POLYPUBLIE
Polytechnique Montréal

POLYTECHNIQUE MONTRÉAL
UNIVERSITÉ D'INGÉNIERIE

Titre: Développement d'un système de recommandation de produits dans un contexte Business-to-Business
Title:

Auteur: Thamer Saraei
Author:

Date: 2022

Type: Mémoire ou thèse / Dissertation or Thesis

Référence: Saraei, T. (2022). Développement d'un système de recommandation de produits dans un contexte Business-to-Business [Mémoire de maîtrise, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/10705/
Citation:

Document en libre accès dans PolyPublie
Open Access document in PolyPublie

URL de PolyPublie: https://publications.polymtl.ca/10705/
PolyPublie URL:

Directeurs de recherche: Maha Ben Ali, & Jean-Marc Frayret
Advisors:

Programme: Maîtrise recherche en génie industriel
Program:

Ce fichier a été téléchargé à partir de PolyPublie, le dépôt institutionnel de Polytechnique Montréal
This file has been downloaded from PolyPublie, the institutional repository of Polytechnique Montréal

https://publications.polymtl.ca

POLYTECHNIQUE MONTRÉAL

affiliée à l'Université de Montréal

Développement d'un système de recommandation de produits dans un contexte Business-to-Business

THAMER SARAEI

Département de mathématiques et de génie industriel

Mémoire présenté en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*

Génie industriel

Décembre 2022

© Thamer Saraei, 2022.

POLYTECHNIQUE MONTRÉAL

affiliée à l'Université de Montréal

Ce mémoire intitulé :

Développement d'un système de recommandation de produits dans un contexte Business-to-Business

présenté par **Thamer SARAEI**

en vue de l'obtention du diplôme de de *Maîtrise ès sciences appliquées*

a été dûment accepté par le jury d'examen constitué de :

Julie CARREAU, présidente
Maha BEN ALI, membre et directrice de recherche
Jean-Marc FRAYRET, membre et codirecteur de recherche
Michel DESMARAIS, membre

DÉDICACE

À mon cher père, ma chère mère et ma chère épouse Nedra, Ma profonde et sincère gratitude pour toute la confiance et l'amour que vous m'avez donné et pour votre soutien sans fin.

À mon futur bébé, Zayd,

À l'âme de mon cousin MAKRAM qui nous a quittés trop tôt,

À mes frères et sœurs,

À mon cher ami Rzouga,

À tous mes proches, je dédie ce projet.

S.Thamer

REMERCEMENTS

Tout d'abord, je tiens à remercier mon Dieu, qui m'a guidé sur le bon chemin et m'a donné la force et la patience pour accomplir ce travail.

Ensuite, je tiens à remercier l'équipe de Mitacs qui nous a apporté un soutien financier, des ateliers, des conseils et des conditions favorables tout au long de notre stage.

Je tiens également à remercier Mme Maha et M. Jean-Marc, mes directeurs de recherche, qui m'ont préparé un environnement favorable à la recherche. Ils ont su me guider, me conseiller, tout en me laissant libre de mes choix. Leur rigueur scientifique et leur encadrement m'ont permis de mener à bien ce travail.

Mes remerciements vont à tous les professeurs pour les connaissances et le savoir-faire acquis au cours de ma formation à l'École Polytechnique de Montréal.

Mon dernier mot s'adresse à tous les membres du jury pour l'honneur qu'ils me font de participer à l'examen de mon travail. Tous vos commentaires ne pourront qu'améliorer la qualité de ce projet.

RÉSUMÉ

Le filtrage collaboratif est une méthode permettant de recommander des items à un utilisateur actif en se basant sur les évaluations (feedbacks) précédemment fournies par des utilisateurs ayant des intérêts similaires (utilisateurs voisins). Étant donné que le filtrage collaboratif est incapable de découvrir avec précision les utilisateurs voisins similaires lorsque le nombre des évaluations fournies par les utilisateurs est très limité (problème de sparsité des données), la précision de la prédiction des préférences des utilisateurs sera faible. Cette situation se traduit par des recommandations de mauvaise qualité. Dans ce mémoire, notre objectif principal est de développer, pour un partenaire industriel du marché Business-to-Business, un système de recommandation basé sur le filtrage collaboratif à partir de feedbacks implicites uniquement. Pour surmonter le problème de la sparsité des données, nous avons proposé une nouvelle méthodologie, basée sur la création automatique d'une nouvelle taxonomie d'items et l'extraction de règles d'association, pour des recommandations basées sur des profils d'utilisateurs créés à partir de feedbacks implicites. De plus, nous avons utilisé une méthode d'initialisation de l'algorithme de la factorisation matricielle afin d'améliorer la précision des recommandations. Les résultats expérimentaux ont montré que notre technique atteint une meilleure performance que les autres techniques en termes de précision. Cette amélioration est due à l'avantage significatif de la technique proposée, qui fournit suffisamment d'informations sur les préférences de l'utilisateur à analyser à partir de son comportement d'achat.

ABSTRACT

Collaborative filtering is a method for recommending items to an active user based on the ratings (feedbacks) previously provided by users with similar interests (neighboring users). Since collaborative filtering is unable to accurately discover similar neighboring users when the number of user-provided ratings is extremely limited (data sparsity problem), the accuracy of the prediction of user preferences will be low. This results in mediocre quality recommendations. In this work, our main objective is to develop, for an industrial partner in the Business-to-Business market, a recommendation system based on collaborative filtering from implicit feedbacks only. To overcome the problem of data sparsity, we proposed a new methodology, based on the automatic creation of a new item taxonomy and the extraction of association rules, for recommendations based on user profiles created from implicit feedbacks. In addition, we used an initialization method for the matrix factorization algorithm to improve the accuracy of the recommendations. Experimental results showed that our technique achieves better performance than other techniques in terms of accuracy. This improvement is due to the significant advantage of the proposed technique, which provides enough information about the user's preferences to be analyzed from his buying behavior.

TABLE DE MATIÈRES

DÉDICACE iii

REMERCIMENTS iv

RÉSUMÉ v

ABSTRACT vi

TABLE DE MATIÈRES vii

LISTE DE TABLEAUX x

LISTE DE FIGURES x

CHAPITRE 1 INTRODUCTION 1

1.1 Introduction générale 1

1.2 Problématique 4

1.3 Objectif 5

1.4 Structure du rapport 6

CHAPITRE 2 CONCEPTS ET TECHNIQUES DE BASE 7

2.1 Systèmes de recommandations (SRs) 7

2.2 Différentes approches de recommandation 9

2.2.1 Filtrage collaboratif (FC) 10

2.2.1.1 Approches basées sur un modèle 11

2.2.1.2 Approches basées sur les voisinages 12

2.2.2 Filtrage basé sur le contenu 14

2.3 Limites des systèmes de recommandation 16

2.4 Défis des feedback implicites 17

2.5 Comparaison entre filtrage collaboratif et filtrage basé sur contenu 18

2.6 Concepts préliminaires 20

2.6.1 Taxonomies dans la recommandation 20

2.6.2 Extraction des règles d'association 22
2.6.3 Initialisation de la méthode de la factorisation matricielle 24
2.7 Travaux connexes 25
2.8 Positionnement des contributions 33
2.9 Conclusion 33
CHAPITRE 3 MÉTHODOLOGIE 35
3.1 Approche proposée 35
3.2 Création de nouvelle taxonomie 37
3.2.1 Prétraitement des données 37
3.2.2 Utilisation des taxonomies 41
3.3 Renforcement de la matrice Utilisateur-Item 45
3.3.1 Extraction les règles d'associations 45
3.3.2 Renforcement de la matrice utilisateur-item 47
3.4 Initialisation et Application de l'algorithme i-ALS 48
3.4.1 Initialisation de l'algorithme i-ALS 48
3.4.2 Application de l'algorithme i-ALS 49
3.5 Conclusion 50
CHAPITRE 4 EXPÉRIMENTATION 52
4.1 Application de l'approche proposé au cas d'études 52
4.2 Application d'autres approches conventionnelles 57
4.3 Paramètres de l'expérience 57
4.3.1 Division des données 57
4.3.2 Évaluation de la performance des recommandations 58
4.3.3 Spécifications du modèle 60
4.4 Plan de l'expérience 62
4.5 Conclusion 66

CHAPITRE 5 RÉSULTAS ET DISCUSSION 67

5.1 Résultats 67

5.1.1 Comparer la performance de l'algorithme i-ALS selon les trois niveaux de taxonomies 67

5.1.2 Évaluer l'effet d'introduire l'ARM sur la performance de l'algorithme 69

5.1.3 Évaluer l'effet de l'initialisation de l'algorithme i-ALS 71

5.1.4 Comparaison avec les méthodes conventionnelles 72

5.2 Réflexions managériales 73

5.3 Conclusion 74

CHAPITRE 6 CONCLUSION , DISCUSSION ET RECOMMANDATIONS 75

6.1 Discussion 75

6.2 Recommandations 76

6.3 Limitations 76

6.4 Perspectives 77

RÉFÉRENCES 78

LISTE DE TABLEAUX

Tableau 2.1 Avantages et inconvénients de FC et CBF 19

Tableau 3.1 Extrait de la base de données *'Orders'* 38

Tableau 3.2 Extrait de la base de données 'Produits' 39

Tableau 3.3 Extrait de l'ensemble de données finales 39

Tableau 4.1 Statistiques des jeux de données 55

Tableau 4.2 Hyperparamètres pour l'algorithme *i-ALS* 62

Tableau 4.3 Plan d'expérience 65

Tableau 5.1 Comparaison de niveau de sparsité des matrices 68

Tableau 5.2 Sparsité des matrices 70

Tableau 5.3 Effet de l'introduction de l'ARM sur la sparsité 70

LISTE DE FIGURES

Figure 2.1 Illustration de la hiérarchie des systèmes de recommandation 9

Figure 2.2 Système de recommandation collaboratif (Jannach et al., 2010) 10

Figure 2.3 Système de recommandation basé sur le contenu (Jannach et al., 2010) 15

Figure 2.4 Extrait de la taxonomie des livres d'Amazon (Ziegler & Lausen, 2009) 20

Figure 3.1 Étapes de l'approche proposée 36

Figure 3.2 La distribution cumulative des achats des items. 42

Figure 3.3 Étapes de regroupement les items 44

Figure 4.1 Diagnostique de la colonne Description-ITEM 53

Figure 4.2 Création de colonne ITEM_NAME 53

Figure 4.3 Résultats d'agrégation : création de la colonne ITEM_FAMILY 54

Figure 4.4 Taxonomie des items proposée 52

Figure 5.1 Comparaison la performance de l'algorithme *i-ALS* en fonction de différents niveaux de taxonomie 68

Figure 5.2 Comparaison de l'effet du niveau de confiance sur la performance de *i-ALS* 69

Figure 5.3 Évaluation de l'effet de l'initialisation 72

Figure 5.4 Évaluation de l'effet de l'initialisation 72

Figure 5.5 Résultats de l'évaluation des modèles avec les hyperparamètres optimaux 73

CHAPITRE 1 INTRODUCTION

1.1 Introduction générale

Au cours des dernières décennies, les technologies de l'information (TI) ont révolutionné la façon dont les clients (utilisateurs du site web de vente en ligne) font leurs achats et la façon dont les utilisateurs et les entreprises interagissent. Notamment, cette révolution numérique a changé la culture du commerce, influençant la manière dont les entreprises se rapprochent des consommateurs et révolutionnant l'organisation des entreprises dans le monde entier (Beheshti et al., 2020). Selon un rapport récent du CEFRIO, les entreprises québécoises réalisaient en moyenne 20 % de leurs ventes totales en ligne (Beaudoin et al., 2016).

Alors que la vente en ligne gagne en popularité, un défi important consiste à aider les utilisateurs à trier une grande variété de produits proposés pour trouver facilement ceux qui les intéressent le plus. L'un des outils qui sont développés à cet effet est le système de recommandation (SR), qui suscite beaucoup d'intérêt ces dernières années. Les SRs sont une sous-classe de systèmes de filtrage de l'information qui cherchent à prédire l'évaluation ou la préférence que les utilisateurs attribueraient à un item (Hioud, 2017). Les SRs ont été élaborés pour prédire quels sont les produits ou services les plus appropriés, en fonction des intérêts et des contraintes des utilisateurs, tout en minimisant le temps associé à la recherche. Afin de mener à bien ces tâches informatiques, les SRs recueillent les préférences (intérêts) des utilisateurs, qui sont soit explicitement exprimées, comme les évaluations de produits, soit déduites en interprétant les actions de l'utilisateur au cours de leur processus d'achat. Les exemples les plus connus sont les SRs pour des films (Netflix)[1], de la musique (last.fm)[2] et des livres (Amazon)[3]. Les principales recherches dans le domaine des SRs se sont historiquement concentrées sur des scénarios d'application dans lesquels les déclarations de

---

[1] www.netiflix.com

[2] www.lastfm.com

[3] www.amazon.com

`

préférence des utilisateurs sous forme *d'évaluations explicites* sont disponibles (Hioud, 2017). Cela a mené au développement d'algorithmes très sophistiqués capables de prédire avec une grande précision l'évaluation qu'un utilisateur est susceptible de donner à un certain item. La puissance de ces algorithmes repose sur l'existence de grands ensembles de données d'évaluations historiques dans lesquels des dizaines d'évaluations explicites existent pour chaque utilisateur.

Même s'il existe un certain nombre de plateformes web sur lesquelles les utilisateurs peuvent évaluer des films, des livres, des restaurants ou d'autres items, il existe également de nombreux domaines d'application dans lesquels les évaluations explicites sont très rares. Par exemple, si certains items populaires sur *Amazon* reçoivent de nombreuses évaluations, la plupart des items n'en ont aucune. Dans un contexte Business-To-Business où les activités commerciales de l'entreprise ciblent une clientèle d'entreprises, contrairement au cas Business-To-Customer (B2C) où on cible des individus, les items ne peuvent généralement pas être explicitement évalués (Lee et al., 2006a). Lors de la création de recommandations personnalisées dans de tels domaines d'application, nous devons nous appuyer sur des moyens indirects pour évaluer les intérêts et les préférences des utilisateurs en examinant et en interprétant leurs actions et leur comportement historique (Lee et al., 2006a). Dans la littérature, ces observations des actions d'un utilisateur qui sont interprétées comme des déclarations sur la pertinence d'un item particulier sont appelées *feedback implicite*. Ainsi, le terme *"implicite"* est utilisé parce que les utilisateurs n'expriment pas explicitement leurs préférences, mais celles-ci sont dérivées de leurs actions observées. Dans un contexte classique de vente en ligne, un exemple d'action de l'utilisateur pouvant indiquer une préférence pour un item est la consultation de la description détaillée d'un item, l'ajout de cet item à une liste de souhaits ou tout simplement l'achat d'un item (Gao et al., 2019). Sur les plateformes de streaming média, la consommation répétée d'un titre ou d'une vidéo musicale peut être interprétée comme un intérêt ou une préférence de l'utilisateur pour le titre lui-même, l'artiste du titre ou le genre.

Les systèmes de recommandation dans un contexte Business-to-Business (B2B) se sont relativement peu développés par rapport à ceux dans un contexte Business-to-Customer (B2C). Selon le dernier rapport de recherche B2B *d'Accenture Company*, la satisfaction des clients du B2B est inférieure à celle du B2C, ce qui indique que les entreprises du marché B2B n'ont pas répondu efficacement aux besoins de leurs clients (Keinänen & Kuivalainen, 2015). En plus des défis

`

généraux rencontrés par les systèmes de recommandation, les systèmes dans un contexte B2B doivent faire face à plusieurs difficultés. Comme les clients sont des entreprises et ne sont pas des individus comme dans le cas B2C, les algorithmes pour ceux-ci peuvent ne pas s'appliquer (Zhang & Wang, 2005).

Les recommandations dans un contexte B2B ne sont pas générées pour des consommateurs individuels, mais pour des petites ou grandes entreprises et organisations, qui sont les clients de l'entreprise (Lee et al., 2006a). Pour ces entités clientes, une multitude d'informations étendues peuvent être extraites à partir des données publiées en ligne. Cela permettrait les systèmes de recommandation d'exploiter non seulement l'historique des achats, mais aussi des informations qui couvrent pratiquement tous les aspects des activités du client.

Un autre défi présent dans les SRs est la sparsité des données, en particulier pour les systèmes utilisant le feedback explicite ou les systèmes ayant un grand volume d'utilisateur et des items (Guo et al., 2017). Dans ces cas, les utilisateurs évaluent et interagissent avec un petit nombre d'items, ce qui nuit à la performance des systèmes de recommandation. Particulièrement dans les systèmes filtrage collaboratif, il est difficile de recommander des items avec peu ou pas d'interaction avec l'utilisateur ; de même, il est difficile de produire des recommandations pour des utilisateurs avec très peu de notes ou peu d'historique d'interaction avec les items.

L'objectif de ce projet est de développer une approche de recommandation adaptée au contexte B2B afin de prendre en compte le feedback implicite, basé sur l'historique des comportements d'achat des utilisateurs et certaines informations extraites de données en ligne ayant un impact sur les préférences des clients vis-à-vis des produits. Pour surmonter les problèmes de sparsité des données, nous suggérons une solution basée sur l'exploration de règles d'association (ARM en anglais) pour enrichir la matrice UTILISATEUR-ITEM et la rendre plus dense. De plus, afin d'améliorer la performance de notre approche proposée, nous appliquons une telle configuration optimale pour l'algorithme de la factorisation matricielle. En fait, la configuration consiste à initialiser les matrices de facteurs latents en décomposant la matrice USER-ITEM avec un tel algorithme de décomposition en valeur singulière (SVD) proposé pour la première fois par Kalman (1996).

`

Ce projet est motivé par le besoin du partenaire industriel, un distributeur dans le domaine de la quincaillerie et des outils industriels, de valoriser ses données clientèles afin d'augmenter ses ventes et de développer ses marchés stratégiques. Notre partenaire est une entreprise québécoise de vente et de distribution de produits professionnels pour des clients industriels (B2B) qui existe depuis 1998. Il exploite un réseau de 500 fournisseurs et 6 entrepôts de distribution à partir desquels les produits sont expédiés à leurs clients par transport routier dans toute la province de Québec.

1.2 Problématique

Les systèmes de recommandation se révèlent être une fonctionnalité valorisante qui enrichit l'expérience de l'utilisateur, ainsi qu'un outil considérable entre les mains du vendeur. La plupart des recherches dans ce domaine se sont concentrées sur l'application de ces systèmes pour assister les consommateurs dans le commerce électronique (e-commerce) Business-to-Customer (B2C), mais les participants au commerce Business-to-Business (B2B) peuvent également bénéficier d'une assistance utile grâce au système de recommandation. Cependant, le développement d'un système de recommandation dans le domaine des services électroniques B2B dans le monde réel pose toujours deux défis : 1) les items (produits) présentent souvent des structures arborescentes compliquées dans les applications commerciales, qui ne peuvent pas être traitées par les mesures normales de similarité des items et 2) les préférences des utilisateurs sont souvent vagues et floues, et ne peuvent pas être traitées par les méthodes de recommandation existantes dans le commerce *B2C*.

La première question de recherche est alors comment adapter un système de recommandation B2C aux caractéristiques et aux habitudes des utilisateurs du marché B2B.

Les utilisateurs utilisent souvent les systèmes de commerce électronique parce qu'ils peuvent faire leurs achats avec beaucoup de confort et de facilité. De toute évidence, la conception d'un site de commerce électronique doit être réalisée de manière à ne pas harceler les utilisateurs en leur posant constamment des questions sur leurs préférences et en les obligeant à fournir un feed-back sur leurs achats. De plus, le système doit rendre l'ensemble du processus d'achat facile et rapide, sans quoi les ventes et la fidélité risquent de diminuer. Ces implémentations souffrent manifestement du manque d'informations et de commentaires explicites de la part des utilisateurs. La plupart des

`

systèmes se sont concentrées sur des cas d'utilisation où l'utilisateur fournit des informations de manière explicite. En revanche, il est possible de concevoir et d'améliorer les algorithmes de recommandation qui sont basés uniquement sur des informations implicites recueillies en surveillant le comportement de l'utilisateur. De cette façon, les systèmes de commerce électronique peuvent offrir des recommandations personnalisées sans détériorer l'expérience de leurs utilisateurs.

Dans notre cas, le système de recommandation est réalisé pour une plateforme de vente en ligne opérant à Montréal au Canada. Chaque utilisateur du système est identifié de manière unique, ce qui permet de suivre toutes ses actions. Lorsqu'un utilisateur passe une commande, une transaction d'achat est créée. Chaque transaction contient des informations sur chaque produit acheté par l'utilisateur. Malheureusement, le système ne demande pas à l'utilisateur d'évaluer les items, ce qui signifie qu'aucun feedback explicite n'est disponible.

La deuxième question de recherche est comment trouver une solution pour la fluctuation du système de recommandation afin de lui donner une adaptation aux données implicites.

Ces informations implicites, même si elles sont facilement recueillies en surveillant le comportement de navigation de l'utilisateur et en exploitant son historique d'achat, concernent souvent un petit nombre d'items plutôt que l'ensemble de l'offre du fournisseur. Ainsi, un défi supplémentaire qui devrait être relevé est la rareté des données dont souffrent ces implémentations.

La troisième question de recherche est comment surmonter le problème de rareté.

1.3 Objectif

L’objectif général de ce projet de recherche est de développer et évaluer une approche de recommandations adaptée au contexte B2B considérant les feedbacks implicites basé sur le comportement d'achat du client et les informations sur le client extraites en ligne.

Pour atteindre l’objectif général, nous visons suivre la méthodologie suivante :

1- Faire une revue de littérature sur les systèmes de recommandation en récapitulant par un tableau des avantages et des inconvénients de chaque système;

`

2- Analyser et identifier les besoins de l'entreprise partenaire pour laquelle le système est conçu en termes d'outils d'aide à la décision;
3- Explorer plusieurs algorithmes pour déterminer celles qui répondent le mieux aux besoins de l'entreprise;
4- Proposer une approche permettant de valoriser l'historique d'achat des clients du partenaire et de faire des recommandations appropriées.

1.4 Structure du rapport

Ce rapport est structuré comme suit. Le chapitre 2 présente les différentes approches des systèmes de recommandation, la manière dont elle est utilisée pour prédire les préférences des utilisateurs et faire des recommandations, et donne un aperçu de l'état actuel de la recherche. Nous aborderons également dans le chapitre 2 certaines techniques d'exploration de données qui sont pertinentes pour l'approche proposée. Le chapitre 3 expose la méthodologie. Ces méthodes comprennent l'identification des technologies, les étapes de collecte des données et les méthodes d'analyse des données. Dans le chapitre 4, nous présentons une analyse et une discussion détaillées des résultats obtenus. Le chapitre 5 renferme les conclusions et les recommandations pour les avancées futures de cette recherche et discute des limites.

CHAPITRE 2 CONCEPTS ET TECHNIQUES DE BASE

Ce chapitre présente un aperçu des concepts et les techniques de base utilisées dans cette étude. Il est divisé en 4 parties. La première partie présente un bref historique des systèmes de recommandation. Dans la deuxième partie nous citerons quelques domaines d'application de ces systèmes ainsi que de certaines limites des systèmes de recommandation. Dans la troisième et la quatrième étape, nous présentons la définition des techniques de base de notre approche.

2.1 Systèmes de recommandations (SRs)

"Les ordinateurs nous ont promis une fontaine de sagesse mais nous ont livré un flot de données», (Frawley et al., 1992)

La quantité de données et la complexité des caractéristiques de tout environnement et tâche informatique augmentent rapidement. D'énormes quantités de données brutes et souvent complexes poussent les grandes organisations et les acteurs du marché à vouloir extraire des connaissances précieuses de leurs collections de données. Cette demande incite les scientifiques à trouver de nouvelles méthodes de comprendre les données et de trouver les connaissances latentes derrière les données brutes. Parmi ces méthodes, les systèmes de recommandation prennent sa place pour aider à valoriser les données brutes des achats historique. Les systèmes de recommandation (SRs) sont apparus dans le début des années 1990 pour répondre au problème de surcharge d'information et de choix. Ces systèmes visent à proposer à un utilisateur actif des éléments (l'unité de contenu) susceptibles de l'intéresser. Les SRs identifient les contenus spécifiques qui correspondent aux intérêts personnels des utilisateurs au sein d'énormes collections de contenus. La pertinence d'un élément par rapport à un utilisateur est prédite par des algorithmes de recommandation ; les éléments ayant les scores de prédiction les plus élevés sont affichés à l'utilisateur (Shokeen & Rana, 2020). Dans le reste de ce mémoire, nous avons utilisé le terme d'item pour designer l'élément.

Une classification typique divise les algorithmes de recommandation en deux approches principales : le filtrage basé sur le contenu (FBC) et le filtrage collaboratif (FC) (Herlocker et al., 2004). Les algorithmes de FBCs utilisent les métadonnées de l'utilisateur (par exemple, les données

`

démographiques) et les métadonnées des items (par exemple, l'auteur, le genre, etc.) pour prédire la préférence de l'utilisateur en fonction de ces attributs. En revanche, les méthodes de FC n'utilisent pas les métadonnées, mais uniquement les données des interactions entre l'utilisateur et l'item. Mais il y'a une méthode hybride qui est une combinaison des deux filtrages précédents.

En fonction de la nature des interactions, les algorithmes peuvent également être classés en méthodes basées sur le feedback explicite et implicite. Dans le premier cas, les utilisateurs fournissent des informations explicites sur leurs préférences en matière d'items, habituellement sous la forme d'évaluations. Dans le second cas, les préférences des utilisateurs sont fournies de manière totalement abstraite à travers leurs activités. Les algorithmes de feedback implicite utilisent les interactions de l'utilisateur, comme la consultation et l'achat, récupérées par exemple dans les logs d'utilisation des sites web.

Données implicites : ce sont des données qui ne sont pas générées par aucune action spécifique de la part de l'utilisateur. Typiquement, ces données sont collectées en observant les activités des utilisateurs sur la plateforme qu'ils utilisent. Il peut notamment s'agir de données sur les flux de clics (données représentant le parcours de l'utilisateur sur un site web), les sites web visités, l'historique des achats, les coordonnées GPS, les données de transaction, les quantités d'items achetés, etc, (Ricci et al., 2015). Dans quelques domaines d'application, tels que la recherche de films, les applications de voyage et le streaming vidéo, les utilisateurs sont motivés pour fournir des évaluations afin d'obtenir un meilleur service. Cependant, les utilisateurs de boutiques ou de services de commerce électronique n'ont pas tendance à fournir des évaluations sur les items, même si une telle option est disponible, puisque, au moment de l'achat, ils n'ont aucune information sur leur taux de satisfaction, aussi, ils ne sont pas motivés pour revenir plus tard sur le système pour le faire (Pilászy et al., 2010). Dans ce contexte, les préférences de l'utilisateur ne peuvent être déduites qu'en interprétant ses actions (également appelées événements) (Pilászy et al., 2010). Par exemple, un système de recommandation peut considérer la navigation vers une page d'item particulière comme un signe implicite de préférence pour l'item présenté sur cette page (Ricci et al., 2011). L'historique de l'utilisateur spécifique aux items est donc considéré comme un retour implicite sur les goûts de l'utilisateur. En revanche, les données de feedback implicite peuvent être mal interprétées et ne reflètent pas nécessairement la satisfaction de l'utilisateur. Par exemple, un item

`

acheté par un utilisateur ne signifie pas nécessairement un feedback favorable, car l'utilisateur pourrait déçu de cet achat.

Les algorithmes FCs se sont avérés plus précis que les algorithmes FBCs, si des données de préférence suffisantes sont disponibles (Takács et al., 2009). Si ce n'est pas le cas, le problème du démarrage à froid se pose. Le démarrage à froid se traduit notamment par l'introduction d'un nouvel élément (item et/ou utilisateur) dans le système de recommandation (Lika et al., 2014). Dans le cas d'un système de recommandation basé sur le filtrage collaboratif, ce nouvel élément ne sera pas intégré dans les recommandations, puisque le système ne dispose d'aucune information sur cet élément (Xiaoyuan Su & Taghi M Khoshgoftaar, 2009; Takács et al., 2009).

2.2 Différentes approches de recommandation

En général, les systèmes de recommandation prédisent les préférences des utilisateurs et fournissent une liste de recommandations en utilisant l'une des trois approches suivantes (voir Figure 2.1) : le filtrage collaboratif, le filtrage basé sur le contenu et les systèmes de recommandation hybrides qui combine à la fois le filtrage collaboratif et le filtrage basé sur le contenu (Tadlaoui, 2018).

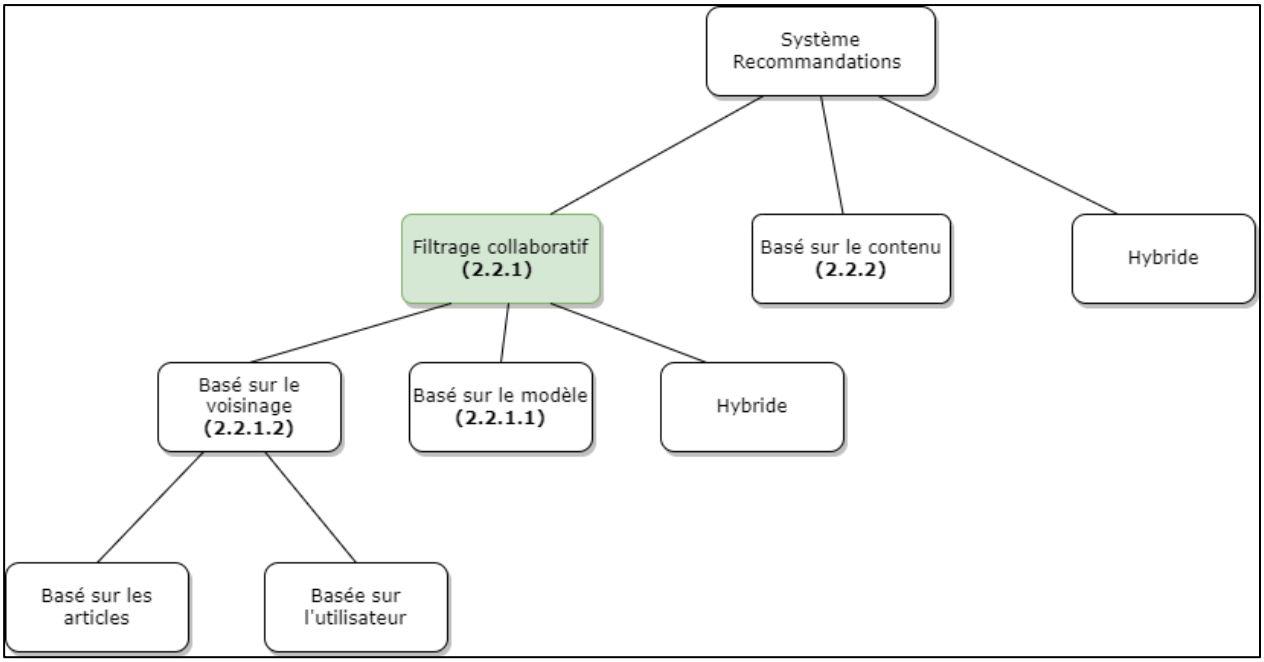

Figure 2.1 Illustration de la hiérarchie des systèmes de recommandation

`

2.2.1 Filtrage collaboratif (FC)

Ces dernières années, l'approche de filtrage collaboratif est la technique la plus populaire utilisée dans les SRs (Yang et al., 2016). Les systèmes basés sur le filtrage collaboratif fournissent des recommandations en mesurant la **similarité** entre les préférences d'un utilisateur et celles d'autres utilisateurs. Ces systèmes ne sont pas destinés à analyser ou à comprendre le contenu des items à recommander (Nilashi et al., 2018). La méthode consiste à faire des prédictions automatiques sur les intérêts d'un utilisateur en recueillant les diverses évaluations des autres utilisateurs. Cette approche est basée sur l'hypothèse suivante : les personnes qui étaient d'accord, en termes de préférences, avec un utilisateur dans le passé le seront également dans le futur (Aciar et al., 2007). Un exemple simple de filtrage collaboratif peut être décrit comme suit : imaginez deux utilisateurs qui ont des évaluations similaires pour certains items. Le filtrage collaboratif part du principe que ces similitudes s'étendent aux items évalués par un seul des deux utilisateurs. Par conséquent, pour apprendre le modèle d'un utilisateur, les méthodes de filtrage collaboratif tentent de réunir des informations sur les préférences d'utilisateurs similaires (c'est-à-dire d'utilisateurs dont les préférences pour des items évalués en commun sont similaires) (Nilashi et al., 2018).

Une description de base de l'approche FC pour expliquer l'hypothèse précédente est illustrée dans la Figure 2.2.

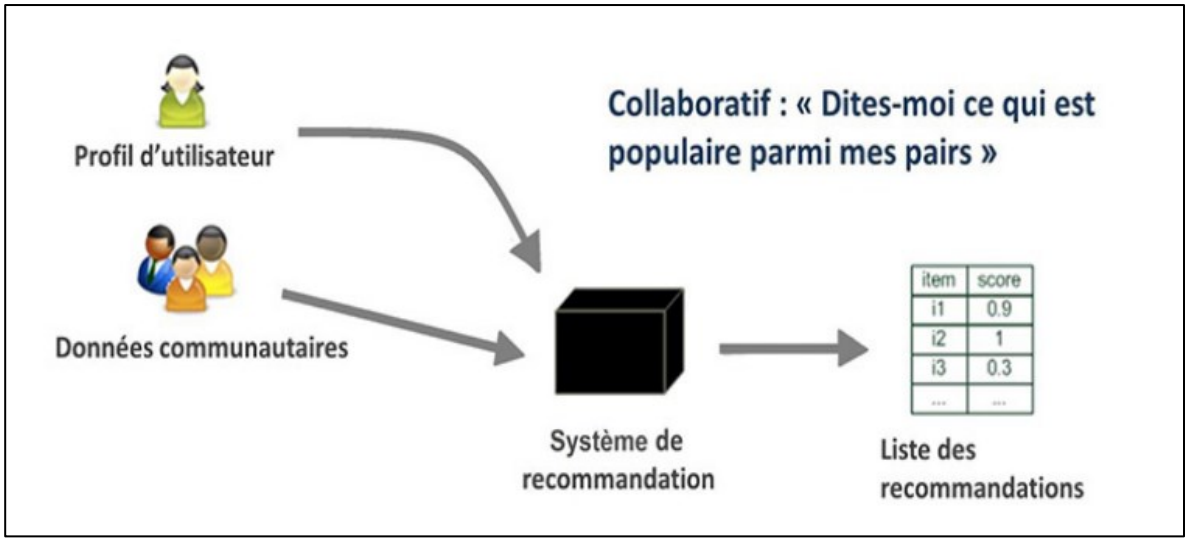

Figure 2.2 Système de recommandation collaboratif (Jannach et al., 2010)

`

Les préférences forment une matrice d'évaluations de deux dimensions : utilisateurs et items, où chaque ligne figure un utilisateur et chaque colonne figure un item. La similarité entre deux utilisateurs dépend de la similarité de l'historique des évaluations de ces deux utilisateurs (Tadlaoui, 2018). Ekstrand et al., (Ekstrand et al., 2011) énoncent que les différents systèmes de recommandation, y compris les systèmes basés sur le filtrage collaboratif, effectuent deux tâches : la première est la prédiction et la dernière est celle de recommandation. La prédiction qui essaye de prédire les valeurs manquantes dans la matrice d'évaluations. La recommandation génère une liste d'items les mieux classées en fonctions des préférences de l'utilisateur.

Le FC implique l'analyse des relations entre les utilisateurs et des interdépendances entre les items pour identifier de nouvelles associations *utilisateur-item*, de sorte que l'utilisateur découvre de nouveaux intérêts sur la base de son comportement passé (Yang et al., 2016). L'un des avantages de le FC est que les utilisateurs peuvent obtenir une exposition plus large à de nombreux items appropriés (García-Cumbreras et al., 2013). Un autre avantage du FC est qu'il n'y a pas de dépendance à la connaissance du domaine. Simultanément, le FC doit faire face à plusieurs défis majeurs : la scalabilité, la sparsité, la confidentialité et les recommandations structurées (Al-Bashiri et al., 2017).

Selon (Nilashi et al., 2014), les approches de filtrage collaboratif sont divisées en deux taxonomies : les approches basées sur le voisinage (section 2.2.1.1) et les approches basées sur le modèle (section 2.2.1.1).

2.2.1.1 Approches basées sur un modèle

Les approches basées sur un modèle émettent des recommandations en estimant les paramètres des modèles d'apprentissage. Tout d'abord, les évaluations passées des utilisateurs sont collectées et utilisées pour apprendre un modèle. Ce modèle est ensuite utilisé pour faire des prédictions d'évaluation qui sont généralement rapides et précises (Dadouchi & Agard, 2018).

Les techniques basées sur des modèles utilisent les informations des évaluations (score de 1 à 5 par exemple) disponibles pour construire un modèle de dimensionnalité réduite qui est utilisé pour un traitement ultérieur et la prédiction de l'évaluation (Xiaoyuan Su & Taghi M. Khoshgoftaar, 2009). Les modèles basés sur des facteurs latents sont l'une des modèles les plus

populaires (Koren, 2008). Les modèles à facteurs latents sont basés sur l'hypothèse que le choix d'un utilisateur pour un item est dirigé par une série de facteurs appelés facteurs latents. Par exemple, dans le cas des recommandations des livres, ces modèles peuvent capturer des caractéristiques comme le genre, l'auteur et la langue, tandis que pour les recommandations des destinations touristiques, ils encapsulent des caractéristiques déterminantes comme la catégorie (par exemple plage, montagne, etc.) ou les contraintes budgétaires (Koren et al., 2009). Les techniques basées sur les facteurs latents tentent de représenter à la fois les utilisateurs et les items par des vecteurs capturant leur association/affinité avec ces facteurs latents (Elahi et al., 2016 ; Koren et al., 2009). L'évaluation donnée par un utilisateur à un item est la quantification de l'interaction entre les deux vecteurs.

La factorisation matricielle (FM) vise à condenser la dimensionnalité de la matrice d'interaction utilisateur-item (Koren et al., 2009). Par conséquent, l'algorithme de factorisation divise la matrice originale des évaluations des utilisateurs en deux matrices distinctes, de sorte que lorsque leur produit scalaire est calculé, il se rapproche de la matrice originale sans perdre aucune information. Ainsi, il prévoit les évaluations absentes dans la matrice originale *utilisateur-item* (Koren et al., 2009).

Différentes méthodes FMs sont utilisées pour calculer ces vecteurs en approchant la matrice d'évaluation partiellement connue en utilisant les moindres carrés alternatifs (*ALS*) (Bell & Koren, 2007), la méthode de descente de gradient (Takács et al., 2007), la méthode de descente des coordonnées (Pilászy, 2010), la méthode du gradient conjugué (Takács et al., 2011) ou la décomposition de la valeur singulière (Koren, 2008).

2.2.1.2 Approches basées sur les voisinages

Le type de FC basé sur les voisinages est un algorithme heuristique qui calcule les corrélations entre les utilisateurs et les items pour produire un score de préférence qui prédit la probabilité qu'un utilisateur acquière un item dans le futur et fournit des recommandations appropriées (Sharma & Mann, 2013). Les algorithmes basés sur l'utilisateur et sur l'item sont les types les plus courants de méthodes de recommandation basées sur les voisinages (Ricci et al., 2011). Les méthodes basées sur l'utilisateur génèrent des recommandations en fonction des similarités entre les utilisateurs

`

(Sarwar et al., 2001), tandis que les méthodes basées sur les items calculent les similarités dans un espace d'items pour trouver des relations fortes avec les items qui ont déjà été évalués par un utilisateur actif (Zeng et al., 2014).

- **Les systèmes basés sur l'utilisateur (user-based systems)** :

Le FC basé sur l'utilisateur, aussi appelé filtrage collaboratif utilisateur-utilisateur, repose sur l'idée centrale que les utilisateurs qui s'intéressent aux mêmes items et qui ont des évaluations similaires auront donc des préférences similaires (Resnick et al., 1994). Compte tenu d'un comportement d'évaluation semblable, le système de recommandation est capable de prédire si un utilisateur peut être intéressé par un item non vu (Shardanand & Maes, 1995). Le processus d'un filtre collaboratif utilisateur-utilisateur typique est généralement divisé en trois étapes) :

1- Mesure de la similarité : le système de recommandation calcule les similarités entre l'utilisateur actif et les autres utilisateurs qui ont évalué les mêmes items. Cette étape est un domaine bien étudié dans le domaine des systèmes de recommandation (Adomavicius & Tuzhilin, 2005).

Dans la littérature, plusieurs mesures de similarité entre les utilisateurs (ou les items) ont été proposées. Selon Beliakov et al. (Beliakov et al., 2011) et Breese et al. (Breese et al., 1998), les deux types de mesures de similarité les plus connus sont la similarité vectorielle (ou basée sur le cosinus) et le coefficient de corrélation de Pearson. Il existe également d'autres mesures telles que la différence quadratique moyenne (Shardanand & Maes, 1995) et le coefficient de corrélation de Spearman (Herlocker et al., 2002).

2- Voisinage : les utilisateurs similaires sont regroupés dans un sous-ensemble, le "voisinage" de l'utilisateur actif. Typiquement, ce voisinage est constitué des utilisateurs les plus similaires calculés lors de l'étape 1.

3- Prédiction et génération de recommandations : dans la troisième et dernière étape, le système rassemble les informations du voisinage de l'utilisateur actif. Ensuite, un algorithme génère une liste de *N* items à recommander. Dans le contexte réel des sites de commerce électronique, on l'appelle la liste de recommandation *"Top-N"*. En outre, une prédiction d'évaluation peut être calculée pour un item spécifique. Dans leur étude, Herlocker et al. (Herlocker et al., 2002) présentent plusieurs algorithmes pour l'étape de génération de recommandations. Le

`

plus grand avantage du filtrage collaboratif utilisateur-utilisateur est son indépendance vis-à-vis du domaine.

- **Les systèmes basés sur les items (item-based systems) :** Le filtrage collaboratif basé sur les items, appelé aussi filtrage collaboratif item-item, est en quelque sorte similaire au filtrage collaboratif utilisateur-utilisateur et peut être considéré comme la même approche, mais du point de vue de l'item. Le principe est le suivant : les éléments qui ont été évalués de la même manière sont susceptibles de partager des caractéristiques similaires, ainsi les utilisateurs qui aiment l'un d'entre eux devraient aimer les autres qui sont évalués de manière similaire (Herlocker et al., 2002).

Amazon est probablement le site de commerce électronique le plus connu pour l'utilisation du filtrage collaboratif item-item (Linden et al., 2003). Amazon, grâce à sa technologie de recommandation item-item très efficace et fonctionnant bien, a réalisé une forte augmentation des gains. Comme il ne nécessite que les données d'évaluation, le filtrage collaboratif item-item présente les mêmes avantages et inconvénients que le filtrage collaboratif utilisateur-utilisateur.

2.2.2 Filtrage basé sur le contenu

De façon très simple, la recommandation basée sur le contenu (*FBC*) construit un modèle (ou profil) des intérêts des utilisateurs sur la base des caractéristiques des items qu'ils ont déjà évalué (Dadouchi & Agard, 2018). Pour ce faire, le système analyse la description des items précédemment évalués.

Le processus de recommandation consiste essentiellement à trouver de bonnes correspondances entre le profil de l'utilisateur et les caractéristiques des items. En effet, il génère le profil de l'utilisateur à partir des items sélectionnés auparavant en caractérisant l'utilisateur selon les caractéristiques de l'item et recommande des items à l'utilisateur en fonction des items ayant des caractéristiques similaires aux items que l'utilisateur a évalués auparavant (García-Cumbreras et al., 2013). Il caractérise chaque utilisateur sans avoir à comparer ses préférences à celles des autres utilisateurs. En d'autres termes, elle n'utilise pas d'informations sur les préférences des autres utilisateurs ou sur les similarités avec les autres utilisateurs (Aciar et al., 2007 ; García-Cumbreras et al., 2013).

`

Une description simple de cette approche est illustrée dans Figure 2.3.

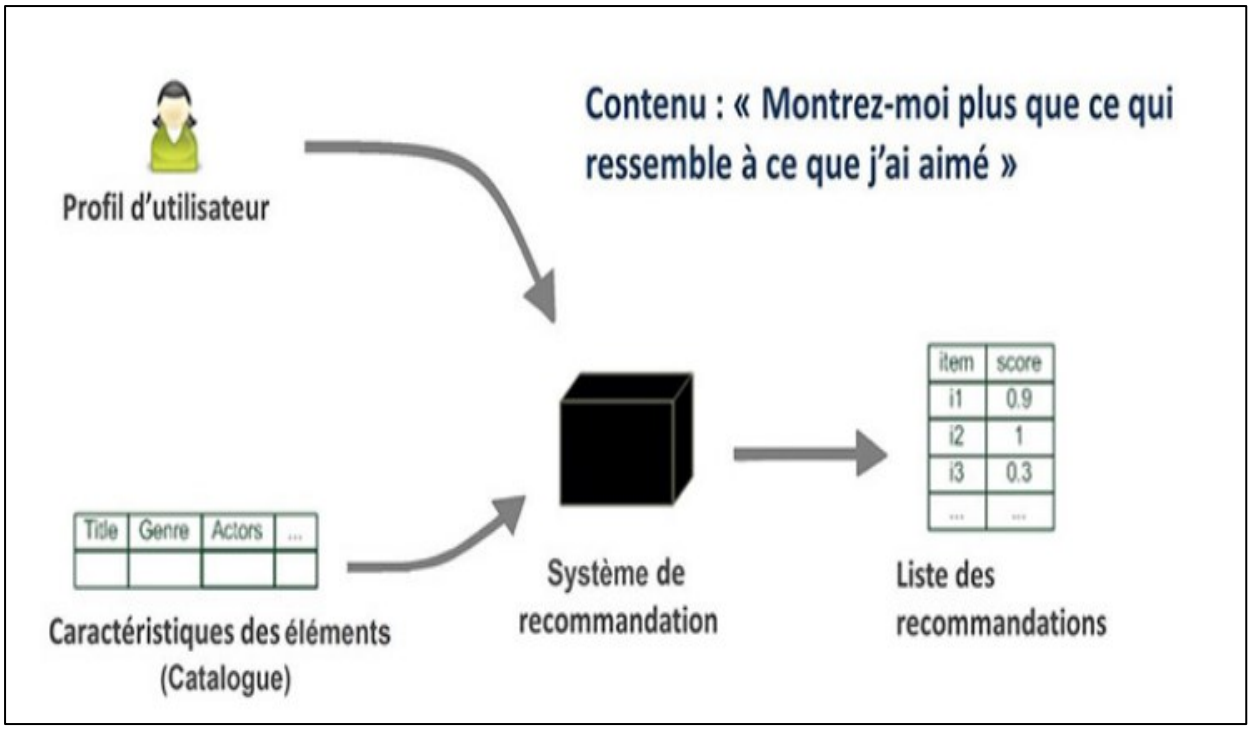

Figure 2.3 Système de recommandation basé sur le contenu (Jannach et al., 2010)

Le processus peut être résumé par les étapes suivantes :

- **Représentation de l'item** : la source d'information de la description de l'item est utilisée pour extraire les caractéristiques de l'item en question afin de produire la représentation structurée de celui-ci (Hdioud et al., 2013).
- **Apprentissage du profil de l'utilisateur** : un profil d'utilisateur est généré à partir des comportements de l'utilisateur précédents (c.-à-d., les feedbacks explicites et implicites) tels que le fait d'aimer ou de ne pas aimer un item, d'attribuer une évaluation à un item (*score*), ou d'écrire une opinion textuelle sur un item (*commentaire*), le fait d'acheter un item (*feedback implicite*), les informations sur la navigation des utilisateurs dans le site web, etc (Cantador et al., 2008).
- **Génération de recommandations** : une liste d'items est recommandée à l'utilisateur en comparant les caractéristiques de l'item avec le profil de l'utilisateur. Les items les plus

`

susceptibles d'être attrayants pour l'utilisateur sont ajoutés à la liste des items à recommander (Blanco-Fernández et al., 2008).

Cette technique présente des avantages tels que l'indépendance vis-à-vis de l'utilisateur, puisque les systèmes FBCs n'utilisent que les évaluations de l'utilisateur actif pour construire leur modèle, contrairement aux techniques de filtrage collaboratif, qui reposent sur les "voisins". De plus, lorsqu'un nouvel élément apparaît et qu'il n'a pas encore été évalué, les systèmes FBCs sont capables de le recommander.

2.3 Limites des systèmes de recommandation

Les systèmes de recommandation se basant sur les méthodes expliquées précédemment ont certaines limitations (Adomavicius & Tuzhilin, 2005).

- **Démarrage à froid** : ce problème est largement abordé dans la littérature et concerne les recommandations pour de nouveaux utilisateurs ou items (Lika et al., 2014). Dans le cas de nouveaux utilisateurs, le système ne dispose d'aucune information sur leurs préférences et ne peut donc rien leur recommander. Dans le cas de nouveaux items, le système ne dispose d'aucune évaluation pour ces items et ne sait pas à qui les recommander. Pour atténuer le démarrage à froid Zhang et al (Pilászy; Zhang et al., 2014) utilisent un modèle probabiliste pour extraire les caractéristiques latentes de la représentation de l'item. En utilisant les caractéristiques latentes, ils génèrent des pseudo-évaluations précises, même dans une situation de démarrage à froid lorsque peu ou pas d'évaluations sont fournies. Un autre exemple est fourni par Chughtai et al. (Chughtai et al., 2013) où les auteurs tentent de résoudre le problème du démarrage à froid des nouveaux utilisateurs dans le domaine de l'apprentissage en ligne en combinant le FC avec les métadonnées des contenus d'apprentissage. Le problème du démarrage à froid est également traité par Basiri et al. (Basiri et al., 2010) où les auteurs fusionnent les résultats pondérés de différentes stratégies de recommandation en utilisant la moyenne pondérée ordonnée.
- **Rareté (Sparsité) de données :** ce problème provient du fait que les utilisateurs évaluent généralement un nombre très limité d’items disponibles. Il en résulte une matrice d'évaluation utilisateur-item a faible densité avec des données insuffisantes pour identifier

`

les utilisateurs ou les items similaires, ce qui a un impact négatif sur la qualité des recommandations. Dans Wang et al. (Wang et al., 2006), nous trouvons une tentative de résoudre la sparsité des données en traitant chaque évaluation de l'item-utilisateur comme un prédicteur des autres évaluations manquantes. Les auteurs estiment les évaluations finales en fusionnant les évaluations du même item par d'autres utilisateurs, les évaluations d'items différents faites par le même utilisateur et les évaluations d'autres utilisateurs similaires sur différents items items similaires (Zuva et al., 2012).

- **Problème de synonymie :** la synonymie consiste généralement à ce que des items très similaires aient des noms ou des descriptions différentes. En effet, lorsqu'un certain nombre d'items identiques ou très semblables portent un nom différent et que les systèmes de recommandation ne parviennent pas à découvrir cette association latente, ils traitent alors ces items différemment (Liphoto et al., 2016). La plupart des systèmes de filtrage collaboratif ont du mal à faire la distinction entre des items étroitement semblables, comme la différence entre, par exemple, "*Backpack*" et "*Knapsack*". Ils ne trouvent généralement aucune correspondance entre les deux termes pour pouvoir calculer leur similarité (Zuva et al., 2012).
- La principale limitation de l'approches basée sur le contenu est qu'elles ne peuvent recommander que des ressources similaires à celles qu'un utilisateur (donné) a apprécié, ce qui empêche de recommander d'autres ressources que ce même utilisateur pourrait également apprécier (Zuva et al., 2012).

2.4 Défis des feedback implicites

La recommandation basée sur les feedbacks implicites (c'est-à-dire seul l'historique de l'utilisateur est disponible, mais qu'il n'y a pas d'évaluation) est une tâche beaucoup plus difficile que la recommandation basée sur les feedbacks explicites, en raison de l'incertitude inhérente à l'interprétation de ces feedbacks de l'utilisateur. Récemment, le problème feedback implicite a reçu plus d'attention, car la recherche orientée vers les applications devient plus attrayante dans ce domaine.

Lorsque l'on s'appuie sur le feedback implicite, un certain nombre de défis doivent être relevés.

`

- Dans la plupart des domaines, les feedbacks implicites pour les quelques items très populaires sont prédominants, tandis que celles pour les items de niche peut être très rare. Par conséquent, la distribution des feedbacks est biaisée en forme de *longue queue*. La construction de modèles de recommandation basés sur de telles données peut facilement conduire à un fort biais de popularité.
- Lorsqu'un feedback explicite est disponible, il peut être plus facile pour l'utilisateur de comprendre la raison pour laquelle les recommandations ont été fournies. Les recommandations qui résultent de feedbacks implicites peuvent ne pas être aussi évidentes ou crédibles pour l'utilisateur. Par exemple, une recommandation présentée à un utilisateur avec l'explication "parce que vous avez donné 5 étoiles à [film A]" peut être plus crédible que l'explication "parce que vous avez regardé [film A]", car dans ce dernier cas, l'utilisateur peut ne pas avoir aimé le film A finalement.
- Un autre point est que le feedback implicite dépend du domaine. Par exemple, dans un système de recommandation de films, un utilisateur peut regarder un acteur 10 fois, mais cela ne signifie pas qu'il a une préférence relativement élevée pour cet acteur. Il se peut que l’acteur fasse partie d'une série que l'utilisateur regarde régulièrement.

2.5 Comparaison entre filtrage collaboratif et filtrage basé sur contenu

Le tableau ci-après illustre les points forts et les points faibles des approches traditionnelles utilisées par les systèmes de recommandation, basées sur le filtrage collaboratif et sur le contenu:

`

Tableau 2.1 Avantages et inconvénients de FC et CBF

Techniques	**Avantages**	**Inconvénients**
Filtrage collaboratif	- Ne demande aucune connaissance sur le contenu de l'item ni sa sémantique. - Le système n'a, dans une certaine mesure, besoin que de la matrice de feedback pour entraîner un modèle de factorisation matricielle. - Le système n'a pas besoin de caractéristiques contextuelles. - Plus les nombre d'utilisateurs est grand plus la recommandation est meilleure.	- Démarrage à froid. - La complexité des algorithmes est proportionnelle à la taille de la matrice donc d'ordre $O(l*c)$ (l : nombre de lignes ; c : nombre de colonnes). - Probleme de sparsité - Le problème du mouton gris : les utilisateurs ayant des préférences et des goûts particuliers (sortant de l'ordinaire) auront peu d'utilisateurs voisins. Il sera donc difficile de faire des recommandations pertinentes pour ces utilisateurs (Ghazanfar & Prügel-Bennett, 2014).
Filtrage basé sur le contenu	- Le modèle n'a pas besoin de données sur les autres utilisateurs, puisque les recommandations sont spécifiques à cet utilisateur. - Plus facile de l'adapter à un grand nombre d'utilisateurs. - Le modèle peut capturer les intérêts spécifiques d'un utilisateur et recommander des items spécifiques auxquels très peu d'autres utilisateurs s'intéressent. - Pas besoin d'information sur les autres utilisateurs.	- L'analyse du contenu est nécessaire pour faire une recommandation. - Problème de recommandation des images et de vidéos sans Métadonnées. - Nécessite du profil d'utilisateur.

`

2.6 Concepts préliminaires

2.6.1 Taxonomies dans la recommandation

Une taxonomie peut être définie comme une collection de catégories selon lesquelles les items peuvent être organisés de façon hiérarchique (Cho & Kim, 2004 ; Weng et al., 2008). De plus, tout élément doit appartenir à une ou plusieurs catégories de la taxonomie. Par exemple, dans la taxonomie d'Amazon présentée à Figure 2.4, le livre "City of Bones" appartient aux catégories fiction et mythologie. Selon certaines études, les taxonomies sont attribuées par des humains et capturent la connaissance du domaine, fournissant ainsi des preuves solides sur la sémantique des items (Ahmed et al., 2015 ; Cho & Kim, 2004 ; Ziegler, Lausen, et al., 2004).

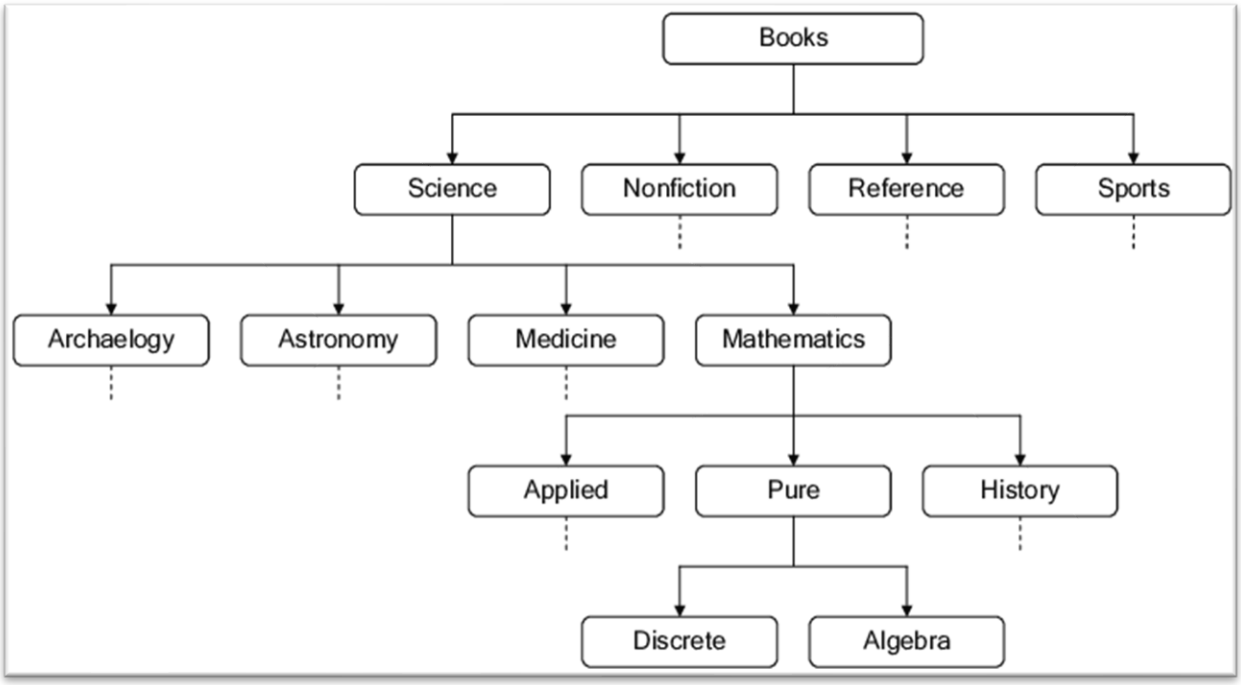

Figure 2.4 Extrait de la taxonomie des livres d'Amazon (Ziegler & Lausen, 2009)

La taxonomie a attiré une attention considérable dans de nombreux domaines d'application, en raison de son utilité fondamentale et de sa hiérarchie arborescente (Yang et al., 2020 ; Zhang et al., 2018). Les méthodes précédentes de recommandation utilisent généralement les données de la taxonomie pour résoudre le problème de sparsité et de calcul coûteux. De nombreux systèmes de recommandation impliquent des items avec des descriptions textuelles, et une taxonomie pour ces

items peut aider le système à mieux comprendre les intérêts de l'utilisateur pour faire des recommandations plus précises (Zhang et al., 2014). Par exemple, C.N. Ziegler et al, (Ziegler, Schmidt-Thieme, et al., 2004) a exploité les connaissances taxonomiques de base pour déduire efficacement le profil des utilisateurs. J. Gao et al, (Gao et al., 2019) ont proposé de générer des explications appropriées pour les résultats de la recommandation à l'aide des données de la taxonomie. Cho et Kim (Cho & Kim, 2004) ont utilisé des taxonomies pour réduire la dimension de la matrice d'évaluation afin d'améliorer les recommandations basées sur les recherches de plus proches voisins. Ziegler et al. (Ziegler, Lausen, et al., 2004) et Weng et al. (Weng et al., 2008) ont représenté les utilisateurs par leur intérêt pour les catégories d'une taxonomie des items. Kanagal et al. (Kanagal et al., 2012) ont proposé un modèle de facteur latent tenant compte de la taxonomie, qui combine les taxonomies et les facteurs latents en utilisant des modèles additifs. De même, Ahmed et al. (Ahmed et al., 2015) ont proposé un modèle additif hiérarchique pour exploiter les préférences des utilisateurs envers les attributs des items catalogués dans différentes catégories.

Outre les taxonomies, de nombreuses études (Ma et al., 2019; Wang et al., 2019) basées sur les graphes de connaissances sont appliquées pour atténuer le problème de la sparsité. Bien que les graphes de connaissances et les taxonomies puissent être considérés comme des informations secondaires pour les systèmes de recommandation, les graphes couvrent diverses relations sémantiques tandis que les taxonomies se concentrent sur les relations hiérarchiques. Par rapport aux graphes, les avantages de l'incorporation d'une taxonomie sont les suivants : (1) les graphes couvrent de nombreuses relations qui ne sont pas pertinentes pour la tâche de recommandation, dont l'incorporation aveugle peut entraîner un coût de calcul élevé et même nuire aux performances de recommandation, et (2) en se concentrant sur les relations hiérarchiques, nous pouvons tirer parti des structures arborescentes des taxonomies et des espaces hyperboliques pour mieux organiser l'intégration utilisateur/item et obtenir des recommandations plus précises et interprétables.

Bien que les avantages de la taxonomie semblent éminents dans la recommandation, les méthodes récentes tenant compte de la taxonomie ne peuvent pas fonctionner lorsque les données de la taxonomie ne sont pas disponibles. Des travaux en traitement automatique des langues qui utilisent des algorithmes d'apprentissage automatique pour construire des taxonomies à partir de données non structurées sont aussi proposés pour construire automatiquement une taxonomie pour

`

assurer la disponibilité de la connaissance structurelle (Downey et al., 2015; Zhang et al., 2018). Zhang et al. (Zhang et al., 2014) a proposé de découvrir une taxonomie à partir de données d'achat automatiquement. Les auteurs proposent une formulation purement basée sur la description des items pour identifier tous les termes descriptifs qui sont informatifs des catégories de la taxonomie auxquelles un item appartient.

2.6.2 Extraction des règles d'association

La technique d'extraction de règles d'association (Association Rules Mining ***ARM***) a été appliquée pour représenter les intérêts des utilisateurs dans divers domaines afin de fournir des modèles de recommandation puissants. Cela est dû à sa capacité à s'adapter à de grands ensembles de données et à atteindre une précision élevée (Tyagi & Bharadwaj, 2013). Kardan et Ebrahimi (2013) ont employé la technique d'extraction de règles d'association dans un système de recommandation hybride pour calculer la similarité entre les utilisateurs à partir de données implicites collectées dans un groupe de discussion. Feng et al. (2015) ont proposé une nouvelle technique de recommandation dans laquelle les facteurs temporels de l'intérêt des utilisateurs ont été incorporés à la fois dans la détection de communautés chevauchantes basée sur une génération de graphe utilisateur-utilisateur avec des liens pondérés dans le temps et dans la génération de règles d'association basée sur des communautés chevauchantes temporelles pour modéliser l'intérêt des utilisateurs dans le temps.

La recherche de FC utilisant l'extraction de règles d'association sont encore limitées (Tyagi & Bharadwaj, 2013). Par exemple, les techniques d'extraction de règles d'association utilisées par Feng et al. (2015), Kardan et Ebrahimi (2013) et Tyagi et Bharadwaj (2013) n'ont pas pris en compte le problème de la répétition d'items dans une transaction ou achats multiples par transaction. Aussi, les règles d'association ne sont pas utilisées directement pour renforcer la matrice utilisateur-item en ajoutant des items complémentaires.

Dans ce projet, nous adaptons la technique d'extraction des règles d'association pour traiter efficacement les données massives et identifier l'historique d'achat similaire des utilisateurs sur le même ensemble de transactions pour prédire les préférences inconnues des utilisateurs en les ajoutant des items complémentaires. Inspire par le travail de *Loannis Schoinas et Christos Tjortjis*

`

(2017), nous proposons de renforcer la matrice utilisateur-item par les règles d'associations générées.

- **Aperçu de l'extraction de règles d'association**

L'objectif de la technique d'extraction des règles d'association est de découvrir les relations intéressantes parmi un grand ensemble de données en décrivant les relations entre les chemins de navigation typiques des utilisateurs passés. En général, les intérêts des utilisateurs peuvent être identifiés par l'extraction de règles d'association sous la forme "A→B" (où A et B sont des intérêts ou des éléments de l'utilisateur), ce qui signifie que les utilisateurs intéressés par "A" sont susceptibles d'être intéressés par "B" ou, en d'autres termes, l'occurrence de l'élément A (côté antécédent de la règle) entraîne l'occurrence de l'élément B (côté conséquent de la règle) sur la base des transactions des utilisateurs d'un ensemble de données. Par exemple, dans une analyse marketing, une règle d'association "Banane → Nutella" indique que les utilisateurs qui achètent de la banane sont enclins à acheter des chocolats (Nutella). En se basant sur l'idée des règles d'association sur l'analyse des données, des règles telles que "lorsqu'un client achète des items X, il peut également acheter des items Y avec une probabilité de p%" ou "les utilisateurs qui ont aimé les items X, ont également aimé les items y" peuvent être découvertes.

Il y a deux concepts importants liés à la technique d'extraction de règles d'association, les mesures de support et de confiance. Les règles sont évaluées à l'aide de ces mesures. Seules les règles ayant un niveau (donné) de support et/ou confiance sont sélectionnées comme règles utiles.

Le support et la confiance d'une règle d'association A→B sont définis par :

$$supportA \rightarrow B(\%) = \frac{nombre\ des\ transactions\ incluant\ A\ et\ B}{nombre\ total\ des\ transactions} \tag{2.1}$$

$$confianceA \rightarrow B(\%) = \frac{nombre\ des\ transactions\ incluant\ A\ et\ B}{nombre\ des\ transactions\ incluant\ A} \tag{2.2}$$

La motivation pour appliquer les règles d'association dans le système de recommandation est basée sur l'idée de cette technique qui découvre les associations entre les items ou les utilisateurs en se basant sur l'association des données de l'utilisateur actif avec les données des autres

`

utilisateurs ou en associant les données des items qui intéressent l'utilisateur actif avec les données des autres items disponibles sur le système. En fait, l'extraction de règles d'association décrit la probabilité des associations entre les éléments traçables (utilisateurs ou items) dans la base de données. Les règles générées par les règles d'association sont faciles à interpréter, et peuvent donc être facilement utilisées dans la pratique.

2.6.3 Initialisation de la méthode de la factorisation matricielle

Une autre approche intéressante visant à augmenter la performance est via l'initialisation de l'algorithme de factorisation matricielle.

- **Aperçu sur la technique de factorisation matricielle :**

Nous supposons que nous avons une matrice de données (dite matrice utilisateur-item) $X \in R^{n*p}$ résumant les préférences de n utilisateurs envers p items comme illustré dans la matrice ci-dessous.

$$X = \begin{bmatrix} r_{1,1} & \cdots & r_{1,p} \\ \vdots & \ddots & \vdots \\ r_{n,1} & \cdots & r_{n,p} \end{bmatrix}$$

Le principe de la technique de factorisation matricielle, comme dans le cas de l'algorithme *i-ALS*, est d'apprendre un modèle latent (s'appelle aussi facteur latent) d'utilisateurs $U \in R^{n*k}$ et d'items $I \in R^{p*k}$ tel que la reconstruction $F_{ij} = U_i \,.\, M_j$ entre un utilisateur i et un item j estime le terme X_{ij} ou son utilité (score). U_i désigne la i-ième ligne de la matrice U et I_j désigne la j-ième colonne de la matrice I. Ainsi, F_{ij} peut être exploitée pour délivrer une recommandation en sélectionnant les N premiers éléments ayant le score le plus élevé pour un utilisateur donné. Le but est donc d'identifier des variables latentes permettant de déterminer un profil. Ces facteurs latents U et I sont généralement obtenus en minimisant une fonction objective qui provient de fonctions de coût régularisées (Takács et al., 2009) ou de modèles probabilistes (Krohn-Grimberghe et al., 2012). Afin d'apprendre ces facteurs latents, une initialisation de ces derniers est nécessaire (Koren et al., 2009).

Smilde et al. (Smilde et al., 2005) a démontré qu'une bonne initialisation peut améliorer la vitesse et la précision des approches basées sur la factorisation matricielle, car elle peut produire

`

une convergence plus rapide vers un minimum local. Dans les travaux (Johnson, 2014 ; Koren, 2008), l'approche de la factorisation matricielle implique une initialisation aléatoire des matrices de facteurs latents (d'utilisateur et d'item) et la mise en œuvre d'une méthode d'optimisation répétée sur les matrices. À chaque itération, les matrices sont mises à jour en minimisant l'erreur d'une fonction de coût. Après quelques itérations, ces méthodes convergent vers un optimum local qui dépend du point de départ (l'initialisation). L'objectif de l'approche proposée dans ces deux items est de présenter une autre façon d'initialiser les matrices de facteurs. L'hypothèse est qu'une initialisation appropriée permet aux méthodes de factorisation de produire des matrices facteurs plus précis et donc de conduire à une précision accrue de l'algorithme. Pour cette tâche, l'implémentation de SVD est proposée par ces auteurs.

Donoho, D. et al, (Donoho & Grimes, 2003) ont montré qu'un bon choix des valeurs initiales affecte non seulement la propriété de convergence de l'algorithme mais en particulier la qualité de la solution vers laquelle il converge. Dans notre travail, nous proposons une autre technique de décomposition plus avancée qui s'appelle Recherche Lineaire par Retour arrière (appelé aussi *Backtracking Line Search SVD++* ou ***BLS-SVD++*** en anglais***)*** (Wang et al., 2020). La motivation pour l'utilisation d'une initialisation basée sur la variante ***BLS-SVD++*** de la technique *SVD* est que Wang S et al. (2020) montrent avec des expériences sur les ensembles de données réelles comme *FilmTrus*[4] et *MovieLens 1M*[5] que l'algorithme ***BLS-SVD++*** proposé surpasse le *SVD* et d'autres algorithmes de recommandation en termes de performance et de temps de convergence.

2.7 Travaux connexes

Dans cette section, nous présentons divers articles de recherche étroitement liés à notre projet. Certains de ces articles sont discutés ici sur le travail dans le filtrage collaboratif et le problème du feedback implicite et de sparsité.

---

[4] http://konect.cc/networks/librec-filmtrust-trust/

[5] https://grouplens.org/datasets/movielens/

`

Dans le travail **(Lee et al., 2006b**), les auteurs ont développé un système de recommandation basé sur une ontologie pour les marchés B2B (*Business-to-Business*). Leur méthode est basée sur les mots-clés et indépendante de la structure physique sous-jacente de l'ontologie des produits. Plus précisément, leur méthode est basée sur une technique de recommandation basée sur le contenu qui représentait les données sur les items dans des graphes ontologiques.

De nombreuses options différentes ont été proposées pour exprimer au mieux les intérêts de l'utilisateur. Selon une étude de 2011 sur la relation entre l'évaluation implicite et explicite **Pradel et al. (2011)**, le nombre de fois où l'utilisateur a interagi avec un item a la corrélation significative sur la façon dont il évalue l'item. Dans le cas d'évaluation implicite, les auteurs affirment que plus l'utilisateur a acheté un item, plus il est susceptible d'attribuer une note élevée à cet item.

Le problème de recommandations en utilisant le feedback implicite a été abordé dans la littérature. **Koren et al. (2008)** abordent les cas où le feedback explicite n'est pas disponible et présente un algorithme produisant des recommandations basées strictement sur des informations recueillies implicitement. L'algorithme est classé dans la catégorie du filtrage collaboratif et plus précisément dans la catégorie de la factorisation matricielle. En outre, l'algorithme fournit des explications sur les recommandations qu'il produit. L'item présente une méthode de facteur latent pour le filtrage collaboratif d'ensembles de données où les seules informations disponibles sur les préférences des utilisateurs proviennent de données sur leur comportement (feedback implicite). Il suggère également une méthode permettant de donner aux utilisateurs une explication des recommandations qui leur sont faites. Les facteurs latents, sur lesquels les utilisateurs et les items sont projetés, sont déterminés en minimisant l'erreur quadratique entre la préférence souhaitée de l'utilisateur p_{ui} pour chaque paire (utilisateur u, item i) (déterminée à partir des données collectées) et celle générée par le modèle factoriel (déterminé par le produit scalaire de deux matrices des facteurs latents de l'utilisateur et de l'item, x_u et y_i, respectivement). Cette fonction objective est pondérée par un facteur de confiance de l'observation pour chaque paire c_{ui} et régularisée par λ pour éviter un sur-apprentissage excessif lors de la phase d'entrainement de l'algorithme. La fonction objective résultante est:

`

$$\min_{x,y} \sum_{u,i} c_{ui}(p_{ui} - x_u^T y_i)^2 + \lambda \left(\sum_u |x_u|^2 + \sum_i |y_i|^2 \right) \quad (2.3)$$

L'ensemble de données utilisé pour tester l'algorithme est constitué de l'historique de visionnage des utilisateurs de 300 000 terminaux d'un service de télévision numérique. Chaque information indique combien de fois un utilisateur a regardé un programme pendant une période de quatre semaines. L'algorithme est comparé à deux autres modèles, un qui produit une recommandation basée sur la popularité et un algorithme de filtrage collaboratif item-item. L'évaluation a montré que cette méthode produisait les meilleurs résultats parmi les trois. L'implémentation proposée donne de bons résultats et établit les règles de base pour le traitement des recommandeurs à feedback implicite.

Similairement à **Koren et al. (2008)**, **Guo et al. (2017)** se concentre sur les feedbacks implicites et en traitant le problème de la sparsité des données dont souffrent les algorithmes de filtrage collaboratif. Le problème de la sparsité des données concerne les ensembles de données avec un grand nombre d'utilisateurs et d'items et un petit nombre d'interactions entre eux. Ce problème affecte la précision des modèles de filtrage collaboratif, en particulier les approches de reconnaissance de voisinage qui, dans de tels cas, sont incapables de regrouper avec précision les utilisateurs et les items. Ce travail vise à résoudre le problème de la sparsité des données en incorporant diverses informations collectées implicitement, plutôt que d'utiliser uniquement les informations sur les achats. Les diverses informations implicites sont appelées données auxiliaires multitypes. Les données auxiliaires sont considérées comme les différentes façons dont l'utilisateur interagit avec les items. Les requêtes de recherche d'items, l'ajout à la liste de souhaits, le partage sont quelques exemples de données auxiliaires.

En lien avec **Koren et al. (2008) et Guo et al. (2017), JinHyun et al. (Jooa et al., 2016)** ont concentré sur les problèmes des feedbacks implicites. Dans cette recherche, une approche hybride pour un système de recommandation incorporant le filtrage collaboratif ainsi que les algorithmes d'extraction de règles d'association ont été proposés. En outre, les données utilisées dans la recherche actuelle sont également recueillies de manière implicite. L'objectif est de prédire efficacement les vendeurs ou les entreprises locales que le client a de la chance de visiter. Enfin, comme le moteur de recommandation est prévu pour fonctionner sur des appareils mobiles, il

`

intègre le GPS (*Global Positioning System*) des appareils et prend en compte la proximité du vendeur par rapport au client pour la recommandation finale. Au cours de la première phase du système, la partie d'exploration des règles d'association a lieu. L'utilisation de l'algorithme *K-means* pour l'extraction de règles permet de prédire les catégories des vendeurs qui pourraient intéresser le client. Au cours de la phase suivante, la partie de filtrage collaboratif est mise en œuvre. Dans cette phase, le système recommande des vendeurs jamais visités par le client. La prédiction est réalisée en synergie avec les règles dérivées de la phase précédente. La similarité entre les utilisateurs est calculée en fonction de leurs choix passés sur les vendeurs. De plus, le coefficient de corrélation de Pearson a été utilisé pour mesurer la similarité entre les utilisateurs. Les évaluations des utilisateurs similaires sur les vendeurs non visités par le client spécifique sont utilisées pour créer les recommandations. L'implémentation présentée ne vise pas à améliorer l'algorithme des systèmes de recommandation, mais plutôt à mettre en œuvre une combinaison des deux approches dans le contexte d'une application mobile recommandant des vendeurs.

Contrairement aux qui utilise le filtrage collaboratif avec des feedbacks implicites, dans **(Dutta & Mukhopadhyay, 2011)**, le système proposé vise à créer des recommandations en utilisant les évaluations explicites des utilisateurs. L'item introduit un algorithme hybride mettant en place le filtrage collaboratif et les règles d'association pour produire des recommandations. Plus le feedback explicite, le système considère également les informations de feedback implicites en proposant une démarche de transformation des informations de transaction d'achat en un système de notation implicite. En outre, il présente une façon dont la reconnaissance des séquences peut améliorer les performances du système. Le filtrage collaboratif utilise la similarité cosinus pondérée sur les évaluations des utilisateurs. La similarité en cosinus mesure la similarité entre deux vecteurs d'un espace de produit scalaire. Elle est mesurée par le cosinus de l'angle entre deux vecteurs et détermine si deux vecteurs pointent approximativement dans la même direction (Han et al., 2012). Le système vise à produire des recommandations supplémentaires en utilisant l'historique des achats pour extraire des règles d'association. En mesurant le support et la confiance trouvés dans l'historique des achats des utilisateurs (lors de l'extraction des règles d'association), le système est capable de déterminer si un utilisateur est susceptible d'acheter un produit en se basant uniquement sur ses achats précédents. Enfin, le système prend en compte la reconnaissance des séquences. Il enregistre l'ordre des items achetés et si un item recommandé n'est pas vendu après la séquence

`

spécifique d'items achetés précédemment, il n'est pas inclus dans les recommandations. Les résultats expérimentaux démontrent la précision supérieure des évaluations explicites par rapport aux évaluations implicites. Bien que l'amélioration de la précision en utilisant une combinaison de feedback implicite et explicite ne soit pas significative, elle constitue néanmoins une amélioration. De plus, la précision est augmentée par l'implémentation des recommandations de règles d'association dans le système. En outre, l'article présente un système qui produit des multiples recommandations, certains items étant recommandés via le filtrage collaboratif et d'autres à partir de règles d'association. Enfin, l'article montre comment l'analyse des séquences d'achat des items peut profiter aux systèmes de recommandation.

Bien que la méthode de filtrage collaboratif présente généralement de bonnes performances, elle est confrontée à deux problèmes majeurs : la sparsité des données et la scalabilité. **Choonho Kim et al. (2003)** ont traité ces deux problèmes. L'approche proposée applique des règles d'association entre les catégories dans des taxonomies des items à plusieurs niveaux pour aborder la problématique de la sparsité des données. L'approche peut être appliquée à des items organisés en une structure hiérarchique de catégories, comme la classification des items dans un grand magasin (la hiérarchie *"is-a"*). En fait, dans le cas où les données d'achat sont utilisées comme préférence des utilisateurs, le nombre de règles d'association applicables est très faible en raison du fait que chaque utilisateur n'achète habituellement qu'un petit nombre d'items. Dans ce cas, il est impossible de prédire la préférence pour la plupart des items et la performance de la recommandation devient très faible. Pour surmonter ce problème, cette approche introduit une méthode de recommandation utilisant des informations supplémentaires - les règles d'association entre les catégories de niveau supérieur - si les items sont organisés en une structure de catégories hiérarchiques. L'utilisation d'une telle hiérarchie d'items augmente le nombre d'items recommandables pour les utilisateurs dont les données de préférence connues sont tellement limitées qu'il n'est pas possible de produire des recommandations. La méthode consiste d'abord à trouver la catégorie c_i à laquelle appartient l'item préféré. S'il existe une règle d'association de catégorie de la forme $c_i \Rightarrow c_j$, alors elle accorde un certain degré de préférence à tous les éléments qui appartiennent à la catégorie c_j. Les expériences réalisées à l'aide de deux jeux de données *MovieLens* (http://www.cs.umn.edu/Research/GroupLens/index.html) et *KDD*

`

(http://www.ecn.purdue.edu/KDDCUP) montrent que l'algorithme proposé augmente la précision de la recommandation par rapport à l'application de règles d'association à un seul niveau. L'algorithme proposé présente également de meilleures performances par rapport au filtrage collaboratif de base lorsque le nombre de recommandations augmente.

Dans **Pradel et al. (2011)**, nous avons un système de recommandation utilisé par les vendeurs dans un magasin comme un outil pour recommander des items aux utilisateurs plutôt que l'implémentation habituelle d'un système de recommandation en ligne. Les algorithmes utilisés dans cet item prennent en compte l'historique des achats et les règles d'association pour produire ses recommandations. De plus, les résultats soulignent l'importance des systèmes de recommandation sensibles au contexte. Les informations utilisées pour l'apprentissage du modèle proposé utilisent une matrice utilisateur-item. Chaque ligne de la matrice comprend les informations d'achat d'un client pour chaque item sur une période spécifique. Pour évaluer l'hypothèse selon laquelle le contexte peut augmenter la précision de la prédiction, trois algorithmes ont été mis en œuvre et évalués. Le premier algorithme met en œuvre un algorithme collaboratif basé sur les items (*Item-Based Colaborative Fieltring Prediction*), le deuxième met en œuvre un algorithme de factorisation matricielle en utilisant la décomposition en valeurs singulières (SVD) (*Prediction with Matrix Method*) et le troisième crée une matrice basée sur l'extraction de règles d'association (*Prediction with Association Rules*). L'évaluation des trois algorithmes sur deux paramètres temporels différents (historique d'achat complet et une limite de deux semaines d'achat) a montré que l'algorithme le plus précis lors de l'utilisation de l'historique d'achat complet est l'algorithme de factorisation matricielle contrairement aux cas de limite de deux semaines imposées, l'algorithme basé sur les règles d'association présentent les meilleurs résultats. La comparaison des performances du même algorithme dans les différentes limites de temps montre que les trois algorithmes ont été affectés positivement. Il convient toutefois de noter que dans le cas de SVD, la différence est considérée comme insignifiante puisqu'elle ne dépasse pas 0,02 %. Les règles d'association ont présenté une augmentation de la précision de 3 à 4 %. Finalement, Le filtrage collaboratif basé sur les items présente également des avantages significatifs, de l'ordre de 3,5 % à 4 %. On observe que les techniques de factorisation matricielle ne bénéficient pas beaucoup de la conscience du temps dans les données d'entrée.

`

En abordant le même problème que **Kim et al. (2003), Nasiri et al. (2016)** propose une nouvelle approche. En effet ce travail vise à améliorer la précision des algorithmes classiques de filtrage collaboratif en abordant le problème de la sparsité des données. Plus précisément, les auteurs traitent de l'optimisation de la technique de factorisation matricielle. Dans l'approche classique de la factorisation matricielle, l'initialisation aléatoire des matrices de facteurs n'est pas optimale pour la méthode d'optimisation. L'objectif de l'approche proposée dans cette étude est de présenter une autre façon d'initialiser les matrices de facteurs et ainsi d'améliorer la précision de l'algorithme. Pour cette tâche, l'implémentation de SVD est proposée par les auteurs. La méthode proposée comporte deux sections. Tout d'abord, la première section factorise la matrice utilisateur-item via l'algorithme de factorisation matricielle (SVD). Cela permet d'obtenir les matrices de facteurs latents des utilisateurs et des items. Ensuite, dans la deuxième section, ils définissent ces matrices comme matrices initiales de l'algorithme principal du système de recommandation. Ainsi, l'algorithme principal est un algorithme de factorisation matricielle qui se résout par des méthodes d'optimisation. Pour l'évaluation, l'approche proposée a été testée contre un algorithme de factorisation matricielle initialisé de façon aléatoire avec la même méthode d'optimisation. Les résultats montrent que la méthode proposée a amélioré le système. Plus précisément, elle a réduit la mesure d'évaluation ***RMSE*** (*Root Median Squared Error*) de 0,06. En outre, l'initialisation des matrices de facteurs a permis à la méthode d'optimisation de nécessiter moins d'itérations pour trouver les matrices avec le moins d'erreur. En particulier, il est signalé que pour les cas où le nombre de facteurs à découvrir est augmenté, la méthode proposée était deux fois plus rapide que l'algorithme non initialisé.

Dans **Yuchen Zhang et al. (2014)**, les auteurs ont abordé le problème de l'inférence d'une taxonomie pour les systèmes de recommandation. Le lissage de modèles de facteurs latents sur une taxonomie permet de lutter contre la sparsité. Cependant, dans de nombreuses situations, une taxonomie induite par l'homme n'est pas disponible, par exemple lorsque de nouveaux items arrivent, lorsqu'un nouveau marchand est ajouté au système (éventuellement avec des items d'une culture/langue différente), ou lorsque les utilisateurs ne fournissent pas de catégories pour les nouveaux items (comme dans les vidéos *youtube*). Ce probleme est traité par cet item. En se basant sur le fait qu'il est toujours possible d'obtenir une description textuelle des items, les auteurs ont décrit une méthode non supervisée et non paramétrique qui apprend la structure de la taxonomie

`

sur les items à partir de la description textuelle des items et des données d'achat. L'objectif dans la conception d'un modèle facteurs latents basé sur l'hiérarchique est de générer automatiquement une catégorisation hiérarchique pour tous les items sur la base de leurs descriptions et de leurs données d'achat, afin de pouvoir apprendre les facteurs latents item/utilisateur conjointement. La phase de conception de l'arbre des catégories est la suivante : considérer un item arbitraire *i* que l'on veut rattacher à l'arbre des catégories. Pour cela, l'algorithme génère un chemin dans l'arbre de la racine au nœud feuille qui représente *i*. En partant du nœud v, la probabilité de sélectionner un enfant existant (ou nouveau) est :

$$\mathrm{P}(\mathrm{y} \rightarrow \mathrm{z}) = \begin{cases} \dfrac{n_z}{n_y + \alpha} & si\ z \in C(y) \\ \dfrac{\alpha}{n_y + \alpha} & pour\ un\ nouveau\ nœud \end{cases}$$

Où n_y est le nombre d'éléments appartenant à la catégorie y et n_z est le nombre d'éléments qui appartiennent à la catégorie z. Le paramètre α contrôle la probabilité de créer un nouvel enfant pour y. Une fois qu'un nœud enfant est sélectionné, le processus est répété jusqu'à ce qu'un chemin complet soit défini. L'expérience se déroule en deux parties : la première consiste à comparer la méthode proposée et trois algorithmes de référence de l'état de l'art basés sur le facteur latent. La deuxième partie se concentre sur la comparaison entre les taxonomies entièrement ou partiellement induites par des humains et les taxonomies générées automatiquement par l'algorithme présenté précédemment. Tout d'abord, les auteurs ont montré que les performances du modèle proposé sont supérieures, favorablement, à celles des algorithmes de base. Les résultats ont également montré qu'en utilisant uniquement les descriptions textuelles des items, le modèle des auteurs surpasse le modèle de facteur latent qui repose sur une taxonomie induite par les humains.

Contrairement aux travaux précédemment présentés qui travaille sur le système de recommandation dans une contexte *Business-to-Customer*, **Lee et al. (2006b)** ont développé un système de recommandation des produits basé sur l'ontologie pour les plateformes commerciales Business-to-Business (B2B). Plus précisément, leur méthode repose sur une technique de recommandation basée sur le contenu qui représente les données relatives aux items dans des graphes ontologiques. La discussion dans cet article montre qu'un système de recommandation basé sur l'ontologie de produit fournit une meilleure précision que les systèmes basés sur la

`

technique de recherche par texte. L'approche présentée dans l'article permet aux utilisateurs de référencer l'ontologie de produit directement par le biais d'une interface simple de mots-clés.

2.8 Positionnement des contributions

Dans la littérature, peu d'attention a été portée sur l'utilisation de la taxonomie automatiquement générée et les règles d'association pour aborder le problème de sparsité. Dans ce projet, nous proposons de combler ce gap de littérature en :

1) Nous prétraitons les bases de données des commandes facturées et des items de l'entreprise partenaire et adaptons des données au problème du système de recommandation.
2) Nous proposons une nouvelle méthode, basée sur l'extraction des règles d'association, qui permet de renforcer la matrice Utilisateur-Item afin d'améliorer la qualité de recommandations en traitant le problème de sparsité de données ;
3) De plus, une approche destinée à augmenter la précision des algorithmes de factorisation matricielle via l'initialisation de l'algorithme de factorisation matricielle est implémentée.
4) Nous étudions le problème de la sparsité des données. En particulier, nous utilisons la description détaillée du produit comme source discriminante en sélectionnant des termes de type taxonomique, ce qui nous permet de générer automatiquement une nouvelle taxonomie. Nous utilisons cette taxonomie avec le modèle basé sur les facteurs latents renforcé par la technique d'extraction des règles d'association pour augmenter l'efficacité de nos recommandations.
5) Nous effectuons des expériences avec une grande variété de modèles et paramètres en utilisant un ensemble de données réelles.

2.9 Conclusion

Nous avons essayé à travers ce chapitre de mettre l'accent sur les systèmes de recommandation, ainsi que leurs principaux types et leurs points forts et faibles. Un point sur les problèmes des

`

systèmes de recommandation a été aussi abordé. Nous avons présenté également les concepts préliminaires de notre solution proposée. En outre, nous avons discuté des travaux connexes de la littérature, en les reliant aux concepts de base de notre solution.

Dans le chapitre suivant, nous présentons notre solution pour une recommandation basée sur les feedbacks implicites.

`

CHAPITRE 3 MÉTHODOLOGIE

Dans ce travail, nous nous intéressons à un système de recommandation des items dans le contexte du commerce électronique business-to-business. Il s'agit d'une approche de recommandation basée uniquement sur des informations collectées implicitement. La principale technique utilisée est un algorithme de factorisation matricielle basé sur une taxonomie d'item générée automatiquement à partir des textes de description des items. En outre, la technique d'extraction de règles d'association est utilisée pour optimiser le système, ainsi que pour fournir des préférences supplémentaires afin de surmonter le problème de la sparsité des données. Enfin, pour améliorer les performances de l'algorithme FM, une initialisation des matrices de facteurs latents est proposée.

3.1 Approche proposée

Notre approche concerne un système de recommandation des items dans le contexte du commerce B2B. Il s'agit d'une approche de recommandation basée uniquement sur des informations collectées de manière implicite. La principale technique utilisée est un algorithme de factorisation matricielle, *implicit-ALS* (alias *i-ALS*). Dans la première phase, nous avons généré une nouvelle taxonomie pour surmonter le problème de sparsité. Ensuite l'exploration de règles d'association est utilisée pour l'optimisation du système, ainsi que pour fournir des recommandations supplémentaires en parallèle à la méthode principale (Schoinas & Tjortjis, 2019). Finalement, une initialisation de l'algorithme *i-ALS* sera effectuée pour améliorer la précision.

Un diagramme de l'approche proposée est présenté dans la Figure 3.1.

`

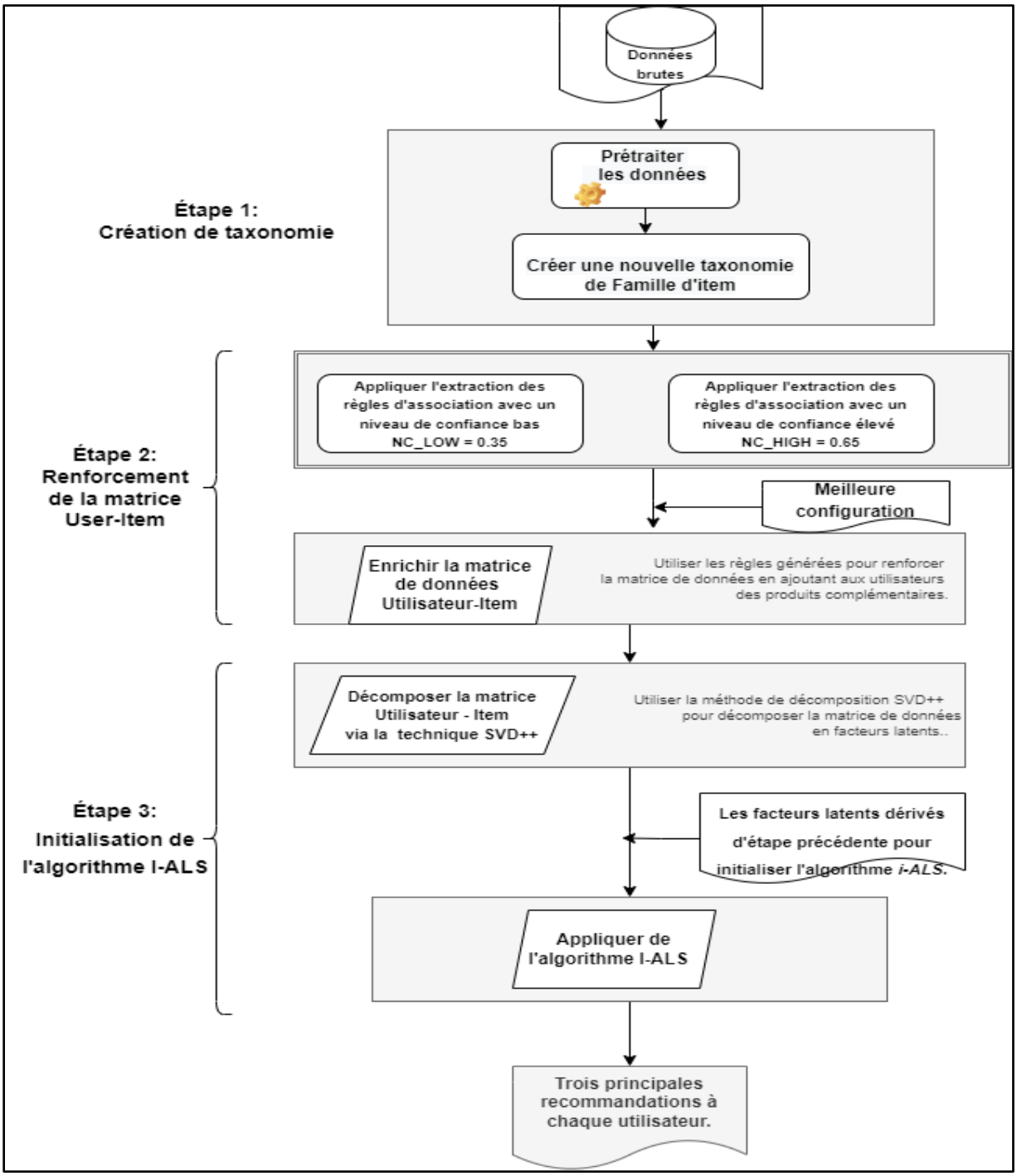

Figure 3.1 Étapes de l'approche proposée

Dans la première étape, en se basant seulement sur la description textuelle des items, nous générons automatiquement une nouvelle taxonomie appelée *Famille d'item*. Ensuite, en utilisant cette nouvelle taxonomie, nous construisons une matrice utilisateur-item (M_{ui}) de préférence

`

déduite en observant le comportement d'achat du client (dans ce cas chaque élément *mui* de la matrice représente la fréquence d'achat d'une famille d'item *i* donnée par un utilisateur *u*). Dans la deuxième étape, nous exploitions les résultats de la technique de l'extraction des règles d'association (Sharma & Mann, 2013) pour résoudre le problème de la sparsité des données présentes dans notre ensemble de données. Finalement, dans la troisième étape, nous visons à augmenter davantage les performances de recommandations en proposant une configuration optimale de l'algorithme principal utilisé en initialisant les matrices de facteurs latents par la décomposition de la matrice M_{ui} en appliquant l'algorithme de décomposition en valeurs singuliers (***BLS-SVD***++) (Wang et al., 2020).

L'algorithme principal que nous utilisons est une variation de filtrage collaboratif utilisant les Moindres carrés alternatifs (applee aussi ***Alternating Least Square*** ou ***ALS*** en anglais) pour les feedback implicites, appelé *i-ALS*. L'i-ALS est un modèle de recommandation basé sur les algorithmes décrits dans l'article "Collaborative Filtering for Implicit Feedback Datasets" avec des optimisations de performance décrites dans "Applications of the Conjugate Gradient Method for Implicit Feedback Collaborative Filtering" (Koren 2009 ; Volinsky, 2008 ; Takács, 2011). L'idée clé derrière leur approche est d'attribuer une valeur de préférence r_{ui} et un niveau de confiance à chaque élément de la matrice *utilisateur-item*. La préférence prend généralement les valeurs 0 et 1, indiquant si un utilisateur a acheté un item particulier ou non. Le niveau de confiance est lié à la mesure dans laquelle nous sommes convaincus des préférences de l'utilisateur. Si par exemple il a acheté l'item dix fois, nous pouvons être plus sûrs de leur préférence.

3.2 Création de nouvelle taxonomie

3.2.1 Prétraitement des données

Avant d'entrer dans les détails algorithmiques, nous présentons l'ensemble de données du partenaire industriel. Dans ce projet, nous avons pu explorer deux principales bases de données : les commandes facturées (*Orders.csv*) et les items courants (*Produits.csv*). La base de données *"Ordres"* (voir Tableau 3.1) contient environ 1,2 million de données transactionnelles historiques de janvier 2007 à octobre 2020. Elle comporte plus de 50 champs tels que l'emplacement géographique, la date de vente, le numéro de commande régional, la quantité achetée, le client qui

`

fait la transaction, etc. La base de données *"Produits"* (voir Tableau 3.2) contient plus de 40 000 items uniques et possède des informations détaillées sur les items offerts par le Groupe JSV, notamment les catégories, la quantité, les descriptions textuelles en anglais et en français, etc.

L'ensemble de données est filtré à l'aide d'une requête en langage de requête structuré (SQL). L'ensemble de données final contient les informations suivantes : l'ID de la transaction, l'ID du client, l'ID du produit, les descriptions en anglais et en français des items, et le nombre de fois qu'un client a acheté un item.

Tableau 3.1 Extrait de la base de données *'Orders'*

ORDER_ID	**ITEM_ID**	**DESC_ITEM**	**CLIENT_CODE**	**CATEGORY**	**SUB_CATEGORY**
355615	spaSS-270	"SQ HD SET SCRW CUP PT 5/8"" - 11 UNC X"	CC10468	Quincailleries	Divers
958159	BARPJ014	CORDE TORSADEE POLYPROPYLENE JAUNE 1/4"	CS10454	Quincailleries	Cables.cordes. Ficelles. Broches
305633	INF178017N	ECROU HEXAGONAL GR8 NC. 7/8-9 NOIR	CV10188	Boulonnerie	Écrous acier
719470	INF103018P	BOULON HEX GR 5 NC 1/4-20 X 3/4" ZINC	CSM002700	Boulonnerie	Boulons de carrosserie inox

`

Tableau 3.2 Extrait de la base de données ‘Produits’

ORDER_ID	ORDER_DATE	ITEM_ID	USER_ID	QUANTITY
355615	2022-07-14	spaSS-270	CC10468	156,9735427
958159	2011-05-14	BARPJ014	CS10454	86,18423634
305633	2012-01-05	INF178017N	CV10188	493,6560814
719470	2007-07-02	INF103018P	CSM002700	313,5242675
229060	2014-11-18	HX1FW037T20	CH10567	105,0067001

Afin d'obtenir un ensemble de données unifié, les ensembles de données ont été fusionnées par identifiant de transaction *(ORDER_ID),* de sorte que chaque ligne de l'ensemble de données s'est vu attribuer une transaction correspondant à la fréquence d'achat d’item par un utilisateur comme s’est illustrée dans *Tableau 3.3*. Nous partons de ce fichier principal pour la suite, car il contient toutes les informations nécessaires pour procéder à la factorisation de la matrice *Utilisateur-Item* et les différentes étapes.

Tableau 3.3 Extrait de l’ensemble de données finales

ORDER_ID	USER_ID	ITEM_ID	DESC_ITEM	FREQ_ACHAT
355615	CC10468	spaSS-270	"SQ HD SET SCRW CUP PT 5/8"" - 11 UNC X"	21
958159	CS10454	BARPJ014	CORDE TORSADEE POLYPROPYLENE JAUNE 1/4" (PRIX AU PIED) 1300'	110
305633	CV10188	INF178017N	ECROU HEXAGONAL GR8 NC. 7/8-9 NOIR (VR200/BT20)	35
719470	CSM002700	INF103018P	BOULON HEX GR 5 NC 1/4-20 X 3/4" ZINC (VR2700/BT100)	11

`

Après cette manipulation, les données ont été transformées sous forme d'une matrice de valeurs réelle pour la factorisation matricielle. La bibliothèque *Sparse* de *SciPy*[6] a été utilisée pour convertir les données en une matrice creuse (*CSR* pour ***C**ompresed **S**parse **R**ow*). La matrice *CSR* est un moyen plus optimal en termes de mémoire pour stocker une matrice régulière, qui ne stocke pas les valeurs manquantes via des zéros et stocke plutôt chaque valeur en mémoire via des indices. Les valeurs de cette matrice représentent des interactions entre 5 312 utilisateurs et 8 541 items. Une interaction est une fréquence d'achat de 0 à 2 111. Pour les données qui alimentent la matrice *i-ALS*, il est recommandé dans Koren (2008) de convertir ces interactions en des valeurs binaires. Les auteurs proposent cette représentation binaire afin de gérer le cas où ils ne sont pas sûrs de leurs exemples négatifs (items que les utilisateurs n'aiment pas). En effet, l'approche *i-ALS* apprend une représentation matricielle factorisée en utilisant différents niveaux de confiance sur les préférences binaires : les items non vus sont traités comme négatifs avec une faible confiance, tandis que les items achetés sont traités comme positifs avec une confiance beaucoup plus élevée. Pour les convertir et d'après l'implémentation originale de l'algorithme ***i-ALS***, nous transformons chaque valeur en une valeur binaire, r_{ui}^{imp} qui vaut 1 si l'utilisateur u a acheté l'item i, 0 sinon. r_{ui}^{imp} défini comme :

$$r_{ui}^{imp} = \begin{cases} 1 \; si \; f_{ui} \geq 5 \\ 0 \; sinon \end{cases}$$

Avec u : désigne l'utilisateur, i : désigne le produit et f_{ui} : fréquence d'achat d'item i par un utilisateur u. En d'autres termes, si un utilisateur u a acheté un item i plus de 5 fois (f_{ui} >= 5), alors nous avons une indication que u aime i (r_{ui}= 1). En revanche, si u n'a acheté un item i que 4 fois ou moins, nous ne croyons pas à une préférence (r_{ui} = 0). De manière analogues à Chen Gao et al. [2020], nous ne retenons que les utilisateurs et les items ayant au moins 20 interactions. La matrice *utilisateur-item* finale est composée de 1 160 359 interactions entre 4 909 utilisateurs et 5 646 items et avec une sparsité de 99,51%. Cela signifie que sur toutes les interactions (n_utilisateurs * n_items = 4 909 * 5 646) possibles entre l'utilisateur et l'item, seulement 0,49 % étaient présentes

[6] https://docs.scipy.org/doc/scipy/reference/sparse.html

`

ce qui rend la tâche de recommandation plus difficile, car une sparsité élevée est un problème critique dans ce contexte [G. Guo et al., 2017].

3.2.2 Utilisation des taxonomies

La recherche sur les algorithmes de filtrage collaboratif a atteint un pic dû à la compétition de recommandation de films de Netflix[7], qui a coûté un million de dollars. Les approches de filtrage collaboratif basées sur la factorisation matricielle, telles que la décomposition en valeurs singulières régularisées, obtiennent de bons résultats dans cette compétition, bien meilleurs que ceux de l'algorithme de Netflix qui utilise le coefficient de corrélation de Pearson (les plus proches voisins). Malgré ce succès considérable, l'approche du filtrage collaborative rencontre des difficultés spécifiques dans la recommandation des items pour les achats en ligne (C. A. Gomez-Uribe et al., 2016). En effet, sur un site Web de vente au détail typique, il existe un effet qui s'appelle en anglais *long tail*. Cela signifie que non seulement la plupart des utilisateurs n'achètent qu'un petit nombre d'items, mais aussi que la majorité des items ne sont que rarement achetés.

Dans notre cas, nous pouvons facilement détecter cet effet dans la Figure 3.2, qui démontre la présence de l'effet de *long tail* présenté dans l'ensemble des données.

[7] https://netflixprize.com

`

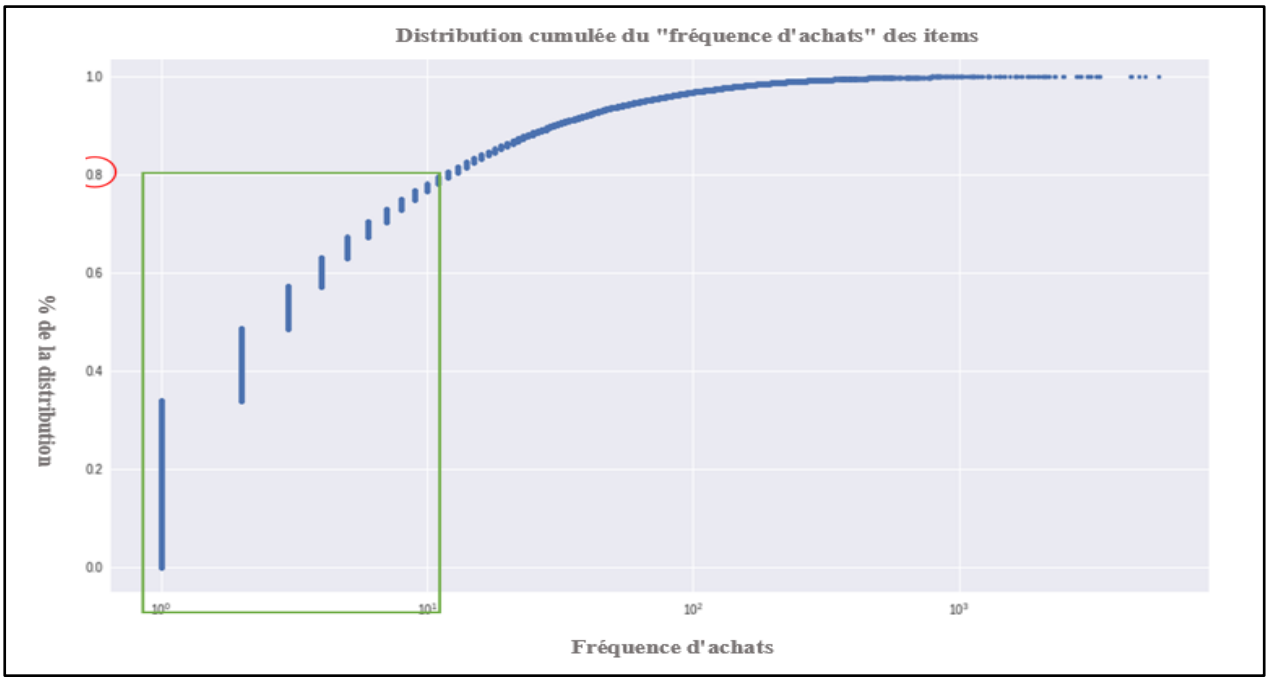

Figure 3.2 La distribution cumulative des achats des items.

Ainsi, nos utilisateurs achètent en moyenne 5.6 items et plus de 75% des items sont achetés par moins de 10 utilisateurs. Cette rareté des informations sur les comportements d'achats des utilisateurs rend difficile l'apprentissage de l'algorithme basé sur la factorisation matricielle, comme l'algorithme *i-ALS* utilisé dans notre approche.

Afin de résoudre le problème de la sparsité dans les achats en ligne, (Kanagal 2012) utilisent une taxonomie d'origine humaine qui associe chaque item à un nœud de l'arbre des catégories. Le modèle du filtrage collaboratif tenant compte de la taxonomie qu'ils proposent suppose que le facteur latent associé à chaque nœud de l'arbre est échantillonné à partir de son nœud parent, généralisant ainsi les données d'achat d'un item individuel aux items appartenant à la même catégorie. Des études expérimentales montrent que cela améliore considérablement les performances pour les données d'achat en ligne. Contrairement au (Kanagal 2012) qui nécessitent une taxonomie existante, nous proposons une nouvelle approche qui découvre automatiquement la taxonomie à partir de données d'achat en ligne et apprend conjointement un système de recommandation basé sur la taxonomie à partir de données brutes. En effet, il peut être trop coûteux de créer une taxonomie manuelle et de rattacher chaque item à une catégorie. Nous proposons

`

d’exploiter le texte de description des items (lorsqu'il est disponible) et les transactions des achats pour générer et rattacher un item à une famille. Ces deux sources sont beaucoup plus faciles à collecter.

Notre motivation de créer une nouvelle taxonomie qui, en basant sur la description, regroupe les items similaires est que le filtrage collaboratif est incapable d'identifier les *synonymes*. Ici, les "*synonymes*" font référence à des items semblables étiquetés ou nommés différemment. Le filtrage collaboratif était incapable de découvrir l'association latente entre les synonymes, il traitera donc ces items différemment. Par exemple, le filtrage collaboratif ne peut pas comprendre que les items apparemment différents "*Backpack*" et "*Knapsack*" font en fait référence au même item. Une approche automatique reposant sur la similarité des textes qui décrivent les items est proposée pour agréger les items qui sont très semblables avec une petite différence de caractéristiques, dans une même famille. Si nous avons deux items qui ont deux Identifiants Unique (IDs) distincts, mais qui sont très semblables au niveau de la description textuelle, ils seront regroupés dans la même famille. Exemple :

- Item 1 : ***ID = 1001*** et appelé "*Boulonne Acier Taille 10 * 12 * 15* ",

- Item 2 : ***ID = 1005*** et appelé "*Boulonne Acier Noir Taille 12 * 12 * 20* ".

- Famille (Item 1, Item 2) = ***Boulonne Acier***

Les deux items sont bien identiques dans leur description (il s'agit de boulonne en acier) mais que dans la base de données, ils sont considérés comme deux items distincts puisqu'ils ont deux identifiants uniques différents. Dans ce cas, un système de recommandation peut dériver et la performance du système sera réduite considérablement.

De point de vue pratique, nous avons procédé au nettoyage des textes (transformation en minuscules, suppression de la ponctuation, suppression des caractères latins, etc.) puis à l’extraction automatique de taxonomies à partir de descriptions des items en utilisant deux modules de traitement nommés Fragmentation du texte (*Text Splitting)* et Regroupement des phrases (*Sentence Clustering)* (voir Figure 3.3). Le premier divise chaque texte d'une description en mots. Le second module identifie et regroupe les phrases similaires. Le résultat est un ensemble de clusters qui regroupe les textes similaires.

`

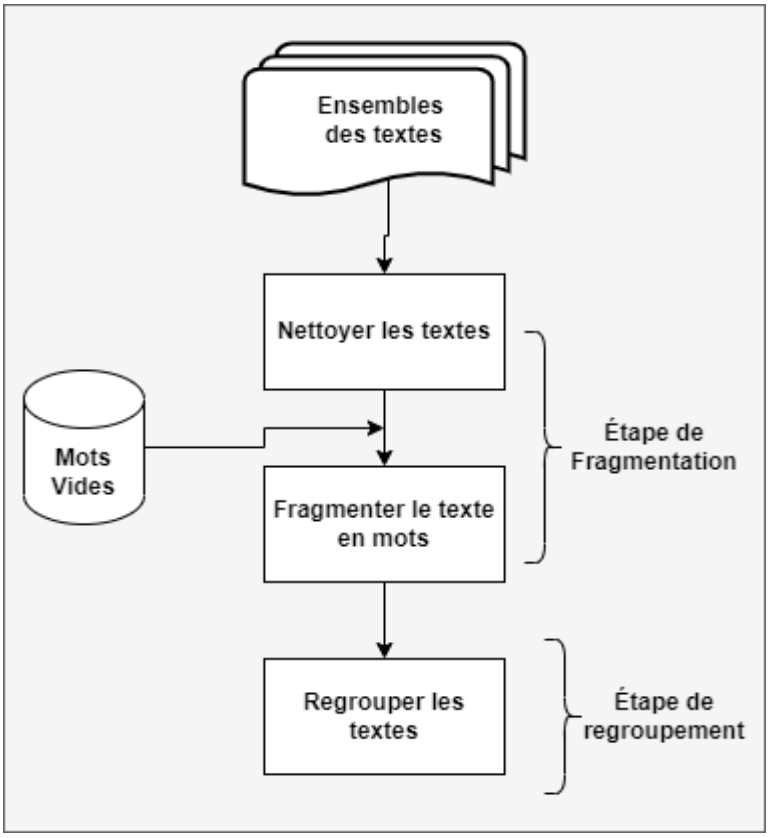

Figure 3.3 Étapes de regroupement les items

La fragmentation du texte a été effectué à l'aide d'un outil de découpage de texte appelé *Tokneizer* de la bibliothèque *NTLK*[8]. La bibliothèque NLTK est l'une des bibliothèques Python les plus anciennes et les plus utilisées pour le traitement du langage naturel. NLTK supporte la suppression des mots vides (*Stop Words*). Ce sont les mots qui sont généralement filtrés avant le traitement d'un langage naturel. Ils représentent en fait les mots les plus courants de toute langue (tels que les items, les prépositions, les pronoms, les conjonctions, etc.) et n'ajoutent pas beaucoup d'informations au texte. Quelques exemples de mots vides en français sont : " le/la", " un/une ", " ainsi ", " ce ". En supprimant ces mots, nous éliminons les informations de bas niveau de texte afin de donner plus d'importance aux informations importantes. Pour supprimer les mots vides d'une phrase, nous

[8] https://www.nltk.org/api/nltk.tokenize.html

`

divisons le texte en mots, puis nous supprimons le mot s'il figure dans la liste des mots vides fournie par *NLTK*.

Le regroupement du texte utilise un algorithme de clustering apprend d'abord la représentation des mots, puis les organise en cluster basé sur la similarité de leur représentation. Le premier cluster est créé en sélectionnant le premier texte de l'ensembles des descriptions des items. À chaque itération, l'algorithme décide si une nouvelle phrase d'entrée doit être insérée dans un cluster existant ou s'il doit en créer un nouveau (D & Kuttiyannan, 2003). Cette décision est basée sur une condition spécifiée par la fonction de similarité employée, c'est-à-dire si la similarité calculée par cette fonction entre deux textes passe un seuil de similarité, nous pouvons le regrouper dans le même groupe. La fonction de la similarité est basée sur la métrique *WO* pour *Word Overlap* (Radev et al., 2008), qui calcule le nombre de mots communs entre une texte D et un cluster C, normalisé par le nombre total de mots dans D et C te qu'illustré par l'équation (3.1).

$$WO(D,C) = \frac{\#MotsCommun(D,C)}{\#Mots(D) + Mots(C)} \tag{3.1}$$

3.3 Renforcement de la matrice Utilisateur-Item

3.3.1 Extraction les règles d'associations

Bien que nous ayons utilisé la taxonomie pour surmonter le problème de sparsité, nous sommes toujours confrontés au problème. En utilisant les règles d'association pour développer le profil d'utilisateur, nous introduisons pour chaque utilisateur d'autres familles complémentaires (de point de vue des règles extraites) à ceux qu'il a montré intérêt. À partir des profils d'utilisateurs, nous construisions un ensemble de données transactionnelles. Ainsi, chaque transaction est constituée des familles d'items qui intéressent un utilisateur. Nous explorons ensuite ces données transactionnelles pour trouver des modèles fréquents et dériver des règles d'association à partir de ces modèles. Cela nous donne des règles d'association entre les familles d'item qui intéressent les utilisateurs. Ces règles nous permettent de découvrir les familles qui apparaissent fréquemment ensemble dans le cadre des intérêts d'un utilisateur. Cet ensemble de règles sera ensuite utilisé pour élargir les profils des utilisateurs afin de résoudre le problème de la densité de données.

`

De nombreux des systèmes de recommandations ont été développés à l'aide de règles d'association. L'extraction de règles d'association vise à extraire des modèles à partir des transactions des utilisateurs. Ces règles sont extraites considérant un support et une confiance élevée. Le support étant le nombre de fois où la règle s'applique par rapport au nombre total de transactions et la confiance (illustrée par l'équation (2.2)) indique la probabilité que l'antécédent et le conséquent d'une règle apparaissent dans la même transaction. La confiance est la probabilité conditionnelle du conséquent étant donné l'antécédent. Par exemple, un produit A peuvent apparaître dans 150 transactions ; 90 de ces 150 transactions peuvent également contenir du produit B. La règle serait la suivante : *"A (antécédent) implique B (Conséquence) avec 60% de confiance"*

Nous avons utilisé, les règles d'association extraites pour améliorer un algorithme de filtrage collaboratif en renforçant la matrice *utilisateur-item*. Pour extraire les règles d'association, nous utilisons l'algorithme *Apriori* (Agrawal & Srikant, 1994) qui est capable d'extraire des règles, en fonction des données d'achat fournies. L'algorithme *Apriori* présenté par Agrawal et al. (1994) est l'algorithme le plus efficace. Cet algorithme fonctionne en deux étapes : dans une première étape, les ensembles fréquents sont déterminés. Il s'agit d'ensembles d'éléments qui ont au moins le support minimum donné (c'est-à-dire qui apparaissent au moins dans un pourcentage donné de toutes les transactions). Dans la deuxième étape, les règles d'association sont générées à partir des ensembles d'éléments fréquents trouvés dans la première étape. En général, la première étape est la plus importante, car elle représente la plus grande partie du temps de traitement (Wang et al., 2009).

Les règles d'association au niveau des items sont extraites sur la base des identifiants des items, en traitant chaque transaction séparément. Elles fournissent des informations sur les items qui sont généralement achetés ensemble tout au long de l'historique d'achat. Avec un niveau de confiance donné, cette méthode peut fournir beaucoup ou moins de règles à partir de l'ensemble de données utilisé. Dans ce projet, nous avons testé l'algorithme *Apriori* avec deux niveaux de confiance : Niveau de confiance bas (égal à 0.35) et élevé (égal à 0.65).

`

3.3.2 Renforcement de la matrice utilisateur-item

L'un des problèmes courants, auxquels les algorithmes de filtrage collaboratif sont confrontés, est la sparsité des données. Ce problème survient lorsqu'il y a un grand nombre d'utilisateurs et des items et que les utilisateurs n'interagissent qu'avec une petite portion des items. Les techniques de factorisation matricielle visent à calculer les vecteurs factoriels des items de l'utilisateur sur la base de la matrice des items de l'utilisateur. Plus la matrice est dense, plus les résultats sont précis. Une mise en œuvre intéressante est décrite dans (Osadchiy et al., 2019) où l'exploration de règles d'association est utilisée afin de rendre la matrice de préférence *utilisateur-item* plus dense en extrayant des règles et en remplissant les valeurs en conséquence. Ainsi, les règles extraites seront utilisées pour compléter les valeurs manquantes de la matrice utilisateur-item.

Une fois que les règles d'association ont été extraites et que les données brutes ont été prétraitées, ce qui fournit la matrice *utilisateur-item*, nous proposons de renforcer la matrice *utilisateur-item* en se basant sur les règles extraites précédemment. Ainsi, sur la base d'une règle extraite, le système remplit la matrice, pour un utilisateur spécifique, avec des items pour lesquels l'utilisateur a acheté l'item de base, mais pas l'item complémentaire. Plus clairement, une règle est de la forme A ➔ B avec un niveau de confiance *nc* : A est l'item de base et B est l'item complémentaire). C'est-à-dire que dans *nc%* des transactions où apparaît l'item A, l'item B apparaît également.

Nous avons attribué à l'utilisateur qui a acheté A et n'a jamais acheté B, l'item B avec un score spécifique (le score signifie la fréquence d'achat). Le score attribué au couple *item-utilisateur* non attribué préalablement doit prendre en compte le score de l'item de base. Ainsi, nous pouvons modéliser les cas où l'utilisateur a acheté l'item de base plus d'une fois. En outre, comme la confiance fournit des informations sur la corrélation entre les items de base et les items complémentaires, il faut pareillement en tenir compte dans le score (T. Osadchiy et al, 2018).

Tel que présenté dans l'équation (3.6), le score de l'item ajouté, inspire par le travail de Loannis Schoinas et Christos Tjortjis (2017), est égal au score de l'item de base multiplié par la confiance de la règle *nc*. En notant l'utilisateur avec u, les items avec i, i_a sont les items complémentaires, i_b sont les items de base et *nr* est un score nouvellement attribué qui était auparavant absent pour le couple spécifique *utilisateur-item*.

`

$$nr_{ui_a} = r_{ui_b} * nc \tag{3.6}$$

Après avoir renforcé la matrice des interactions utilisateur-item, le système procède à la découverte les facteurs latents de l'item et utilisateur de la matrice renforcée et des autres étapes de l'approche proposée.

3.4 Initialisation et Application de l'algorithme i-ALS

3.4.1 Initialisation de l'algorithme i-ALS

Dans la dernière phase de notre approche, nous visons à augmenter encore les performances du système en initialisant les vecteurs factoriels qui caractérise l'utilisateur et l'item avant les introduire dans l'algorithme *i-ALS*. Comme décrit précédemment, les méthodes de factorisation matricielle visent à découvrir les vecteurs de facteurs latents de l'utilisateur et de l'item et à utiliser leur produit scalaire pour produire des recommandations. La plupart de ces méthodes sont des algorithmes itératifs qui partent d'un point aléatoire : les vecteurs factoriels des items et des utilisateurs sont initialisés de manière aléatoire. Après quelques itérations, ces méthodes convergent vers un optimum local qui dépend du point de départ.

Les techniques d'initialisation traditionnelles sont basées sur des méthodes purement aléatoires, où les matrices sont initialisées avec des nombres aléatoires uniformément distribués, par exemple entre 0 et 1. Bien que ce type d'initialisation soit très simple, il ne fournit pas toujours une bonne solution. Notre hypothèse est qu'une initialisation appropriée de ces vecteurs permet aux méthodes de factorisation matricielle de produire des vecteurs plus précis et donc de donner des prédictions plus précises. La méthodologie proposée dans (Koren, 2008) utilise des valeurs aléatoires pour l'initialisation de l'algorithme *i-ALS*. L'importance d'une bonne initialisation a été reconnue pour certains algorithmes de factorisation matricielle comme la factorisation matricielle non négative (NMF). Il a été démontré dans (Smilde, 2004) qu'une bonne initialisation peut améliorer la vitesse et la précision des algorithmes, car elle peut produire une convergence plus rapide.

Dans la solution proposée, à l'aide de l'algorithme de décomposition en valeurs singulières ***BLS-SVD++*** (Wang et al., 2020), notre système décompose la matrice *utilisateur-item* et en extrait les

`

vecteurs des facteurs latents utilisateur et item qui sont ensuite utilisés comme initiaux pour l'algorithme *i-ALS* au lieu de partir avec des initiaux aléatoires.

3.4.2 Application de l'algorithme i-ALS

Inspiré de Koren et al. (2008]), nous proposons d'utiliser l'algorithme *i-ALS* qui prend en compte le feedback implicite au lieu des évaluations communément utilisées (*Rating*). Pour la mise en œuvre de l'algorithme, les données brutes doivent être transformées en une matrice *utilisateur-item* avec des valeurs représentant le feedback implicite associé au couple *utilisateur-item*. Comme le jeu de données fournit des observations sur le comportement d'achats de différents utilisateurs, la première étape de l'algorithme consiste à prétraiter les données brutes pour les transformer en une matrice *utilisateur-item*.

Dans Koren (2008), le jeu de données utilisé concerne les transactions des visiteurs et la matrice *utilisateur-item* requise pour la factorisation matricielle est construite en agrégeant le nombre de fois qu'un client a vu un spectacle spécifique. L'intérêt d'un utilisateur pour un spectacle est déduit du nombre de fois qu'il l'a vu. Dans notre cas, nous déduisons l'intérêt de l'utilisateur pour un item à partir de son comportement historique d'achat. La différence est qu'au lieu de calculer le nombre de fois qu'un utilisateur a acheté un item spécifique, nous en considérons une agrégation au niveau des items pour créer une taxonomie de famille d'items. Ainsi, le score d'interaction est construit en agrégeant le nombre de fois qu'un utilisateur a acheté des items de famille spécifique et est donné par l'équation (3.2)

$$r_{uf} = \sum_{i \in I(f)} n_{ui} \tag{3.2}$$

Où u : désigne un utilisateur donné, f : désigne une famille d'item donné, $I(f)$: l'ensemble des des items d'une famille d'item f, n_{ui} : désigne le nombre de fois que l'utilisateur u a acheté un item i.

Dans l'algorithme proposé, la préférence est considérée comme une variable binaire qui prend la valeur 1 pour les cas où un utilisateur a interagi (acheté 5 fois) avec un item et 0 sinon. Avec r_{uf}, nous notons le score d'interaction d'un utilisateur u avec famille f.

$$p_{uf} = \begin{cases} 1 & si\ r_{uf} > 5 \\ 0 & si\ r_{uf} = 0 \end{cases} \tag{3.3}$$

`

La deuxième étape de l'algorithme, après la création de la matrice *utilisateur-item*, consiste à calculer les matrices de facteurs latents. Pour cela, nous procédons au calcul des matrices en minimisant la fonction de coût proposée par Koren et al. (2008) présentée par l'équation (3.4).

$$\min_{x,y} \sum_{u,f} c_{uf}\left(p_{uf} - x_u^T y_f\right)^2 + \lambda\left(\sum_u |x_u|^2 + \sum_f |y_f|^2\right) \tag{3.4}$$

On désigne par x la matrice des facteurs latents utilisateur et par y la matrice des facteurs latents famille. λ est le régulateur pour réduire le surajustement, c_{uf} représente la confiance de chaque paire *utilisateur-famille*. Elle est donnée par : $c_{uf} = 1 + \alpha * p_{uf}$ où α est un régulateur. Ici, la confiance est calculée en utilisant les données de feedback, ce qui nous donne une confiance plus élevée au fur et à mesure qu'un utilisateur a acheté une famille. Le taux d'augmentation de la confiance est fixé par le régulateur α. Nous ajoutons également 1 pour avoir une confiance minimale même si $\alpha * p$ est nul. Dans Koren et al. (2008), ils ont trouvé que $\boldsymbol{\alpha = 40}$ fonctionnait bien.

Après avoir déterminé les matrices de facteurs latents, nous pouvons obtenir une prédiction pour la paire *utilisateur-famille* en calculant le produit scalaire de leurs vecteurs de facteurs latents selon l'équation (3.5).

$$rec_{uf} = x_u^T . y_f \tag{3.5}$$

Où rec_{uf} est le score de préférence prédit pour un utilisateur spécifique envers une famille spécifique. La dernière étape est le classement de toutes les familles pour un utilisateur et la génération d'une liste ordonnée par préférence, comme recommandations.

3.5 Conclusion

Dans ce chapitre, nous avons détaillé l'approche utilisée, qui est basée principalement sur l'algorithme de factorisation matricielle *i-ALS*, l'utilisation de la taxonomie, l'extraction des règles d'association et l'initialisation de l'algorithme *i-ALS*. Nous avons également présenté un cadre de création d'une nouvelle taxonomie automatiquement en se basant sur les textes de description des items.

`

Dans le chapitre suivant, nous présentons la réalisation des expériences et discutons le plan d'expérience.

`

CHAPITRE 4 EXPÉRIMENTATION

Ce chapitre fournit des détails sur l'application des algorithmes sélectionnés pour l'analyse des données du partenaire industriel. Les techniques de nettoyage et de prétraitement des données sont décrites. Les critères d'évaluation des algorithmes choisis sont également présentés. Nous avons aussi discuté des spécifications du modèle et de la sélection des hyperparamètres pour les algorithmes sélectionnés. La section 4.1 décrit l'application de l'approche proposée au cas d'études. La section 4.2 présente les approches conventionnelles de la littérature. La section 4.3 décrit les spécifications du modèle et les hyperparamètres pour l'algorithme *i-ALS*. Dans la dernière section (4.4), nous résumons le plan de l'expérience.

4.1 Application de l'approche proposée au cas d'études

Les détails spécifiques de l'ensemble de données ne peuvent pas être résumés directement dans ce mémoire dans la mesure où les données sont confidentielles. Néanmoins, les détails généraux de l'analyse exploratoire des données sont fournis en ce qui concerne la façon dont l'ensemble de données a été préparé. Comme défini dans le chapitre précédent, l'ensemble de données fourni par la société se compose de treize années de transactions de vente, de janvier 2007 à octobre 2020, pour tous les utilisateurs. L'ensemble de données final contient les informations suivantes : l'ID du client, l'ID du produit, la description des items et le nombre de fois où un client a acheté un produit (évaluations).

- *CLIENT_CODE* : L'Identifiant Unique du client
- *ITEM_ID* : L'Identifiant Unique d'item
- *ITEM_DESC* : Description ou nom d'item
- *ITEM_SOUS_CATEGORY* : La sous-catégorie à laquelle appartient un item
- *COUNT* : Le nombre de fois où un client a acheté un item

Dans la première étape de l'approche, les items similaires sont identifiés et regroupés dans une nouvelle taxonomie (famille d'item). Nous avons deux niveaux de taxonomies d'items dans le jeu de données (Catégorie et Sous-catégorie). Un diagnostic de notre jeu de données montre que nous avons un problème de descriptions d'items mal saisies. Ainsi, nous avons constaté qu'un identifiant

`

d'item donné a, en moyenne, plus de deux descriptions différentes. L'exemple dans la Figure 4.1 montre qu'il y a quatre identifiants d'items différents qui ont quasiment les mêmes descriptions, mais avec de petites variations comme la taille ou la couleur.

	ITEM_ID	DESC_ITEM_y
0	CCB12T9400524	CROCHET CLEVIS
1	CCB13T9400624	CROCHET CLEVIS OUVERT
2	CCB21T9500424	CROCHET CLEVIS FERME
3	CCB22T9500524	CROCHET CLEVIS FERME
4	CCB57	CROCHET CLEVIS FERME JAUNE
5	CCB58T9503415	CROCHET CLEVIS FERME JAUNE
6	CCB59T9503515	CROCHET CLEVIS FERME JAUNE
7	CCB60	CROCHET CLEVIS FERME
8	CCB61	CROCHET CLEVIS OUVERT JAUNE

Figure 4.1 Diagnostique de la colonne Description-ITEM

Pour résoudre ce problème, en se basant sur le cadre présenté dans le chapitre précèdent, nous avons tout d'abord agrégé les descriptions d'items semblables dans le même colonne *ItemName*. Par exemple, comme indiqué dans la Figure 4.2, les *ITEM_IDs*, nous les avons mis dans le même *ItemName* appelé *CLEVIS HOOK*. De même, ces *ItemName* peuvent être regroupés dans une nouvelle taxonomie nommée Famille (ou ***ITEM_FAMILY***) en se basant sur notre

	ITEM_ID	DESC_ITEM_y	ItemName
0	CCB12T9400524	CROCHET CLEVIS	CROCHET CLEVIS GR
1	CCB13T9400624	CROCHET CLEVIS OUVERT	CROCHET CLEVIS GR
2	CCB21T9500424	CROCHET CLEVIS FERME	CROCHET CLEVIS GR
3	CCB22T9500524	CROCHET CLEVIS FERME	CROCHET CLEVIS GR
4	CCB57	CROCHET CLEVIS FERME JAUNE	CROCHET CLEVIS GR
5	CCB58T9503415	CROCHET CLEVIS FERME JAUNE	CROCHET CLEVIS GR
6	CCB59T9503515	CROCHET CLEVIS FERME JAUNE	CROCHET CLEVIS GR

Figure 4.2 Création de colonne ITEM_NAME

méthode décrite dans la section (3.2.2)**.** Ainsi, à la fin, nous avons les données telles que présentées dans la Figure 4.3.

`

	ITEM_ID	ITEM_DESC	ITEM_NAME	ITEM_FAMILY
0	CCB12T9400524	CROCHET CLEVIS	CROCHET CLEVIS GR	Manutention/Grappins/CROCHET
1	CCB13T9400624	CROCHET CLEVIS OUVERT	CROCHET CLEVIS GR	Manutention/Grappins/CROCHET
2	CCB21T9500424	CROCHET CLEVIS FERME	CROCHET CLEVIS GR	Manutention/Grappins/CROCHET
3	CCB22T9500524	CROCHET CLEVIS FERME	CROCHET CLEVIS GR	Manutention/Grappins/CROCHET
4	CCB23T9500624	CROCHET CLEVIS FERME	CROCHET CLEVIS GR	Manutention/Grappins/CROCHET
5	CCB25T9500824	CROCHET CLEVIS FERME	CROCHET CLEVIS GR	Manutention/Grappins/CROCHET
6	CCB57	CROCHET CLEVIS FERME JAUNE	CROCHET CLEVIS GR	Manutention/Grappins/CROCHET

Figure 4.3 Résultats d'agrégation : création de la colonne ITEM_FAMILY

Enfin, la taxonomie des items que nous avons créée est donnée par l'arbre de Figure 4.4. Ainsi, nous avons testé trois jeux de données, selon notre hiérarchie d'items créés, pour tester notre approche : Le jeu de données ITEM_FAMILY dont le profil des utilisateurs est calculé en fonction de leur intérêt pour un ITEM_FAMILY donné, le jeu de données ITEM_SOUS_CATEGORY dont le profil des utilisateurs est calculé en fonction de leur intérêt pour une sous-catégorie donnée. La taxonomie existante dans l'ensemble de données est utilisée pour organiser les items en catégories et sous-catégories. Par exemple, une taxonomie pour une catégorie *Outils de Coupe* a des sous-catégories pour *Couteaux, Coupe-Tuyaux*, etc. Finalement, le jeu de données ITEM_ID dont le profil des utilisateurs est calculé en fonction de leur intérêt pour un ITEM_ID spécifique.

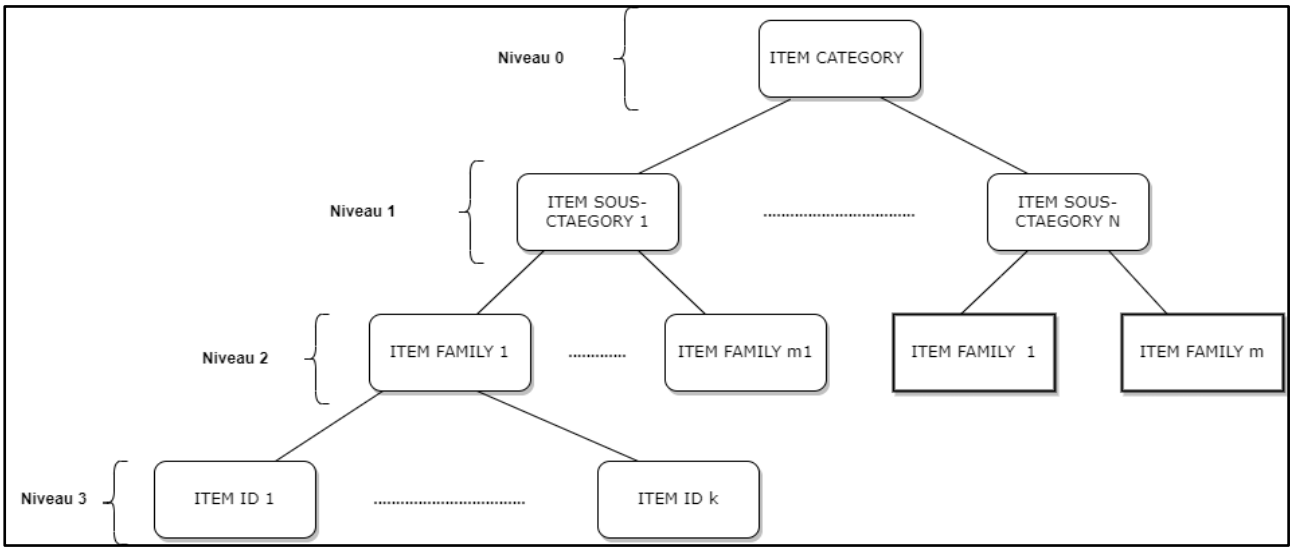

Figure 4.4 Taxonomie des items proposée

`

Finalement, comme un résultat de cette phase, nous obtenons une arborescence des items. Par exemple, les deux items "Tire Fond 5-16-x-2-304" et "Tire Fond 15-32-x-4-102" appartiennent à la famille d'item "TIRE FOND", qui fait partie de la sous-catégorie "TETE HEX INOX " de catégorie "BOULONNERIE ".

La matrice d'interaction comprend des utilisateurs en lignes et des items en colonnes, les valeurs binaires dans les cellules représentant une interaction. Le niveau de sparsité d'un ensemble de données représente le nombre total de **1** que l'on peut trouver dans cette matrice d'interaction. La sparsité est calculée par la formule (4.1).

$$sparsité\ (\%)\ = 1\ - \frac{nombre\ d'\ interactions}{n_{items} * n_{users}} \tag{4.1}$$

Où n_{items} *et* n_{users}, respectivement, sont le nombre des items et le nombre des utilisateurs dans l'ensemble des données. Il s'agit simplement du nombre total d'interactions uniques entre les utilisateurs et les items, divisé par le nombre total d'interactions possibles. Par exemple, l'ensemble de données utilisé, l'ITEM_ID a une densité (définie comme ***1 - sparsité***) de ***1.30%***, ce qui signifie qu'il y a **98,70%** de zéros dans la matrice d'interaction. Les statistiques de nos ensembles de données sont présentées dans le tableau ci-dessous.

Tableau 4.1 Statistiques des jeux de données

Datasets	***ITEM_ID***	***ITEM_FAMILY***	***ITEM_SOUS_CATG***
#utilisateurs	*4 498*	*4 498*	*4 498*
#items	*12 000*	*3 115*	*315*
#sparsité	***98.70%***	***99.14%***	***91.18%***

`

L'utilisation de la taxonomie permet de réduire la densité par : la matrice que nous utilisons ITEM_FAMILY agrège les préférences de l'utilisateur envers les items de la même famille. Par exemple, si un utilisateur achète un item I1 de la famille F10 et n'achète pas l'item I101 de la même famille, alors dans cette matrice nous considérons qu'il s'intéresse à la famille F10 même si un utilisateur n'a pas d'interaction avec l'item I101. Ainsi, les interactions avec les familles sont nombreuses par rapport aux items et par conséquent la densité de la matrice est plus élevée.

Dans la deuxième étape, nous avons utilisé la fonction python *Apriori* de la bibliothèque *mlxtend* pour appliquer l'algorithme d'extraction *Apriori* sur le jeu de données de transactions d'achat afin d'obtenir les ensembles de règles les plus fréquentes. Les tâches de cette étape sont effectuées dans l'ordre suivant : (1) Importer le jeu de données. (2) Appliquer l'algorithme d'*Apriori* en utilisant la fonction *apriori()*. Cette fonction fournit l'infrastructure pour représenter, manipuler et analyser les données pour l'extraction de caractéristiques fréquentes et l'application de règles d'association. Dans cette étape, l'ensemble des règles sont extraites avec les critères suivants :

- Longueur maximale de la règle est égale à 3. Ce paramètre détermine le nombre maximal des items qui apparaissent dans une règle d'association. Les règles générées comprennent au maximum deux items dans le corps de la règle et un item dans la tête de la règle. La règle générée sera présentée comme suit : $\{A, B\} \rightarrow \{C\}$.

- Niveau de confiance de la règle : 0,35 minimum et 0,65 maximum.

Comme défini précédemment, le niveau de confiance représente une mesure de la fiabilité de la règle sur une échelle de 0 à 1. Étant donné que le nombre de règles d'association extraites dépend du niveau de confiance, ce paramètre a un réel impact sur les résultats de notre modèle de recommandation.

À la fin de l'étape précédente, pour chaque niveau de confiance (*NC_Low* et *NC_High*), nous avons sélectionné les meilleures règles à utiliser dans la suite de l'expérience. Ces règles nous fournissent des informations sur les items qui ont tendance à être achetés ensemble. Dans cette étape, le système renforce la matrice telle décrite dans le chapitre 3.

`

4.2 Application d’autres approches conventionnelles

Pour évaluer la performance de l’approche proposée dans le chapitre 3, nous la comparons à d’autres approches connues en systèmes de recommandation.

1. **Aléatoire :**

   Tout simplement, une liste aléatoire de N items est recommandée à l'utilisateur.

2. **Le plus populaire**

   C’est un algorithme plus compétitif que l'algorithme aléatoire. Il renvoie tout simplement à l'utilisateur les N items les plus populaires. Il s’agit d’un point de référence couramment utilisé pour la tâche de recommandation des items basée sur le feedback implicite.

3. **Bayesian Personalized Ranking (Rendle & Schmidt-Thieme, 2008)**

   Nous parlons d'un algorithme de recommandation spécialement conçu pour le feedback implicite. Cette technique optimise le modèle de factorisation matricielle avec une technique bayésienne de perte de classement par paires (*pairwise ranking loss*). Il utilise la descente de gradient stochastique pour apprendre les préférences des utilisateurs à partir des données de feedback implicite. Nous avons évalué l’algorithme avec une liste de facteurs latents [8 ; 16 ; 32 ; 64] avec un taux d'apprentissage fixe, et nous rapportons la meilleure performance.

4. **NeuMF** (He et al., 2017) : Cette méthode est un modèle de factorisation matricielle avec une architecture de réseau neuronal. Nous utilisons l'implémentation originale qui est déjà destinée au problème de recommandation avec les feedbacks implicite.

4.3 Paramètres de l'expérience

4.3.1 Division des données

Dans les systèmes de recommandation, la division des données en ensembles d'apprentissage et de test requiert une attention particulière. Par exemple, dans le cas d'un système de recommandation qui recommande des items à des utilisateurs sur la base des similarités entre les évaluations

`

collaboratives des utilisateurs, il convient d'utiliser le même ensemble des utilisateurs pour la phase d'apprentissage et de test du modèle afin de disposer de mesures d'évaluation statistiquement fiables. La stratégie de fractionnement affecte de manière significative les recommandations d'items et doit toujours être prise en compte par les praticiens (Meng et al., 2020).

Nous proposons de diviser l'ensemble de données en un ensemble d'apprentissage et un ensemble de test à l'aide d'une fonction personnalisée appelée *python_stratified_split*, où le fractionnement est stratifié de manière que le même groupe d'utilisateurs et d'items apparaisse dans les ensembles d'apprentissage et de test. La fonction personnalisée comporte trois paramètres essentiels : "*filter_by*", "*min_rating*", et "ratio". Le paramètre "*filter_by*" prend deux entrées, soit "*utilisateur*", soit "*item*", en fonction de l'entrée sur laquelle nous voulons fractionner l'ensemble de données. Le paramètre "*min_rating*" prend un nombre entier en entrée, où il filtre l'ensemble de données sur la base du nombre minimum d'évaluations pour l'utilisateur ou le produit. Le "*ratio*" n'est rien d'autre que la fraction apprentissage/test. Par exemple, si le paramètre "*filter_by*" est "*utilisateur*" et que le "ratio" est de 0,7, cela signifie que les premiers 70% des évaluations pour chaque utilisateur dans les données seront placées dans l'ensemble d'apprentissage tandis que les autres 30% seront dans l'ensemble de test. Après avoir discuté avec les représentants du partenaire industriel, en fonction de leurs objectifs commerciaux, nous avons choisi "*min_rating*" = 5, " *filter_by* " = "*user*" et " *ratio* " = 0,8, c'est-à-dire que l'ensemble de données est fractionné en 80 % d'ensemble d'apprentissage et 20 % d'ensemble de test, et est filtré par les utilisateurs qui ont acheté un item au moins 5 fois.

4.3.2 Évaluation de la performance des recommandations

Pour évaluer les modèles de recommandation, nous avons recours à la technique de Validation Croisée (K-fold Cross Validation), soit utiliser plusieurs ensembles d'apprentissage attribués de manière aléatoire. Habituellement, les systèmes de recommandation sont évalués soit en ligne, soit hors ligne. L'évaluation en ligne mesure essentiellement les taux d'acceptation des recommandations dans les systèmes de recommandation implémentés sur un site e-commerce (Shani & Gunawardana, 2011). Les taux d'acceptation sont généralement mesurés par le taux de clics (CTR), autrement dit le rapport entre les recommandations cliquées et les recommandations affichées. Par exemple, si un système de recommandation affiche 1 000 recommandations et que 12 sont cliquées, le CTR est de 1,2%. D'autres mesures incluent le ratio d'items achetés. Le taux

`

d'acceptation est habituellement interprété comme une mesure implicite de la satisfaction de l'utilisateur. L'évaluation hors ligne, quant à elle, mesure habituellement la précision d'un système de recommandation sur une base de données de référence. Elle est aussi utilisée pour évaluer des aspects tels que la nouveauté ou la sérendipité des recommandations (Ge et al., 2010). Pour un aperçu complet des évaluations hors ligne, y compris les paramètres d'évaluation et les problèmes potentiels, est présenté par (Karatzoglou et al., 2013) et (Konstan & Riedl, 2012).

Bien que l'évaluation hors ligne soit plus limitée, car elle ne peut mesurer que le pouvoir prédictif d'un algorithme de recommandation (Shani & Gunawardana, 2011), elle était la seule option disponible pour ce projet, parce que nous n'avions accès qu'à des données hors ligne. L'efficacité de la liste de recommandations est mesurée par deux mesures communes largement utilisées dans le problème de recommandation, à savoir le gain cumulatif actualisé normalisé (Yilmaz et al., 2008) et la précision à N *(P@N)* (Yue et al., 2007). Selon Shani et al. (2011), la métrique *P@N* est mieux adaptée aux situations où seul un nombre limité d'items peut être recommandé à un utilisateur. C'est généralement le cas pour les détaillants en ligne, car un site Web ne dispose que d'un espace suffisant pour répertorier un nombre limité d'options (par exemple, seuls cinq items peuvent être affichés). Cette métrique se concentre sur les items corrects, les plus susceptibles d'être achetés par un utilisateur dans le top *N* des items recommandés. Après avoir discuté avec représentants du partenaire industriel, nous avons opté pour la métrique *P@N* avec valeur de 3 pour *N*. Dans ce cas, une liste de trois items seront recommandés pour chaque utilisateur.

Un système de recommandation recommande à l'utilisateur courant *u* une liste *RL(u)* contenant *N* items qui pourraient potentiellement être intéressants pour lui. La mesure de la précision est alors la capacité du système à décider si un item est pertinent ou non pour l'utilisateur *u*. Telle que définie par l'équation (4.2).

$$P = \frac{|\mathrm{RL} \cap \mathrm{SL}|}{|\mathrm{RL}|} \tag{4.2}$$

Où *RL* est l'ensemble des items recommandés à un utilisateur *u* et *SL* est l'ensemble des items sélectionnés par cet utilisateur. Elle mesure combien d'items recommandés ont été effectivement sélectionnés par l'utilisateur. Elle reflète donc l'exactitude globale de toutes les recommandations

`

et ne tient pas compte de leur ordre. Comme nous voulons connaître la précision jusqu'au *k-ème* item recommandé, nous avons utilisé la précision à la position *k*, qui est définie comme suit :

$$\mathrm{P@k} = \frac{1}{|\mathrm{U}|} \times \sum_{\mathrm{u} \in \mathrm{U}} \frac{SL(u)@k}{k} \tag{4.3}$$

Où *SL(u)@k* est l'ensemble des items sélectionnés par l'utilisateur *u* jusqu'au *k-ème* item recommandé. Nous pouvons voir dans l'équation (4.3) que plus la valeur de *P@k* est élevée, plus les résultats de la recommandation sont précis jusqu'au *k-ème* item recommandé.

4.3.3 Spécifications du modèle

Tel qu'indiqué dans le chapitre 3, nous utiliserons la technique de factorisation matricielle comme le modèle principal de l'approche proposée dans ce projet. Ce modèle nécessite des spécifications avant de faire l'entrainer. Les spécifications du modèle comprennent les procédures d'entraînement du modèle et l'optimisation des hyperparamètres (Abadi et al., 2016). Les modèles sont développés à l'aide des bibliothèques d'apprentissage automatique tel que *LightFM* (Kula, 2015) du langage de programmation *Python,* puis entraînés sur un serveur doté de 6 cœurs de CPU et de 125 Go de RAM. Comme mentionné dans la section 4.3.1, l'ensemble de données a été divisé en ensembles d'entraînement et de test. Nous avons utilisé l'ensemble de données d'entraînement pour entraîner le modèle et l'ensemble de test pour tester les recommandations ou les prédictions faites par le modèle.

L'apprentissage automatique comporte également un aspect essentiel appelé optimisation des hyperparamètres, qui consiste à choisir un ensemble de paramètres optimaux pour le modèle. Il existe trois façons standard de procéder à l'optimisation des hyperparamètres : la recherche par grille , la recherche aléatoire et l'optimisation bayésienne (Liashchynskyi & Liashchynskyi, 2019). La recherche par grille fonctionne en recherchant de manière exhaustive un sous-ensemble spécifié d'hyperparamètres. La recherche aléatoire consiste à rechercher un sous-ensemble spécifié d'hyperparamètres de manière aléatoire au lieu d'utiliser la force brute. L'approche bayésienne utilise les choix passés effectués pour faire un choix intelligent des hyperparamètres pour le prochain ensemble de valeurs à évaluer, grâce auquel elle réduit le coût de la recherche de paramètres. Dans ce projet, nous avons utilisé la recherche par grille pour l'optimisation des

`

hyperparamètres car il s'agit d'une méthode systématique de recherche d'hyperparamètres dans l'espace de recherche, qui crée toutes les combinaisons possibles, quels que soient les effets des éléments du processus d'optimisation. Tous les paramètres ont la même probabilité d'avoir un impact sur ce processus (Liashchynskyi & Liashchynskyi, 2019).

Pour l'algorithme *i-ALS*, nous inspirons de la mise en œuvre de Koren et al. (2008). Nous utilisons l'ensemble d'entrainement pour construire le modèle et nous construisons les évaluations pour chaque item en utilisant les achats faites par un utilisateur. Ces évaluations sont estimées le nombre de fois que les items ont été achetés par l'utilisateur.

Pour implémenter la factorisation matricielle avec l'algorithme *i-ALS* (Koren, 2008), deux hyperparamètres α et λ doivent être réglés. α est un hyperparamètre d'ajustement, si α augmente, le modèle donne plus de poids aux valeurs non nulles de la matrice originale qu'aux valeurs nulles. L'hyperparamètre λ régularise le modèle, de sorte que le modèle ne se sur-ajuste pas pendant la phase d'entraînement. Inspiré de Koren, et al. (2008), nous avons utilisé une valeur de 40 pour l'hyperparamètre α et à une valeur de 0.0001 pour l'hyperparamètre λ. Le nombre de facteurs latents (*k*) est un autre hyperparamètre important dans les méthodes basées sur la factorisation matricielle. En effet, cette dernière est une méthode algébrique linéaire qui décompose une matrice par sa factorisation en un produit de matrices de facteurs latents de dimension *k*. Un nombre plus élevé de facteurs latents peut entraîner une augmentation des performances du modèle, mais rend l'algorithme de recommandation plus lent (Koren, 2008). Pour hyperparamètre *k*, nous avons tester un ensemble des valeurs entre 15 et 60 et nous sommes arrivées à nombre de facteurs latents optimal $k=50$.

Les valeurs d'hyperparamètres considérées dans ce projet sont résumées dans le tableau 4.2.

`

Tableau 4.2 Hyperparamètres pour l'algorithme *i-ALS*

Hyperparamètre	**Définition**	**Valeurs**
Nombre de facteurs latents (k)	Hyperparamètre qui contrôle la dimension de l'espace latent	50
Régularisation (λ)	Hyperparamètre qui Contrôle la régularisation dans la fonction objective	0.0001
Taux d'apprentissage (par défaut)	Hyperparamètre qui détermine la taille du pas à chaque itération	0.01
Itérations (par défaut)	Nombre d'itérations que l'algorithme fait sur l'ensemble de données d'apprentissage	80
Facteur d'ajustement (α)	Hyperparamètre d'ajustement	40

4.4 Plan de l'expérience

Trois séries d'expériences seront à réaliser pour évaluer et comparer différentes configurations. La première série vise à évaluer si l'algorithme *i-ALS* donne de meilleurs résultats en considérant des différents niveaux de taxonomies (Niveau 1: Sous-Catégorie noté par ITEM_SS_CATG, Niveau 2: Famille d'item noté par ITEM_FAMILY) ou en ne prenant en compte que les Identifiant Unique (IDs) d'item (Niveau 3: ITEM_ID). Pour cela, trois matrices *utilisateur-item* sont construites :

- La matrice notée par R_u^i (*utilisateur/ITEM_ID)* est construite à partir de la base de données originale en considérant pour chaque utilisateur u, la fréquence d'achat d'item d'identifiant ITEM_ID. La matrice est présentée comme suite :

`

$$R_u^i = \begin{bmatrix} r_{u_1}^{i_1} & \cdots & r_{u_1}^{i_n} \\ \vdots & \ddots & \vdots \\ r_{u_n}^{i_1} & \cdots & r_{u_n}^{i_n} \end{bmatrix}$$

Où r_u^i est définie par la fréquence d'achat d'utilisateur *u* d'item *i*.

- La matrice notée par R_u^{ss} (*utilisateur/ITEM_SS_CATG)* est construite à partir de la base de données originale en considérant pour chaque utilisateur u, la fréquence d'achat des items d'une sous-catégorie ITEM_SS_CATG. La matrice est présentée comme suite :

$$R_u^{ss} = \begin{bmatrix} r_{u_1}^{ss_1} & \cdots & r_{u_1}^{ss_n} \\ \vdots & \ddots & \vdots \\ r_{u_n}^{ss_1} & \cdots & r_{u_n}^{ss_n} \end{bmatrix}$$

Où r_u^{ss} est définie par la fréquence d'achat d'utilisateur *u* des items d'une sous-catégorie *ss*.

- La matrice notée par R_u^f (*utilisateur/ITEM_FAMILY)* est construite à partir de la base de données originale en considérant pour chaque utilisateur u, la fréquence d'achat des items d'une famille ITEM_FAMILY. La matrice est présentée comme suite :

$$R_u^f = \begin{bmatrix} r_{u_1}^{f_1} & \cdots & r_{u_1}^{f_n} \\ \vdots & \ddots & \vdots \\ r_{u_n}^{f_1} & \cdots & r_{u_n}^{f_n} \end{bmatrix}$$

Où r_u^f est définie par la fréquence d'achat d'utilisateur *u* des items (items) d'une famille d'item *f*.

Pour la suite de l'expérience, les matrices R_u^i, R_u^{ss} et R_u^f seront nommées, respectivement, *ALS_ITEM_ID, ALS_ITEM_SSCATG* et *ALS_ITEM_FAMILY*. Avec chacune de ces matrices, nous évaluons la performance de l'algorithme *i-ALS*. À la fin de cette première série, nous sélectionnons parmi ces trois matrices celle avec laquelle l'algorithme présente la meilleure performance.

La deuxième série d'expérience sera dédiée à explorer si la performance de l'algorithme *i-ALS* avec la matrice sélectionnée dans la série précédente pourrait être améliorée. Trois approches différentes seront testées : l'approche améliorée par l'utilisation de règles d'association, l'approche

`

par l'initialisation l'algorithme *i-ALS* et l'approche combinée (l'amélioration par les règles d'association et l'initialisation l'algorithme *i-ALS*).

Pour la deuxième approche (celle en utilisant juste les règles d'association), les règles extraites ont été utilisées tel que décrit dans la section 3.2. Au total, 54 règles et 26 règles ont été extraites en utilisant un niveau de confiance de 0,35 et 0,65 respectivement. Les règles extraites auront comme rôle de renforcer la matrice *Utilisateur-Item* afin de surmonter le problème de sparsité de données. L'idée est d'ajouter à partir les règles extraites des items complémentaires aux utilisateurs qui ont acheté les items de base. Cette idée sera applicable aux utilisateurs qui ont acheté les items de base de la règle et ils n'ont acheté pas les items complémentaires de la même règle. Cette mise en application de changements a conduit respectivement à ajouter 1206 et 414 changements (lignes) à la matrice.

Pour initialiser les matrices de facteurs latents (voir section 3.3), l'algorithme ***BLS-SVD*++** est utiliser pour décomposer la matrice initiale *Utilisateur-Item* sans aucun remplissage préalable des valeurs manquantes afin de rechercher des matrices facteurs latents pour l'utilisateurs et l'item. Ces matrices seront utilisées pour la suite comme initiales de l'algorithme *i-ALS* au lieu d'utiliser des initiales aléatoires.

Lors de la création des matrices, les utilisateurs qui n'ont pas acheté plus de 5 items ont été considérés comme du bruit donc retirés de l'ensemble d'apprentissage, car les informations fournies sont trop faibles pour produire des recommandations significatives. En outre, cela réduit la taille totale de l'ensemble d'apprentissage évalué et réduit sa sparsité.

Pour finir, les meilleures configurations retenues seront comparées aux approches de la littérature définies dans la section 4.2, soit l'approche aléatoire, l'approche les plus populaires, l'approche bayésienne *BPR* et l'approche basée sur les réseaux de neurones *NeuMF*. Cela sera la troisième et la dernière série des expériences.

Les expériences que nous avons réalisées sont résumées dans Tableau 4.3.

`

Tableau 4.3 Plan d'expérience

Phases	**Objectif**	**Configurations testées**	**Résultats**
1	**Comparer la performance de l'algorithme i-ALS selon les trois niveaux de taxonomies**	*Taxonomies ITEM_ID, ITEM_SS_CATG et ITEM_FAMILY*	Calculer P@3 de l'algorithme pour chaque configuration et la meilleure configuration sera utilisée pour le reste des expériences.
2	**Évaluer l'effet d'introduire l'ARM sur la performance de l'algorithme**	*ARM avec un niveau de confiance faible NC = 0.35* *ARM avec un niveau de confiance élevé NC = 0.65*	Calculer P@3 de l'algorithme pour chaque configuration et la meilleure configuration sera utilisée pour le reste des expériences.
3	**Évaluer l'effet de l'initialisation de l'algorithme *i-ALS***	*Avec Initialisation* *Sans Initialisation (Aléatoire)*	Calculer P@3 de l'algorithme pour chaque configuration et la meilleure configuration sera utilisée pour le reste des expériences.
Finale	**Comparaison avec des approches conventionnelles de la littérature**		

`

4.5 Conclusion

Dans ce chapitre, nous avons décrit et expliqué les différentes étapes de l'application de l'approche proposée à travers une étude de cas. Nous avons retenu la métrique de précision P@N pour évaluer les différentes étapes de l'expérience. Nous avons également discuté des spécifications du modèle et des hyperparamètres pour l'algorithme i-ALS. Enfin, nous avons présenté les différentes étapes de l'expérience et nous les avons résumées dans un plan d'expérience.

Dans le chapitre suivant, nous discutons et analysons les résultats de l'approche proposée et les différentes configurations adoptées en utilisant le jeu de données de notre partenaire industriel.

`

CHAPITRE 5 RÉSULTAS ET DISCUSSION

Dans ce chapitre, nous présentons les résultats expérimentaux réalisés pour évaluer les différentes configurations du chapitre précédent. Les évaluations concernent la factorisation matricielle proposée et les différentes améliorations adoptées. Nous utilisons la métrique de précision *P@3* pour évaluer la pertinence du système proposé.

5.1 Résultats

La métrique précision P@3 a été utilisée pour calculer la performance.

5.1.1 Comparer la performance de l'algorithme i-ALS selon les trois niveaux de taxonomies

Rappelons les matrices suivantes :

- Matrice UTILISATEUR-ITEM basée sur la taxonomie famille d'item est nommée *Matrice ITEM_FAMILY.*
- Matrice UTILISATEUR-ITEM basée sur la taxonomie sous-catégorie est nommée *Matrice ITEM_SS_CATG.*
- Matrice UTILISATEUR-ITEM basée sur l'ID d'item est nommée *Matrice ITEM_ID.*

En comparant les résultats obtenus en appliquant l'algorithme *i-ALS* sur les trois matrices telles qu'illustrés par la Figure 5.1, il apparaît que l'algorithme *i-ALS* est plus performant avec la *matrice ITEM_FAMILY*. Plus précisément, par rapport au cas standard (*matrice ITEM_ID*), les deux matrices basées sur les niveaux de taxonomie, *matrice ITEM_FAMILY* et *matrice SS_CATG*, sont respectivement 23,25% et 14,34% plus performantes. La raison pour laquelle ces deux matrices sont plus performantes n'est pas exclusivement due au fait qu'elles constituent un meilleur modèle. La principale différence entre les trois matrices est le niveau de sparsité des données comme présentée par le Tableau 5.1. En effet, la sparsité affecte grandement les techniques de factorisation

`

de la matrice. Plus concrètement, la *matrice ITEM_FAMILY* a un degré de sparsité de *97.84%*, ce qui signifie que seulement *2.16%* de toutes les interactions possibles sont remplies.

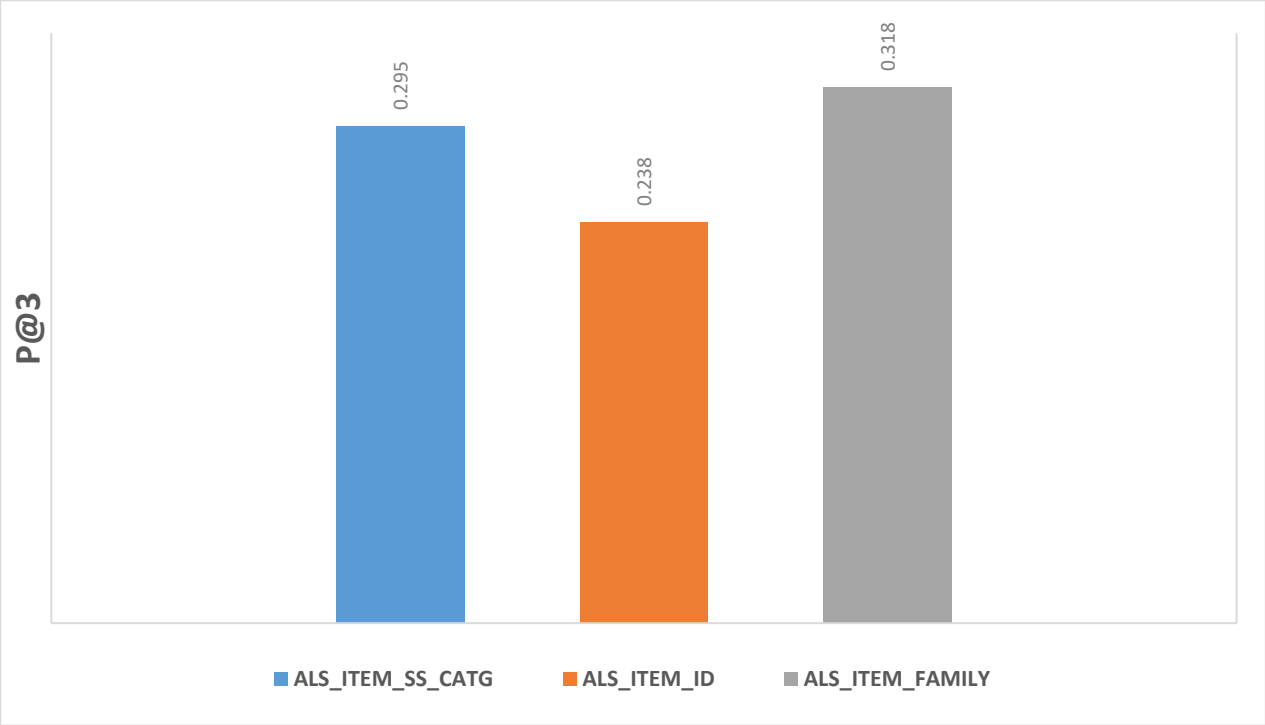

Figure 5.1 Comparaison la performance de l'algorithme *i-ALS* en fonction de différents niveaux de taxonomie

Tableau 5.1 Comparaison de niveau de sparsité des matrices

	Ensemble de données		
	ITEM_FAMILY	***ITEM_SS_CATG***	***ITEM_ID***
#items	*4113*	*350*	*12 000*
Sparsité (%)	***99.34***	***91.18***	***98.70***

Vu que la matrice ITEM_FAMILY a la meilleure performance, nous l'utilisons pour le reste de

`

l'expérience. Dans la prochaine phase, nous testons l'approche basée sur l'extraction des règles d'association.

5.1.2 Évaluer l'effet d'introduire l'ARM sur la performance de l'algorithme

Dans cette phase, nous évaluions l'effet d'introduire l'ARM sur la performance de l'algorithme i-ALS. Pour cela, nous construisons les algorithmes suivants :

- Algorithme testé avec la **M**atrice (Matrix ITEM_**F**AMILY) **S**tandard : **MFS**
- Algorithme testé avec la **M**atrice (Matrix ITEM_**F**AMILY) **R**enforcé par les règles d'association avec un niveau de confiance faible NC_**L**OW = 0.35 : **MFRL**
- Algorithme testé avec la **M**atrice (Matrix ITEM_**F**AMILY) **R**enforcé par les règles d'association avec un niveau de confiance élevé NC_**H**IGH = 0.65 : **MFRH**

Les résultats de cette phase sont présentés dans la Figure 5.2.

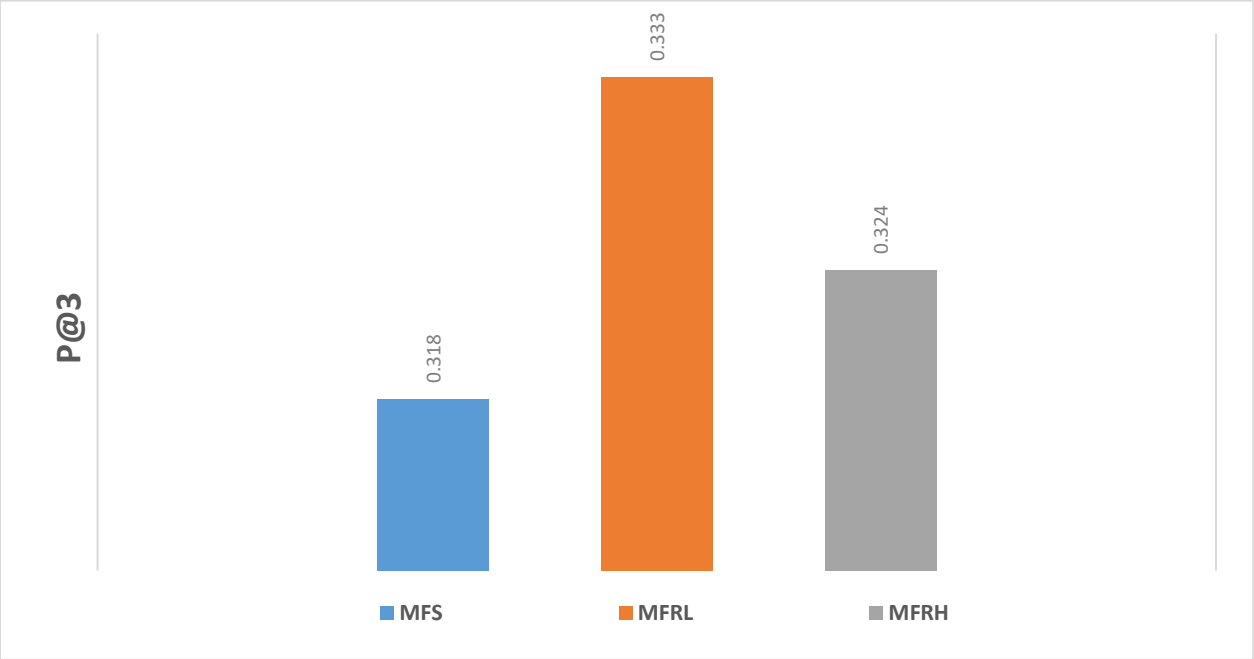

Figure 5.2 Comparaison de l'effet du niveau de confiance sur la performance de l'algorithme *i-ALS*

Dans ce cas, les deux configurations avec renforcement de la matrice ITEM_FAMILY donne les meilleurs résultats par rapport à la configuration standard. Notamment, le renforcement avec

`

l'ARM configuré avec un faible niveau de confiance a fourni la plus grande amélioration de ***4,7 %***. La configuration de l'ARM avec un niveau de confiance élevé a amélioré la performance méthodologie standard juste de *1,88 %*.

L'application de l'ARM pour renforcer la matrice améliore les performances de l'algorithme. Ce résultat s'explique par le fait que l'ajout des items à la liste des utilisateurs sur la base de règles ayant un bon niveau de confiance, diminue significativement la sparsité de la matrice standard telle que montre par le Tableau 5.2.

Tableau 5.2 Sparsité des matrices

	Matrice		
	MTS	***MTRL***	***MTRH***
Sparsité (%)	99.34	98.74	99.14

De manière inattendue, la configuration à faible niveau de confiance est plus performante que celle à fort niveau de confiance. Ceci pourrait être expliqué par le fait qu'en utilisant un faible niveau, nous avons généré plus de règles et donc nous avons plus renforcé la matrice, ce qui permet d'améliorer la sparsité (voir Tableau 5.3), de sorte que la performance de l'algorithme s'améliorera.

Tableau 5.3 Effet de l'introduction de l'ARM sur la sparsité

	Niveau Confiance bas = 0.65	*Niveau de confiance élevé = 0.35*
Nb. Règles générés	*24*	***54***
Sparsité (%)	*99.14*	***98.74***

`

5.1.3 Évaluer l'effet de l'initialisation de l'algorithme *i-ALS*

Dans la troisième phase, nous évaluons l'effet de l'initialisation de l'algorithme *i-ALS* avec le résultat de la technique de décomposition ***BLS-SVD*++**. Pour ce faire, nous comparons les quatre configurations suivantes :

- Configuration Standard : consiste à appliquer l'algorithme *i-ALS* avec la matrice ITEM_FAMILY. L'initialisation dans ce cas sera aléatoire.
- Configuration Améliorée par l'ARM : consiste à appliquer l'algorithme i-ALS avec la matrice ITEM_FAMILY renforcé par les règles d'association;
- Configuration Initialisée par ***BLS-SVD*++** : consiste à appliquer l'algorithme *i-ALS* initialisée par les résultats de l'algorithme ***BLS-SVD*++** en utilisant la matrice *ITEM_FAMILY*;
- Configuration Améliorée & initialisé : cette configuration est la combinaison de deux précédentes méthodes.

Comme l'illustre Figure 5.3, toutes les méthodes appliquées pour augmenter la précision du système l'ont considérablement amélioré. Ainsi, la combinaison des deux méthodes d'amélioration a réalisé la plus grande amélioration de ***12.47%***. La méthodologie standard est décrite dans le chapitre 3 basée sur l'initialisation aléatoire. Nous avons comparé cette méthode d'initialisation avec notre méthode d'initialisation proposée qui se base sur le résultat de l'algorithme ***BLS-SVD*++**. La même figure montre que notre méthode d'initialisation a amélioré la méthode standard de près de trois pour cent, ***2.83*** précisément.

Pour La suite, nous retenons la configuration suivante :

- Basée sur l'algorithme i-ALS initialisé (Phase trois montre que l'initialisation améliore la précision)
- En utilisant l'agrégation ITEM_FAMILLE (Phase une montre que la matrice ITEM_FAMILLE donne la meilleure précision par rapport les autres agrégation)
- En renforçant notre donnée par les résultats d'extraction des règles d'associations (Phase deux)

`

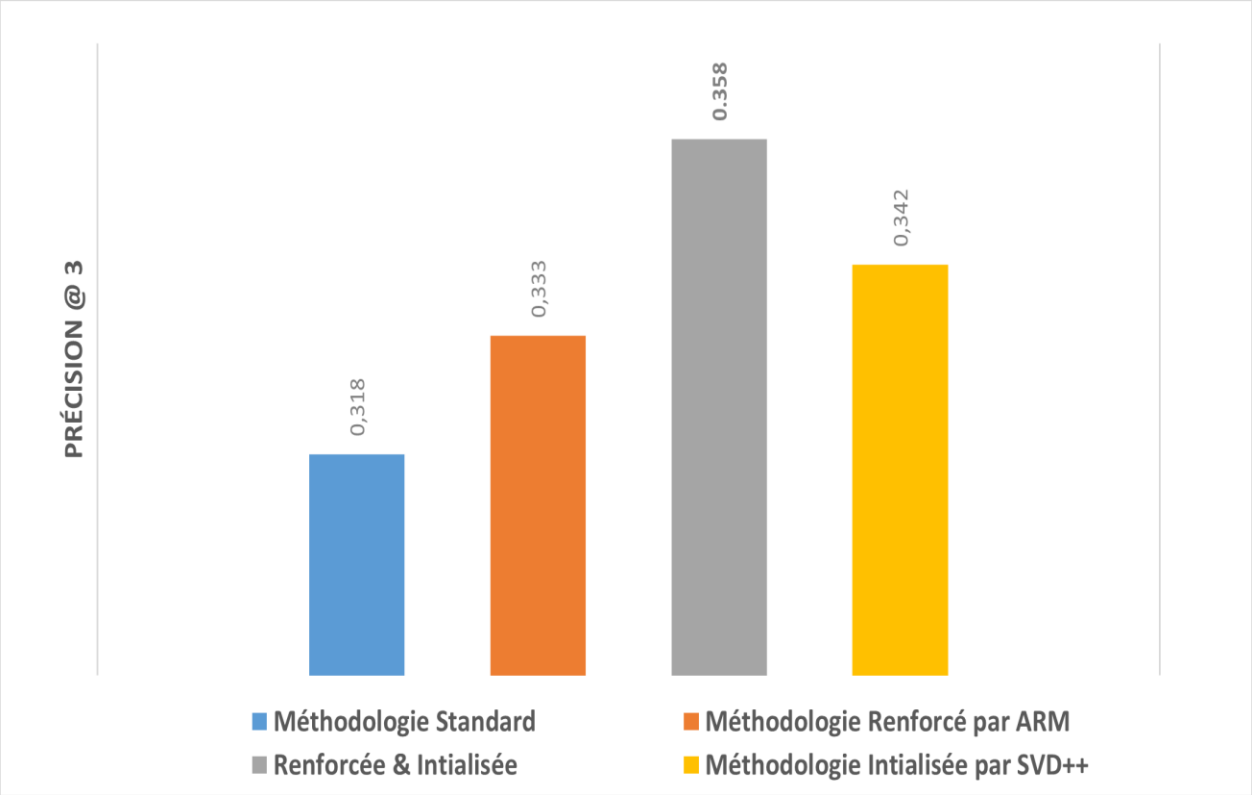

Figure 5.3 Évaluation de l'effet de l'initialisation

5.1.4 Comparaison avec les méthodes conventionnelles

Dans cette section, nous comparons l'approche retenue dans la section 5.1.3 avec les approches de littérature que nous les avons définis dans la section 4.4 soit l'approche aléatoire, approche le plus populaire, l'algorithme *BPR* et l'approche FM basée sur les réseaux de neurones *NeuMF*. Les mesures de performance moyennes des prédictions des approches testées sur l'ensemble du jeu de données de test sont présentées dans Figure 5.4. Les résultats montrent que l'approche proposée a réalisé la meilleure performance sur l'intervalle de test. Ce qui signifie que le modèle peut produire des recommandations personnalisées plus précises.

Ainsi, nous pouvons conclut, en nous basant sur une base de données réelle, que la création automatique d'une nouvelle taxonomie basée sur la description textuelle des items permet de surmonter les deux principaux défis du filtrage collaboratif : le problème de sparsité et le problème

`

des synonymies. De plus, l'introduction de règles d'association pour renforcer la matrice *utilisateur-item* améliore les performances en réduisant la sparsité de la matrice. Une autre hypothèse a

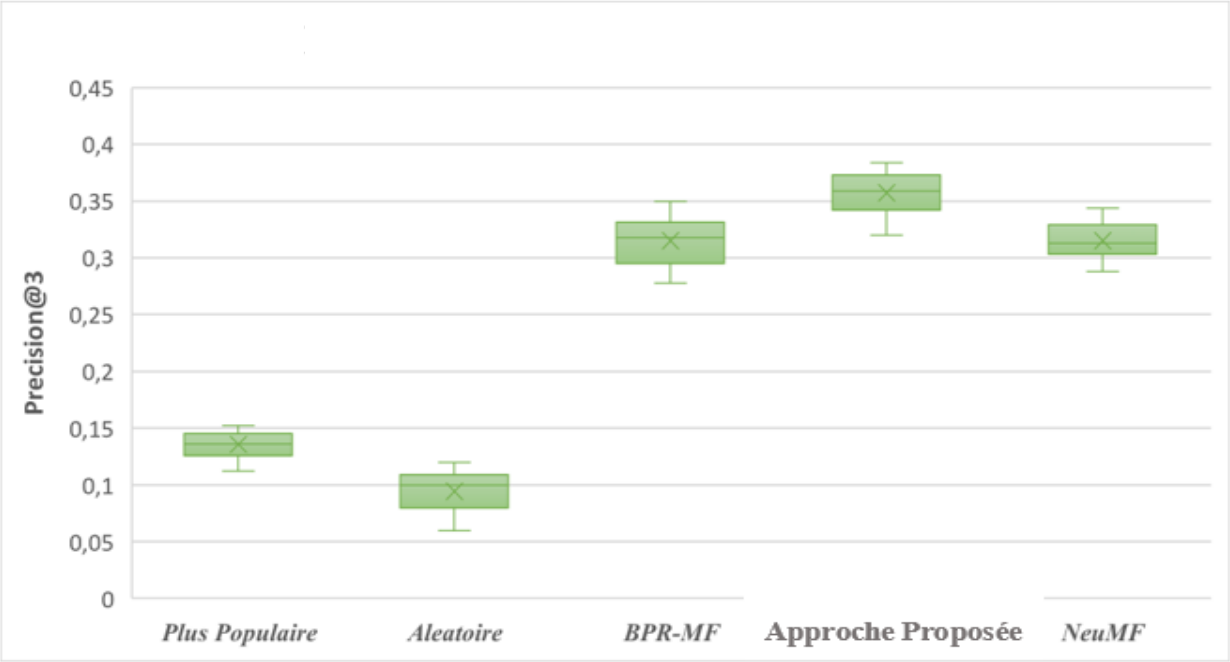

Figure 5.4 Résultats de l'évaluation des modèles avec les hyperparamètres optimaux

également été testée impliquant une initialisation non aléatoire, mais plutôt basée sur le résultat de l'algorithme ***BLS-SVD***++. Nos expériences montrent qu'une telle initialisation améliore significativement la précision de la technique de factorisation matricielle.

5.2 Réflexions managériales

Il est évident que les systèmes de recommandation ne sont pas uniquement bénéfiques aux utilisateurs du système, une entreprise peut réaliser également de grands bénéfices en mettant en place un système de recommandation efficace (Schafer et al., 2001). Du point de vue managérial, le système de recommandation est considéré comme un outil d'aide à la décision qui utilise de grandes données hétérogènes et des mécanismes générant des recommandations liées à la stratégie de vente de l'entreprise. Ainsi, les résultats d'un système de recommandation, tel notre système proposé, peuvent être bénéfique de différentes manières.

`

Tout d'abord, en utilisant nos résultats, notre partenaire ne vise pas seulement à fournir des items intéressants à leurs utilisateurs, mais également à augmenter les ventes. Ainsi, sur la base de la modélisation des besoins et des intérêts de chaque utilisateur, notre système peut être capable de guider ou de promouvoir les items les plus susceptibles d'être consommés par l'utilisateur en question. En outre, les managers peuvent exploiter ces résultats pour augmenter les ventes croisées. En effet, Les recommandations peuvent aider les utilisateurs qui ne sont pas sûrs de leurs goûts ou intérêts à identifier leurs véritables favoris. En outre, des suggestions peuvent être faites sur la base des items que l'intention d'acheter et trouver des items complémentaires possibles.

Deuxièmement, notre système proposé peut aider notre partenaire à accroître la satisfaction et la fidélité des utilisateurs en le déployant sur son site web. En effet, au fil du temps, il accumule de plus en plus de données sur les utilisateurs et les items et peut fournir des recommandations de plus en plus précises. Avec des recommandations intéressantes et pertinentes, l'utilisateur vivra une expérience agréable. L'utilisateur sera plus enclin à préférer le service à l'avenir. En ce sens, une relation entre le site web et l'utilisateur se forme, car plus le système est adapté aux besoins de l'utilisateur, plus ce dernier est satisfait. Cela conduit à une augmentation de la fidélité, parce que les utilisateurs continuellement satisfaits apprécient l'expérience qu'ils vivent et reviennent sur les sites avec lesquels ils aiment interagir.

5.3 Conclusion

Nous avons présenté dans cette dernière partie les résultats d'expérimentations de nos contributions qui sont basées essentiellement sur l'utilisation de la taxonomie automatiquement créée pour surmonter le problème de sparsité et le problème de synonymie du filtrage collaboratif. D'un autre côté l'introduction de l'extraction des règles d'association nous a permis de diminuer la mesure de sparsité et d'atteindre une meilleure qualité de recommandation. Finalement, et afin d'optimiser l'algorithme de factorisation matricielle, nous avons appliqué une initialisation non aléatoire, cette solution donne des résultats satisfaisants. Par critères d'efficacité et de simplicité du langage Python que nous avons utilisé, les résultats obtenus sont encourageants par rapport aux résultats antérieurs dans le domaine de filtrage collaboratif.

`

CHAPITRE 6 CONCLUSION, DISCUSSION ET RECOMMANDATIONS

Dans ce chapitre, nous présentons un bref résumé de ce qui a été fait dans ce projet de recherche. En outre, les limites de notre étude seront discutées et quelques perspectives d'études futures seront suggérées.

6.1 Discussion

Dans ce projet, nous avons abordé le sujet du développement du système de recommandation basé sur le filtrage collaboratif pour le commerce électronique B2B en utilisant le feedback implicite. Bien que la technique de filtrage collaboratif soit la méthode la plus utilisée dans le cadre de la recommandation, elle souffre du problème de sparsité des données. En effet, la matrice *utilisateur - item* est généralement peu dense et contient plusieurs données manquantes. Ceci affecte la qualité et la précision de la recommandation.

Pour pallier le problème de la sparsité des données, nous avons proposé une nouvelle approche, basée sur l'utilisation de la taxonomie d'items et l'extraction des règles d'associations. En effet, nous proposons tout d'abord de générer automatiquement une nouvelle taxonomie explicite basée uniquement sur les descriptions textuelles des items. Nous avons également renforcé la matrice utilisateur-item en exploitant les résultats de la technique d'extraction de règles d'association. Basé sur une règle, nous avons ajouté des items complémentaires aux utilisateurs en se basant sur leurs achats des items de base de la règle. Finalement, nous avons proposé une initialisation de l'algorithme de factorisation matricielle utilisé dans notre approche. En effet, les matrices de facteurs latents ont été initialisés avec des valeurs calculées résultant de la technique de décomposition en valeurs singulières (**BLS-SVD++**).

Les résultats que nous avons obtenus à travers des expériences avec des données réelles sont encourageants et valident notre idée de bénéficier de la taxonomie et des techniques d'extraction des règles d'association afin de combler le problème de la sparsité. Nous avons montré à travers d'une expérience que notre initialisation de l'algorithme principal a produit de meilleurs résultats du point de vue précision par rapport à l'initialisation aléatoire.

Les résultats expérimentaux ont également démontré que notre approche obtient de meilleures performances par rapport à d'autres approches de la littérature en termes de précision. Cette

`

amélioration est due à l'avantage majeur de notre approche proposée qu'est capable considérablement de surmonter le problème de sparsité.

6.2 Recommandations

Afin de construire la matrice de données pour générer des recommandations, nous avons sélectionné une seule mesure implicite, qui représente l'intérêt de l'utilisateur pour un item donné, à savoir la fréquence d'achat. Les données utilisées pour dériver ces intérêts étaient uniquement des données historiques stockées dans une base de données statique en raison de l'inexistence d'un site de commerce électronique chez notre partenaire industriel. Dans un tel cas, des enjeux peuvent être posés dans la conception et la mise en œuvre d'un tel système de recommandation. En effet, il existe d'autres informations implicites telles que le nombre de clics de souris et le temps passé sur la page web que nous pouvons exploiter si nous avons un site web bien implanté. Loannis Schoinas et Christos Tjortjis (2017) ont montré que l'utilisation de plusieurs informations implicites améliore significativement la qualité de la recommandation. De plus, nous pouvons construire des profils d'utilisateurs dynamiques qui sont mis à jour lorsque l'utilisateur effectue un achat ou navigue sur le site web. Dans ce sens, nous suggérons à notre partenaire de mettre en place un site de commerce électronique afin de mettre en œuvre notre système proposé. Dans ce cas, il sera possible de faire une évaluation en ligne de notre système de recommandation et d'apporter des améliorations appropriées si nécessaire.

6.3 Limitations

Ce travail de recherche a souffert de certaines limitations liées i) aux mesures implicites et ii) la méthode d'évaluation.

Mesures implicites : Afin de construire la matrice utilisée pour la factorisation matricielle, nous avons sélectionné une seule mesure implicite, qui mesure l'intérêt de l'utilisateur pour un item. Cependant, il existe d'autres mesures implicites comme le nombre de clics de souris et le temps passé sur la page web. Malheureusement, ces données n'ont pas été collectées dans le cadre de ce projet, et n'ont donc pas pu être utilisées pour construire la matrice. Ces mesures supplémentaires permettraient d'obtenir une plus grande certitude qu'un utilisateur est intéressé par un élément, et pourraient donc construire une meilleure matrice.

`

Évaluation hors ligne : Dans ce projet, seule une évaluation hors ligne pu être réalisé sur des données pré-collectées, et tenté de simuler l'interaction de l'utilisateur avec un système de recommandation. Ce type d'évaluation mesure le pouvoir prédictif d'un système de recommandation, tandis que les expériences en ligne peuvent mesurer l'influence d'un système de recommandation sur le comportement de l'utilisateur. Malheureusement, cela n'a pas pu être testé dans ce projet, car le système de recommandation basé sur la factorisation matricielle n'a pas pu être implémenté sur le site web du partenaire industriel.

6.4 Perspectives

À partir des limitations dégagées dans la section 6.3, ce travail pourrait être amélioré en se concentrant sur l'utilisation d'autres mesures implicites supplémentaires pour construire la matrice *utilisateur-item*. Ainsi, les recherches futures devraient examiner les différentes manières d'inclure différentes informations implicites afin d'avoir une meilleure précision des recommandations.

Les ensembles de données implicites qui offrent un historique des transactions offrent des possibilités d'analyser de séquences d'achat. Dans notre travail, nous n'avons considéré que les règles d'association et nous avons laissé l'analyse des motifs séquentiels comme l'une des meilleures pistes d'amélioration.

La question de la sparsité des données devrait être examinée plus en détails. Les règles d'association seules ont réussi à diminuer la sparsité de manière significative. Un moyen possible pourrait être la similarité des items basée sur les caractéristiques des items. Les informations sur les items sont également disponibles dans l'ensemble de données, ce qui permet de mettre en œuvre des techniques de filtrage basées sur le contenu.

En outre, étant donné que le prix d'item est un critère de base dans la décision d'achat, l'impact du prix sur les préférences des utilisateurs devrait être inclus dans les recherches futures. De même, de nombreux items ne conviennent qu'à des périodes spécifiques, comme les saisons. Ainsi, le facteur temps devrait également être pris en compte.

`

RÉFÉRENCES

Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., & Zhang, L. (2016). Deep learning with differential privacy. Proceedings of the 2016 ACM SIGSAC conference on computer and communications security,

Aciar, S., Zhang, D., Simoff, S., & Debenham, J. (2007). Informed recommender: Basing recommendations on consumer product reviews. *IEEE Intelligent systems*, *22*(3), 39-47.

Adomavicius, G., & Tuzhilin, A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE transactions on knowledge and data engineering*, *17*(6), 734-749.

Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules. Proc. 20th int. conf. very large data bases, VLDB,

Ahmed, E., Gani, A., Khan, M. K., Buyya, R., & Khan, S. U. (2015). Seamless application execution in mobile cloud computing: Motivation, taxonomy, and open challenges. *Journal of Network and Computer Applications*, *52*, 154-172.

Al-Bashiri, H., Abdulgabber, M. A., Romli, A., & Hujainah, F. (2017). Collaborative filtering recommender system: Overview and challenges. *Advanced Science Letters*, *23*(9), 9045-9049.

Basiri, J., Shakery, A., Moshiri, B., & Hayat, M. Z. (2010). Alleviating the cold-start problem of recommender systems using a new hybrid approach. 2010 5th International Symposium on Telecommunications,

Beaudoin, J., Lefebvre, G., Normand, M., Gouri, V., Skerlj, A., Pellerin, R., Rivest, L., & Danjou, C. (2016). *Prendre part à la révolution manufacturière? Du rattrapage technologique à l'Industrie 4.0 chez les PME*. Centre francophone d'informatisation des organisations (CEFRIO).

Beheshti, A., Yakhchi, S., Mousaeirad, S., Ghafari, S. M., Goluguri, S. R., & Edrisi, M. A. (2020). Towards cognitive recommender systems. *Algorithms*, *13*(8), 176.

Beliakov, G., Calvo, T., & James, S. (2011). Aggregation of preferences in recommender systems. In *Recommender systems handbook* (pp. 705-734). Springer.

Bell, R. M., & Koren, Y. (2007). Scalable collaborative filtering with jointly derived neighborhood interpolation weights. Seventh IEEE international conference on data mining (ICDM 2007),

Blanco-Fernández, Y., Pazos-Arias, J. J., Gil-Solla, A., Ramos-Cabrer, M., López-Nores, M., García-Duque, J., Fernández-Vilas, A., Díaz-Redondo, R. P., & Bermejo-Munoz, J. (2008). A flexible semantic inference methodology to reason about user preferences in knowledge-based recommender systems. *Knowledge-Based Systems*, *21*(4), 305-320.

`

Breese, J. S., Heckerman, D., & Kadie, C. (1998). Empirical Analysis of Predictive Algorithms for Collaborative Filtering. In Proceedings of Fourteenth Conference on Uncertainty. *Artificial Intelligence, Madison, Wisconsin. Morgan Kaufmann*.

Cantador, I., Bellogín, A., & Castells, P. (2008). A multilayer ontology-based hybrid recommendation model. *Ai Communications*, *21*(2-3), 203-210.

Cho, Y. H., & Kim, J. K. (2004). Application of Web usage mining and product taxonomy to collaborative recommendations in e-commerce. *Expert Systems with Applications*, *26*(2), 233-246.

Chughtai, M. W., Ghani, I., Semalat, A., & Jeong, S. R. (2013). Goal-based Framework for cold-start problem using multi-user personalized similarities in e-Learning scenarios. 2013 Second International Conference on E-Learning and E-Technologies in Education (ICEEE),

D, A. K., & Kuttiyannan, D. T. (2003). *A SINGLE PASS INCREMENTAL CLUSTERING ALGORITHM FOR DATA MINING*.

Dadouchi, C., & Agard, B. (2018). Lowering penalties related to stock-outs by shifting demand in product recommendation systems. *Decision Support Systems*, *114*, 61-69.

Donoho, D. L., & Grimes, C. (2003). Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *Proceedings of the National Academy of Sciences*, *100*(10), 5591-5596.

Downey, D., Bhagavatula, C., & Yang, Y. (2015). Efficient methods for inferring large sparse topic hierarchies. Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers),

Dutta, R., & Mukhopadhyay, D. (2011). Offering A Product Recommendation System in E-commerce. *arXiv preprint arXiv:1109.4257*.

Ekstrand, M. D., Riedl, J. T., & Konstan, J. A. (2011). Collaborative filtering recommender systems. *Foundations and Trends® in Human–Computer Interaction*, *4*(2), 81-173.

Elahi, M., Ricci, F., & Rubens, N. (2016). A survey of active learning in collaborative filtering recommender systems. *Computer Science Review*, *20*, 29-50.

Feng, H., Tian, J., Wang, H. J., & Li, M. (2015). Personalized recommendations based on time-weighted overlapping community detection. *Information & Management*, *52*(7), 789-800.

Frawley, W. J., Piatetsky-Shapiro, G., & Matheus, C. J. (1992). Knowledge discovery in databases: An overview. *AI magazine*, *13*(3), 57-57.

Gao, J., Wang, X., Wang, Y., & Xie, X. (2019). Explainable recommendation through attentive multi-view learning. Proceedings of the AAAI Conference on Artificial Intelligence,

García-Cumbreras, M. Á., Montejo-Ráez, A., & Díaz-Galiano, M. C. (2013). Pessimists and optimists: Improving collaborative filtering through sentiment analysis. *Expert Systems with Applications*, *40*(17), 6758-6765.

`

Ge, M., Delgado-Battenfeld, C., & Jannach, D. (2010). Beyond accuracy: evaluating recommender systems by coverage and serendipity. Proceedings of the fourth ACM conference on Recommender systems,

Ghazanfar, M. A., & Prügel-Bennett, A. (2014). Leveraging clustering approaches to solve the gray-sheep users problem in recommender systems. *Expert Systems with Applications*, *41*(7), 3261-3275.

Guo, G., Qiu, H., Tan, Z., Liu, Y., Ma, J., & Wang, X. (2017). Resolving data sparsity by multi-type auxiliary implicit feedback for recommender systems. *Knowledge-Based Systems*, *138*, 202-207.

Han, J., Kamber, M., & Pei, J. (2012). Outlier detection. *Data mining: concepts and techniques*, 543-584.

Hdioud, F., Frikh, B., & Ouhbi, B. (2013). Multi-criteria recommender systems based on multi-attribute decision making. Proceedings of international conference on information integration and web-based applications & services,

He, X., Liao, L., Zhang, H., Nie, L., Hu, X., & Chua, T.-S. (2017). Neural collaborative filtering. Proceedings of the 26th international conference on world wide web,

Herlocker, J., Konstan, J. A., & Riedl, J. (2002). An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. *Information retrieval*, *5*(4), 287-310.

Herlocker, J. L., Konstan, J. A., & Riedl, J. (2000). Explaining collaborative filtering recommendations. Proceedings of the 2000 ACM conference on Computer supported cooperative work,

Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS)*, *22*(1), 5-53.

Hioud, F. (2017). *Industrial Recommendation Systems in Big Data Context: Efficient Solutions for Cold-Start issues* Faculté de Sciences et Techniques de Fès (FSTF)].

Jannach, D., Zanker, M., Felfernig, A., & Friedrich, G. (2010). *Recommender systems: an introduction*. Cambridge University Press.

Johnson, C. C. (2014). Logistic matrix factorization for implicit feedback data. *Advances in Neural Information Processing Systems*, *27*(78), 1-9.

Jooa, J., Bangb, S., & Parka, G. (2016). Implementation of a recommendation system using association rules and collaborative filtering. *Procedia Computer Science*, *91*, 944-952.

Kalman, D. (1996). A singularly valuable decomposition: the SVD of a matrix. *The college mathematics journal*, *27*(1), 2-23.

Kanagal, B., Ahmed, A., Pandey, S., Josifovski, V., Yuan, J., & Garcia-Pueyo, L. (2012). Supercharging recommender systems using taxonomies for learning user purchase behavior. *arXiv preprint arXiv:1207.0136*.

`

Karatzoglou, A., Baltrunas, L., & Shi, Y. (2013). Learning to rank for recommender systems. Proceedings of the 7th ACM Conference on Recommender Systems,

Kardan, A. A., & Ebrahimi, M. (2013). A novel approach to hybrid recommendation systems based on association rules mining for content recommendation in asynchronous discussion groups. *Information Sciences*, *219*, 93-110.

Keinänen, H., & Kuivalainen, O. (2015). Antecedents of social media B2B use in industrial marketing context: customers' view. *Journal of Business & Industrial Marketing*.

Kim, C., & Kim, J. (2003). A recommendation algorithm using multi-level association rules. Proceedings IEEE/WIC International Conference on Web Intelligence (WI 2003),

Konstan, J. A., & Riedl, J. (2012). Recommender systems: from algorithms to user experience. *User modeling and user-adapted interaction*, *22*(1), 101-123.

Koren, Y. (2008). Factorization meets the neighborhood: a multifaceted collaborative filtering model. Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining,

Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, *42*(8), 30-37.

Krohn-Grimberghe, A., Drumond, L., Freudenthaler, C., & Schmidt-Thieme, L. (2012). Multi-relational matrix factorization using bayesian personalized ranking for social network data. Proceedings of the fifth ACM international conference on Web search and data mining,

Kula, M. (2015). Metadata embeddings for user and item cold-start recommendations. *arXiv preprint arXiv:1507.08439*.

Lee, T., Chun, J., Shim, J., & Lee, S.-g. (2006a). An ontology-based product recommender system for B2B marketplaces. *International Journal of Electronic Commerce*, *11*(2), 125-155.

Lee, T., Chun, J., Shim, J., & Lee, S.-g. (2006b). An Ontology-Based Product Recommender System for B2B Marketplaces. *International Journal of Electronic Commerce / Winter*, *7*, 125-155. https://doi.org/10.2753/JEC1086-4415110206

Liashchynskyi, P., & Liashchynskyi, P. (2019). Grid search, random search, genetic algorithm: a big comparison for NAS. *arXiv preprint arXiv:1912.06059*.

Lika, B., Kolomvatsos, K., & Hadjiefthymiades, S. (2014). Facing the cold start problem in recommender systems. *Expert Systems with Applications*, *41*(4), 2065-2073.

Linden, G., Smith, B., & York, J. (2003). Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing*, *7*(1), 76-80.

Liphoto, M., Du, C., & Ngwira, S. (2016, 28-29 Nov. 2016). A survey on recommender systems. 2016 International Conference on Advances in Computing and Communication Engineering (ICACCE),

Ma, W., Zhang, M., Cao, Y., Jin, W., Wang, C., Liu, Y., Ma, S., & Ren, X. (2019). Jointly learning explainable rules for recommendation with knowledge graph. The world wide web conference,

`

Meng, Z., McCreadie, R., Macdonald, C., & Ounis, I. (2020). Exploring data splitting strategies for the evaluation of recommendation models. Fourteenth ACM conference on recommender systems,

Nasiri, M., & Minaei, B. (2016). Increasing prediction accuracy in collaborative filtering with initialized factor matrices. *The Journal of Supercomputing*, *72*(6), 2157-2169.

Nilashi, M., Ibrahim, O., & Bagherifard, K. (2018). A recommender system based on collaborative filtering using ontology and dimensionality reduction techniques. *Expert Systems with Applications*, *92*, 507-520. https://doi.org/https://doi.org/10.1016/j.eswa.2017.09.058

Nilashi, M., Ibrahim, O. b., & Ithnin, N. (2014). Hybrid recommendation approaches for multi-criteria collaborative filtering. *Expert Systems with Applications*, *41*(8), 3879-3900. https://doi.org/https://doi.org/10.1016/j.eswa.2013.12.023

Osadchiy, T., Poliakov, I., Olivier, P., Rowland, M., & Foster, E. (2019). Recommender system based on pairwise association rules. *Expert Systems with Applications*, *115*, 535-542.

Pilászy, I. (2010). Factorization-based large scale recommendation algorithms.

Pilászy, I., Zibriczky, D., & Tikk, D. (2010). Fast als-based matrix factorization for explicit and implicit feedback datasets. Proceedings of the fourth ACM conference on Recommender systems,

Pradel, B., Sean, S., Delporte, J., Guérif, S., Rouveirol, C., Usunier, N., Fogelman-Soulié, F., & Dufau-Joel, F. (2011). A case study in a recommender system based on purchase data. Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining,

Radev, D., Otterbacher, J., & Zhang, Z. (2008). Cross-document relationship classification for text summarization. *tangra, si, umich, edu/~ radev/papers/progress/p1, ps (último acesso: 13/04/2009)*.

Rendle, S., & Schmidt-Thieme, L. (2008). Online-updating regularized kernel matrix factorization models for large-scale recommender systems. Proceedings of the 2008 ACM conference on Recommender systems,

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994). Grouplens: An open architecture for collaborative filtering of netnews. Proceedings of the 1994 ACM conference on Computer supported cooperative work,

Ricci, F., Rokach, L., & Shapira, B. (2011). Introduction to recommender systems handbook. In *Recommender systems handbook* (pp. 1-35). Springer.

Ricci, F., Rokach, L., & Shapira, B. (2015). Recommender systems: introduction and challenges. In *Recommender systems handbook* (pp. 1-34). Springer.

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. Proceedings of the 10th international conference on World Wide Web,

Schafer, J. B., Konstan, J. A., & Riedl, J. (2001). E-commerce recommendation applications. *Data mining and knowledge discovery*, *5*(1), 115-153.

`

Schoinas, I., & Tjortjis, C. (2019). MuSIF: a product recommendation system based on multi-source implicit feedback. IFIP International Conference on Artificial Intelligence Applications and Innovations,

Shani, G., & Gunawardana, A. (2011). Evaluating recommendation systems. In *Recommender systems handbook* (pp. 257-297). Springer.

Shardanand, U., & Maes, P. (1995). Social information filtering: Algorithms for automating "word of mouth". Proceedings of the SIGCHI conference on Human factors in computing systems,

Sharma, M., & Mann, S. (2013). A survey of recommender systems: approaches and limitations. *International Journal of Innovations in Engineering and Technology*, *2*(2), 8-14.

Shokeen, J., & Rana, C. (2020). Social recommender systems: techniques, domains, metrics, datasets and future scope. *Journal of Intelligent Information Systems*, *54*(3), 633-667.

Smilde, A. K., Geladi, P., & Bro, R. (2005). *Multi-way analysis: applications in the chemical sciences*. John Wiley & Sons.

Su, X., & Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques. *Advances in artificial intelligence*, *2009*.

Su, X., & Khoshgoftaar, T. M. (2009). A Survey of Collaborative Filtering Techniques. *Advances in artificial intelligence*, *2009*, 421425. https://doi.org/10.1155/2009/421425

Tadlaoui, M. (2018). *Système de recommandation de ressources pédagogiques fondé sur les liens sociaux: formalisation et évaluation* Université de Lyon; Université Abou Bekr Belkaid (Tlemcen, Algérie)].

Takács, G., Pilászy, I., Németh, B., & Tikk, D. (2007). Major components of the gravity recommendation system. *Acm Sigkdd Explorations Newsletter*, *9*(2), 80-83.

Takács, G., Pilászy, I., Németh, B., & Tikk, D. (2009). Scalable collaborative filtering approaches for large recommender systems. *The Journal of Machine Learning Research*, *10*, 623-656.

Takács, G., Pilászy, I., & Tikk, D. (2011). Applications of the conjugate gradient method for implicit feedback collaborative filtering. Proceedings of the fifth ACM conference on Recommender systems,

Tyagi, S., & Bharadwaj, K. K. (2013). Enhancing collaborative filtering recommendations by utilizing multi-objective particle swarm optimization embedded association rule mining. *Swarm and Evolutionary Computation*, *13*, 1-12.

Wang, J., De Vries, A. P., & Reinders, M. J. (2006). Unifying user-based and item-based collaborative filtering approaches by similarity fusion. Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval,

Wang, P.-j., Shi, L., Bai, J.-n., & Zhao, Y.-l. (2009). Mining association rules based on apriori algorithm and application. 2009 International Forum on Computer Science-Technology and Applications,

`

Wang, S., Sun, G., & Li, Y. (2020). SVD++ recommendation algorithm based on backtracking. *Information*, *11*(7), 369.

Wang, X., Wang, D., Xu, C., He, X., Cao, Y., & Chua, T.-S. (2019). Explainable reasoning over knowledge graphs for recommendation. Proceedings of the AAAI conference on artificial intelligence,

Weng, L.-T., Xu, Y., Li, Y., & Nayak, R. (2008). Exploiting item taxonomy for solving cold-start problem in recommendation making. 2008 20th IEEE International Conference on Tools with Artificial Intelligence,

Yang, B., Lei, Y., Liu, J., & Li, W. (2016). Social collaborative filtering by trust. *IEEE transactions on pattern analysis and machine intelligence*, *39*(8), 1633-1647.

Yang, C., Zhang, J., & Han, J. (2020). Co-embedding network nodes and hierarchical labels with taxonomy based generative adversarial networks. 2020 IEEE International Conference on Data Mining (ICDM),

Yilmaz, E., Kanoulas, E., & Aslam, J. A. (2008). A simple and efficient sampling method for estimating AP and NDCG. Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval,

Yue, Y., Finley, T., Radlinski, F., & Joachims, T. (2007). A support vector method for optimizing average precision. Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval,

Zeng, W., Zeng, A., Liu, H., Shang, M.-S., & Zhou, T. (2014). Uncovering the information core in recommender systems. *Scientific reports*, *4*(1), 1-8.

Zhang, C., Tao, F., Chen, X., Shen, J., Jiang, M., Sadler, B., Vanni, M., & Han, J. (2018). Taxogen: Unsupervised topic taxonomy construction by adaptive term embedding and clustering. Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining,

Zhang, X., & Wang, H. (2005). Study on recommender systems for business-to-business electronic commerce. *Communications of the IIMA*, *5*(4), 8.

Zhang, Y., Ahmed, A., Josifovski, V., & Smola, A. (2014). Taxonomy discovery for personalized recommendation. Proceedings of the 7th ACM international conference on Web search and data mining,

Ziegler, C.-N., & Lausen, G. (2009). Making product recommendations more diverse. *IEEE Data Eng. Bull.*, *32*(4), 23-32.

Ziegler, C.-N., Lausen, G., & Schmidt-Thieme, L. (2004). Taxonomy-driven computation of product recommendations. Proceedings of the thirteenth ACM international conference on Information and knowledge management,

Ziegler, C.-N., Schmidt-Thieme, L., & Lausen, G. (2004). Exploiting semantic product descriptions for recommender systems. Proceedings of the 2nd ACM SIGIR Semantic Web and Information Retrieval Workshop,

`

Zuva, T., Ojo, S., Ngwira, S. M., & Zuva, K. (2012). A Survey of Recommender Systems Techniques, Challenges and Evaluation Metrics.