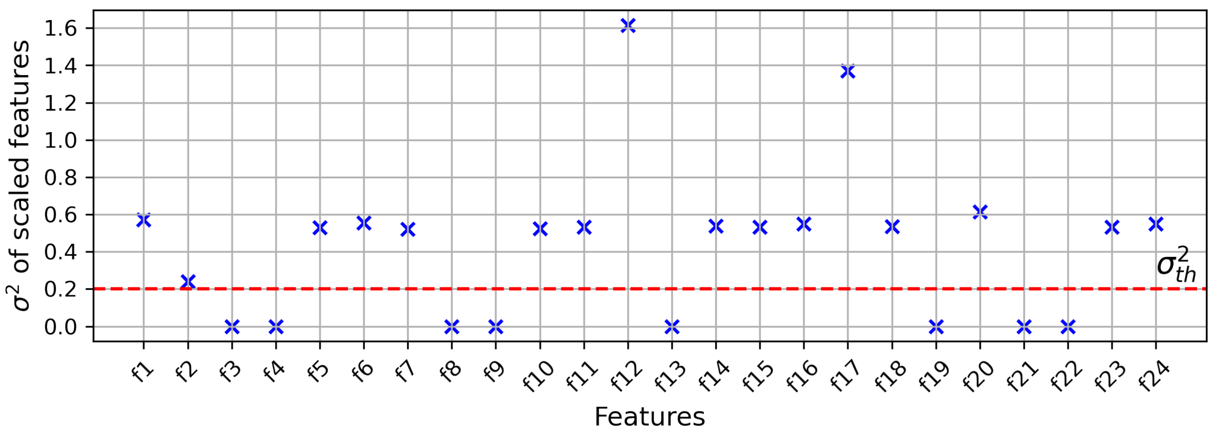


Figure 5.6 Variance of the scaled values of data features

Table 5.2 Results of voting-based feature selection mechanism

	f1	f2	f5	f6	f7	f10	f11	f12	f14	f15	f16	f17	f18	f20	f23	f24
LASSO	0	0	1	1	1	1	1	1	1	1	1	0	1	1	1	1
Gradient Boosting	0	0	1	0	1	1	0	1	1	1	0	1	1	0	1	0
Random Forests	0	0	1	0	1	1	0	1	1	1	0	1	1	0	1	1
Voting score	0	0	3	1	3	3	1	3	3	3	1	2	3	1	3	2
Selected Features	-	-	Y	-	Y	Y	-	Y	Y	Y	-	Y	Y	-	Y	Y

moving standard deviation, and percentage change from the 10-cycle moving average. These statistical properties can change over time in the case of the process is out-of-control. As such changes should be captured by the model if exist. This step resulted in having 30 new features and 40 features in total.

Reapplying Feature Selection In this last step of feature engineering, the same voting-based feature selection mechanism was reapplied on the total 40 features. Finally, 13 features were selected with majority votes from two classifiers. The final 13 selected features with their description are illustrated in Table 5.3.

Table 5.3 Final selected features

Symbol	Description
f5-1	Moving average of Sensor 2
f7-1	Moving average of Sensor 4
f10-1	Moving average of Sensor 7
f12	Sensor 9
f12-1	Moving average of Sensor 9
f12-2	Moving standard deviation of Sensor 9
f14	Sensor 11
f14-1	Moving average of Sensor 11
f15-1	Moving average of Sensor 12
f17-1	Moving average of Sensor 14
f18-1	Moving average of Sensor 15
f23-1	Moving average of Sensor 20
f24-1	Moving average of Sensor 21

5.5.2 DA-LASD Results

A model of DA-LASD was generated using the selected features in Table 5.3, we called it DA-LASD-FE (DA-LASD with feature engineering). Another model was generated from the data without any feature engineering process, we called it DA-LASD. The trends of the different decayed accuracy measures along the data stream, as well as the number of generated patterns, are displayed for both of the two models in Figure 5.7. As expected, sudden drop in decayed sensitivity occurred after the concept drift point nearly to observation 10K. This is happened because the out-of-control threshold was shifted from 15 to 30 cycles preceding a complete failure, and hence more out-of-control observations are considered in the stream. It is obvious that DA-LASD-FE model was able to adapt to the new concept which is represented by an uptrend of the decayed sensitivity. Despite DA-LASD model without feature engineering generated three times patterns more than DA-LASD-FE after the concept drift, it was not able to adapt to this change and enhance its sensitivity. This shows how the proposed DA-LASD-FE is quite sustainable in dynamically adapting to the change in the data stream, and consequently maintaining high accuracy measures, specifically, the sensitivity which is very important for monitoring and control industrial processes and fault prognosis applications. In addition, DA-LASD-FE shows a high resilience before and after the concept drift with consistency in the evaluation criteria. Moreover, the small number of the generated patterns by DA-LASD-FE, compared to DA-LASD, makes DA-LASD-FE has better interpretability [39]. Table 5.4 shows two of the most powerful patterns presenting the out-of-control process states. At end of the stream, the protected homogeneity values $\hat{h}$ for these two patterns are 0.94 and 0.98 respectively. And the coverage index values δ are 1 and 0.99, respectively. For example, if the model predict an out-of-control state and pattern 1 is a covering pattern, DA-LASD will guide the controlling system by demanding one or more of following corrective actions: (1) decrease the total temperature at LPT outlet until get moving average lower than 1411 $\circ$R, (2) increase the total pressure at HPC outlet until get moving average higher than 553 psia, (3) decrease the physical core speed until get moving average lower than 9048 rpm, or (4) increase the physical fan speed until get moving average 47.62 rpm, and hence the engine will be back to in-control state.

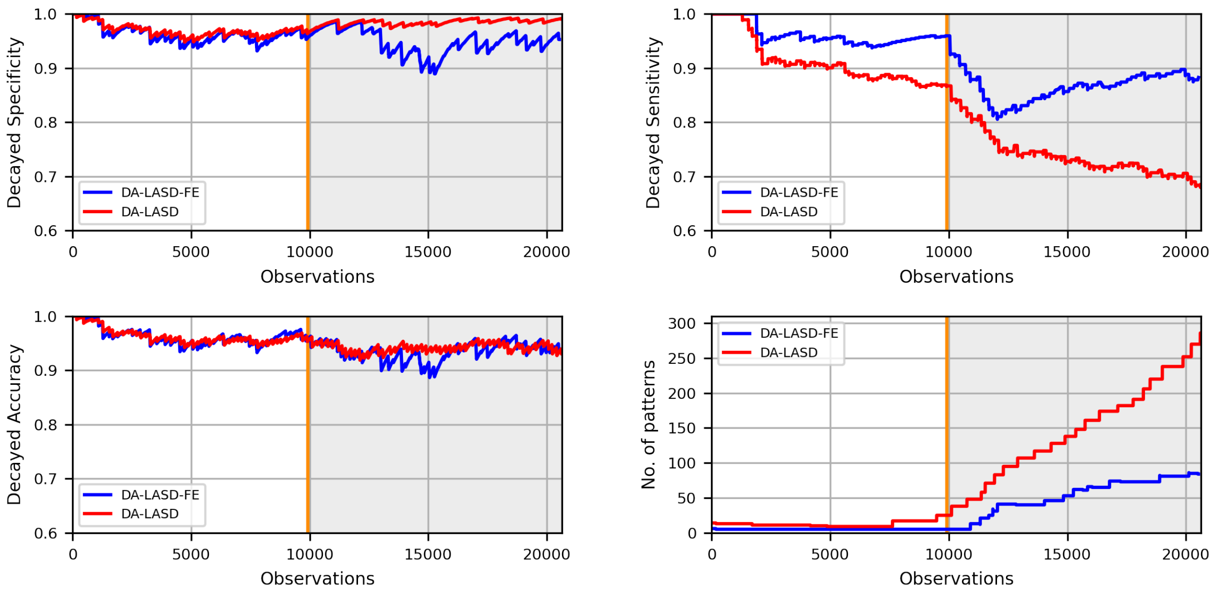


Figure 5.7 DA-LASD-FE and DA-LASD performances

Table 5.4 Two of the most powerful patterns for out-of-control states

Symbol	Description	Pattern 1	Pattern 2
f5-1	Moving average of Sensor 2	–	> 642.84
f7-1	Moving average of Sensor 4	> 1411.12	–
f10-1	Moving average of Sensor 7	< 553.17	< 553.07
f12-1	Moving average of Sensor 9	> 9048	–
f12-2	Moving STD of Sensor 9	–	> 3.445
f14-1	Moving average of Sensor 11	< 47.6205	–
f24-1	Moving average of Sensor 21	–	< 23.21

5.5.3 Comparative Analysis

Finally, Table 5.5 and Figure 5.8 provide a comparison between the proposed DA-LASD-FE model and other machine learning techniques capable of handling streaming data, namely Hoeffding Tree (HT) [15], Hoeffding Adaptive Tree (HAT) [16] and Self Adjusting Memory K nearest neighbors (SAMKNN) [17]. However other ML techniques provide higher average decayed accuracies than DA-LASD-FE through the stream, that is considered a deceiving evaluation as the stream is imbalanced and their average decayed sensitivity are low, i.e, their abilities to detect out-of-control states are low. The ability to detect these states is the most important feature in the context of process control. In order to statistically compare the resilience and the sustainability of DA-LASD-EN with the different models in terms of the sensitivity through the stream, the Friedman test, a non-parametric statistical test, is performed between them. A decayed sensitivity value is recorded for each model each one thousand received observations through the stream for this statistical test. The test is conducted in two phases. Phase 1 is to evaluate the significant difference between the means of the sensitivity for all the models. Phase 2 is to pairwise evaluate the significant difference between DA-LASD-FE and the others. Table 5.6 shows the result of phase 1 that indicates there is a high significant difference between the means. Table 5.7 presents the results of phase 2 which indicate that the proposed DA-LASD-FE outperforms statistically other ML models in terms of the decayed sensitivity. That means DA-LASD-FE statistically is more resilient and sustainable through the data stream. What makes the proposed DA-LASD-FE quite promising is how it combines these resilience and sustainability with its distinctive interpretability power, which is essential in providing physical meanings and describing hidden phenomena behind the out-of-control cases and help to intelligently guide the controlling systems to take automated corrective actions. This is particularly important in advanced industrial applications, especially in Industry 4.0 paradigm.

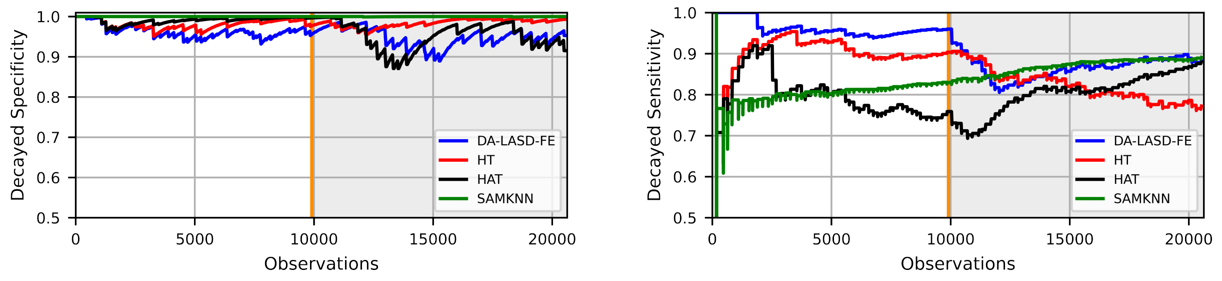


Figure 5.8 Performances of DA-LASD and other ML techniques

Table 5.5 Comparison between DA-LASD-FE and different ML techniques

Model	*Avg. D. Sensitivity*	*Avg. D. Specificity*	*Avg. DA*
DA-LASD-FE	0.913	0.955	0.958
DA-LASD	0.824	0.977	0.961
HT	0.863	0.982	0.963
HAT	0.844	0.971	0.957
SAMKNN	0.839	0.999	0.985

Table 5.6 Sensitivity - Friedman test phase 1

Model	Sum of ranks (R)	Mean of ranks
DA-LASD-FE	29	1.61
DA-LASD	69	3.83
HT	45	2.50
HAT	77	4.27
SAMKNN	50	2.77
$F_r = 32.8 > F_{cr} = 9.49$ $p\text{-}value = 1.31\text{E}{-}06 < 0.05$		$\rightarrow$ highly significant

Table 5.7 Sensitivity - Friedman test phase 2

Model 1	Model 2	*p-value*	Significance
DA-LASD-FE	DA-LASD	7.63E−06	Yes
DA-LASD-FE	HT	4.19E−04	Yes
DA-LASD-FE	HAT	7.62E−06	Yes
DA-LASD-FE	SAMKNN	2.36E−02	Yes

5.6 Conclusion

In this paper, a resilient and sustainable Dynamic and Adaptive Logical Analysis of Streaming Data (DA-LASD) model reinforced with feature engineering mechanism is proposed to develop an intelligent and interpretable monitoring system for turbofan engine. This monitoring system helps in controlling the operational process of the turbofan engine. The feature engineering mechanism aims to determine the most relevant/significant variables of the process. It consists of two main steps, feature selection and time series feature extraction. Feature selection compromises a voting mechanism of three classifiers; random forests, gradient boosting classifier and LASSO. The concept of decayed evaluation measures is used to have an indicator for resilience and sustainability of a given model. Through an aerospace case study of turbojet engine failure detection, the proposed model DA-LASD with feature engineering (DA-LASD-FE) showed high resilience and sustainability, and was able to dynamically adapt to the concept drift. The proposed model statistically outperformed, in this context, the DA-LASD model without a feature engineering. The proposed feature engineering determined the most 13 relevant/significant features that enhanced the DA-LASD-FE sustainability and made it be able to capture the new drift with a less number of new patterns. In addition to the provided interpretable patterns that are most important for automated corrective actions in intelligent process control systems, DA-LASD-FE model outperformed statistically other machine learning techniques that handle streaming data in terms of resilience and sustainability. This makes the proposed DA-LASD-FE quite promising in industrial advanced process controls.

Further research The DA-LASD-FE introduced for turbofan jet engine in this paper could be investigated in another industrial process control applications, specifically in Industry 4.0 application where the autonomous and intelligent controlling systems are demanded. Other direction to extend this model would be to enhance the interpretability by extracting an online feature importance based on their appearance frequency in the available pattern set and how efficient and decayed these patterns. As such ability will give the controlling system the most important features that could be targeted to control the process.

CHAPTER 6 GENERAL DISCUSSION

The ultimate objective in this research work is achieved by providing robust, resilient, sustainable, and interpretable classification model that able to be utilized in the demanded autonomous controlling systems in the different applications in Industry 4.0. The classification model is based on Logical Analysis of Data (LAD) which is a robust classification technique that provides the needed interpretability through generating patterns containing structural knowledge that explain the hidden phenomena under study. This research overcomes the computational time limitation of LAD by developing accelerated ensemble LAD (LAD-ENS) maintaining LAD's robustness and interpretability in high levels. This mechanism allows LAD to be implemented in distributed computing systems and to handle large datasets.

In order to enhance the resilience and sustainability of LAD, a dynamic and adaptive version of LAD called DA-LASD is developed. DA-LASD is able to process and learn from streaming data. Such version is able to monitor its performance while it is on the service and able to detect any drift in the concept through the streaming data, and hence adapt itself to the new data. The high resilience and sustainability allowed LAD to be applied on applications of Industry 4.0, specifically the industrial process control. In order to evaluate the developed DA-LASD on practical, high-dimensional, and dynamic data; this research work applied DA-LASD on industrial dataset to develop an intelligent and interpretable monitoring system for turbofan engine operational process by reinforcing DA-LASD with feature engineering. The model gives accurate detection of faults and provide high resilience and sustainability, specifically after a concept drift.

These modifications and developed models, LAD-ENS and DA-LASD, allow LAD to be utilized in applications where large volume of data or data stream exists, and interpretability is required, namely, in industry 4.0 applications thanks to their robustness, resilience, sustainability, and interpretability.

CHAPTER 7 CONCLUSION AND RECOMMENDATIONS

7.1 Summary of Works

In this doctoral research, an ensemble LAD system has been developed and called LAD-ENS to enhance computation time and to process large volumes of data. This ensemble system was built based on stratified sampling without a replacement technique, in addition to a proposed combining mechanism that integrates all patterns that are provided by the individual LAD classifiers. This mechanism combines the knowledge at the level of patterns and then formulates a new ensemble discriminant function. This mechanism preserves the explanatory power of LAD patterns and does not reduce their interpretability like the voting mechanism does. By means of this system, LAD-ENS was successfully run on cloud computing clusters. The LAD-ENS was evaluated in terms of computational time, accuracy, pattern quality and comprehensibility. The statistical Friedman tests revealed that LAD-ENS significantly outperforms classical LAD models in terms of computational performance. Moreover, Friedman tests revealed that very competitive accuracy is obtained. Although the patterns may lose their purity, the patterns of LAD-ENS have lower degrees than those of classical LAD, which enhanced the comprehensibility index.

In addition, a Dynamic and Adaptive Logical Analysis of Streaming Data (DA-LASD) framework was proposed. It is a LAD-based classifier capable of processing data streams. The DA-LASD consists of three main modules. Module I, LAD-CM, is the LAD classifier model, which classifies observations based on the set of patterns it contains. However, this set of patterns has to be continuously updated to adapt to the changes in the data streams. The update is done by modifying the characteristics of existing patterns, eliminating decayed and inefficient patterns, and added new patterns. The first two steps are done in the Module II: Dynamic Update of Existing Patterns, DUEP. The third step is taken care of in the Module III: Extracting New Patterns, ENP. In order to correctly detect decayed and inefficient patterns, several modifications on the classical pattern characteristics are proposed. Particularly, the covering and opposite covering characteristics are normalized, and the homogeneity is normalized and protected. This enables the framework to take into account any data imbalance that may exist. Moreover, a new characteristic is proposed, namely the covering index, δ, which is an updated measure of the pattern's ability to cover observations of its own class as they continue to flow in. The proposed DA-LASD is evaluated on several synthetic data streams generated by different generating functions and containing different types and numbers of concept drifts. The detailed analysis of the experimentation results shows how the proposed framework dynamically adapts the model, and successfully improves any performance measures that start to decline, i.e., successfully maintains high levels resilience and sustainability. This accuracy of the DA-LASD is found to be competitive when compared with other machine learning techniques that handle streaming data.

Additionally, a resilient and sustainable Dynamic and Adaptive Logical Analysis of Streaming Data reinforced with feature engineering (DA-LASD-FE) is proposed for is proposed to develop an intelligent and interpretable monitoring system for turbofan engine. This monitoring system helps in controlling the operational process of the turbofan engine. The feature engineering aims to determine the most relevant/significant attributes of the system. It consists of two main steps, feature selection and time series feature extraction. Feature selection compromises a voting mechanism of three classifiers; random forests, gradient boosting classifier and LASSO. Through an aerospace case study of turbojet engine failure detection, the proposed model DA-LASD with feature engineering (DA-LASD-FE) showed high resilience and sustainability, and was able to dynamically adapt to the concept drift and statistically outperformed, in this context, the DA-LASD model without a feature engineering. The proposed feature engineering determined the most 13 relevant/significant features that enhanced the DA-LASD-FE performance and made it be able to capture the new drift with a less number of new patterns. In addition to the provided interpretable patterns that are most important for automated corrective actions in intelligent process control systems, DA-LASD-FE model outperformed statistically other machine learning techniques that handle streaming data. This makes the proposed DA-LASD-FE quite promising in industrial advanced process controls. Figure 7.1 shows the main outcomes and their abilities.

7.2 Limitations

LAD-ENS and DA-LASD still have relatively high computational time comparing to other machine learning techniques. In addition, DA-LASD must have initial patterns before starting to process data streams. That limits DA-LASD to be applied directly before having some historical data.

7.3 Future Research

For the ensemble LAD system introduced in research, it could be extended into many different directions. One of these directions would be to use various sampling methods to enhance the quality of the data subsets provided to the pattern generation processes. The feature level could be another direction, focusing mainly on the features of original data in order

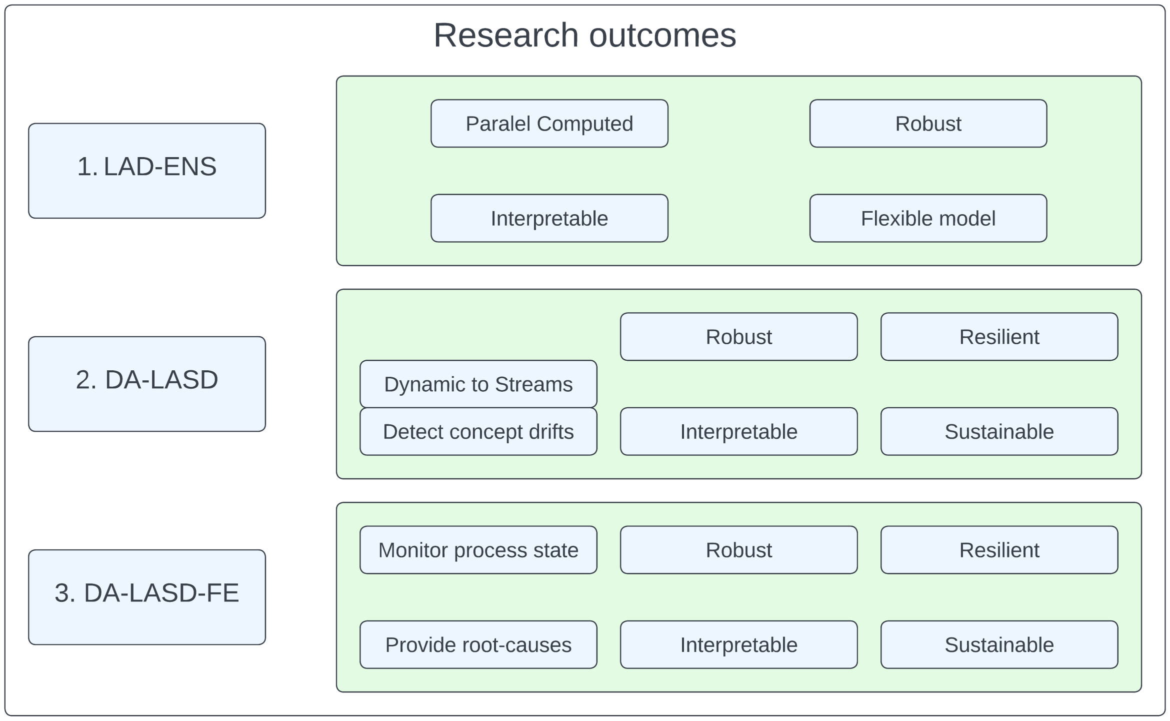


Figure 7.1 Research summary and outcomes

to select appropriate subsets, or sample the features in subsets and provide them to the pattern generation processes with an aim to enhance accuracy. The combining mechanism level is another direction, which would focus mainly on enhancing the combining process and selecting the most appropriate patterns. This research direction could enhance the explanatory power by reducing the number of patterns.

For the DA-LASD framework could be extended to another industrial process control applications, specifically in Industry 4.0 application where the autonomous and intelligent controlling systems are demanded. Other direction to extend this framework to enhance the interpretability would be by extracting an online feature importance based on their appearance frequency in the available pattern set and how efficient and decayed these patterns. As such ability will give the controlling system the most important features that could be targeted to control the process.

LAD-ENS and DA-LASD are developed only for classification problems. They do not consider regression problems in this research. The can be extended for regression problems in order to tackle different varieties of Industry 4.0 applications.

REFERENCES

[1] J. Xu and L. Xu, *Integrated System Health Management: Perspectives on Systems Engineering Techniques.* Academic Press, 2017.

[2] O. Elfar, H. Osman, S. Yacout, and A. Mohammed, "Dynamic and adaptive logical analysis of streaming data with concept drifts," *Manuscript submitted for publication*, 2022.

[3] H. Lasi, P. Fettke, H.-G. Kemper, T. Feld, and M. Hoffmann, "Industry 4.0," *Business & information systems engineering*, vol. 6, no. 4, pp. 239–242, 2014.

[4] M. Ghobakhloo, "Industry 4.0, digitization, and opportunities for sustainability," *Journal of cleaner production*, vol. 252, p. 119869, 2020.

[5] A. Bifet, R. Gavalda, G. Holmes, and B. Pfahringer, *Machine learning for data streams: with practical examples in MOA.* MIT press, 2018.

[6] S. Alexe and P. L. Hammer, "Accelerated algorithm for pattern detection in logical analysis of data," *Discrete Applied Mathematics*, vol. 154, no. 7, pp. 1050–1063, 2006.

[7] P. Hansen and C. Meyer, "A new column generation algorithm for logical analysis of data," *Annals of Operations Research*, vol. 188, no. 1, pp. 215–249, 2011.

[8] C.-A. Chou, T. O. Bonates, C. Lee, and W. A. Chaovalitwongse, "Multi-pattern generation framework for logical analysis of data," *Annals of Operations Research*, vol. 249, no. 1, pp. 329–349, 2017.

[9] V. S. Moertini, G. W. Suarjana, L. Venica, and G. Karya, "Big data reduction technique using parallel hierarchical agglomerative clustering." *IAENG International Journal of Computer Science*, vol. 45, no. 1, 2018.

[10] L. Zhou, S. Pan, J. Wang, and A. V. Vasilakos, "Machine learning on big data: Opportunities and challenges," *Neurocomputing*, vol. 237, pp. 350–361, 2017.

[11] Y. Zhang, S. Ren, Y. Liu, and S. Si, "A big data analytics architecture for cleaner manufacturing and maintenance processes of complex products," *Journal of cleaner production*, vol. 142, pp. 626–641, 2017.

[12] H. Li, R. Liu, J. Wang, and Q. Wu, "An enhanced and efficient clustering algorithm for large data using mapreduce," *IAENG International Journal of Computer Science*, vol. 46, no. 1, pp. 61–67, 2019.

[13] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[14] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, pp. 785–794.

[15] P. Domingos and G. Hulten, "Mining high-speed data streams," in *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2000, pp. 71–80.

[16] G. Hulten, L. Spencer, and P. Domingos, "Mining time-changing data streams," in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, 2001, pp. 97–106.

[17] J. Read, A. Bifet, B. Pfahringer, and G. Holmes, "Batch-incremental versus instance-incremental learning in dynamic and evolving data," in *International symposium on intelligent data analysis*. Springer, 2012, pp. 313–323.

[18] C. C. Aggarwal *et al.*, *Data mining: the textbook*. Springer, 2015, vol. 1.

[19] H. M. Gomes, J. P. Barddal, F. Enembreck, and A. Bifet, "A survey on ensemble learning for data stream classification," *ACM Computing Surveys (CSUR)*, vol. 50, no. 2, pp. 1–36, 2017.

[20] H. M. Gomes, J. Read, A. Bifet, and R. J. Durrant, "Learning from evolving data streams through ensembles of random patches," *Knowledge and Information Systems*, vol. 63, no. 7, pp. 1597–1625, 2021.

[21] J. S. Oakland, *Statistical process control*. Routledge, 2007.

[22] A. Apsemidis, S. Psarakis, and J. M. Moguerza, "A review of machine learning kernel methods in statistical process monitoring," *Computers & Industrial Engineering*, vol. 142, p. 106376, 2020.

[23] R. M. Khalifa, S. Yacout, and S. Bassetto, "Developing machine-learning regression model with logical analysis of data (lad)," *Computers & Industrial Engineering*, vol. 151, p. 106947, 2021.

[24] Y. Zhao, S. Wang, and F. Xiao, “Pattern recognition-based chillers fault detection method using support vector data description (svdd),” *Applied Energy*, vol. 112, pp. 1041–1048, 2013.

[25] Y. Zhao, F. Xiao, J. Wen, Y. Lu, and S. Wang, “A robust pattern recognition-based fault detection and diagnosis (fdd) method for chillers,” *HVAC&R Research*, vol. 20, no. 7, pp. 798–809, 2014.

[26] G. Li, Y. Hu, H. Chen, L. Shen, H. Li, M. Hu, J. Liu, and K. Sun, “An improved fault detection method for incipient centrifugal chiller faults using the pca-r-svdd algorithm,” *Energy and Buildings*, vol. 116, pp. 104–113, 2016.

[27] Y. Chetouani, “A neural network approach for the real-time detection of faults,” *Stochastic Environmental Research and Risk Assessment*, vol. 22, no. 3, pp. 339–349, 2008.

[28] K. Patan and T. Parisini, “Identification of neural dynamic models for fault detection and isolation: the case of a real sugar evaporation process,” *Journal of Process Control*, vol. 15, no. 1, pp. 67–79, 2005.

[29] K. Wang, M. G. Forbes, B. Gopaluni, J. Chen, and Z. Song, “Systematic development of a new variational autoencoder model based on uncertain data for monitoring nonlinear processes,” *IEEE Access*, vol. 7, pp. 22 554–22 565, 2019.

[30] J. I. Mireles Gonzalez, “Deep recurrent neural networks for fault detection and classification,” Master’s thesis, University of Waterloo, 2018.

[31] Y. Tharrault, G. Mourot, and J. Ragot, “Fault detection and isolation with robust principal component analysis,” in *2008 16th Mediterranean Conference on Control and Automation.* IEEE, 2008, pp. 59–64.

[32] B. Schölkopf, A. Smola, and K.-R. Müller, “Nonlinear component analysis as a kernel eigenvalue problem,” *Neural computation*, vol. 10, no. 5, pp. 1299–1319, 1998.

[33] I. Jaffel, O. Taouali, M. F. Harkat, and H. Messaoud, “Moving window kpca with reduced complexity for nonlinear dynamic process monitoring,” *ISA transactions*, vol. 64, pp. 184–192, 2016.

[34] H. Lahdhiri, M. Said, K. B. Abdellafou, O. Taouali, and M. F. Harkat, “Supervised process monitoring and fault diagnosis based on machine learning methods,” *The International Journal of Advanced Manufacturing Technology*, vol. 102, no. 5, pp. 2321–2337, 2019.

[35] S. Yacout, "Fault detection and diagnosis for condition based maintenance using the logical analysis of data," in *The 40th International Conference on Computers & Indutrial Engineering*. IEEE, 2010, pp. 1–6.

[36] M.-A. Mortada, S. Yacout, and A. Lakis, "Diagnosis of rotor bearings using logical analysis of data," *Journal of Quality in Maintenance Engineering*, 2011.

[37] A. Ragab, M. El-Koujok, B. Poulin, M. Amazouz, and S. Yacout, "Fault diagnosis in industrial chemical processes using interpretable patterns based on logical analysis of data," *Expert Systems with Applications*, vol. 95, pp. 368–383, 2018.

[38] Y. Shaban, S. Yacout, M. Balazinski, and K. Jemielniak, "Cutting tool wear detection using multiclass logical analysis of data," *Machining Science and Technology*, vol. 21, no. 4, pp. 526–541, 2017.

[39] O. Elfar, S. Yacout, and H. Osman, "Accelerating logical analysis of data using an ensemble-based technique." *Engineering Letters*, vol. 29, no. 4, 2021.

[40] I. Odun-Ayo, C. Okereke, and H. Orovwode, "Cloud computing and internet of things: Issues and developments," in *Lecture Notes in Engineering and Computer Science: Proceedings of The World Congress on Engineering 2018*, 2018.

[41] S. R. Safavian and D. Landgrebe, "A survey of decision tree classifier methodology," *IEEE transactions on systems, man, and cybernetics*, vol. 21, no. 3, pp. 660–674, 1991.

[42] B. Schölkopf, A. J. Smola, F. Bach *et al.*, *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[43] M. T. Hagan, H. B. Demuth, and M. Beale, *Neural network design*. PWS Publishing Co., 1997.

[44] E. Boros, P. L. Hammer, T. Ibaraki, and A. Kogan, "Logical analysis of numerical data," *Mathematical programming*, vol. 79, no. 1, pp. 163–190, 1997.

[45] E. Boros, P. L. Hammer, T. Ibaraki, A. Kogan, E. Mayoraz, and I. Muchnik, "An implementation of logical analysis of data," *IEEE Transactions on knowledge and Data Engineering*, vol. 12, no. 2, pp. 292–306, 2000.

[46] P. L. Hammer and T. O. Bonates, "Logical analysis of data—an overview: From combinatorial optimization to medical applications," *Annals of Operations Research*, vol. 148, no. 1, pp. 203–225, 2006.

[47] A. Bennane and S. Yacout, “Lad-cbm; new data processing tool for diagnosis and prognosis in condition-based maintenance,” *Journal of Intelligent Manufacturing*, vol. 23, no. 2, pp. 265–275, 2012.

[48] M.-A. Mortada, S. Yacout, and A. Lakis, “Fault diagnosis in power transformers using multi-class logical analysis of data,” *Journal of Intelligent Manufacturing*, vol. 25, no. 6, pp. 1429–1439, 2014.

[49] A. Ragab, S. Yacout, M.-S. Ouali, and H. Osman, “Multiple failure modes prognostics using logical analysis of data,” in *2015 Annual Reliability and Maintainability Symposium (RAMS)*. IEEE, 2015, pp. 1–7.

[50] A. Ragab, M.-S. Ouali, S. Yacout, and H. Osman, “Remaining useful life prediction using prognostic methodology based on logical analysis of data and kaplan–meier estimation,” *Journal of Intelligent Manufacturing*, vol. 27, no. 5, pp. 943–958, 2016.

[51] S. Jocelyn, Y. Chinniah, M.-S. Ouali, and S. Yacout, “Application of logical analysis of data to machinery-related accident prevention based on scarce data,” *Reliability Engineering & System Safety*, vol. 159, pp. 223–236, 2017.

[52] A. Ghasemi, S. Esmaeili, and S. Yacout, “Development of equipment failure prognostics model based on logical analysis of data (lad).” *Engineering Letters*, vol. 21, no. 4, 2013.

[53] Y. Crama, P. L. Hammer, and T. Ibaraki, “Cause-effect relationships and partially defined boolean functions,” *Annals of Operations Research*, vol. 16, no. 1, pp. 299–325, 1988.

[54] M. Lejeune, V. Lozin, I. Lozina, A. Ragab, and S. Yacout, “Recent advances in the theory and practice of logical analysis of data,” *European Journal of Operational Research*, vol. 275, no. 1, pp. 1–15, 2019.

[55] L. M. Moreira, “The use of boolean concepts in general classification contexts,” EPFL, Tech. Rep., 2000.

[56] S. Yacout, D. Salamanca, and M.-A. Mortada, “Tool and method for fault detection of devices by condition based maintenance,” Nov. 21 2017, uS Patent 9,824,060.

[57] D. M. Tax and R. P. Duin, “Using two-class classifiers for multiclass classification,” in *Object recognition supported by user interaction for service robots*, vol. 2. IEEE, 2002, pp. 124–127.

[58] C. Zhang and Y. Ma, *Ensemble machine learning: methods and applications.* Springer, 2012.

[59] D. J. Henderson and C. F. Parmeter, *Applied nonparametric econometrics.* Cambridge University Press, 2015.

[60] J. R. Quinlan, *C4. 5: programs for machine learning.* Elsevier, 2014.

[61] H.-L. Nguyen, Y.-K. Woon, and W.-K. Ng, "A survey on data stream clustering and classification," *Knowledge and information systems*, vol. 45, no. 3, pp. 535–569, 2015.

[62] P. L. Hammer, A. Kogan, and M. A. Lejeune, "Reverse-engineering banks' financial strength ratings using logical analysis of data," *Rutgers Center for OR*, 2007.

[63] M.-A. Mortada, T. Carroll, S. Yacout, and A. Lakis, "Rogue components: their effect and control using logical analysis of data," *Journal of Intelligent Manufacturing*, vol. 23, no. 2, pp. 289–302, 2012.

[64] G. Büchi, M. Cugno, and R. Castagnoli, "Smart factory performance and industry 4.0," *Technological Forecasting and Social Change*, vol. 150, p. 119790, 2020.

[65] D. Jacob, "Quality 4.0 impact and strategy handbook: getting digitally connected to transform quality management," *LNS research*, 2017.

[66] D. Brzezinski, L. L. Minku, T. Pewinski, J. Stefanowski, and A. Szumaczuk, "The impact of data difficulty factors on classification of imbalanced and concept drifting data streams," *Knowledge and Information Systems*, vol. 63, no. 6, pp. 1429–1469, 2021.

[67] A. Bifet, G. Holmes, B. Pfahringer, P. Kranen, H. Kremer, T. Jansen, and T. Seidl, "Moa: Massive online analysis, a framework for stream classification and clustering," in *Proceedings of the first workshop on applications of pattern analysis.* PMLR, 2010, pp. 44–50.

[68] W. N. Street and Y. Kim, "A streaming ensemble algorithm (sea) for large-scale classification," in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, 2001, pp. 377–382.

[69] R. Davies, "Industry 4.0 digitalisation for productivity and growth," *European parliamentary research service*, vol. 10, p. 2018, 2015.

[70] L. Thames and D. Schaefer, "Industry 4.0: an overview of key benefits, technologies, and challenges," *Cybersecurity for Industry 4.0*, pp. 1–33, 2017.

[71] H. A. Spang III and H. Brown, “Control of jet engines,” *Control Engineering Practice*, vol. 7, no. 9, pp. 1043–1059, 1999.

[72] Z. Yiyang, J. HUANG, P. Muxuan, and Z. Wenxiang, “Direct thrust control for multivariable turbofan engine based on affine linear parameter-varying approach,” *Chinese Journal of Aeronautics*, 2021.

[73] H. A. Taha, S. Yacout, and Y. Shaban, “Autonomous self-healing mechanism for a cnc milling machine based on pattern recognition,” *Journal of Intelligent Manufacturing*, pp. 1–21, 2022.

[74] V. Mathew, T. Toby, V. Singh, B. M. Rao, and M. G. Kumar, “Prediction of remaining useful lifetime (rul) of turbofan engine using machine learning,” in *2017 IEEE International Conference on Circuits and Systems (ICCS).* IEEE, 2017, pp. 306–311.

[75] A. Muneer, S. M. Taib, S. M. Fati, and H. Alhussian, “Deep-learning based prognosis approach for remaining useful life prediction of turbofan engine,” *Symmetry*, vol. 13, no. 10, p. 1861, 2021.

[76] J. Kazemitabar, A. Amini, A. Bloniarz, and A. S. Talwalkar, “Variable importance using decision trees,” *Advances in neural information processing systems*, vol. 30, 2017.

[77] X.-w. Chen and J. C. Jeong, “Enhanced recursive feature elimination,” in *Sixth International Conference on Machine Learning and Applications (ICMLA 2007).* IEEE, 2007, pp. 429–435.

[78] R. Tibshirani, “Regression shrinkage and selection via the lasso,” *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.

[79] L. Li, Y. Wu, Y. Ou, Q. Li, Y. Zhou, and D. Chen, “Research on machine learning algorithms and feature extraction for time series,” in *2017 IEEE 28th annual international symposium on personal, indoor, and mobile radio communications (PIMRC).* IEEE, 2017, pp. 1–5.

[80] NASA, “Prognostics center of excellence - data repository,” 2008. [Online]. Available: https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/

[81] A. Saxena, K. Goebel, D. Simon, and N. Eklund, “Damage propagation modeling for aircraft engine run-to-failure simulation,” in *2008 international conference on prognostics and health management.* IEEE, 2008, pp. 1–9.