

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université des Sciences et de la Technologie Houari Boumediene
Faculté d'électronique et d'informatique
Département d'informatique



Mémoire de Master

Domaine : Informatique

Spécialité : Systèmes Informatiques Intelligents

Thème

Recommandation par Filtrage Collaboratif et Sémantique
basée sur une classification optimisée

Présenté par :

- BEZGALI Meriem
- CHABAANE Nouar

Proposé et dirigé par :

-Mme. BERKANI

Devant le jury composé de :

- M.Aissani
- Mme.Bouchene

Président
Membre

Projet N° : 064/2019

Table des matières

| | | |
|----------|---|-----------|
| 1 | Recommandation personnalisée | 12 |
| 1.1 | Introduction | 13 |
| 1.2 | Recommandation | 13 |
| 1.2.1 | Les systèmes de recommandations | 14 |
| 1.2.2 | Le fonctionnement des systèmes de recommandation | 14 |
| 1.2.3 | Les caractéristiques d'un SR | 15 |
| 1.3 | Techniques de recommandation | 15 |
| 1.3.1 | Filtrage cognitif | 16 |
| 1.3.2 | Filtrage collaboratif | 19 |
| 1.3.3 | Filtrage hybride | 24 |
| 1.4 | Autre Système de recommandation | 24 |
| 1.4.1 | Filtrage sémantique | 24 |
| 1.5 | Construction du profil utilisateur | 28 |
| 1.5.1 | Caractéristiques du profil utilisateur | 28 |
| 1.5.2 | Modèle de représentation du profil de l'utilisateur | 29 |
| 1.6 | Travaux liés | 31 |
| 1.6.1 | Recommandation hybride par Filtrage sémantique et FC multicritères . | 31 |
| 1.6.2 | Un système de recommandation basé sur une clustering multivues de similarité et de confiance | 31 |
| 1.7 | Conclusion | 32 |
| 2 | Classification et optimisation | 33 |

| | | |
|----------|--|-----------|
| 2.1 | Introduction | 34 |
| 2.2 | classification | 34 |
| 2.2.1 | Modèles supervisés | 34 |
| 2.2.2 | Modèles non supervisés (<i>Clustering</i>) | 36 |
| 2.3 | Optimisation avec <i>métaheuristiques</i> | 39 |
| 2.3.1 | Bee Swarm Optimisation (BSO) | 40 |
| 2.3.2 | Optimisation dans les systèmes de recommandations | 41 |
| 2.4 | Conclusion | 42 |
| 3 | Conception | 43 |
| 3.1 | Introduction | 44 |
| 3.2 | Description de l'approche de recommandation par classification | 44 |
| 3.3 | Filtrage sans classification | 45 |
| 3.3.1 | Filtrage collaboratif | 45 |
| 3.3.2 | Filtrage sémantique standard | 51 |
| 3.3.3 | Filtrage hybride | 57 |
| 3.4 | Filtrage avec classification | 65 |
| 3.4.1 | Filtrage avec classification supervisée | 66 |
| 3.4.2 | Filtrage avec classification non supervisée | 70 |
| 3.4.3 | Filtrage avec classification non supervisée optimisée | 72 |
| 3.5 | Exemple de calcul de prédiction | 78 |
| 3.6 | Ingénierie du système | 79 |
| 3.7 | Conclusion | 81 |
| 4 | Implémentation et expérimentation | 82 |
| 4.1 | Introduction | 83 |
| 4.2 | Environnement de travail | 83 |
| 4.2.1 | Langage de programmation <i>Python</i> | 83 |
| 4.2.2 | Outils et logiciels | 84 |
| 4.3 | Présentation de notre système de recommandation | 84 |
| 4.3.1 | Description générale | 84 |
| 4.3.2 | Illustration des fonctionnalités développées | 85 |
| 4.4 | Expérimentation | 90 |

| | | |
|----------------------|--|------------|
| 4.4.1 | Dataset | 90 |
| 4.4.2 | Métriques d'évaluation | 92 |
| 4.4.3 | Résultats des évaluations | 92 |
| 4.5 | Récapitulatif des différents résultats des évaluations | 117 |
| 4.6 | Conclusion | 118 |
| Appendices | | 119 |
| Bibliographie | | I |

Table des figures

| | | |
|------|--|----|
| 1.1 | classification principale des système de recommandation de [5] | 16 |
| 1.2 | SR basé sur le contenu [6] | 18 |
| 1.3 | Matrice d’usage | 20 |
| 1.4 | Matrice d’usage | 20 |
| 1.5 | Une ontologie du concept animal [29] | 25 |
| 1.6 | Exemple de taxonomie pour les mesures de similarité basées sur les arcs | 27 |
| 1.7 | Exemple de hiérarchie de concepts [39] | 27 |
| 1.8 | Exemple de profil utilisateur représenté par le modèle d’ontologies avec le processus de mise à jour des poids des concepts [28] | 30 |
| 2.1 | Schéma expliquant le fonctionnement de BSO [35] | 41 |
| 3.1 | Description générale de notre système de recommandation | 45 |
| 3.2 | Formule SVD | 47 |
| 3.3 | Exemple des caractéristiques d’un user [32] | 48 |
| 3.4 | Exemple de Singular Value decomposition [32] | 49 |
| 3.5 | Exemple d’une ligne de la matrice <i>item-catégorie</i> | 52 |
| 3.6 | Génération de la matrice de distance <i>item-item</i> | 54 |
| 3.7 | Génération de la matrice de distance <i>user-user</i> | 55 |
| 3.8 | Matrice de distance user-user | 57 |
| 3.9 | Pondération des matrices de distances | 58 |
| 3.10 | Différence entre sémantique standard et sémantique basé collaboratif | 62 |
| 3.11 | Exemple de filtrage collaboratif-sémantique | 63 |
| 3.12 | Sélection des matrices de distance <i>user-user</i> pour l’application de la classification | 65 |

| | | |
|------|--|-----|
| 3.13 | Schéma du filtrage multiview K-NN | 68 |
| 3.14 | Génération des <i>search areas</i> à partir de <i>Sref</i> | 76 |
| 3.15 | Exemple de matrice d'usage | 78 |
| 3.16 | Exemple de matrice de distance | 78 |
| 3.17 | Les voisins de l'utilisateur U_1 | 79 |
| 3.18 | Diagramme de cas d'utilisation de notre système de recommandation | 80 |
| 4.1 | Exemple de structuration d'un fichier exploitable dans notre système de recommandation | 85 |
| 4.2 | Capture d'écran de l'onglet <i>Data preprocessing</i> | 86 |
| 4.3 | Capture d'écran de l'onglet <i>Filtering data</i> | 87 |
| 4.4 | Capture d'écran de l'onglet <i>Test without classification</i> | 88 |
| 4.5 | Capture d'écran de l'onglet <i>Classification and optimisation</i> | 89 |
| 4.6 | Architecture de la base de données RED | 92 |
| 4.7 | Évaluation MAE du FC standard en fonction du seuil | 94 |
| 4.8 | Évaluation RMSE du FC standard en fonction du seuil | 94 |
| 4.9 | Évaluation MAE du FC avec classification en fonction du seuil | 96 |
| 4.10 | Évaluation RMSE du FC avec classification en fonction du seuil | 97 |
| 4.11 | Évaluation MAE et RMSE du FSem en fonction du seuil de similarité | 99 |
| 4.12 | Évaluation MAE avec les algorithmes de FSem basés classification | 100 |
| 4.13 | Évaluation RMSE avec les algorithmes de FSem basés classification | 101 |
| 4.14 | Évaluation MAE de l'hybridation sans classification des algorithmes FC et FSem | 103 |
| 4.15 | Évaluation RMSE de l'hybridation sans classification des algorithmes FC et FSem | 103 |
| 4.16 | Évaluation MAE des algorithmes hybrides basés classification | 105 |
| 4.17 | Évaluation RMSE des algorithmes hybrides basés classification | 106 |
| 4.18 | Évaluation MAE du FC avec classification sur RED | 111 |
| 4.19 | Évaluation RMSE du FC avec classification sur RED | 111 |
| 4.20 | Évaluation MAE du FSem avec classification sur RED | 113 |
| 4.21 | Évaluation RMSE du FSem avec classification sur RED | 113 |
| 4.22 | Évaluation MAE du filtrage hybride avec classification sur RED | 115 |
| 4.23 | Évaluation RMSE du filtrage hybride avec classification sur RED | 115 |

Liste des tableaux

| | | |
|------|---|-----|
| 4.1 | Récapitulatif des trois meilleurs résultats (MAE et RMSE) pour chaque FC sans classification | 95 |
| 4.2 | Récapitulatif des trois meilleurs résultats (MAE et RMSE) pour l'algorithme de FC avec classification | 97 |
| 4.3 | Les valeurs de seuils correspondant aux indices du graphes 4.11 | 99 |
| 4.4 | Récapitulatif des trois meilleurs résultats (MAE et RMSE) pour l'algorithme de FSem sans classification | 99 |
| 4.5 | Récapitulatif des trois meilleurs résultats (MAE et RMSE) pour l'algorithme de FSem avec classification | 101 |
| 4.6 | Tableau des différents seuils combiné à Alpha du FHyb pondéré | 104 |
| 4.7 | Récapitulatif des trois meilleurs résultats (MAE et RMSE) pour le FHyb pondéré | 104 |
| 4.8 | Récapitulatif des trois meilleurs résultats (MAE et RMSE) pour les différents FHyb avec classification | 107 |
| 4.9 | Récapitulatif des meilleurs résultats (MAE et RMSE) pour tous les filtrages conçus | 109 |
| 4.10 | Les meilleurs résultats (MAE et RMSE) trouver à partir de [43] pour le dataset MovieLens 100k | 109 |
| 4.11 | Récapitulatif des trois meilleurs résultats (MAE et RMSE) pour le FC avec classification du dataset RED | 112 |
| 4.12 | Récapitulatif des trois meilleurs résultats (MAE et RMSE) pour le FSem avec classification du dataset RED | 114 |
| 4.13 | Récapitulatif des trois meilleurs résultats (MAE et RMSE) pour le FHyb avec classification du dataset RED | 116 |

| | | |
|------|---|-----|
| 4.14 | Récapitulatif des meilleurs résultats (MAE et RMSE) pour les différents fil- trage conçu et testés sur RED | 117 |
| 2 | Résultats de MAE et RMSE pour l'algorithme de FC item-item | 120 |
| 3 | Résultats de MAE et RMSE pour l'algorithme de FC item-item-IA | 120 |
| 4 | Résultats de MAE et RMSE pour l'algorithme de FC item-item-SVD | 120 |
| 5 | Résultats de MAE et RMSE pour l'algorithme de FC item-item-UA | 120 |
| 6 | Résultats de MAE et RMSE pour l'algorithme de FC user-user | 120 |
| 7 | Résultats de MAE et RMSE pour l'algorithme de FC user-user-IA | 120 |
| 8 | Résultats de MAE et RMSE pour l'algorithme de FC user-user-UA | 120 |
| 9 | Résultats de MAE et RMSE pour l'algorithme de FC user-user-SVD | 121 |
| 10 | Résultats de MAE et RMSE pour l'algorithme de FC user-user-Kmedoids . . . | 121 |
| 11 | Résultats de MAE et RMSE pour l'algorithme de FC user-user-KNN | 122 |
| 12 | Résultats de MAE et RMSE pour l'algorithme de FSem user-user | 122 |
| 13 | Résultats de MAE et RMSE pour l'algorithme de FSem user-user-Kmedoids . | 123 |
| 14 | Résultats de MAE et RMSE pour l'algorithme de FHyb-alpha | 124 |
| 15 | Résultats de MAE et RMSE pour l'algorithme de FHyb alpha-Kmedoids . . . | 124 |
| 16 | Résultats de MAE et RMSE pour l'algorithme de FHyb alpha-KNN | 125 |
| 17 | Résultats de MAE et RMSE pour l'algorithme de FHyb FC basé Sem | 125 |
| 18 | Résultats de MAE et RMSE pour l'algorithme de FHyb FC basé Sem et Kme- doids | 126 |
| 19 | Résultats de MAE et RMSE pour l'algorithme de FHyb FC basé Sem et K-NN | 127 |
| 20 | Résultats de MAE et RMSE pour l'algorithme de FHyb Sem basé FC | 127 |
| 21 | Résultats de MAE et RMSE pour l'algorithme de FHyb Sem basé FC et K- medoids | 128 |
| 22 | Résultats de MAE et RMSE pour l'algorithme de FHyb Sem basé FC et K-NN | 129 |
| 23 | Résultats de MAE et RMSE pour l'algorithme de FHyb multivues K-NN . . . | 130 |
| 24 | Résultats de MAE et RMSE pour l'algorithme FC basé K-medoids sur le dataset RED | 131 |
| 25 | Résultats de MAE et RMSE pour l'algorithme FC basé K-NN sur le dataset RED | 132 |
| 26 | Résultats de MAE et RMSE pour l'algorithme FC basé K-medoids et BSO sur le dataset RED | 133 |

| | | |
|----|---|-----|
| 27 | Résultats de MAE et RMSE pour l'algorithme FSem basé K-medoids sur le dataset RED | 133 |
| 28 | Résultats de MAE et RMSE pour l'algorithme FSem basé K-medoids et BSO sur le dataset RED | 134 |
| 29 | Résultats de MAE et RMSE pour l'algorithme FSem basé K-NN sur le dataset RED | 135 |
| 30 | Résultats de MAE et RMSE pour l'algorithme FHyb FC basé Sem et K-medoids sur le dataset RED | 136 |
| 31 | Résultats de MAE et RMSE pour l'algorithme FHyb FC basé Sem et K-medoids-BSO sur le dataset RED | 136 |
| 32 | Résultats de MAE et RMSE pour l'algorithme FHyb FC basé Sem et K-NN sur le dataset RED | 137 |
| 33 | Résultats de MAE et RMSE pour l'algorithme FHyb multivues K-NN sur le dataset RED | 138 |

Introduction générale

Avec le développement des technologies de l'information et de la communication et de l'internet, les moyens et les possibilités d'échanges et de partage de données ont largement évolués. Cependant, le volume de données créé chaque jour est en croissance exponentiel. Devant cette masse importante de données, plusieurs problèmes sont engendrés dont la difficulté à retrouver la bonne information adéquate selon les besoins et préférences des utilisateurs. Par conséquent, l'utilisation des systèmes de recommandation devient une nécessité et un moyen incontournable pour avoir les ressources les plus appropriés à un utilisateur donné. Les systèmes de recommandation sont en expansion continue depuis les années 1990. Ils offrent aux utilisateurs la possibilité d'accéder aux contenus qui leur sont les plus adaptés et appropriés parmi les différents contenus disponibles, de manière personnalisée selon leurs profils leurs préférences et centres d'intérêts. . . . Depuis leur apparition, les systèmes de recommandation ont beaucoup évolué, de la définition d'approches de recommandation simples à la mise en place d'approches hybrides qui combinent plusieurs algorithmes de recommandation afin de gagner en performance et de pallier les limitations des algorithmes standards. Cependant, les systèmes de recommandation ont encore besoin d'être améliorés afin d'augmenter la satisfaction des utilisateurs. Nous pouvons trouver actuellement des systèmes combinant les systèmes classiques avec d'autres informations telles que l'information sémantique ou encore sociale dans les réseaux sociaux par exemple.

Dans ce même contexte, nous nous intéressons dans le cadre de ce projet de fin d'études de master, à la proposition d'une approche de recommandation de contenu en considérant la représentation sémantique. Dans un premier temps, nous nous sommes basées sur la combinaison du filtrage collaboratif (FC), l'un des algorithmes les plus utilisés, avec le filtrage sémantique (FSem), en proposant trois hybridations différentes. Nous avons considéré dans l'hybridation différentes variantes du FC (FC basé utilisateur, FC basé item, FC avec SVD. . .). Puis, dans un second temps, nous avons appliqué deux techniques de classification sur le FC et FSem : classification supervisée (K-NN - K plus proches voisins) et non supervisée (Kmedoids). L'hybridation dans ce cas a été appliquée sur les algorithmes avec classification. Finalement, afin d'améliorer les performances de l'algorithme Kmedoids, nous avons utilisé

la méta-heuristique de colonies d'abeilles (BSO - Bees Swarm Optimization Algorithm).

En se basant sur l'approche proposée qui inclut plusieurs algorithmes FC standard, FC avec classification et optimisation, FSem standard, FSem avec classification et optimisation, filtrage hybride (FHyb) et FHhyb avec classification et optimisation, nous avons implémenté un prototype de système de recommandation implémentant tous ces algorithmes. Sachant que notre système a été implémenté de manière flexible, permettant ainsi une évaluation selon le choix de l'utilisateur. A titre d'exemple, l'utilisateur peut choisir la variante souhaitée du FC et du FSem et construire sur la base de ce choix son hybridation selon les trois hybridations offertes. Il aura aussi la possibilité d'appliquer une classification avec ou sans optimisation, de choisir une hybridation multi-vues ou non.

Après cette introduction, notre mémoire sera structuré comme suit :

- Chapitre1 : présente notre état de l'art en décrivant quelques notions liées au recommandation.
- Chapitre2 : présente une étude sur les méthodes de classification et d'optimisation.
- Chapitre 3 : décrit en détail notre approche de recommandation basée sur les algorithmes de filtrage collaboratif et sémantique et les techniques de classification et d'optimisation.
- Chapitre 4 : présente les résultats des évaluations des différents algorithmes implémentés en utilisant deux bases de données : la base connue MovieLens-100K et la base Epinions enrichie par l'aspect sémantique (RED- Rich Epinions Dataset).

Finalement, nous terminerons par une conclusion générale et quelques perspectives.

Chapitre 1

Recommandation personnalisée

1.1 Introduction

Aujourd'hui internet est devenu omniprésent dans notre vie quotidienne, si nous avons besoin d'informations nous pensons instantanément à effectuer une recherche rapide sur le net pour qu'un grand nombre de résultats s'affiche et nous aide à effectuer notre travail, avoir des conseils et opinions d'autres personnes ou juste pour consulter les nouvelles dans le monde. Toutefois pour sélectionner et afficher un contenu censé être pertinent pour l'utilisateur est un défi en soi, il existe une multitude de ressources informationnelles et leur flot est incessant. L'une des solutions afin de pallier cette problématique consiste à utiliser un système de recommandation qui effectue un filtrage d'informations en se basant généralement sur les préférences de l'utilisateur ou d'essayer de les deviner. Dans ce chapitre nous allons aborder les points suivants :

- Les principales notions liées aux systèmes de recommandation,
- Les différentes techniques de recommandations,
- L'aspect sémantique dans la recommandation,
- La notion de profil utilisateur dans les systèmes de recommandation,
- Quelques travaux liés à la recommandation hybride,
- Et nous terminerons par une conclusion.

1.2 Recommandation

La recommandation est l'action de signaler favorablement, conseiller ou faire valoir les mérites d'un objet ou d'une personne auprès d'une autre personne, sachant que nul ne possède la science infuse, nous essayons de nous entraider en partageant nos informations et notre expérience avec autrui et nous recommandons les choses que nous avons testés et jugés utiles aux autres en attendant qu'ils fassent de même en retour. Par conséquent des liens se forment entre nous et par la suite des communautés se forment afin d'améliorer la qualité des recommandations et la diversifier. Un des meilleurs outils pour la recommandation et le système de recommandation. Nous allons définir et étudier ses aspects afin de comprendre son fonctionnement et par la suite l'implémenté et amélioré.

1.2.1 Les systèmes de recommandations

Les systèmes de recommandation (SR) sont des outils dont l'objectif est de proposer des items pertinents à l'utilisateur. En d'autres termes, ils tentent de prédire pour un utilisateur l'intérêt d'un item. L'item dans ce contexte peut être un produit à acheter, un morceau de musique à écouter, un film à regarder, un livre à lire, une page web à consulter, ou bien autre chose. Afin de pouvoir fournir des recommandations personnalisées, le SR doit connaître les préférences de chaque utilisateur[1]. Les systèmes de recommandation deviennent indispensables dans de nombreux domaines. Ces domaines d'application sont différents mais partagent tous un même problème : ils offrent un choix de possibilités très important aux utilisateurs, chacun d'eux ayant par ailleurs leurs propres préférences. Dès lors, un moteur de recommandation qui permet de proposer à un utilisateur donné, de manière personnalisée, le sous-ensemble d'éléments qui l'intéresse, devient très utile. Il réduit de manière considérable l'effort que doit fournir l'utilisateur pour accéder à ce qui l'intéresse et participe ainsi à sa satisfaction et à sa fidélisation[2].

1.2.2 Le fonctionnement des systèmes de recommandation

L'entité à laquelle la recommandation est fournie est appelée *user*, et le produit recommandé est appelé *item*. Par conséquent, l'analyse des recommandations est souvent basée sur l'interaction précédente entre les users et les items[3]. En général pour faire une recommandation, il est nécessaire de passer par les étapes suivantes :

- La connaissance de l'utilisateur et sa position par rapport aux autres : on tente dans cette étape d'acquérir les informations nécessaires pour construire les profils des utilisateurs, en exploitant les traces laissées explicitement ou implicitement. Les traces explicites sont fournies volontairement par l'utilisateur lors de l'inscription ou par le remplissage d'un formulaire par exemple, et les traces implicites sont collectées en traçant les actions de l'utilisateur pendant la navigation et la recherche d'items. Par la suite ces connaissances sur les utilisateurs sont représentées et stockées dans un modèle ou une matrice nommée *matrice usage*, qui relie l'appréciation de l'utilisateur sur un item, cette matrice est évidemment mise à jour car la préférence d'un item peut changer.
- Générer les listes de recommandations des items : après la construction des profils des

utilisateurs, le SR tente de prédire les relations manquantes entre l'utilisateur et l'item afin de savoir si ce dernier va intéresser l'utilisateur et le lui recommander.

1.2.3 Les caractéristiques d'un SR

Les caractéristiques que doit contenir un bon SR sont[3] :

- Pertinence : l'objectif opérationnel le plus évident d'un système de recommandation est de recommander des éléments qui sont pertinents pour l'utilisateur. Les utilisateurs sont plus susceptibles de consommer des objets qu'ils trouvent intéressants.
- Nouveauté : les systèmes de recommandation sont vraiment utiles lorsque l'élément recommandé est quelque chose que l'utilisateur n'a pas vu auparavant. Par exemple, les films populaires d'un genre préféré seraient rarement nouveaux pour l'utilisateur. La recommandation répétée d'articles populaires peut également entraîner une réduction de la diversité des ventes.
- Sérendipité : les éléments recommandés doivent être quelque peu inattendus, et cela doit être un peu rare et pas souvent utilisé.
- Augmentation de la diversité des recommandations : les systèmes de recommandation suggèrent généralement une liste des principaux items. Lorsque tous les items recommandés sont très similaires, cela augmente le risque que l'utilisateur n'apprécie aucun.

1.3 Techniques de recommandation

Étant donné le nombre de données que le SR doit traiter afin de fournir l'information adéquate aux personnes qui le désirent, il est nécessaire de lui implémenter certaines techniques de filtrage d'information. Le filtrage est souvent interprété comme l'élimination de données indésirables sur un flux entrant, plutôt que la recherche de données spécifiques sur ce flux [4]. Il existe plusieurs classifications des systèmes de recommandations (voir la figure 1.1). Nous avons choisi de citer et détailler trois types de filtrage selon la classification classique [5] qui est reconnue par trois types de filtrage :

- Filtrage cognitif : calcule la similarité entre les items pour prédire si un item sera jugé pertinent par un utilisateur sachant qu'il a déjà noté un item qui lui est similaire.
- Filtrage collaboratif : basé sur les évaluations collectives des utilisateurs sur les items.

- Filtrage hybride : combine deux types ou plusieurs types de filtrages.

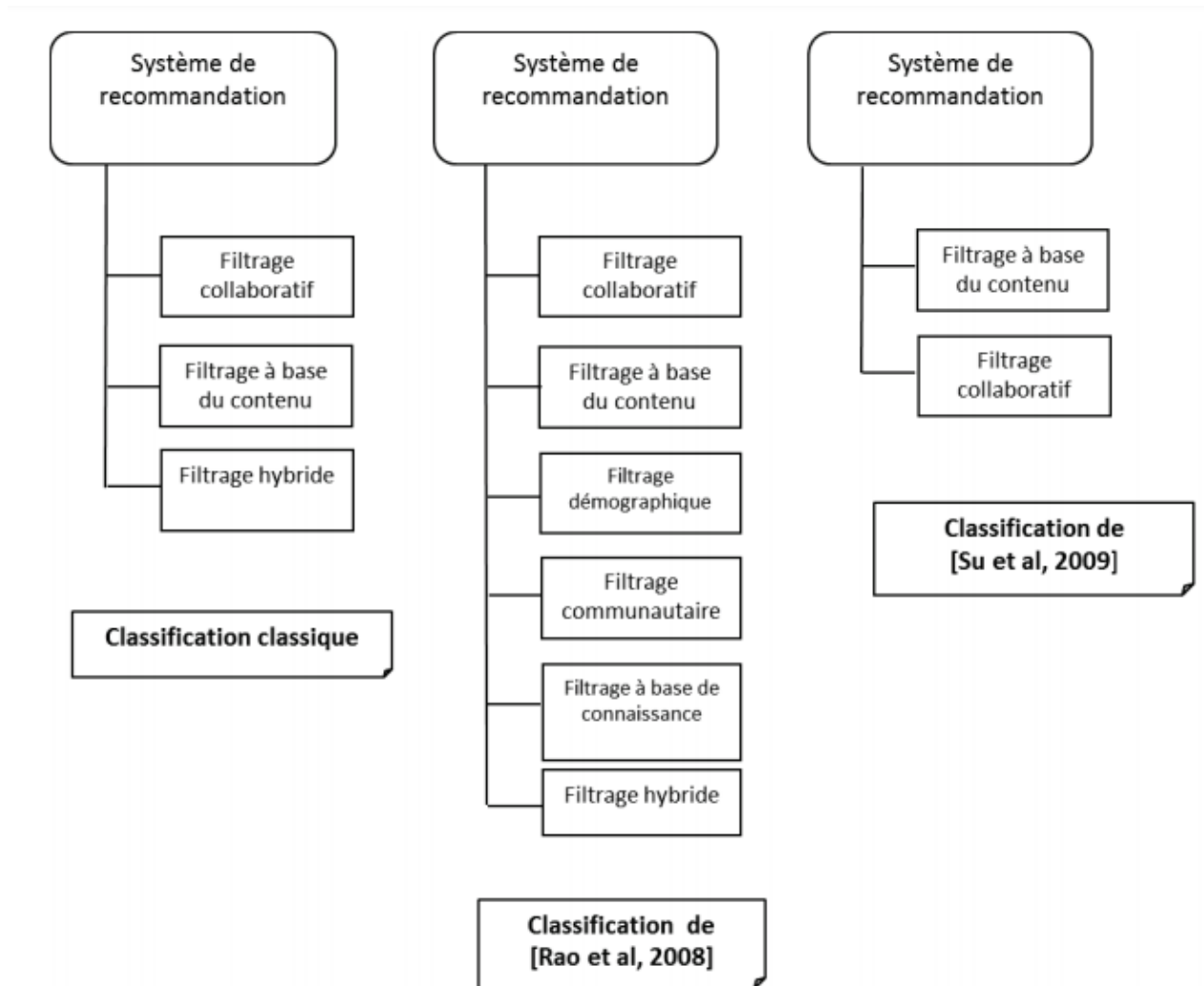


FIGURE 1.1 – classification principale des système de recommandation de [5]

1.3.1 Filtrage cognitif

Dans les systèmes de recommandation basées sur le filtrage cognitif (appelé aussi filtrage basé sur le contenu), le terme « contenu » fait référence aux descriptions des items. Les systèmes de recommandation basées sur le contenu font correspondre les utilisateurs à des items similaires à ceux déjà aimés dans le passé. Selon [3], en général les SR dépendent de deux sources de données :

- La première source de données est une description de divers items. Un exemple d'une telle représentation pourrait être la description textuelle d'un article par le fabricant.

- La deuxième source de données est un profil d'utilisateur, généré à partir des *feedbacks* des utilisateurs sur les divers items, qui peuvent être explicites ou implicites. Les feedback explicites peuvent correspondre à des évaluations, tandis que les feedbacks implicites peuvent correspondre à des actions de l'utilisateur tel qu'achat d'article ou historique de recherche.

Après l'acquisition de ces deux sources de données, le filtrage cognitif s'effectue selon trois étapes [3] :

1. Prétraitement et Extraction de caractéristique des items : les systèmes basés sur le contenu utilisent des descriptifs de type textuel pour les items il faut donc prétraiter ce texte (stemming, suppression de mots vides, extraction de phrase ...) et le convertir en une représentation d'espace vectoriel basée sur des mots-clés ou modèle sémantique à base d'ontologies, qui sont une représentation structurée et facilement exploitable par le système. Il s'agit de la première étape de tout système de recommandation basé sur le contenu et elle est hautement spécifique à un domaine (par exemple : un livre a une représentation différente d'une voiture en vue des propriétés que ces 2 objets possèdent).
2. Apprentissage basé sur le contenu des profils d'utilisateurs : un modèle spécifique est construit pour chaque utilisateur en fonction de leurs antécédents d'achat ou d'évaluation des items, ces 2 derniers sont combinés pour réaliser un profil d'utilisateur qui établit une relation conceptuelle entre les intérêts de l'utilisateur et les attributs des items.
3. Filtrage et recommandation : dans cette étape, le modèle appris de l'étape précédente (profil utilisateur) et les caractéristiques des items sous forme vectoriel sont utilisés pour formuler des recommandations en calculant une certaine similarité entre les deux, cette similarité permet de définir les items qui sont susceptible d'intéresser l'utilisateur.

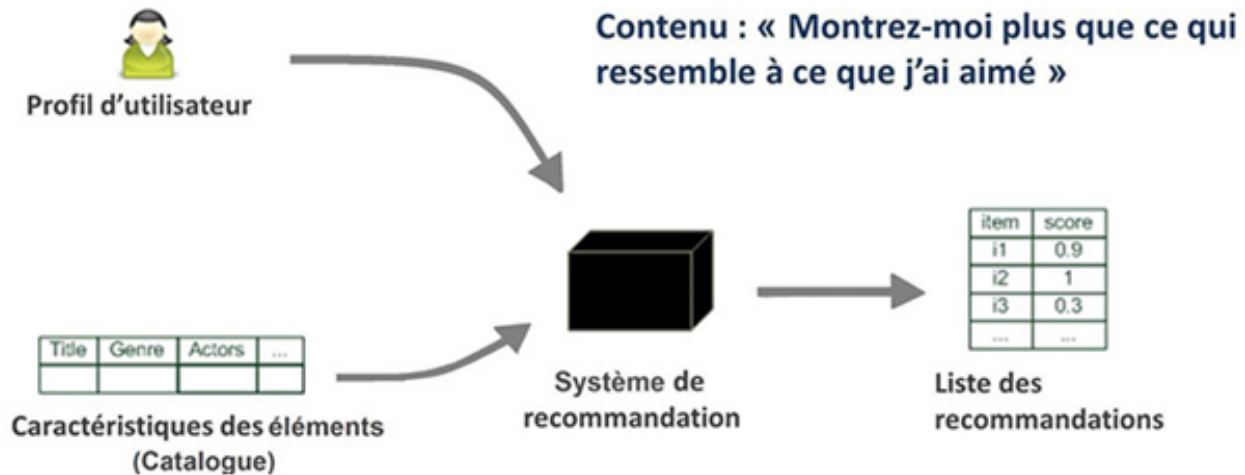


FIGURE 1.2 – SR basé sur le contenu [6]

Avantages et inconvénients du filtrage cognitif

- Avantages
 - Pas besoin d'une communauté d'utilisateurs pour pouvoir effectuer des recommandations aux utilisateurs seul leurs profils suffisent afin de proposer des articles adéquats à leurs besoins.
 - Si de nouveaux items sont ajoutés dans la base de données ou s'ils ne sont pas populaires ils peuvent être recommandés
 - Transparence : la méthode basée sur le contenu peut nous dire sur quoi elle se base pour suggérer des items.
 - Pas de démarrage à froid pour les items : contrairement au filtrage collaboratif, de nouveaux éléments peuvent être suggérés avant d'être notés par un nombre important d'utilisateurs.
- Inconvénients
 - Les systèmes basés sur le contenu ont tendance à rechercher des éléments similaires à ceux que l'utilisateur a vus jusqu'à présent. Ce problème est appelé sur-spécialisation. Il est toujours souhaitable d'avoir une certaine quantité de nouveauté dans les recommandations. [3]

- Même si les systèmes basés sur le contenu aident à résoudre le problème de démarrage à froid des nouveaux items, ils ne permettent pas de résoudre ce problème pour les nouveaux utilisateurs, lorsqu'il n'y a pas assez d'informations pour créer un profil solide, la recommandation ne peut pas être fournie correctement (Démarrage à froid pour les utilisateurs). [7]
- Si la description textuelle des items est insuffisante ou inexistante (cas item de type multimédia) la recommandation ne sera pas précise ou ne s'effectuera pas, ce qui peut souvent se produire .[7]
- Filtrage basé sur la similarité item profil utilisateur donc absence d'autres facteurs comme la qualité de l'item fourni par d'autre utilisateur, le public visé, etc.

1.3.2 Filtrage collaboratif

Les modèles de filtrage collaboratif utilisent la puissance collaborative des évaluations fournies par plusieurs utilisateurs pour formuler des recommandations, ce type de SR est le plus répondu car il se base sur ce que d'autres utilisateurs ont déjà évalué et apprécié, et cela fait resurgir dans les tendances des articles qui sont bien évalués par la majorité des utilisateurs.

Par exemple, considérons deux utilisateurs A et B, qui ont des goûts très similaires, si les évaluations, que les deux ont spécifiées, sont très similaires, leur similarité peut être identifiée et il est très probable que les évaluations dans lesquelles l'un des deux utilisateurs a spécifié une seule valeur sont également susceptibles d'être similaires. Cette similarité peut être utilisée pour déduire des évaluations non spécifiées de A et B pour des items.

Cette approche peut être appliquée à tout type de documents textuels et multimédias car contrairement au filtrage cognitif, elle ne se base pas sur le contenu des items, mais principalement sur l'exploitation des évaluations que les utilisateurs ont effectué afin de recommander ces mêmes items à d'autres utilisateurs [8].

Il existe deux types de méthodes couramment utilisées dans le filtrage collaboratif, appelées méthodes basées sur la mémoire et méthodes basées sur un modèle [3], le choix de l'approche à utiliser dépendant des informations prises en compte lors du calcul de la prédiction.

Les deux méthodes partagent les points communs suivants :

- Les deux utilisent un calcul de similarité entre les lignes et colonnes d'une matrice appelée « matrice d'usage».
- Les deux s'effectuent en deux étapes qui sont le calcul de similarité et la prédiction, que nous allons détailler par la suite.

1) Filtrage collaboratif basé mémoire

Les méthodes basées mémoire sont également appelées algorithmes de filtrage collaboratif basés sur le voisinage. C'étaient parmi les premiers algorithmes de filtrage collaboratif, dans lesquels les évaluations des utilisateur sont prédites sur la base de leurs voisinages[3].

Le voisinage peut être défini de deux manières :

- Filtrage collaboratif basé sur l'utilisateur : l'idée de base est de déterminer les utilisateurs qui sont similaires à l'utilisateur ciblé A (les voisins de A) en comparant les lignes de la matrice d'usage pour prédire l'évaluation que va donner A, en se basant sur le vote de ses voisins sur le même item. Des fonctions de similarité sont calculées entre les lignes de la matrice d'usages pour détecter des utilisateurs similaires.

| | i_1 | i_2 | i_3 | i_4 | i_5 |
|-------|-------|-------|-------|-------|-------|
| u_1 | | 2 | 4 | 3 | 1 |
| u_2 | 3 | 1 | | | |
| u_3 | 3 | | 5 | | |
| u_4 | | 4 | 2 | 3 | 1 |
| u_5 | | 4 | 3 | 1 | 4 |
| u_6 | | 3 | 3 | 1 | 4 |

FIGURE 1.3 – Matrice d'usage

- Filtrage collaboratif basé sur les items : le même processus que le FC basé utilisateur est appliqué ici, sauf que dans ce cas la similarité est calculée entre les items et donc entre les colonnes de la matrice d'usage pour détecter les items similaires et les proposer aux utilisateurs qui ont déjà apprécié un item semblable.

| | i_1 | i_2 | i_3 | i_4 | i_5 |
|-------|-------|-------|-------|-------|-------|
| u_1 | | 2 | 4 | 3 | 1 |
| u_2 | 3 | 1 | | | |
| u_3 | 3 | | 5 | | |
| u_4 | | 4 | 2 | 3 | 1 |
| u_5 | | 4 | 3 | 1 | 4 |
| u_6 | | 3 | | 1 | 4 |

FIGURE 1.4 – Matrice d'usage

- Le calcul de similarité :

Comme vu précédemment nous avons beaucoup cité les termes « similarité » et « calcul de similarité » afin de déterminer le voisinage des utilisateurs et des items, en effet cette étape est jugée très importante dans les SR basé sur le filtrage collaboratif. Dans cette partie nous allons citer les mesures les plus utilisées dans le FC.

1. Calcul similarité utilisateur :

- Corrélation de Pearson (COR) :

Pour le calcul de similarité entre deux utilisateurs u et v , la corrélation de Pearson est donnée par la formule suivante :

$$pearson(u, v) = \frac{\sum_{i \in I} (r_{u,i} - \bar{r}_u) \cdot (r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i \in I} (r_{u,i} - \bar{r}_u)^2 \cdot (r_{v,i} - \bar{r}_v)^2}} \quad (1.1)$$

où :

- * $r_{u,i}$: est l'estimation de l'utilisateur u sur l'item i .
- * $r_{v,i}$: est l'estimation de l'utilisateur v sur l'item i .
- * \bar{r}_u : est la moyenne de toutes les notes de l'utilisateur u .
- * \bar{r}_v : est la moyenne de toutes les notes de l'utilisateur v .

- Cosine (COS) :

Le cosinus entre les vecteurs, est une méthode pour calculer le poids de l'utilisateur u_i par rapport à l'utilisateur actif u_j , il est calculé comme un cosinus entre les vecteurs formés par les évaluations des utilisateurs, comme suit :

$$cosine(u_i, u_j) = \frac{\sum_{k=1}^n r_{u_i,k} \cdot r_{u_j,k}}{\sqrt{\sum_{k=1}^n r_{u_i,k}^2 \cdot \sum_{k=1}^n r_{u_j,k}^2}} \quad (1.2)$$

où :

- * $r_{u_i,k}$: est l'évaluation de l'utilisateur u_i sur l'item i_k
- * $r_{u_j,k}$: est l'évaluation de l'utilisateur u_j sur l'item i_k
- * n : est le nombre d'item dans le système.

- Calcul similarité items :

Le calcul de similarité entre deux items i et j : la corrélation de Pearson est donnée par la formule suivante :

$$pearson(i, j) = \frac{\sum_{u \in U} (r_{u,i} - \bar{r}_i) \cdot (r_{u,j} - \bar{r}_j)}{\sqrt{\sum_{u \in U} (r_{u,i} - \bar{r}_i)^2 \cdot \sum_{u \in U} (r_{u,j} - \bar{r}_j)^2}} \quad (1.3)$$

où :

- * $r_{u,i}$: est l'évaluation de l'utilisateur u sur l'item i
- * $r_{u,j}$: est l'évaluation de l'utilisateur u sur l'item j
- * \bar{r}_i : est la moyenne des évaluations de l'item i par les utilisateurs.
- * \bar{r}_j : : est la moyenne des évaluations de l'item j par les utilisateurs.

2. Calcul des prédictions :

Le calcul de prédiction est une phase importante qui consiste à calculer des prédictions pour générer des recommandations intéressantes à un utilisateur. La plus utilisée pour le calcul de prédictions est la somme pondérée qui considère les plus proches voisins de u_i (phase 1) ayant déjà noté l'item i_k , pour calculer la prédiction de la note de l'utilisateur u_i sur i_k [9].

$$pred(u_i, i_k) = r(\bar{u}_i) + \frac{\sum_{u_j \in U_i} sim(u_i, u_j) \cdot (r_{u_j, i_k} - r(\bar{u}_j))}{\sum_{u_j \in U_i} sim(u_i, u_j)} \quad (1.4)$$

Où :

- $sim(u_i, u_j)$: est la mesure de similarité entre un utilisateur u_i et son voisin u_j , tel que $u_j \in U_i$.

2) Filtrage collaboratif basé modèle

Le filtrage collaboratif basé modèle consiste à élaborer des modèles sur lesquels la prédiction va se baser à partir d'une base de données contenant les évaluations faites par les utilisateurs sur les items et une méthode pour la construction du modèle. Il existe plusieurs méthodes pour la construction du modèle, les plus souvent utilisés dans le FC sont le clustering et les réseaux bayésien. Le modèle à base de cluster repose sur le principe que certains groupes ou type d'utilisateur capturent un ensemble commun de préférences et de goûts. Étant donné un tel groupe, les préférences concernant les différents items (sous la forme d'évaluations) sont indépendantes.

Le modèle à base de réseaux bayésien associe un nœud à chaque item. Les états pour chaque nœud correspondent aux valeurs d'évaluations possible pour chaque item. On inclut également un état correspondant à l'absence d'évaluations pour les domaines où il n'y a pas d'interprétation. On peut alors appliquer un algorithme d'apprentissage de réseau bayésien sur la base d'exemple, où les évaluations manquantes sont associées à une valeur « parent » qui sont les meilleures prédictions de ses évaluations[10].

Il existe aussi plusieurs modèle appliqué au filtrage collaboratif pour faire de la recommandation qui sont issues des domaines de recherche sur l'intelligence artificiel comme les réseau de neurone ou les processus décisionnels de Markov, cependant ces techniques font face à certaines problématique car elle sont sensible à l'arrivée de nouveaux utilisateurs ou insertion d'un nouveau item, la phase d'apprentissage sera ainsi ré-effectuée au fur et à mesure des mise à jour et connaissant la forte complexité de ces algorithmes elle peut s'avérer très coûteuse en temps et en ressource[36].

Avantages et inconvénients du filtrage collaboratif [11] :

- Avantages
 - Le filtrage collaboratif se base principalement sur les évaluations des utilisateurs et pas la thématique du contenu à recommander. Ce type de filtrage résout les problèmes liés au filtrage cognitif et donc il permet de filtrer tout type d'information (textuel, multimédia, ressources physiques, image, etc.). L'efficacité du système augmente donc en fonction du nombre d'utilisateurs.
 - Absence de l'effet entonnoir, car si un item est évalué, il peut alors être recommandé ce qui permet à l'utilisateur de découvrir divers domaines intéressants auxquels il n'avait pas pensé.
 - Possibilité de prendre en considération d'autres facteurs et critères des items tels que la qualité de l'information, le public visé, etc.
- Inconvénients
 - Démarrage à froid : toute nouvelle ressource ne peut être recommandée car elle n'a pas encore été évaluée. De même, pour un nouvel utilisateur, le système ne peut pas lui recommander une ressource car tant qu'il n'a pas effectué d'évaluations il ne pourra pas faire partie d'une communauté d'utilisateurs.

- Pour former de meilleures communautés, le système exige un nombre suffisant d'évaluations en commun entre les utilisateurs pour pouvoir les rapprocher (i.e. décider s'ils appartiennent au même voisinage).

1.3.3 Filtrage hybride

Le filtrage hybride combine les deux algorithmes précédent (et/ou d'autres algorithmes), selon les méthodes suivantes :[12] :

- Pondérée : Les résultats pondérés de plusieurs techniques de recommandation sont combinés pour produire une nouvelle recommandation.
- Permutation : Le système permute entre les différentes techniques de recommandation selon le résultat de la recommandation.
- Mixte : Les recommandations de plusieurs techniques sont présentées en même temps
- Combinaison : Différentes techniques de recommandation sont combinées en un unique algorithme de recommandation
- En cascade : Un système de recommandation raffine les résultats fournis par un autre système.
- Augmentation : Le résultat "output" d'une technique de recommandation est utilisé comme données en entrée "input" pour l'autre technique.
- Méta-niveau : Le modèle appris par une technique de recommandation est utilisé comme données en entrée pour l'autre technique.

1.4 Autre Système de recommandation

Étant donné que les filtres présentés précédemment ne prennent pas en compte l'aspect sémantique existant dans la description des items ou dans les besoins récoltés des évaluations des utilisateurs, nous expliquons dans ce qui suit le fonctionnement du filtrage sémantique et les mesures similarité qu'il utilise.

1.4.1 Filtrage sémantique

Afin de surpasser les limitations du filtrage collaboratif et obtenir de meilleurs résultats, nous proposons une amélioration en combinant ce dernier avec le filtrage sémantique.

La notion de sémantique qui est représentée par les ontologie va nous permettre d'établir une relation entre les utilisateurs et les items, et relier ces derniers entre eux. Ces relations permettent d'avoir un filtrage plus précis.

Définitions d'ontologie

D'après [13], une ontologie est une spécification explicite d'une conceptualisation. Plus concrètement, une ontologie est une description explicite des concepts et les relations entre ces derniers. Dans une ontologie, les concepts sont représentés par une classe décrite par un titre et un ensemble d'attributs. Une classe peut être une sous-classe d'une autre. L'ensemble des classes et les relations entre eux constitue une ontologie.

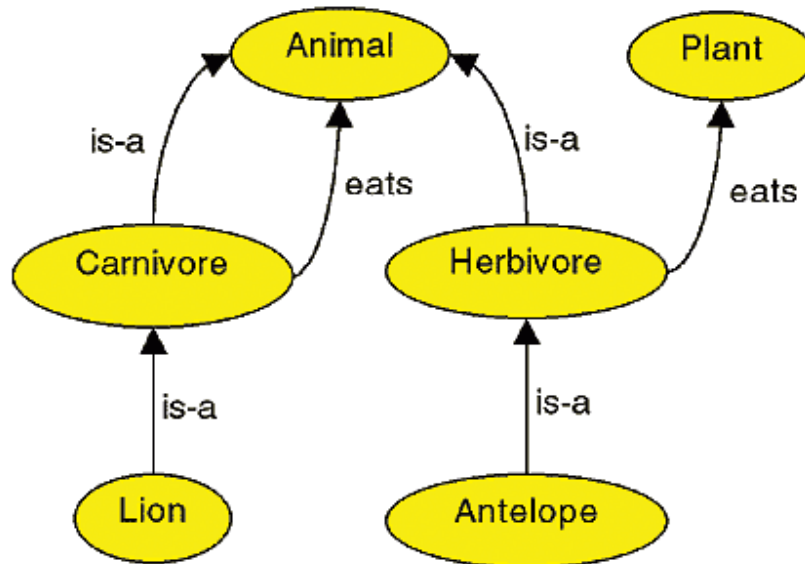


FIGURE 1.5 – Une ontologie du concept animal [29]

Systèmes de recommandation avec modèles sémantiques à base d'ontologie

Dans le domaine du web sémantique, les recherches qui tentent d'améliorer les systèmes de recommandations ont connu une croissance, cette approche construit une hiérarchie des catégories (arborescence des concepts) dont les items sont les feuilles. Elle construit des informations sémantiques sur ces derniers et les profils des utilisateurs afin de représenter les relations entre eux, et entre les items et les utilisateurs.

Méthodes de mesures de similarités sémantiques

Il y a plusieurs méthodes du calcul de similarité sémantique, peuvent être divisées en

quatre catégories majeurs d'après [14] :

1. Comptage des arcs entre deux concepts (*Edge Counting*).
2. Approche basée sur le contenu informationnel (*Information Content*).
3. Méthodes basées sur les fonctionnalités. (*Feature-Based*)
4. Méthodes hybrides.

Dans notre travail nous nous intéressons à la méthode de comptage d'arc, de ce fait nous allons la décrire dans ce qui suit.

Le comptage des arcs entre deux concepts

C'est une méthode qui utilise les ontologies sous forme d'arborescence. Elle consiste à calculer la distance entre deux concepts (le nombre des arcs qui séparent ces derniers). Ce calcul dépend très fortement de la profondeur dans laquelle se trouvent les deux concepts en question, car la similarité entre les concepts spécifiques est bien plus grande que celle des concepts généraux (plus deux concepts sémantiquement proches sont profonds dans l'arborescence plus leur similarité augmente). Il existe plusieurs techniques de comptage d'arcs, parmi ces techniques :

- Rada : Les mesures de similarité sémantique basées sur les arcs ont été introduites par [15]. Elles ont été définies en fonction de la distance qui sépare deux concepts. La mesure est donnée par la formule suivante :

$$Sim_{con} = \frac{1}{1 + dist(c1, c2)} = \frac{1}{1 + N1 + N2} \quad (1.5)$$

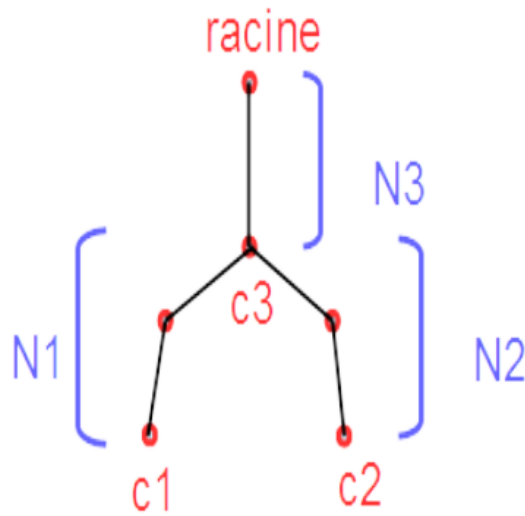


FIGURE 1.6 – Exemple de taxonomie pour les mesures de similarité basées sur les arcs

- Wu et Palmer : Soit C1 et C2 deux concepts dont nous allons calculer la similarité. Le principe de calcul de similarité est basé sur la distance (N1 et N2) séparant C1 et C2 du nœud racine et la distance (N) séparant l'ancêtre commun le plus proche (CS) de C1 et C2 du nœud R (voir figure 1.7). La mesure de similarité de [38] est définie par la formule suivante :

$$Dist_{WP} = \frac{(2 * N)}{(N_1 + N_2 + 2 * N)} \quad (1.6)$$

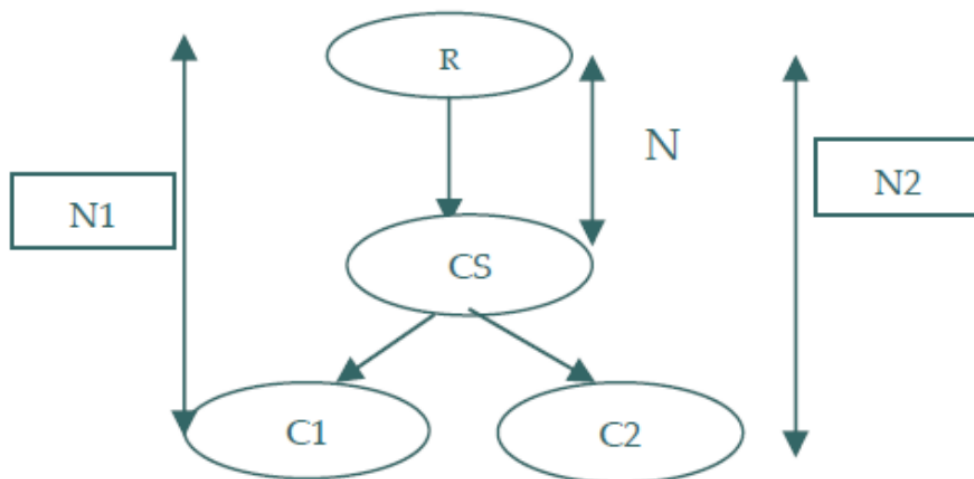


FIGURE 1.7 – Exemple de hiérarchie de concepts [39]

1.5 Construction du profil utilisateur

L'utilisateur est au centre des systèmes de recommandation, il est à la fois consommateur et contributeur. Ainsi, la bonne qualité des informations sur les utilisateurs est une condition nécessaire pour l'efficacité des systèmes de recommandation. Ces informations sont stockées dans le profil utilisateur, qui se définit généralement comme une structure permettant de stocker et modéliser les préférences des utilisateurs et leurs centres d'intérêts[20].

1.5.1 Caractéristiques du profil utilisateur

Le profil utilisateur peut être décomposé en deux catégories [21] :

1. **Caractéristiques explicites** Les caractéristiques explicites sont généralement objectives, elles se basent sur les informations fournies par l'utilisateur directement, indiquant ses centres d'intérêts, ses connaissances, ses préférences, ses objectifs, etc. Exemple : demander à un utilisateur de commenter, taguer, aimer (*liker*), ajouter comme favoris des contenus qui l'intéressent. On utilise souvent une échelle d'évaluations allant d'une étoile (je n'aime pas du tout) à cinq étoiles (j'aime beaucoup), qui sont ensuite transformées en valeurs numériques afin de pouvoir être utilisées par les algorithmes de recommandation [22].
2. **Caractéristiques implicites** : Les caractéristiques implicites sont généralement subjectives et difficiles à capter. Mais, contrairement aux caractéristiques explicites, elles contiennent plus d'information sur les besoins de l'utilisateur. Ces caractéristiques sont détectées à travers des comportements observables recueillis par le système lorsque l'utilisateur interagit avec son environnement [23].
3. Elles sont inférées à partir de :
 - Activités de navigation.
 - Évaluations, annotations (étiquetages, tags).
 - Analyse du réseau social de l'utilisateur.
 - Liste des éléments que l'utilisateur a écoutés, regardés ou achetés en ligne.
 - Documents et pages web consultés et le temps passé sur chaque page, etc.
 - Clique de la souris sur une page (document).

- Enregistrement/impression d'un document.

L'avantage d'utiliser des techniques implicites pour la construction du profil de l'utilisateur est que ce dernier est allégé de certaines actions (définition de ses préférences, se son caractère, etc.) [11].

1.5.2 Modèle de représentation du profil de l'utilisateur

Modéliser l'utilisateur, ses centres d'intérêts, ses préférences et son besoin d'information est une tâche très importante dans les systèmes de recommandation. Pour cela, il faut tout d'abord définir la structure du profil qui nous permettra de stocker toutes les informations qui le concernent. Dans cette section, nous allons décrire les modèles les plus répandus dans la littérature pour représenter et structurer les profils des utilisateurs.

1. Modèle vectoriel :

C'est le modèle basique d'espace vectoriel de [24]. Dans cette représentation, le contenu du profil utilisateur est caractérisé par un ou plusieurs vecteurs de termes pondérés. Ces termes sont obtenus à partir de plusieurs sources d'information recueillies sur l'utilisateur.

Selon [25] la plupart des systèmes de recommandation utilisent le modèle d'espace vectoriel (MEV), qui est une représentation spatiale des documents textuels où chaque document est caractérisé par un vecteur de poids sur des termes appartenant à l'ensemble des termes d'une collection de documents. La pondération des termes est généralement basée sur le format TF-IDF (*term frequency-inverse document frequency*). Le poids associé à chaque terme représente son degré d'importance dans le profil utilisateur.

- $D = (d_1, d_2, \dots, d_N)$: collection de documents.
- $T = (t_1, t_2, \dots, t_N)$: l'ensemble de tous les termes appartenant à la collection de document D .

Chaque document d_j figurant dans les préférences de l'utilisateur sera donc représenté dans son profil par un vecteur de dimension n tel que, $d_j = \{ w_{1j}, w_{2j}, \dots, w_{nj} \}$, ou w_{kj} est le poids associé au terme t_k dans le document.

1.5.2.1 Avantages et inconvénients de ce modèle [27]

- Avantages : Le modèle vectoriel a l'originalité d'être simple à mettre en œuvre, et la prise en compte de plusieurs centres d'intérêts de l'utilisateur en utilisant plusieurs vecteurs.
- Inconvénients : pas d'ordonnement entre les préférences et les centres d'intérêts des utilisateurs, et pas de considération de l'aspect sémantique (i.e. pas de liens entre les termes).

2. Modèle sémantique à base d'ontologies : Les ontologies sont utilisées pour représenter les relations sémantiques entre les unités d'informations constituant le profil utilisateur. Dans ce modèle, le profil utilisateur est vu comme une hiérarchie de concepts pondérés [26], où chaque nœud dans la hiérarchie est un concept associé à un poids représentant l'intérêt porté par l'utilisateur sur ce concept. Le contenu de chaque concept peut être souvent représenté par un vecteur défini dans un espace de termes pondérés. La représentation du profil utilisateur avec le modèle sémantique à base d'ontologie aide à mieux connaître les intérêts des utilisateurs par rapport au modèle vectoriel. De plus la valeur d'intérêt d'un concept peut être propagée vers les autres concepts sémantiquement liés, dans le but de trouver de nouveaux centres d'intérêts.

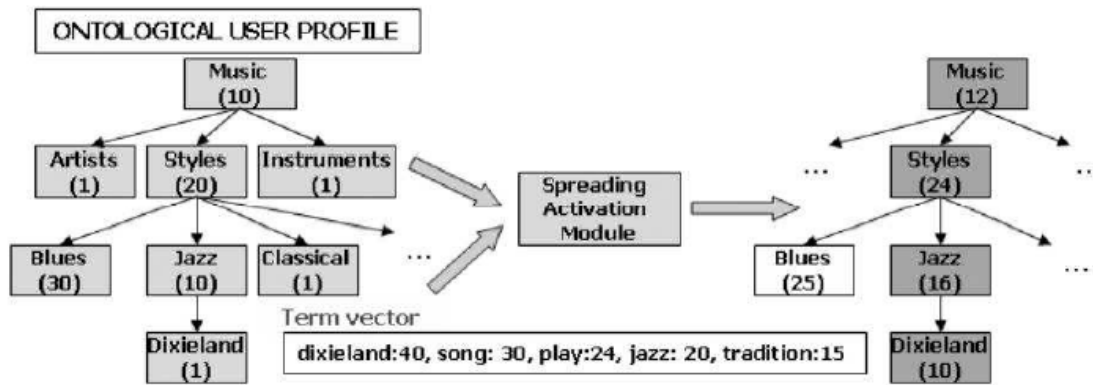


FIGURE 1.8 – Exemple de profil utilisateur représenté par le modèle d'ontologies avec le processus de mise à jour des poids des concepts [28]

1.6 Travaux liés

Dans cette partie nous nous intéressons aux travaux existants (Sémantique, hybridation entre sémantique et collaboratif), afin de proposer par la suite une contribution et la comparé à ces travaux. Nous allons citer deux travaux liés et qui sont :

1.6.1 Recommandation hybride par Filtrage sémantique et FC multicritères

- Le travail présenté par Shambour et Lu [37] propose une hybridation du filtrage sémantique avec le FC multicritères.
- Problématique : Le Filtrage collaboratif (CF) est probablement la technique la plus populaire dans les systèmes de recommandation. Malgré son succès dans diverses applications, les techniques basées sur les FC rencontre une limitation majeure, qui est le démarrage à froid. Par conséquent les chercheurs ont proposé une approche hybride multi-critères sémantiquement renforcée du filtrage FC (MC-SeCF) afin de palier à cette problématique.[37].
- Résultat obtenu : Les résultats expérimentaux vérifient l'efficacité de l'approche hybride pour atténuer le problème de faible densité de données et de démarrage à froid en obtenant une meilleure précision et une plus grande couverture dans les cas du démarrage à froid et l'arrivée de nouvelles données, que les algorithmes de recommandation FC basés sur les items [37].

1.6.2 Un système de recommandation basé sur une clustering multivues de similarité et de confiance

- Le travail présenté par Gu et al. [30] propose un clustering Multivues, une vue similarité et une vue basée sur la confiance.
- Problématique : Bien que le clustering a démontré qu'il est efficace et évolutif pour des ensembles de données à grande échelle, les système de recommandation qui se base sur le clustering souffrent d'une précision et d'une couverture relativement faibles. Pour résoudre ces problèmes, les chercheurs ont développé une méthode de clustering multivues par laquelle les utilisateurs sont mis dans les clusters de manière itérative à

partir des deux vues de similarité et de confiance ente les utilisateurs[30].

- Résultat obtenu : Les résultats expérimentaux ont été effectuer sur trois ensembles de données du monde réel, et démontrent que leurs approche peut effectivement améliorer à la fois l’exactitude et la couverture des recommandations[30].

1.7 Conclusion

Dans ce chapitre nous avons présenté la recommandation, son utilité et les notions liées à cette dernière, suivie des techniques de recommandation les plus utilisés : le filtrage cognitif, collaboratif et hybride, nous avons vu leurs fonctionnements, avantages et inconvénients. La représentation des profils d’utilisateur est une étape importante dans les SR, et elle est souvent effectuée à l’aide de deux modèles : modèle vectoriel et le modèle basé sur les ontologies. Les techniques de filtrage vu précédemment sont très répondu et utilisé, et peuvent t’être améliorées et optimisées en utilisant d’autres technique issue de l’intelligence artificielle comme le clustering et métaheuristique, que nous allons présenter dans le prochain chapitre.

Chapitre 2

Classification et optimisation

2.1 Introduction

Depuis la naissance de l'internet et les réseaux sociaux, le volume des données digitales a connu une croissance rapide et exponentielle. Ces données sont principalement sous forme brute, et sans aucune structure définie avec des volumes de Teras Octets, d'où la nécessité des nouvelles méthodes optimisées pour la classification de ces données et extraire les informations pertinentes. Avec l'IA (Intelligence Artificielle), le monde de la recherche travaille pour introduire et développer de plus en plus ces méthodes pour atteindre de meilleurs résultats.

Dans ce chapitre, nous allons introduire quelques méthodes de classifications et optimisations dont on s'est intéressé pour effectuer notre travail.

La structuration du chapitre est la suivante :

- Classification supervisé et non supervisée,
- Optimisation avec métaheuristiques,
- Optimisation dans les système de recommandation,
- Conclusion.

2.2 classification

La classification est une répartition des êtres vivants, des objets, ou des notions sur plusieurs classes. C'est un processus utilisé dans la science en particulier afin d'effectuer des études et des analyses. En biologie par exemple, la classification est utilisée pour catégoriser les animaux, les végétaux, et plusieurs micro-organismes.

La classification en informatique est un processus qui prend un ensemble de données brutes, et essaye de créer un modèle de classification, qui les répartit d'une façon correcte. Il existe plusieurs types de modèles de classification et qui sont divisés en deux catégories :

2.2.1 Modèles supervisés

Un modèle de classification supervisé est construit en deux étapes :

1. L'apprentissage : dans cette étape on introduit en entrée au modèle un ensemble de données dont les classes sont connues au préalable, et à chaque fois qu'on lui envoie

une nouvelle instance pour qu'il effectue une prédiction de sa classe, on mesure la performance du modèle afin de l'ajuster, on répète ce processus jusqu'à ce qu'il commence à reconnaître les classes des données correctement.

2. Le test : après la fin de l'étape d'apprentissage, on introduit au modèle un ensemble de données qu'il n'a jamais traité auparavant, une prédiction des classes des données est effectuée ensuite une mesure des performances. Quand le modèle reconnaît parfaitement les données qu'on lui donne ou s'il possède un taux d'erreur qui est considéré comme négligeable, on peut dire que le modèle est prêt pour l'utilisation commerciale.

Dans un tel modèle, les classes et leurs nombres sont connus préalablement, donc le modèle ne va jamais reconnaître une instance ayant une nouvelle classe.

Nous nous intéressons dans notre travail au modèle K -NN, de ce fait nous allons le définir et expliquer son fonctionnement.

K-NN :

La méthode des K plus proches voisins est une méthode d'apprentissage supervisé. En abrégé K-NN, de l'anglais *k-nearest neighbors*, inventé par Marcello Pelillo en 1967. Elle peut être utilisée aussi bien pour la régression que pour la classification. Son fonctionnement peut être assimilé à l'analogie suivante "dis moi qui sont tes voisins, je te dirais qui tu es". Pour effectuer une prédiction, l'algorithme K-NN va se baser sur le jeu de données en entier, ceci peut être considéré comme l'ensemble d'entraînement pour l'algorithme, bien qu'un entraînement explicite ne soit pas particulièrement requis. En effet, pour une observation, qui ne fait pas parti du jeu de données, qu'on souhaite prédire, l'algorithme va chercher les K instances du jeu de données les plus proches de notre observation. Ensuite pour ces K voisins, l'algorithme se basera sur leurs variables de sortie (*output variable*) pour calculer la valeur de la variable de l'observation qu'on souhaite prédire.

Par ailleurs :

- Si K-NN est utilisé pour la régression, c'est la moyenne (ou la médiane) des variables des K plus proches observations qui servira pour la prédiction
- Si K-NN est utilisé pour la classification, c'est le mode des variables des K plus proches observations qui servira pour la prédiction[33].

Algorithm 1 K-NN

Entrée : D : Dataset, K : nombre de voisins à considérer ;

Sortie : K clusters ;

Début :

Pour une nouvelle observation X dont on veut prédire sa variable de sortie y **faire** :

 Calculer toutes les distances de cette observation X avec les autres observations du jeu de données D ;

 Retenir les K observations du jeu de données D les proches de X en utilisation la fonction de calcul de distance d ;

 Prendre les valeurs de y des K observations retenues :

 Si on effectue une régression, calculer la moyenne (ou la médiane) de y retenues ;

 Si on effectue une classification , calculer le mode de y retenues ;

 Retourner la valeur calculée dans l'étape 3 comme étant la valeur qui a été prédite par K-NN pour l'observation X ;

Fin.

2.2.2 Modèles non supervisés (*Clustering*)

Contrairement aux modèles supervisés, ils n'effectuent pas d'apprentissage, car ils sont censés deviner les classes des objets en trouvant les similarités et les différences entre les instances des données. On appelle la similarité/différence entre deux instances, la distance entre eux. Un ensemble d'éléments groupés sont appelés un cluster. La notion de cluster varie d'une application à une autre selon les besoins et les nécessités. Elle est généralement reliée au type de dataset, et les mesures de la distance entre les instances. Dans un tel modèle on n'est pas limité par le nombre de classes car on ne connaît pas les classes au préalable.

Il y a plusieurs algorithmes qui font partie du domaine de la fouille de données (*Data Mining*) qui s'adresse au problème de la classification non supervisée (*clustering*). Les techniques et algorithmes de clustering sont très utilisés dans plusieurs domaines pour l'analyse des données.

Les différents types de méthodes de clustering

D'après Han et al.[34], les algorithmes de clustering sont divisés comme suit :

1) Méthodes de partitionnements

C'est l'approche la plus simple, étant donnée un nombre N , elle consiste à partitionner le dataset en N groupes ou clusters initialement. Après, une technique de re-locations est utilisée de façon itérative pour améliorer le partitionnement en déplaçant les instances d'un groupe à un autre. Le critère général pour un bon partitionnement est que les instances d'un même cluster soient proches, et les instances des différents clusters soient loin. Parmi les algorithmes les plus connus : KNN, K-means, K-medoids, CLARANS, ...etc.

2) Méthodes hiérarchiques

Elles consistent à créer une décomposition hiérarchique de l'ensemble de données et est classée comme étant soit agglomération ou division, en fonction de la formation de la décomposition hiérarchique. L'approche agglomérative, également appelée approche ascendante, commence par mettre chaque instance dans un groupe séparé puis des fusions successives sont effectuées sur les instances ou les groupes d'instances proches les uns aux autres jusqu'à ce que tous les groupes soient fusionnés en un seul (le niveau le plus élevé de la hiérarchie) ou une condition de terminaison est remplie. L'approche de division, également appelée la approche descendante, commence par mettre les instances dans un même cluster, puis à chaque itération, un cluster est divisé en clusters plus petits, jusqu'à ce que chaque instance soit finalement dans un cluster, ou une condition de terminaison est valide. Les méthodes de classification hiérarchique peuvent être basées sur la distance ou sur la densité et la continuité. Les méthodes hiérarchiques souffrent du fait qu'une fois l'étape est terminée (fusion ou division), elle ne peut jamais être défaite. Cette rigidité est utile car elle conduit à des calculs plus petits. Parmi les algorithmes les plus connus : AGNES, et DIANA.

3) Méthodes basés sur la densité

C'est une amélioration de la méthode de partitionnement. Dans les méthodes basées sur la densité, un cluster continue à grandir à condition que le nombre d'instances dans un rayon de voisinage est supérieur à un seuil donné. Une telle méthode peut filtrer le bruit et les valeurs aberrantes. Parmi les algorithmes on mentionne *DBSCAN*, et *DENCLUE*.

Quelque Algorithmes de clustering

1) K-means

Étant donné un dataset et un nombre K de clusters, l'algorithme commence par choisir K instances aléatoires qui représentent les centroids initiaux et affecte le reste des instances aux clusters ayant le plus proche centroid. Après, il calcule les nouveaux centroids pour chaque cluster, et refait le processus jusqu'à ce qu'il n'y est plus de changement. Cet algorithme est très sensible aux données aberrantes, par conséquent il ne garantit pas l'optimum global, mais il a l'avantage de converger vers un optimum local quelles que soient les données initiales. Le calcul des centroids se fait en général par la moyenne. Étant donné un cluster, le centroid est une instance dont la valeur de chaque attribut est la moyenne des valeurs du même attribut des instances appartenant à ce cluster [34].

Algorithm 2 k-means

Entrée : D : Dataset, K : nombre de cluster ;

Sortie : K clusters ;

Début :

Choisir les centroids initiaux à partir de D aléatoirement ;

Répéter :

Pour Chaque p de D **faire** :

 Calculer la similarité entre p et les centroids ;

 Affecter p au cluster qui possède le plus proche centroids ;

Fait ;

 Calculer les nouveaux centroids (la moyenne) ;

Fin.

2) K-medoids

C'est une amélioration de K-means qui le rend plus efficace pour traiter les données aberrantes. K-medoid utilise comme centroids, des instances du dataset appelé medoids. La

base de cette approche est de trouver les K instances du dataset qui peuvent représenter les centres des clusters à former.

Une variation populaire de cette approche est *PAM* (*Partitioning Around Medoids*), cet algorithme commence par un choix aléatoire des medoids à partir du dataset initial, et affecte les instances du dataset au cluster possédant le plus proche medoid, par la suite chaque medoid est remplacé soit d'une façon aléatoire ou avec heuristique, s'en suit un calcul du coût de remplacement, si le clustering est amélioré alors la permutation des medoids est gardée, et le processus est répété jusqu'à ce que aucun changement est effectué [34].

Algorithm 3 k-medoids

Entrée : D : Dataset, K : nombre de cluster ;

Sortie : K clusters ;

Début :

initialiser les K medoids aléatoirement ;

Répéter :

Affecter les instances de D au cluster qui convient ;

Pour Chaque p de D **faire :**

Calcul du coût S de swap d'un medoid m_j avec p ;

Si : $S < 0$ **Alors** swap m_j avec p **Fsi** ;

Fait ;

Jusqu'à : aucune changement ;

Fin.

2.3 Optimisation avec *métaheuristiques*

Dans le but d'améliorer les performances du système à implémenter et la qualité des résultats, on a choisi d'appliquer une technique d'optimisation appelée métaheuristiques.

Les métaheuristiques sont utilisées pour trouver des réponses à des problèmes dont on n'a que très peu d'éléments pour le résoudre, c'est à dire que nous ne savons pas à quoi ressemble la solution optimale, ni comment la trouver de manière raisonnée, il y a très peu de renseignements heuristiques et la recherche par force brute est impensable, car l'espace de

recherche est trop grand. Néanmoins, si une solution candidate au problème nous est fournie, nous sommes capables de la tester et d'évaluer sa qualité et de répéter le processus jusqu'à avoir la solution adéquate (taux d'erreur moindre).

Un des algorithmes de métaheuristiques les plus simples, est la méthode de descente *Hill-Climbing* : à partir d'un ensemble de valeurs prises aléatoirement, une petite modification aléatoire est effectuée. Si la modification améliore l'ensemble, on la garde, sinon elle est rejetée. Ce processus est effectué autant qu'il est possible. De nombreuses métaheuristiques sont essentiellement élaborées par des combinaisons de cette méthode et de la recherche aléatoire.

2.3.1 Bee Swarm Optimisation (BSO)

L'algorithme BSO (Bee Swarm Optimisation) est l'algorithme d'optimisation par essaim d'abeilles. BSO se base sur la simulation d'un ensemble d'abeilles qui cherche la nourriture. On suppose que l'espace de recherche est représenté par l'ensemble de solutions possibles, et la richesse de la nourriture par la qualité de la solution. Le processus naturel se déroule comme suit :

- L'abeille éclaireuse qui découvre une quantité de nectar et retourne à la ruche.
- Elle effectue ensuite une danse circulaire, pour signaler qu'elle a trouvé une source de nourriture.
- La danse qu'elle effectue dépend de la qualité de la source, et la distance entre la ruche et la source, ainsi que la direction [35].

Selon [35], voici le déroulement de l'algorithme BSO :

1. On génère une solution aléatoire appelée S_{ref} (solution de référence).
2. On met S_{ref} dans une liste taboue.
3. À partir de S_{ref} , un ensemble de points est déterminé.
4. Chaque point généré est attribué à une abeille (recrutement).
5. Chacune des abeilles effectue une recherche locale commençant par le point qu'est lui y est attribué.

6. Les abeilles placent leurs meilleurs résultats dans un tableau pour les comparer.
7. La meilleure solution dans le tableau est choisie comme le nouveau point S_{ref} , et on réitère le processus depuis (2).

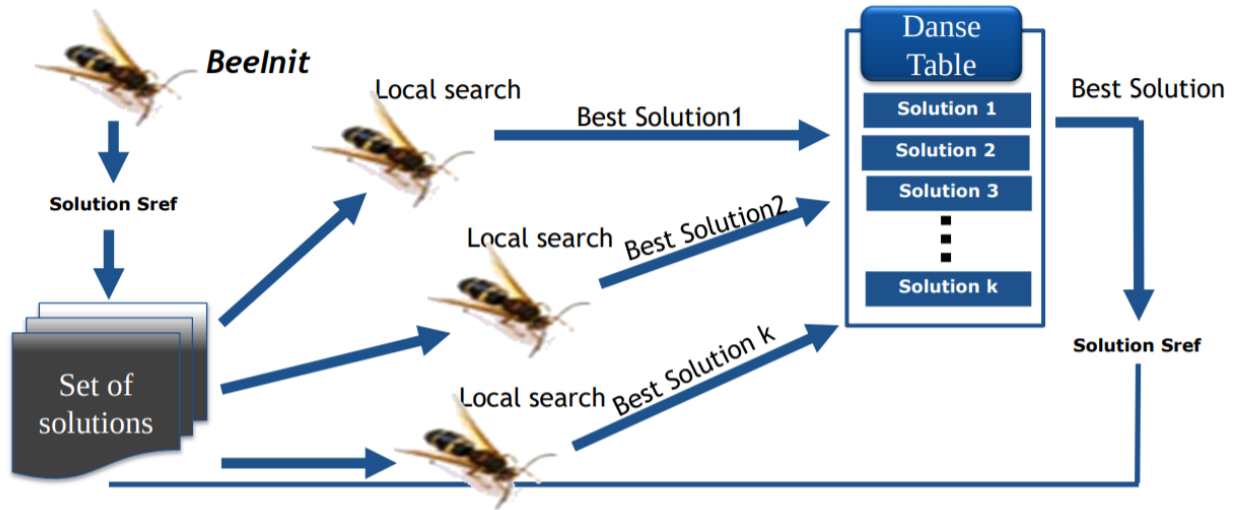


FIGURE 2.1 – Schéma expliquant le fonctionnement de BSO [35]

2.3.2 Optimisation dans les systèmes de recommandations

Il existe plusieurs approches pour l'incorporation des méthodes d'optimisation dans un système de recommandation, afin d'améliorer les résultats. L'une des approches est d'utiliser les caractéristiques des utilisateurs pour un filtrage collaboratif, et donner des priorités à ses caractéristiques en ajoutant des poids. Dans ce cas une métaheuristique peut être utilisée pour trouver les poids qui donnent un résultat optimal. Cette approche a été réalisée par [32] qui ont utilisé l'algorithme PSO pour l'optimisation. Dans notre travail, on utilise les algorithmes de clustering pour la recommandation et afin d'obtenir les résultats les plus optimaux, on utilise aussi une métaheuristique pour affiner leurs paramètres et ainsi améliorer la qualité des prédictions.

2.4 Conclusion

Dans ce chapitre nous avons présenté quelques méthodes de classification et optimisation, et expliqué leurs fonctionnement, dans le prochain chapitre nous allons implémenter ces méthodes avec quelques types de filtrages choisi afin d'améliorer la qualité de prédiction effectuer par ces derniers.

Chapitre 3

Conception

3.1 Introduction

Ayant pour objectif d'augmenter les performances du filtrage collaboratif, sémantique et leurs hybridation nous proposons une approche qui utilise le clustering optimisé afin d'améliorer le partitionnement des utilisateurs (ou des items) en vue d'effectuer des recommandations plus précises. Ce chapitre sera structuré de la façon suivante :

- Description de l'approche de recommandation par classification,
- Les différents types de filtrage conçu et leurs fonctionnement,
- L'aspect de classification ajoutée aux filtres réalisés ;
- L'optimisation de la classification avec une métaheuristique (BSO),
- Conclusion.

3.2 Description de l'approche de recommandation par classification

Les étapes de notre approche peuvent être résumées comme suit :

- Conception du filtrage collaboratif (FC),
- Conception du filtrage sémantique (FSem),
- Réalisation de trois différentes hybridations basées sur les deux filtres FC et FSem,
- Améliorer les algorithmes de filtrage conçu en leur appliquant des techniques de classification,
- Réalisation d'une hybridation multiview (hybridation selon deux ou plusieurs aspects, dans notre cas l'aspect collaboratif et sémantique),
- Optimisation des différents filtres conçus avec une méta-heuristique.

La figure 3.1 illustre les différents algorithmes de notre approche de recommandation, par la suite nous allons présenter en détail chacun de ces algorithmes dans les sections qui suivent.

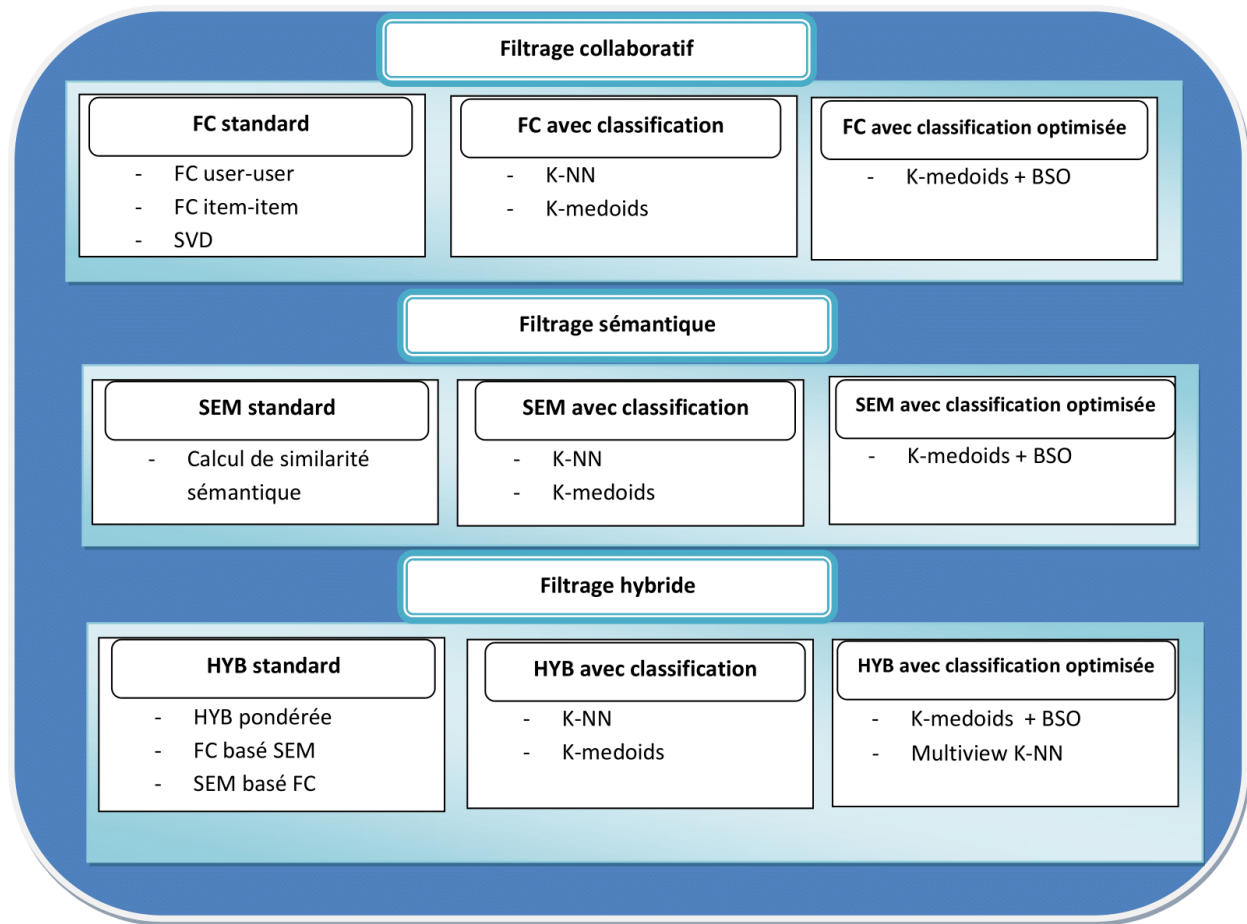


FIGURE 3.1 – Description générale de notre système de recommandation

3.3 Filtrage sans classification

3.3.1 Filtrage collaboratif

Nous avons implémenté le filtrage collaboratif user-user, item-item, et une baseline qui est SVD.

1) Filtrage collaboratif standard

FC user-user

L'idée principale du FC user-user est d'utiliser la matrice d'usage pour déterminer les meilleurs voisins d'un utilisateur (parcours selon les lignes de la matrice), comme expliqué précédemment dans le chapitre 1. Le filtrage collaboratif passe par deux étapes, la première

est le calcul de similarité et la détermination des voisins des utilisateurs, la deuxième consiste à effectuer la prédiction.

L'algorithme 1 ci-dessous décrit le FC basé user-user.

Algorithm 4 FC

Entrée : Matrice_usage, Seuil ;

Sortie : Ensemble de voisins de chaque utilisateur ;

Début :

Pour Chaque *utilisateur* de *Matrice_usage* **faire** :

 Calculer la distance entre cet utilisateur et tout les autres utilisateurs ;

Si : distance <= seuil **Alors** insérer cet utilisateur dans la table voisins du l'utilisateur en cours **Fsi** ;

Fait ;

Fin.

- **Seuil** : Nous déterminons le meilleur seuil pour le FC en évaluant les seuils générés à partir de la formule suivante :

$$Seuil = \frac{Min(matrice) + Max(matrice)}{10} \quad (3.1)$$

Où Min() (respectivement Max()) est une fonction qui retourne la valeur minimum (respectivement maximum) de la matrice. À l'issue de l'application de cette formule nous obtenons dix seuils à tester, et nous choisirons celui qui nous donnera les meilleures prédictions.

- **Complexité de l'algorithme** :

- n : nombre d'utilisateurs.
- Nous effectuons n itérations pour calculer les voisins de chaque utilisateur.
- Donc la complexité de FC = $O(n)$

FC item-item

Comme pour le FC user-user, on détermine les voisins les plus proches des items avec la matrice d'usage mais cette fois-ci selon un parcours colonne par colonne.

SVD (Singular Value decomposition)

Dans le contexte des systèmes de recommandation, SVD est utilisée comme algorithme de filtrage collaboratif pour capturer la similarité entre les utilisateurs et les items en extrayant les caractéristiques cachées qu'un utilisateur ou un item possède et ainsi les faire correspondre selon leurs caractéristiques. Par exemple, pour les films, l'une des caractéristiques peut être le genre auquel le film appartient comme comédie, horreur ou les deux en même temps et pour un utilisateur s'il a un caractère sérieux ou drôle ...ect, (voir figure 3.3). Donc nous effectuons une décomposition en valeurs singulières pour extraire ces caractéristiques et avoir la meilleure décomposition de la matrice d'usage qui permet de prédire les ratings des items non encore évalués par les utilisateurs, souvent la métrique erreur racine moyenne (RMSE) est utilisée pour savoir si la décomposition est bonne ou pas (plus RMSE et bas mieux les performances et la prédiction est meilleure), on précise que nous prenons en compte que les ratings qu'on possède lors de la prédiction dans le calcul d'erreur.

SVD est effectué comme indiqué dans la figure 3.2 ci-dessous.

$$\begin{pmatrix} \hat{X} \\ \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1n} \\ x_{21} & x_{22} & \dots & \\ \vdots & \vdots & \ddots & \\ x_{m1} & & & x_{mn} \end{pmatrix} \\ m \times n \end{pmatrix} \approx \begin{pmatrix} U \\ \begin{pmatrix} u_{11} & \dots & u_{1r} \\ \vdots & \ddots & \\ u_{m1} & & u_{mr} \end{pmatrix} \\ m \times r \end{pmatrix} \begin{pmatrix} S \\ \begin{pmatrix} s_{11} & 0 & \dots \\ 0 & \ddots & \\ \vdots & & s_{rr} \end{pmatrix} \\ r \times r \end{pmatrix} \begin{pmatrix} V^T \\ \begin{pmatrix} v_{11} & \dots & v_{1n} \\ \vdots & \ddots & \\ v_{r1} & & v_{rn} \end{pmatrix} \\ r \times n \end{pmatrix}$$

FIGURE 3.2 – Formule SVD

Où :

- X : désigne la matrice d'utilité.
- U : matrice singulière gauche, représentant la relation entre les utilisateurs et leurs caractéristiques.
- S : matrice diagonale décrivant la force de chaque caractéristique.
- V : matrice transposée singulière droite indiquant la similitude entre les items et leurs caractéristiques.

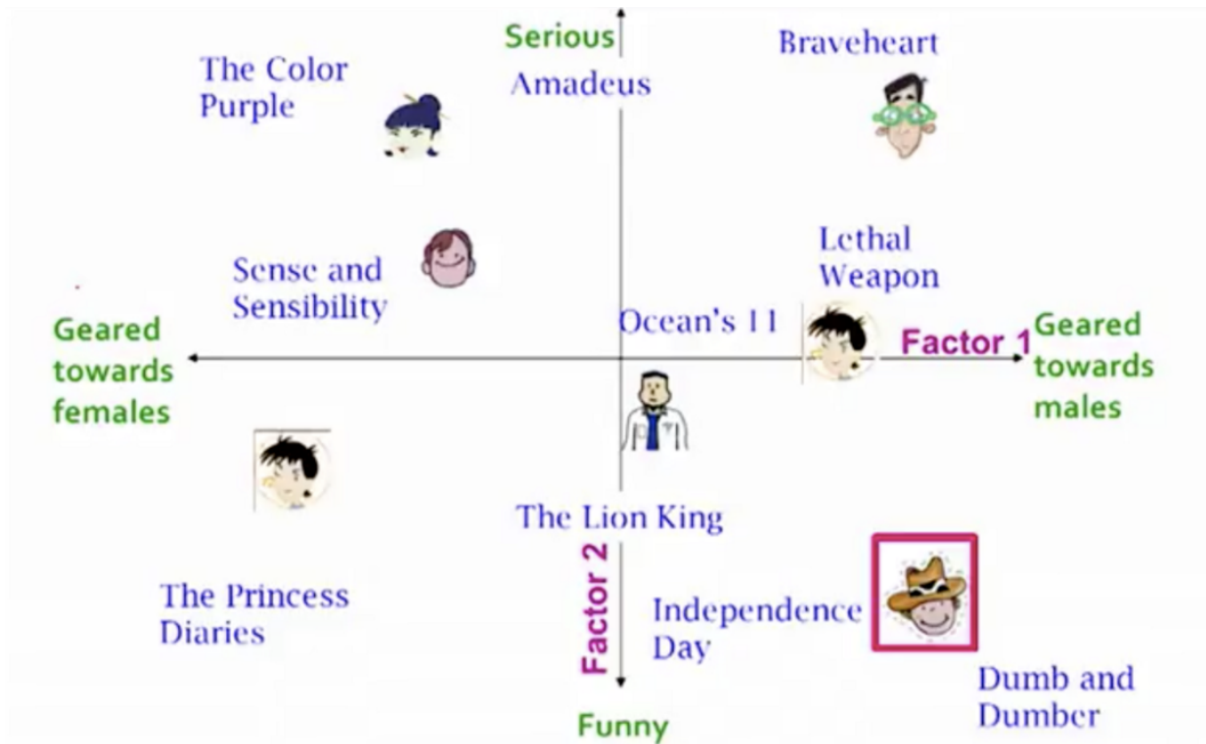


FIGURE 3.3 – Exemple des caractéristiques d'un user [32]

La figure 3.4 ci-dessous montre un exemple de SVD

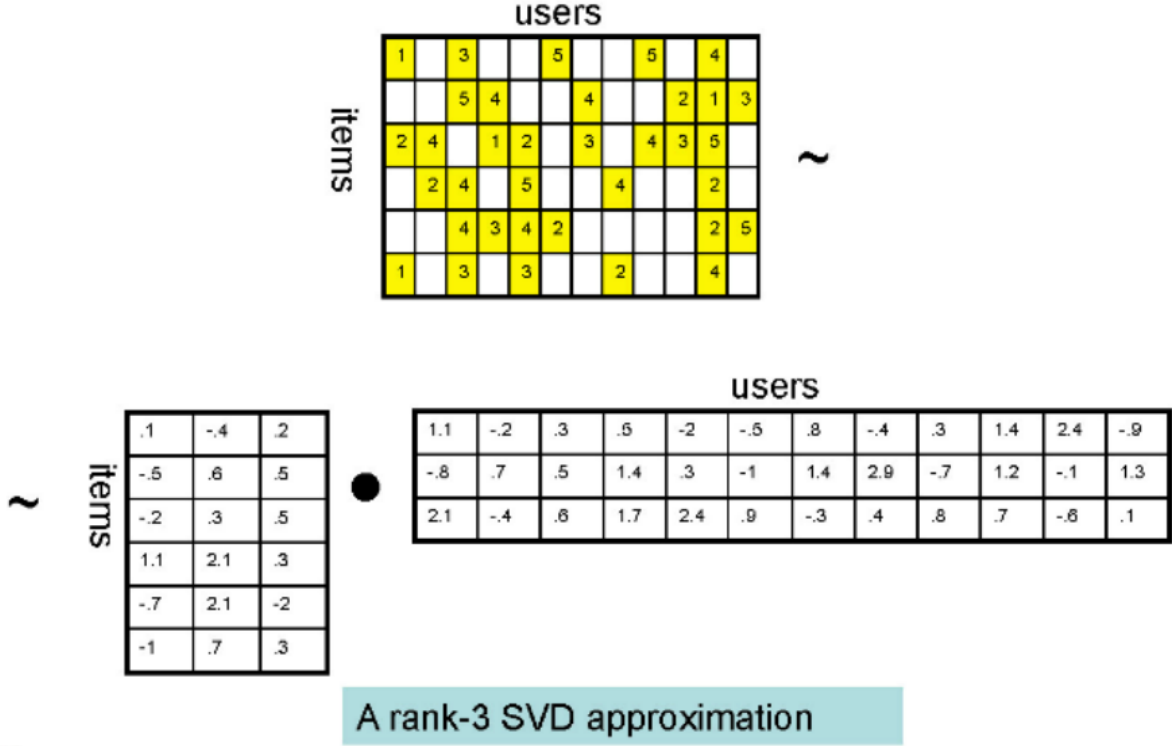


FIGURE 3.4 – Exemple de Singular Value decomposition [32]

Calcul de prédiction et distance pour le filtrage collaboratif

Pour chaque algorithme FC cité précédemment, nous avons utilisé la formule de Pearson afin de calculer la distance entre les utilisateurs et les items, comme suit :

- Calcul distance entre utilisateur :

Pour le calcul de distance entre deux utilisateurs u et v , la corrélation de Pearson est donnée par la formule suivante :

$$pearson(u, v) = 1 - \frac{\sum_{i \in I} (r_{u,i} - \bar{r}_u) \cdot (r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i \in I} (r_{u,i} - \bar{r}_u)^2 \cdot (r_{v,i} - \bar{r}_v)^2}} \quad (3.2)$$

où :

- $r_{u,i}$: est l'estimation de l'utilisateur u sur l'item i .
- $r_{v,i}$: est l'estimation de l'utilisateur v sur l'item i .
- \bar{r}_u : est la moyenne de toutes les notes de l'utilisateur u .
- \bar{r}_v : est la moyenne de toutes les notes de l'utilisateur v .

La distance est interprétée comme suit :

- Si $\text{Pearson}(u,v) = 0$ alors les utilisateurs u et v sont parfaitement identiques.
 - Si $\text{Pearson}(u,v) = 1$ alors les utilisateurs u et v ne sont pas corrélés.
 - Si $0 < \text{Pearson}(u,v) < 1$ alors les utilisateurs u et v sont corrélés positivement (proches) et la distance représente le degrés de corrélation.
 - Si $1 < \text{Pearson}(u,v) < 2$ alors les utilisateurs u et v sont corrélés négativement (loin) et la distance représente le degrés de corrélation.
 - Si $\text{Pearson}(u,v) = 2$ alors les utilisateurs u et v sont parfaitement opposés.
- Calcul distance entre items :

Le calcul de distance entre deux items i et j : la corrélation de Pearson est donnée par la formule suivante :

$$\text{pearson}(i, j) = 1 - \frac{\sum_{u \in U} (r_{u,i} - \bar{r}_i) \cdot (r_{u,j} - \bar{r}_j)}{\sqrt{\sum_{u \in U} (r_{u,i} - \bar{r}_i)^2 \cdot \sum_{u \in U} (r_{u,j} - \bar{r}_j)^2}} \quad (3.3)$$

où :

- $r_{u,i}$: est l'évaluation de l'utilisateur u sur l'item i
- $r_{u,j}$: est l'évaluation de l'utilisateur u sur l'item j
- \bar{r}_i : est la moyenne des évaluations de l'item i par les utilisateurs.
- \bar{r}_j : est la moyenne des évaluations de l'item j par les utilisateurs.

La distance est interprétée comme suit :

- Si $\text{Pearson}(i,j) = 0$ alors les item i et j sont parfaitement identiques.
- Si $\text{Pearson}(i,j) = 1$ alors les item i et j ne sont pas corrélés.
- Si $0 < \text{Pearson}(i,j) < 1$ alors les item i et j sont corrélés positivement (proches) et la distance représente le degrés de corrélation.
- Si $1 < \text{Pearson}(i,j) < 2$ alors les item i et j sont corrélés négativement (loin) et la distance représente le degrés de corrélation.
- Si $\text{Pearson}(i,j) = 2$ alors les item i et j sont parfaitement opposés.

Pour la prédiction des valeurs des évaluations (ratings) des items non encore évalués, nous avons utilisé la formule de la somme pondérée :

$$\text{pred}(u_i, i_k) = r(\bar{u}_i) + \frac{\sum_{u_j \in U_i} \text{sim}(u_i, u_j) \cdot (r_{u_j, i_k} - r(\bar{u}_j))}{\sum_{u_j \in U_i} \text{sim}(u_i, u_j)} \quad (3.4)$$

Où :

- $\text{sim}(u_i, u_j)$: est la mesure de similarité entre un utilisateur u_i et son voisin u_j , tel que $u_j \in U_i$.
- \bar{r}_i : est la moyenne des évaluations de l’item i par les utilisateurs.
- \bar{r}_j : est la moyenne des évaluations de l’item j par les utilisateurs.

3.3.2 Filtrage sémantique standard

Ce type de filtrage utilise la notion sémantique pour déterminer les profils des utilisateurs qui sont similaires. Comme pour le filtrage collaboratif, le filtrage sémantique passe par deux étapes le calcul de similarité entre utilisateurs ensuite la prédication des ratings sur les items non évalués.

Pour déterminer si deux profils d’utilisateurs sont similaires du point de vue sémantique, nous nous intéressons aux items que les deux utilisateurs ont bien notés afin d’identifier les catégories qui sont susceptibles de les intéresser et donc par la suite leur recommander des items de catégories identiques ou similaires.

C’est à partir de la catégorie d’un item que nous allons extraire l’information sémantique, pour cela nous disposons d’une matrice *item-catégorie* qui contient pour chaque item donné la catégorie à laquelle il appartient. Un item peut appartenir à plusieurs catégories en même temps, comme le montre la figure 3.5 ci-dessous :

| | | | | | |
|------|-------------|-------------|-------------|-------|-------------|
| | Catégorie 1 | Catégorie 2 | Catégorie 3 | | Catégorie N |
| Item | 0 | 1 | 0 | | 1 |

Avec :

- ❖ 0 : Item n'appartient pas à la catégorie
- ❖ 1 : Item appartient à la catégorie

FIGURE 3.5 – Exemple d'une ligne de la matrice *item-catégorie*

Pour déterminer la similarité sémantique entre les items, nous devons calculer la distance entre ces derniers en se basant sur leurs représentation. Deux cas de figures se présentent à nous :

- Les items ne sont pas représentés avec des ontologies : Pour chaque item de la matrice *item-catégorie*, nous appliquons la distance de Jaccard afin de déterminer les degrés de similarité entre les items et ainsi obtenir une matrice de distance *item-item*. La distance de Jaccard est comprise entre $[0; 1]$ et a la propriété d'être égale à 0 quand les items sont différents et 1 quand ils sont identiques et est calculée comme suit :

$$J(A, B) = 1 - \frac{|A \cap B|}{|A| + |B| - |A \cup B|} = 1 - \frac{\sum a_i * b_i}{\sum a_i^2 + \sum b_i^2 - \sum a_i * b_i} \quad (3.5)$$

Où :

- A et B : sont deux items,
- a_i, b_i : peuvent être soit 0 ou 1, ils indiquent l'appartenance ou non de l'item A (respectivement B) à une catégorie i .
- Les items sont représentés avec des ontologies : Pour chaque item de la matrice *item-catégorie*, nous appliquons la distance de Wu et Palmer afin de déterminer les degrés de similarité entre les items et ainsi obtenir une matrice de distance *item-item*. La distance

de Wu et Palmer est comprise entre $[0; 1]$. Si deux concepts sont identique la distance est égale à zéro et s'ils sont opposés elle est égale à un, et est calculée comme suit :

$$Dist_{WP} = 1 - \frac{(2 * N)}{(N_1 + N_2 + 2 * N)} \quad (3.6)$$

Où :

- **C1** et **C2** : sont deux concepts.
- **N1** et **N2** : la distance séparant C1 et C2 du nœud racine.
- **N** : distance séparant l'ancêtre commun le plus proche de C1 et C2 du nœud racine.

Où : X (respectivement Y) sont les deux catégories que u (respectivement v) a revu.

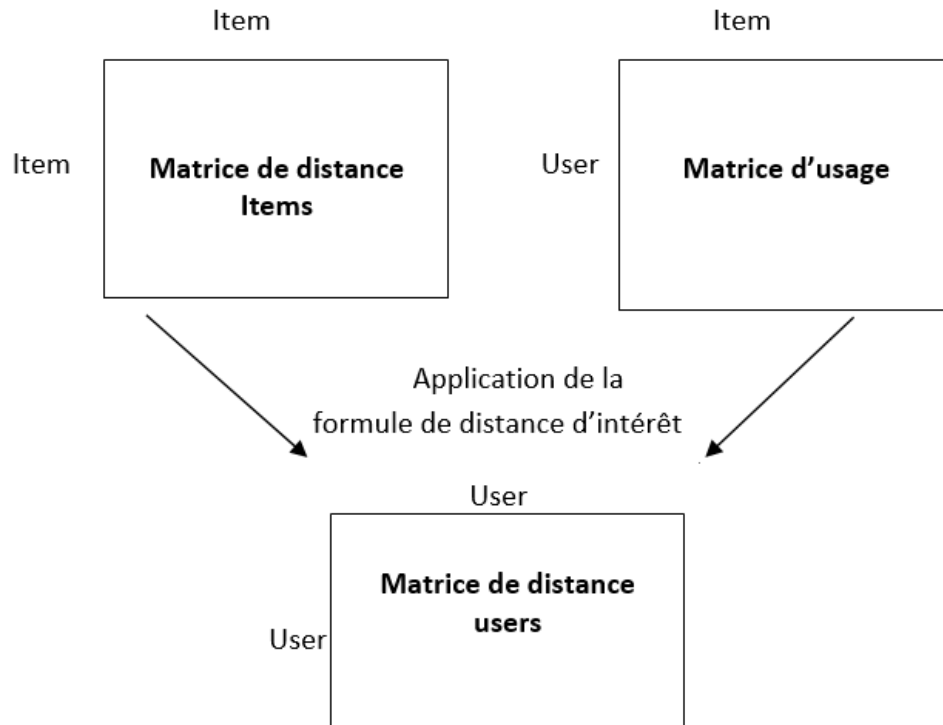


FIGURE 3.7 – Génération de la matrice de distance *user-user*

À partir de la matrice distancée *user-user* nous pouvons déterminer à présent les voisins d'un utilisateur donné en appliquant l'algorithme suivant :

Algorithm 5 Filtrage sémantique

Entrée : Matrice de distance user-user, Seuil ;

Sortie : Ensemble de voisins de chaque utilisateur

Début :

Pour Chaque *utilisateur_i* de *matrice_distance* **faire :**

Pour Chaque *distance_j* de *utilisateur_i* **faire :**

/*récupérer l'indice de la colonne, qui représente l'indice du voisin en cours de traitement*/

utilisateur_j \leftarrow indice(*distance_j*)

Si : *distance_j* \leq Seuil **Alors** insérer *utilisateur_j* dans la table voisins du l'*utilisateur_i* **Fsi ;**

Fait ;

Fait ;

Fin.

- **Seuil :** Le seuil est calculé de la même façon que pour le FC (voir seuil dans la partie FC).

- **Complexité de l'algorithme :**

- n : nombre d'utilisateurs.
- n^2 : taille de la matrice de distance carré.
- À chaque itération on fait un parcours de n^2 (taille de la matrice de distance).

- Donc la complexité du filtrage sémantique = $O(n^2)$

Calcul de prédiction pour le filtrage sémantique

Pour la prédiction des valeurs des ratings sur les items non encore évalués, nous avons utilisé la formule de la somme pondérée citée précédemment.

3.3.3 Filtrage hybride


Dans cette partie, nous allons détailler trois méthodes d'hybridation que nous avons implémentées entre les 2 types de filtrage déjà réalisés précédemment (sémantique, collaboratif), et qui sont :

- Filtrage collaboratif hybride pondéré.
- Filtrage collaboratif basé sémantique.
- Filtrage sémantique basé collaboratif.

Pour chaque type d'hybridation conçu, les deux matrices de distance collaboratif (user-user, item-item) et sémantique jouent un rôle très important et sont calculées de la sorte :

- Matrice distance du filtrage collaboratif :

Nous appliquons la formule Pearson pour calculer la distance d'un utilisateur avec le reste des utilisateurs de la matrice d'usage et nous stockons les valeurs des distances dans une matrice comme le montre la figure 3.8 suivante :



| | User1 | User2 | | | | | User M |
|--------|-------|-------|--|----------------------------------|--|--|--------|
| User1 | | 0.48 | | ... | | | |
| User2 | | | | | | | |
| | | | | . | | | |
| | | | | . | | | |
| | | | | . | | | |
| | | | | Matrice de distance users | | | |
| | | | | . | | | |
| | | | | . | | | |
| User M | | | | ... | | | |

FIGURE 3.8 – Matrice de distance user-user

- Matrice distance du filtrage sémantique : (voir explication dans la partie filtrage sémantique standard.)

1) Filtrage hybride pondéré

L'algorithme de filtrage hybride pondéré associe à chaque matrice de distance collaborative et sémantique respectivement un poids α et β qui représentent les degrés d'importance de chaque type de recommandation (voir figure 3.9) avec :

- $0.1 \leq \alpha \leq 0.9$
- $\beta = 1 - \alpha$
- $\alpha + \beta = 1$

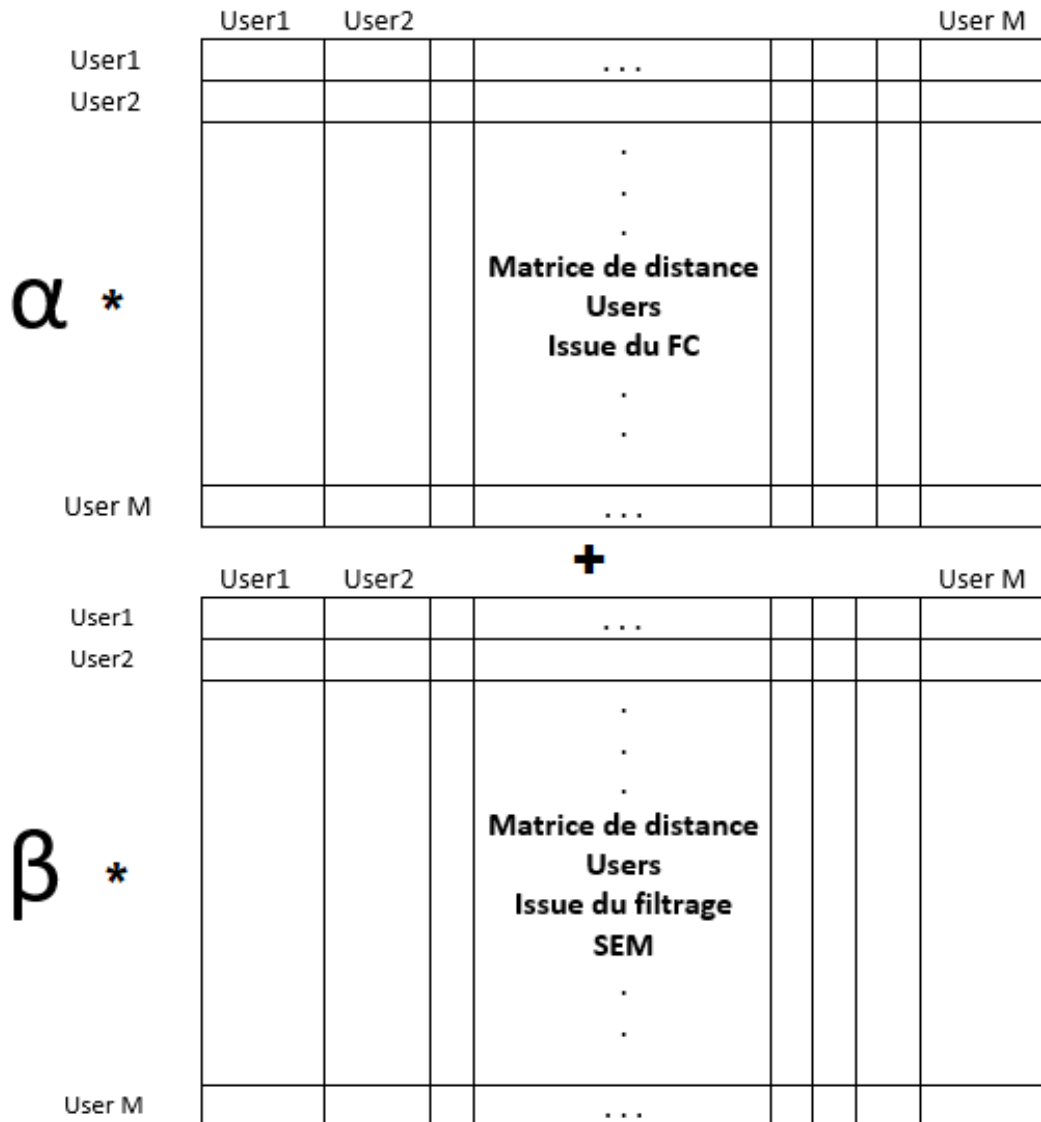


FIGURE 3.9 – Pondération des matrices de distances

Nous utilisons une boucle, dans laquelle nous augmentons la valeur de α de 0.1, et nous

affectons à β la valeur $1 - \alpha$, puis nous multiplions la matrice de distance du filtrage collaboratif par α et la matrice de distance du filtrage sémantique par β , nous évaluons par la suite les prédictions générées à partir de ces deux matrices, nous répétons ce processus jusqu'à ce que nous obtenons les meilleures valeurs possibles pour α et β qui permettent d'accorder l'importance adéquate aux filtres collaboratif et sémantique.

Algorithm 6 Filtrage hybride pondéré

Entrée :

Matrice_distance_FC : Matrice de distance du FC user-user ,
Matrice_distance_SEM : Matrice de distance du filtrage sémantique user-user,
Seuil_Voisin : le seuil d'acceptation d'un voisin ;

Sortie :

Ensemble de voisins pour chaque utilisateur selon meilleur poids Alpha ;

Début :

Alpha \leftarrow 0.1 ;
Beta \leftarrow 1 - Alpha ;
Best_evaluation \leftarrow Float(Inf) ;
BestAlhpa \leftarrow Float(Inf) ;

Tant que *Beta* \geq 0.1 **faire :**

AlphaMatrice \leftarrow matrice_distance_FC * Alpha ;
Beta \leftarrow 1 - Alpha ;
BetaMatrice \leftarrow matrice_distance_SEM * Beta ;
SommeMatrices \leftarrow AphaMatrice + BetaMatrice ;
Voisins_users \leftarrow Voisin (SommeMatrices , Seuil_Voisin) ;
tableau_predicton \leftarrow Prediction (usage_matrice , Voisins_users) ;
evaluation \leftarrow Evaluation (tableau_predicton , reelles_valeurs) ;

Si : Best_evaluation \geq evaluation **Alors**

Swap (Best_evaluation, evaluation) ;
BestAlpha \leftarrow valeurAlpha ;

Fsi ;

Alpha \leftarrow Alpha - 0.1 ;

Fait ;

Fait ;

Fin.

Où :

- Voisins : Fonction qui calcule les voisins pour chaque utilisateur à partir d'un seuil et de la matrice de distance entre utilisateurs.
- Prediction : Fonction qui effectue les prédictions des items non évalués pour chaque utilisateur.
- Evaluation : Fonction qui calcule la qualité d'une prédiction effectuée.

- **Complexité de l'algorithme :**

- n : nombre d'utilisateurs.
- n^2 : taille de la matrice de distance carré.
- Nous avons neuf itérations (de 0.1 jusqu'à 0.9).
- À chaque itération on fait un parcours de n^2 (taille de la matrice de distance).
- Donc la complexité du filtrage hybride pondéré = $O(9 * n^2)$

2) Filtrage hybride sémantique basé collaboratif

Le filtrage sémantique standard décrit précédemment, utilise une matrice de distance entre les items qui est calculée avec la formule Jaccard, ici nous allons la remplacer avec la matrice de distance items issue du filtrage collaboratif (voir figure 3.10) en utilisant la formule Pearson comme suit :

- Soit deux items i et j : la corrélation de Pearson est donnée par la formule suivante :

$$pearson(i, j) = \frac{\sum_{u \in U} (r_{u,i} - \bar{r}_i) \cdot (r_{u,j} - \bar{r}_j)}{\sqrt{\sum_{u \in U} (r_{u,i} - \bar{r}_i)^2 \cdot \sum_{u \in U} (r_{u,j} - \bar{r}_j)^2}} \quad (3.8)$$

Où :

- $r_{u,i}$: est l'évaluation de l'utilisateur u sur l'item i
- $r_{u,j}$: est l'évaluation de l'utilisateur u sur l'item j
- \bar{r}_i : est la moyenne des évaluations de l'item i par les utilisateurs.
- \bar{r}_j : est la moyenne des évaluations de l'item j par les utilisateurs.

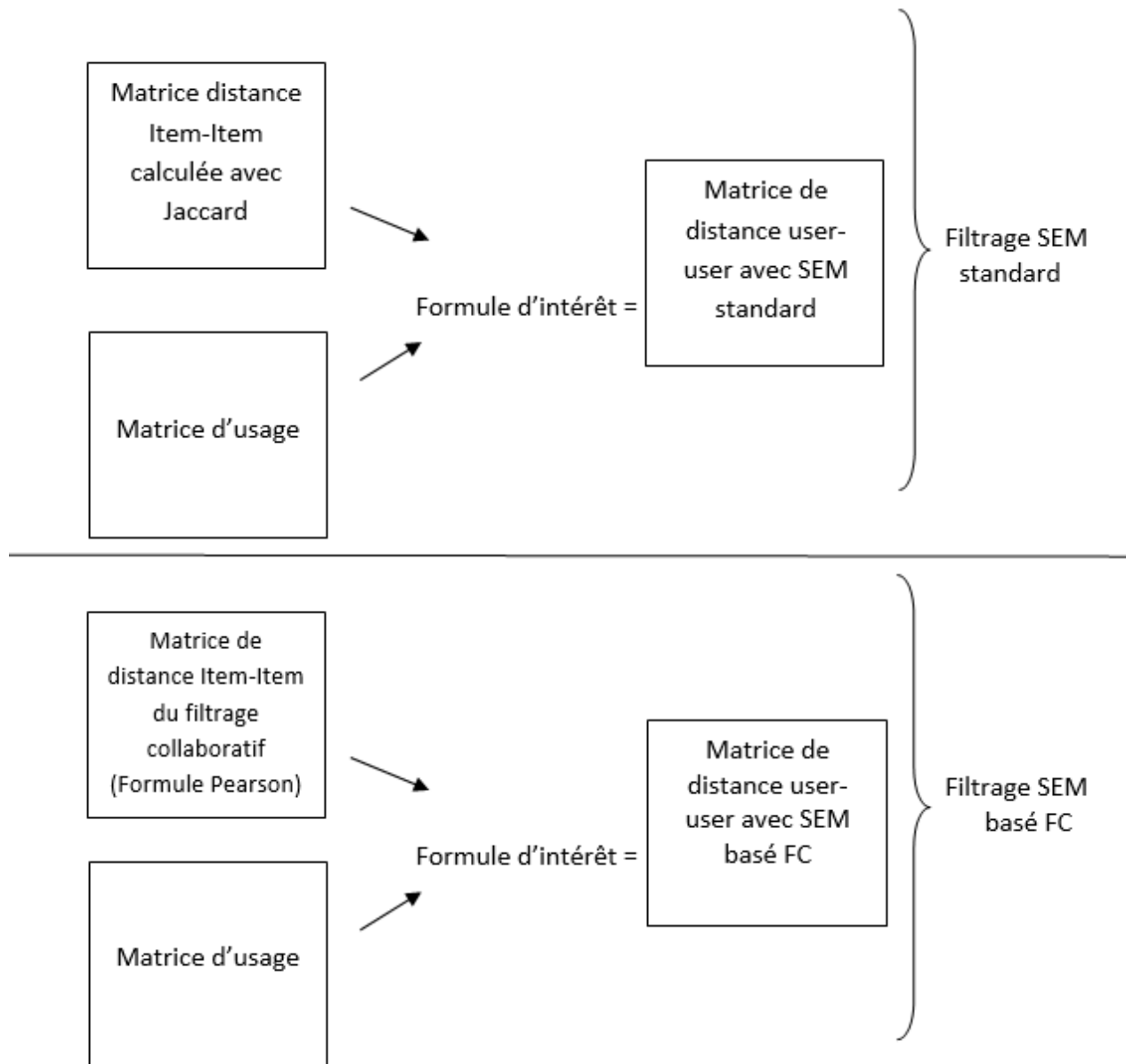


FIGURE 3.10 – Différence entre sémantique standard et sémantique basé collaboratif

Remarque : Après avoir passer en entrée la matrice de distance entre les items issue du FC, le même algorithme du filtrage sémantique est utilisé pour effectuer le filtrage sémantique basé collaboratif.

Calcul de prédiction et distance pour le filtrage hybride sémantique basé collaboratif

Pour la prédiction des valeurs des ratings sur les items non encore évalués, nous avons

utilisé la formule de la somme pondérée citée précédemment. Quant au calcul de distance entre utilisateurs elle s'effectue comme montré dans le filtrage sémantique en utilisant la même formule de calcul de distance d'intérêt (Formule 3.5).

3) Filtrage hybride collaboratif basé sémantique

Dans ce filtrage nous effectuons une sélection des voisins d'un utilisateur en utilisant les deux matrices de distance du FC et du filtrage sémantique. Pour deux utilisateur A et B, B est considéré comme voisin de A ssi :

$$\min(DFC(A, B), DFSEM(A, B)) \leq \text{seuil}$$

Où :

- DFC : distance entre utilisateur A et B selon le filtrage collaboratif.
- DFSEM : distance entre utilisateur A et B selon le filtrage sémantique.
- seuil : seuil d'acceptation d'un voisin pour un utilisateur donnée.

❖ Avec : seuil = 0.3

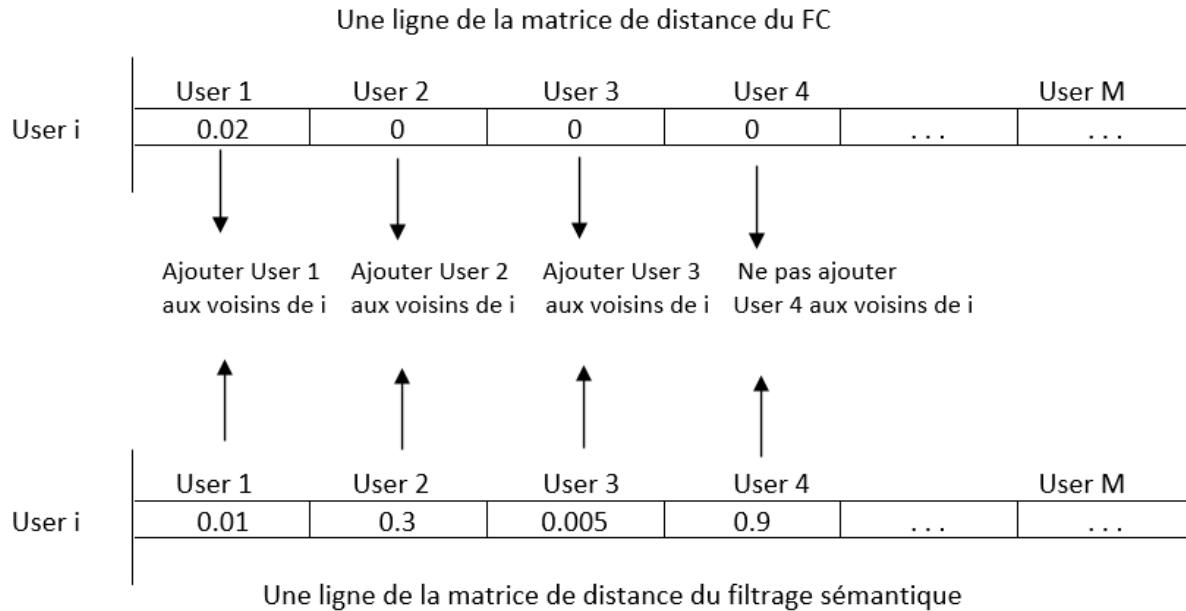


FIGURE 3.11 – Exemple de filtrage collaboratif-sémantique

Algorithm 7 Filtrage hybride collaboratif basé sémantique

Entrée :

Matrice_distance_FC : Matrice de distance du FC user-user ,
Matrice_distance_SEM : Matrice de distance du filtrage sémantique user-user ;
Seuil_Voisin : le seuil d'acceptation d'un voisin ;

Sortie :

Ensemble de voisins de chaque utilisateur ;

Début :

Matrice_distance \leftarrow Min (Matrice_distance_FC , Matrice_distance_SEM) ;

Pour Chaque *utilisateur_i* de Matrice_distance **faire** :

Pour Chaque *distance_j* de *utilisateur_i* **faire** :

 /*récupérer l'indice de la colonne, qui représente l'indice du voisin en cours
 de traitement*/

 utilisateur_j \leftarrow indice(distance_j)

Si : distance_j \leq Seuil **Alors** insérer utilisateur_j dans la table voisins du
 l'utilisateur_i **Fsi** ;

Fait ;

Fait ;

Fin.

Où :

- Min() : Fonction qui prend minimum des distances entre deux matrices et les stocke dans une nouvelle matrice.

- **Complexité de l'algorithme** :

- n : nombre d'utilisateurs.
- n^2 : taille de la matrice de distance carré.
- À chaque itération on fait un parcours de n^2 (taille de la matrice de distance).

- Donc la complexité du filtrage collaboratif basé sémantique = $O(n^2)$

3.4 Filtrage avec classification

Dans cette partie, nous allons expliquer l'amélioration apportée aux différents filtrages conçus jusqu'à présent en leurs ajoutant une technique de classification, on précise que la matrice de distance que nous allons utiliser pour classifier les utilisateurs est le résultat d'une des variantes du FC, notre choix sera basé sur les tests effectués afin de prendre la meilleure variante, donc nous aurons à choisir entre FC item-item, FC user-user ou SVD. Concernant le FSem nous avons conçu une seule variante par conséquent il n'y aura pas de choix à faire. Enfin, pour le filtrage hybride nous incorporons la classification dans les trois méthodes réalisées (filtrage hybride pondéré, collaboratif basé sémantique et sémantique basé collaboratif.(voir figure 3.12)

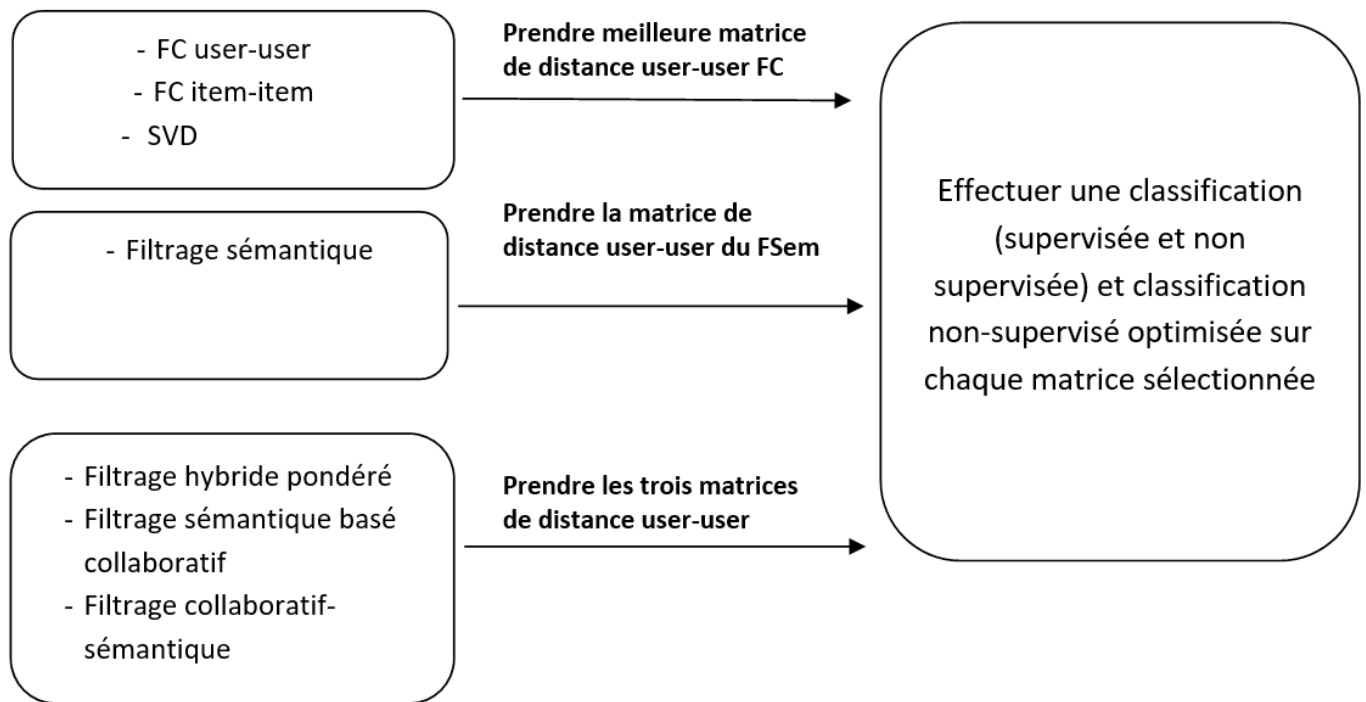


FIGURE 3.12 – Sélection des matrices de distance *user-user* pour l'application de la classification

3.4.1 Filtrage avec classification supervisée

1) K-NN

Comme motionné dans le chapitre 2, K-NN est un algorithme d'apprentissage automatique. Dans notre cas nous l'avons utilisé pour effectuer les prédictions d'un utilisateur donné, en se basant sur ses voisins les plus proches. Nous avons besoin donc des distances entre utilisateurs sous forme de matrice de distance *user-user* issue d'un des type de filtrage et K qui est le nombre de voisins à considérer.

Par exemple pour effectuer un filtrage sémantique basé K-NN, il suffit d'envoyer à l'algorithme ci-dessous la matrice de distance *user-user* issue du filtrage sémantique. Il en est de même pour effectuer :

- Filtrage collaboratif basé K-NN.
- Filtrage hybride pondéré basé K-NN.
- Filtrage collaboratif basé sémantique avec classification K-NN.
- Filtrage sémantique basé collaboratif avec classification K-NN.

Algorithm 8 Filtrage avec classification supervisée (K-NN)

Entrée :

Matrice_distance_user-user : Matrice de distance issue d'un des types de filtrage ,
K : le nombre de voisins à considérer dans les prédictions ;

Sortie :

Ensemble de K voisins de chaque utilisateur ;

Début :

Voisins[] ; /*Liste de listes qui contient les voisins des utilisateurs*/

Pour Chaque *utilisateur_i* de *Matrice_distance_user-user* **faire** :

/*trier selon l'ordre croissant les distances entre l'utilisateur i et le reste des utilisateur du dataset contenu dans la ligne utilisateur_i */

sortedListe \leftarrow Sort(utilisateur_i) ;

/*Récupérer les indices des K plus proches voisins de l'utilisateur_i/

Voisins[utilisateur_i] \leftarrow Min_K_distance(K,indice(sortedListe)) ;

Fin.

- **Complexité de l'algorithme :**

- n : nombre d'utilisateurs.
- CAAT : complexité approximative de l'algorithme de trie.
- À chaque itération on fait un parcours de n (taille de la matrice de distance).
- Donc la complexité du filtrage avec classification supervisée = $O(n * CAAT)$

2) Hybridation mutivues, collaborative et sémantique, basé sur K-NN

Cette approche est inspirée du travail des chercheurs Guibing Guo et al. [30], dans lequel ils effectuent un clustering des utilisateurs en appliquant deux fois consécutives k-medoids (une fois selon la vue similarité et la deuxième fois selon la vue de confiance entre utilisateurs) et par la suite combiner le résultat des deux vues. Nous avons étudié leur travail et put aboutir à l'idée d'utiliser l'algorithme K-NN selon nos deux vues sémantique et collaboratif. Le choix

de l'algorithme K-NN semble adéquat aux données et informations dont on dispose, de plus c'est un algorithme simple et très utilisé dans la recommandation.

L'idée générale de notre approche est d'effectuer une classification K-NN sur la matrice de distance *user-user* du FC et de prendre K utilisateurs ce qui nous donne une classification selon une première vue, par la suite refaire la même chose sur la matrice de distance *user-user* du filtrage sémantique pour avoir la deuxième vue, enfin combiner les deux classifications en exécutant K-NN une autre fois sur les K utilisateurs issues de la première classification et de la deuxième (voir figure 3.13).

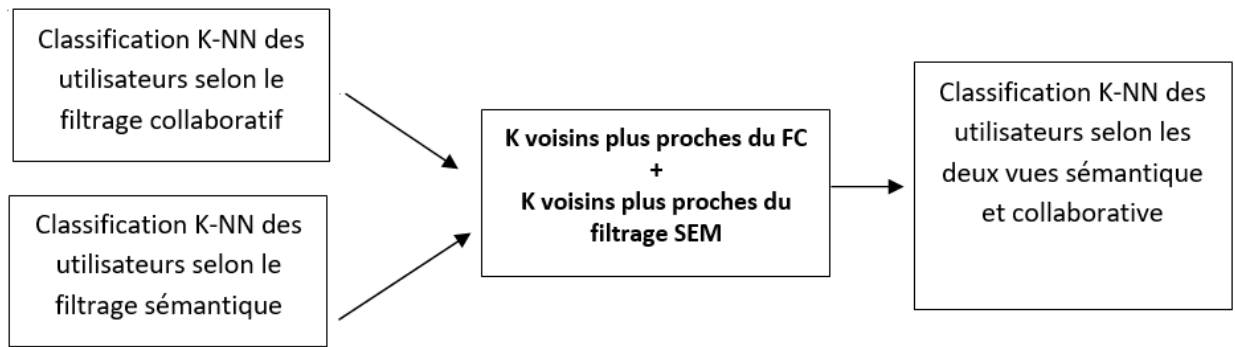


FIGURE 3.13 – Schéma du filtrage multiview K-NN

Algorithm 9 Filtrage avec multiview (K-NN) Partie 1

Entrée :

Matrice_distance_user-user_FC : Matrice de distance issue d'un du filtrage collaboratif,

Matrice_distance_user-user_SEM : Matrice de distance issue d'un du filtrage sémantique,

K : le nombre de voisins à considérer dans les prédictions ;

Sortie :

Ensemble de K voisins de chaque utilisateur selon les deux vues ;

Début :

Voisins[]_CF ; /*Liste de listes qui contient les voisins des utilisateurs selon filtrage collaboratif*/

Voisins[]_SEM ; /*Liste de listes qui contient les voisins des utilisateurs selon filtrage sémantique*/

Pour Chaque *utilisateur_i* de *Matrice_distance_user-user_CF* **faire** :

/*trier selon l'ordre croissant les distances entre l'utilisateur i et le reste des utilisateur du dataset contenu dans la ligne utilisateur_i */

sortedListe \leftarrow Sort(utilisateur_i) ;

/*Récupérer les indices des K plus proches voisins de l'utilisateur_i/

Voisins[utilisateur_i]_CF \leftarrow Min_K_distance(K, indice(sortedListe)) ;

Pour Chaque *utilisateur_i* de *Matrice_distance_user-user_SEM* **faire** :

/*trier selon l'ordre croissant les distances entre l'utilisateur i et le reste des utilisateur du dataset contenu dans la ligne utilisateur_i */

sortedListe \leftarrow Sort(utilisateur_i) ;

/*Récupérer les indices des K plus proches voisins de l'utilisateur_i/

Voisins[utilisateur_i]_SEM \leftarrow Min_K_distance(K, indice(sortedListe)) ;

Algorithm 6 Filtrage avec multiview (K-NN) Partie 2

/*Fusionner les deux vues*/

Voisins[] \leftarrow Voisins[utilisateur_i]_SEM + Voisins[utilisateur_i]_CF;

Pour Chaque *utilisateur_i* de Voisins[] **faire** :

/*trier selon l'ordre croissant les distances entre l'utilisateur i et le reste des utilisateur du dataset contenu dans la ligne utilisateur_i */

sortedListe \leftarrow Sort(utilisateur_i);

/*Récupérer les indices des K plus proches voisins de l'utilisateur_i/

Voisins[utilisateur_i]_Multiview \leftarrow Min_K_distance(K,indice(sortedListe));

Fin.

- **Complexité de l'algorithme** :

- n : nombre d'utilisateurs.
 - CAAT : complexité approximative de l'algorithme de trie.
 - Nous effectuons trois fois l'algorithme K-NN.
 - À chaque itération on fait un parcours de n (taille de la matrice de distance).
- Donc la complexité du filtrage avec classification supervisée = $O(3 * n * CAAT)$

3.4.2 Filtrage avec classification non supervisée

1) Filtrage basé sur K-medoids

Le principe est d'appliquer l'algorithme K-medoids sur l'ensemble des utilisateurs de manière à construire des clusters qui contiennent des utilisateurs possédant des intérêts similaires, le choix des utilisateurs qui seront considérés comme medoids de départ s'effectue de façon aléatoire. Les distances entre les utilisateurs sont stockées dans les matrice distance *user-user* issue d'un des type de filtrage.

Par exemple pour effectuer un filtrage sémantique basé K-medoids, il suffit d'envoyer à l'algorithme ci-dessous la matrice de distance *user-user* issue du filtrage sémantique. Il en est de même pour effectuer :

- Filtrage collaboratif basé K-medoids.
- Filtrage hybride pondéré basé K-medoids.

- Filtrage collaboratif basé sémantique avec clustering K-medoids.
- Filtrage sémantique basé collaboratif avec clustering K-medoids.

Algorithm 7 Filtrage avec classification non supervisé (K-medoids)

Entrée :

Matrice_distance_user-user : Matrice qui contient les distances entre utilisateur issue d'un des types de filtrage ,

K : Le nombre de cluster à former ;

Sortie :

Clustering des utilisateurs en K cluster ;

Début :

Medoids \leftarrow Choisir les K centres des clusters : Désigné aléatoirement K indices de la matrice de distances représentant les indices de K utilisateurs ;

calculer le cout initial ;

Pour Chaque *M-user* de *Medoids* **faire** :

Pour Chaque *N-user* de *non-Medoids* **faire** :

 Swap(M,O) ;

 Distribution(Matrice_distance_user-user, Medoids) ;

 calculerCout() ;

Si : le cout de cette itération est supérieure au cout de l'itération précédente

alors Undo(M-user,N-user) ; **Fsi** ;

Si : Aucune changement alors arrêter l'algorithme **Fsi** ;

Fait ;

Fait ;

Fin.

Où :

- Swap() : Remplace le medoid M-user par N-user

- Undo() : Annule le swap de M-user par N-user
- Distribution() : Assigne les utilisateurs au plus proche cluster en utilisant la matrice de distance et les medoids
- Cout() : Fonction objective qui est destinée à être minimisée

$$Cout = \min \sum_{c \in C} \sum_{u, v \in c} d(u, v) \quad (3.9)$$

Avec : C l'ensemble des clusters résultant de l'algorithme k-medoids, u et v deux utilisateurs, v appartient au cluster et u son medoids.

- **Complexité de l'algorithme :**

- n : nombre d'utilisateurs.
- k : nombre de clusters à former.
- Nous devons trouver la distance entre chacun des $(n-k)$ points de données k fois pour placer les points de données dans le groupe le plus proche.
- Après cela, nous devons remplacer chacun des medoids précédemment supposés par chaque non-medoid et recalculer la distance entre les objets $(n-k)$.
- Donc la complexité du filtrage avec classification non supervisée = $O(k(n-k)^2)$

3.4.3 Filtrage avec classification non supervisée optimisée

Comme nous l'avons vu dans le chapitre précédent l'optimisation par colonie d'abeilles manipule un ensemble d'abeilles où chaque abeille correspond à une solution faisable d'un problème donné. Nous avons décidé d'exploiter le mieux possible BSO dans le but d'optimiser le clustering effectué par K-medoids, et ainsi améliorer la qualité des prédictions.

K-medoids commence initialement par des medoids qui sont sélectionnés aléatoirement, en utilisant BSO nous tentons de trouver les meilleurs medoids de départ qui permettent de faire le meilleur partitionnement possible des utilisateurs, et de ne pas effectuer une sélection aléatoire des medoids car nous avons peu de chance de tomber sur de bons medoids.

Cette classification non supervisée optimisée est appliquée pour les différents types de filtrage conçu (FC, FSem et Filtrage hybride) car nous faisons appel à l'algorithme BSO qui lui fait appel à l'algorithme K-medoids, ce dernier comme vu précédemment (voir Filtrage basé sur K-medoids) à en entrée une matrice de distance issue de l'un des types de filtrages

implémentés.

1) Codification et initialisation de la solution

Afin d'exploiter le mieux possible la métaheuristique et prouver son efficacité il est nécessaire de faire une codification adéquate qui permet de modéliser le problème. Dans notre cas, chaque partitionnement des users effectué par k-medoids représente une solution, donc chaque abeille correspond à un partitionnement faisable.

La solution est représentée par un vecteur de taille égale au nombre d'utilisateurs à classifier dans un dataset, les indices du vecteur représentent les identificateurs des utilisateurs et chaque case du vecteur peut contenir un 0 ou 1 qui signifient respectivement :

- le i ème utilisateur du vecteur solution n'est pas medoid.
- le i ème utilisateur de vecteur solution est medoid.

On démarre l'algorithme avec une solution initiale qui est soit aléatoire en remplissant un vecteur de taille N (N = nombre d'utilisateurs) avec 0 et 1, ou une solution qui est le résultat d'un vecteur construit à partir des clusters sorties de l'algorithme K-medoids qui représente un partitionnement des utilisateurs en K clusters.

2) Pseudo code de l'algorithme BSO adapté au problème de recommandation

Le pseudo-code suivant explique le principe de l'algorithme BSO adapté à notre problème. À noter qu'à chaque ajout d'une solution référence on doit vérifier qu'elle n'est pas déjà dans la liste taboue.

Algorithm 8 Recommandation avec K-medoids optimisée (BSO) partie 1

Entrée :

Matrice_distance_user-user : Matrice qui contient les distances entre utilisateur issue d'un des types de filtrage ,

K : Le nombre de cluster à former,
nbrIteration, flip, local_max_iter ;

Sortie :

Vecteur de solution optimisé par BSO ;

Début :

```
/*Generation aléatoire de Sref (vecteur à valeurs entre 0 et 1)*/  
/*ou appeler K-medoids pour générer une solution avec K-1 medoids*/  
(Sref ← Random-sol() || Sref ← K-medoids(K-1, Matrice_distance_user-user));  
/*Evaluer la solution Sref et la définir comme meilleure évaluation pour le moment*/  
Best_eval = Eval(sref);  
Pour cpt de nbrIteration faire :
```

```
/*Ajouter la solution de référence à tabou liste*/  
taboo-list.append(Sref);  
/*Generer les solutions a partir de Sref avec paramètre flip*/  
areas ← search-area(Sref,flip);  
/* Début de la boucle de locale*/
```

Pour Chaque *area* de *areas* **faire** :

```
/*Recherche local dans le voisinage de la solution courante*/  
current_eval, solution = bee-local-search(area, local_max_iter);  
Si : Best_eval < min_eval alors :  
    Best_eval ← min_eval;  
    Sref ← solution;
```

Fsi;

fait ;

Algorithm 8 Recommandation avec K-medoids optimisée (BSO) partie 2

/*A la sortie de la boucle de la recherche locale, nous aurons une solution S_{ref} qui donne la meilleure évaluation, si $nbiterations$ n'est pas atteint alors nous générons des solutions à partir de S_{ref} et nous re-effectuant une recherche locale, sinon on sort avec la meilleure solution qui est S_{ref}^* */

fait ;

Fin.

Où :

- $search-area()$: génère un espace de recherche à partir de S_{ref} .
- $bee-local-search()$: permet d'effectuer une recherche local à partir d'une solution donnée,
- $Eval()$: évalue la qualité d'une solution donnée.

- **Complexité de l'algorithme** : Soient les variables suivantes :

- GMI : nombre global de max itération.
- LMI : nombre local de max itération.
- EACK : estimation approximative de la complexité de kmedoids.
- flip : paramètre empirique qui détermine le nombre de search areas.

- Donc complexité de BSO = $O(GMI * flip * LMI * EACK)$

3) Flip

Détermine le nombre d'espace de recherche à générer à partir de S_{ref} , une valeur trop grande donnée au $flip$ implique dans notre cas la génération de plusieurs espaces de recherche et donc une bonne diversification et en même temps une intensification de la recherche, car il existerait des solutions qui seront proches dues au fait de la rotation à gauche qu'on effectue pour générer $flip$ solutions à partir de S_{ref} , tant dit qu'un petit $flip$ n'impliquerait qu'une intensification de la recherche dans un voisinage proche (voir figure 3.14).

4) L'espace de recherche

Exemple : Si on suppose que $nbr_medoid = 4$ et $nbr_user = 7$, au départ nous générons un vecteur S_{ref} qui représente une solution de taille égale à 7 avec 3 medoids donc trois 1

seront présents dans le vecteur $Sref$, par la suite nous générons un espace de recherche initiale à partir de $Sref$ en prenant $flip = nbr_medoid / flip$, donc nous aurons $flip$ solutions générées à partir de $Sref$ en effectuant une rotation à gauche $flip$ fois, enfin nous envoyons à chaque abeille ($flip$ abeille) une solution pour qu'elle commence une recherche locale afin de trouver le 4ème medoids qui donne les meilleures évaluations possibles (voir figure 3.14).

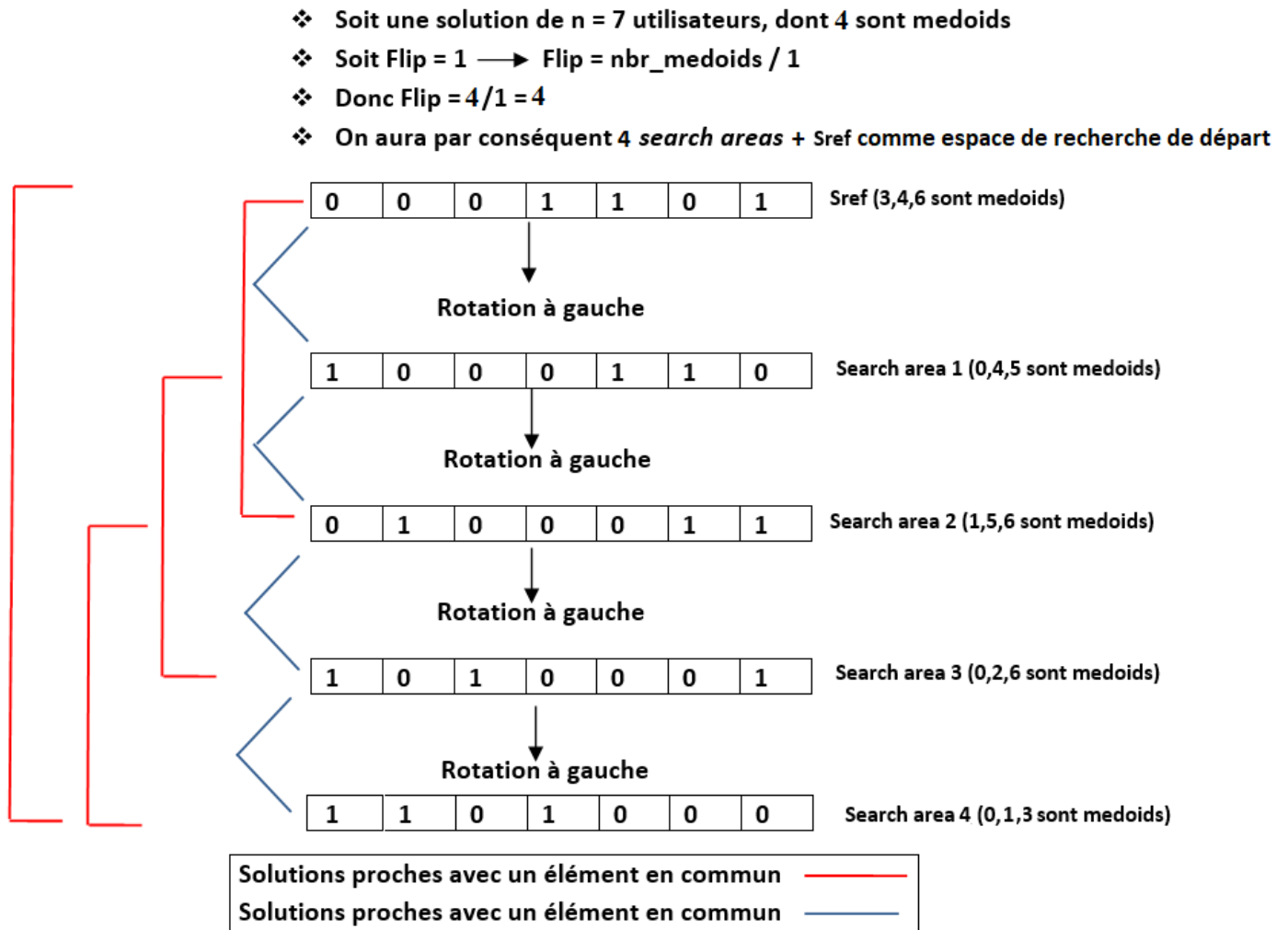


FIGURE 3.14 – Génération des *search areas* à partir de $Sref$

5) Fonction fitness

Pour évaluer la qualité du vecteur de solution on a besoin d'évaluer la qualité de la prédiction effectuée à partir du clustering de K-medoids, pour cela nous avons utilisé les

métriques RMSE/MAE (à minimiser).

$$RMSE = \sqrt{\frac{\sum_{i=1}^N (Predicted_i - Actual_i)^2}{N}} \quad (3.10)$$

$$MAE = \frac{1}{N} \sum_{i=1}^N |Predicted_i - Actual_i| \quad (3.11)$$

Où :

- *Predicted_i* : rating prédit pour l'item *i*.
- *Actual_i* : réel rating pour l'item *i*.
- *N* : nombre d'item total.

6) La recherche locale

La recherche locale est effectuée sur chaque vecteur de solution, pour essayer de trouver le nième medoid optimal. Une itération de la recherche locale est décrite comme suit :

1. Ajouter le nième medoid en changeant la première valeur égale à zéro rencontrée dans le vecteur à un.
2. Évaluer l'ajout du medoid avec la fonction fitness, et enregistrer l'évaluation dans une liste nommée *Liste_Eval*.
3. Défaire l'ajout du nième medoid de l'étape (1).
4. Refaire les étapes (1) (2) et (3) pour le reste des valeurs égale à zéro du vecteur solution.

Après avoir terminer le parcours du vecteur solution, on récupère l'indice du medoid qui a donné la meilleure valeur de fonction fitness à partir de *Liste_Eval*, nous affectons à *Sref* la nouvelle solution trouvée dans la recherche locale, et si *nbriterations* n'est pas atteint, alors nous générons des solutions à partir de *Sref* et nous re-effectuant une recherche locale, sinon on sort avec la meilleure solution qui est *Sref*.

Calcul de prédiction pour le filtrage avec classification et classification optimisée

Pour la prédiction en se basant sur le vote du cluster auquel appartient chaque utilisateur afin de calculer les ratings des items non encore évalués, et cela en utilisant la formule de la somme pondérée citée précédemment.

3.5 Exemple de calcul de prédiction

Soit la matrice d'usage suivante :

| | I1 | I2 | I3 | I4 | I5 | I6 |
|-------|----|----|----|----|----|----|
| U_1 | 5 | 4 | 2 | 3 | 2 | ? |
| U_2 | 4 | 4 | 2 | 3 | 1 | 4 |
| U_3 | 4 | 0 | 2 | 1 | 2 | 4 |
| U_4 | 4 | 5 | 0 | 0 | 2 | 5 |
| U_5 | 1 | 5 | 2 | 1 | 2 | 4 |
| U_6 | 1 | 3 | 5 | 3 | 2 | 1 |

FIGURE 3.15 – Exemple de matrice d'usage

Pour calculer les distances entre les utilisateurs en applique la formule de distance de Pearson suivante :

$$pearson(u, v) = 1 - \frac{\sum_{i \in I} (r_{u,i} - \bar{r}_u) \cdot (r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i \in I} (r_{u,i} - \bar{r}_u)^2 \cdot (r_{v,i} - \bar{r}_v)^2}} \quad (3.12)$$

On aura la matrice de distance entre utilisateur comme suit :

| | I1 | I2 | I3 | I4 | I5 | I6 |
|-------|------|------|------|------|------|------|
| U_1 | 0 | 0.16 | 0.39 | 0.5 | 0.92 | 1.54 |
| U_2 | 0.16 | 0 | 0.39 | 0.13 | 0.62 | 1.42 |
| U_3 | 0.39 | 0.39 | 0 | 0.6 | 0.7 | 1.68 |
| U_4 | 0.5 | 0.13 | 0.6 | 0 | 0.44 | 1 |
| U_5 | 0.92 | 0.62 | 0.7 | 0.44 | 0 | 1.04 |
| U_6 | 1.54 | 1.42 | 1.68 | 1 | 1.04 | 0 |

FIGURE 3.16 – Exemple de matrice de distance

Pour calculer la prédiction de l'utilisateur U_1 sur l'item I_6 , on effectue tout d'abord un des types de filtrage conçu pour trouver les voisins de chaque utilisateur, on suppose que les voisins de l'utilisateur U_1 sont comme suit :

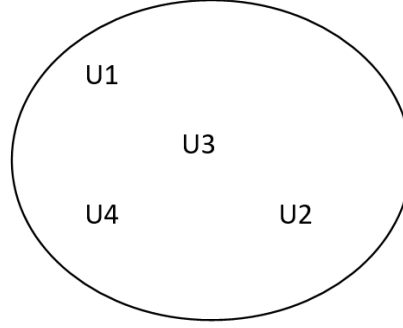


FIGURE 3.17 – Les voisins de l'utilisateur U_1

En utilisant la formule de prédiction de somme pondérée suivante :

$$pred(u_i, i_k) = r(\bar{u}_i) + \frac{\sum_{u_j \in U_i} sim(u_i, u_j) \cdot (r_{u_j, i_k} - r(\bar{u}_j))}{\sum_{u_j \in U_i} sim(u_i, u_j)} \quad (3.13)$$

On aura :

$$pred(u_1, i_6) = 3.2 + \frac{0.16 * (4 - 3) + 0.39 * (4 - 2.6) + 0.5 * (5 - 4)}{0.16 + 0.39 + 0.5} = 4.38 \quad (3.14)$$

3.6 Ingénierie du système

Nous présentons le diagramme de cas d'utilisation qui modélise les différentes fonctionnalités de notre système de recommandation.

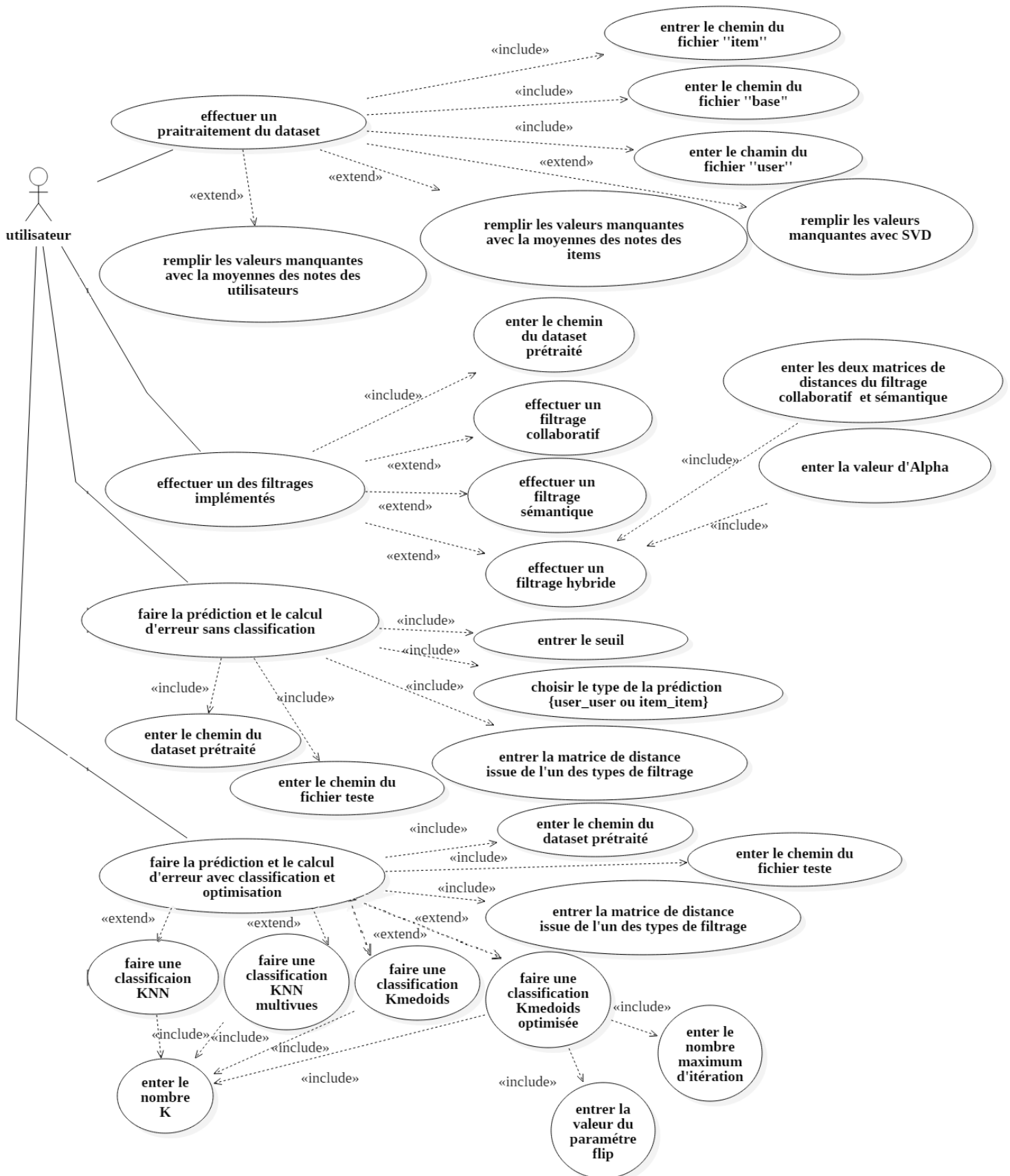


FIGURE 3.18 – Diagramme de cas d'utilisation de notre système de recommandation

3.7 Conclusion

Dans ce chapitre nous avons présenté en détail notre approche de recommandation qui utilise trois types de filtrage : Collaboratif, sémantique et hybride (combinant ces deux algorithmes selon différentes méthodes). Dans un second lieu, nous avons appliqué deux techniques de classification, supervisée (L'algorithme KNN a été utilisé) et non supervisée (l'algorithme K-medoids a été utilisé). Une hybridation basé sur ces algorithmes a été considéré, dont l'hybridation multivues avec KNN. Finalement, afin d'améliorer la qualité de la classification, nous avons appliqué une optimisation en utilisant la métaheuristique BSO (algorithme K-medoids-BSO). Dans le chapitre suivant nous allons évaluer notre système de recommandation et présenter l'interface réalisée qui permet aux utilisateurs de tester notre système et de visualiser les résultats obtenus.

Chapitre 4

Implémentation et expérimentation

4.1 Introduction

Dans le chapitre précédent, nous avons proposé une approche de recommandation incluant plusieurs algorithmes basés sur des techniques différentes (hybridation, clustering, clustering optimisé et clustering multivues). Dans ce chapitre, nous présenterons en détail l'implémentation et l'expérimentation de notre système de recommandation. Ce chapitre est structuré comme suit :

- L'environnement de travail (langage, outils et bibliothèques utilisés),
- L'application réalisée, (son fonctionnement et mode d'emploi),
- L'analyse des performances de l'approche développée et la comparaison de nos résultats avec ceux d'autres techniques déjà existantes.

4.2 Environnement de travail

Nous allons présenter dans cette partie les outils que nous avons utilisés pour réaliser notre système de recommandation.

4.2.1 Langage de programmation *Python*

Créé originellement par le programmeur Guido van Rossum autour de 1990, *Python* est un langage de programmation objet interprété de haut niveau avec une sémantique dynamique. Ses structures de données intégrées de haut niveau, combinées à un typage et liaison dynamique, le rendent très attrayant pour le développement rapide d'applications, ainsi que pour son utilisation en tant que langage de script ou de collage pour connecter des composants existants[40].

Caractéristiques du langage et avantages

- La syntaxe simple et facile à apprendre de Python met l'accent sur la lisibilité et réduit donc le coût de la maintenance du programme.
- Python prend en charge les modules et les packages, ce qui encourage la modularité du programme et la réutilisation du code.

- L'interpréteur Python et la vaste bibliothèque standard sont disponibles gratuitement sous forme binaire ou source pour toutes les principales plates-formes et peuvent être distribués librement.

4.2.2 Outils et logiciels

Système d'exploitation (Linux)

Linux est un noyau monolithique à code source ouvert. Le noyau Linux a été développé à l'origine par Linus Torvalds, qui l'a annoncé dans le groupe de discussion comp.os.minix le 25 août 1991. Depuis lors, il a été porté sur des architectures informatiques comprenant x86-64, x86, ARM, RISC et DEC Alpha. Les développeurs ont accès à tout le code source de Linux et sont autorisés, dans les conditions de la licence, à le modifier et à le distribuer [41].

L'environnement de développement (Visual Studio Code)

Visual Studio Code est un éditeur de code source gratuit et libre de droit, disponible pour Windows, macOS et Linux. Il est livré avec un support intégré pour JavaScript, TypeScript et Node.js et dispose d'un écosystème riche en extensions pour d'autres langages (tels que C++, C#, Java, Python, PHP, Go) [42].

4.3 Présentation de notre système de recommandation

4.3.1 Description générale

Le système que nous avons développé permettra de lancer les différents algorithmes implémentés et de visualiser en détails les performances et les résultats de chaque technique stockés dans des fichiers *.csv*.

Il sera également possible de choisir le dataset (jeux de données) à utiliser. Toutes les techniques implémentées sont généralisées pour traiter n'importe quel dataset possédant la structure montrée dans la figure 4.1. La modularité du code source de l'application permettra facilement d'apporter des mises à jours aux algorithmes afin d'ajouter d'autre techniques.

| Id_user | Id_Item | Rating |
|---------|---------|--------|
| 1 | 1 | 5 |
| 1 | 2 | 3 |
| 1 | 3 | 4 |
| 1 | 4 | 3 |
| 1 | 5 | 3 |
| 1 | 6 | 5 |
| 1 | 7 | 4 |
| 1 | 8 | 1 |

FIGURE 4.1 – Exemple de structuration d’un fichier exploitable dans notre système de recommandation

4.3.2 Illustration des fonctionnalités développées

L’interface de l’application est composée de 4 onglets, qui sont :

- *Data preprocessing* : le prétraitement des données d’un dataset est une étape essentielle avant son utilisation, dans notre cas il existe énormément de valeurs manquantes de rating car les users ne notent pas tous les items existants et par conséquent la prédiction effectuée ne sera pas précise. Pour effectuer un prétraitement du dataset, il faut passer en entrée trois fichiers, qui sont :

- Fichier *u.item* : contient les identificateurs des items.
- Fichier *u.user* : contient les identificateurs des utilisateurs.
- Fichier *u.base* : contient les évaluations faites par les utilisateurs sur les items.

Le traitement des valeurs manquantes s’effectue avec l’une des options suivantes :

- Moyenne de rating des utilisateurs : on remplace les valeurs manquantes en effectuant la moyenne des évaluations des utilisateurs, pour voir en général la note que ces derniers attribuent aux items.
- Moyenne de rating des items : les valeurs manquantes d’un item sont remplacées par la moyenne des évaluations qu’il lui sont attribuées.

- SVD : nous utilisons la technique SVD expliqué dans le chapitre 3, afin d’effectuer une décomposition de la matrice d’usage du dataset pour extraire les caractéristiques cachées des items et des utilisateurs et ainsi les faire correspondre selon leurs caractéristiques, et remplacer les valeurs manquantes des ratings.

On note qu’on peut laisser le fichier tel quel sans prétraitement.

Quand le prétraitement des données est effectué, le résultat est sauvegardé dans un fichier pour une exploitation ultérieure.

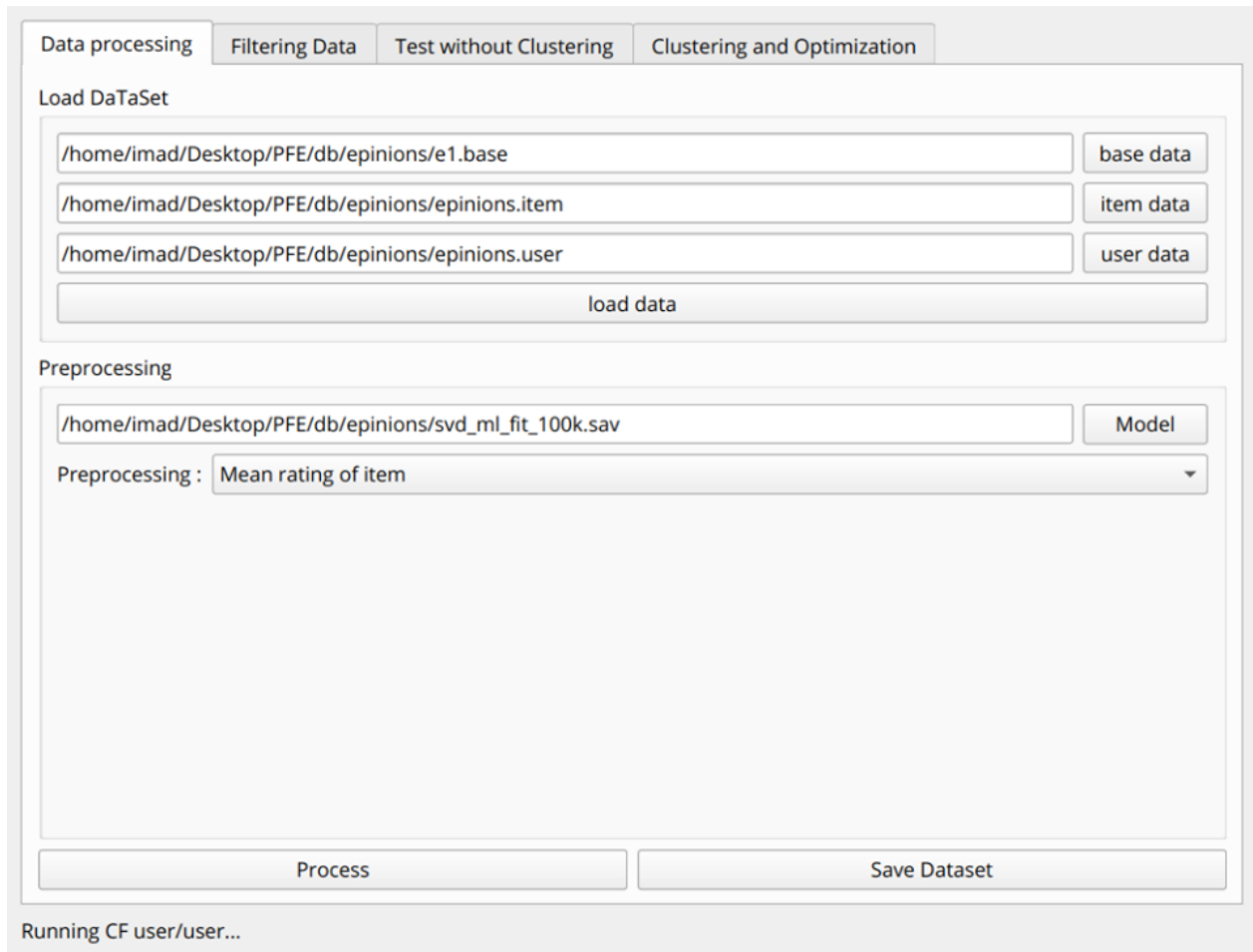


FIGURE 4.2 – Capture d’écran de l’onglet *Data preprocessing*

- *Filtering data* : dans cet onglet on effectue l’un des filtrage conçu (FC, FSem, hybride), le fichier prétraité dans l’étape précédent est importé puis le calcul des distances entre les utilisateurs s’effectue avec les formules de similarité des filtrage et le résultat est sauvegardé dans un fichier (stockage de la matrice des distances dans un fichier).

The screenshot shows a web-based application interface for data filtering. The 'Filtering Data' tab is active. The 'Load Usage Matrix / Item Matrix' section contains three input fields for file paths, with the first field populated with '/home/imad/Desktop/PFE/db/epinions/epinions-svd-100k.sav'. To the right of these fields are buttons labeled 'dataset', 'CF', and 'Semantic'. Below this is the 'Additional parameters' section, which includes a text input for 'Alpha value (Only considered with weighted hybrid technique)'. The 'Apply Filtering Technique CF/Semantic' section features a dropdown menu for 'Filtering Algorithms' currently set to 'Colabortive filtering'. At the bottom of the main content area are 'Process' and 'Save' buttons. A status bar at the bottom of the window displays 'Running CF user/user...'.

FIGURE 4.3 – Capture d’écran de l’onglet *Filtering data*

- *Test without classification* : comme expliqué dans le chapitre précédent, après avoir obtenu la matrice de distances qui est issue de l’un des types de filtrage on peut effectuer les prédictions des évaluations des items en utilisant un seuil qui permet la sélection des voisins des utilisateurs (ou les items pour le cas FC item-item). On commence par introduire le fichier prétraité d’un dataset, car il contient les évaluations des items donnés par les users, puis la matrice de distance du dataset pour la sélection des voisins des users. Le type de la matrice de distance qu’on introduit dépend des cas suivant :
 - Si FC item-item alors introduire matrice de distance item-item calculée avec formule de Pearson.
 - Si FC user-user alors introduire matrice de distance user-user calculée avec formule de Pearson.
 - Si FSem alors introduire matrice de distance user-user calculée avec formule dis-

tance d'intérêt.

- Si filtrage hybride alors introduire matrice de distance hybride calculée avec formule Pearson et distance d'intérêt.

Après avoir importer les fichier du dataset prétraité et sa matrice de distance, on importe le fichier test qui contient les ratings des items pour effectuer les prédictions et vérifier la qualité de cette dernière avec les métriques RMSE et MAE vus dans le chapitre précédent et les affichées dans l'interface.

The screenshot shows a software interface with four tabs: "Data processing", "Filtering Data", "Test without Clustering", and "Clustering and Optimization". The "Test without Clustering" tab is active. It contains a "Load Inputs" section with three text input fields and buttons labeled "Load Dataset", "Distance Matrix", and "Test File". Below this is a "Threshold" section with a text input field and two radio buttons labeled "user/user" and "item/item". The "Results" section has two "TextLabel" placeholders. A "test" button is at the bottom right. A status bar at the bottom left indicates "Running CF user/user..."

FIGURE 4.4 – Capture d'écran de l'onglet *Test without classification*

- *Classification and optimisation* : pour faire un filtrage avec classification ou classification optimisée, les mêmes étapes de l'onglet *test without clustering* sont reeffectuées sauf qu'au lieu d'introduire un seuil de sélection de voisins, on utilise un algorithme de classification sur la matrice de distances des users (respectivement des items pour le FC item-item avec classification). Après avoir introduit les fichiers contenant le dataset, la

matrice de distances et les items de test, on entre les paramétrés selon la technique de classification voulue :

- K-medoids : le nombre K de cluster est requis pour lancer une classification avec K-medoids des utilisateurs.
- K-medoids optimisé : le nombre K de cluster à former et *max iterations* qui est le nombre maximum d'itération à effectuer pour trouver une solution optimisée.
- K-NN : le nombre K de voisins à considérer pour les prédictions des ratings de chaque utilisateur.
- K-NN multiview : (le même paramètre que K-NN est requis pour exécuter K-NN multiview).

Les résultats des évaluations des prédictions sera affiché après exécution des algorithmes dans l'interface est qui sont : RMSE, MAE, rappel, précision.

The screenshot shows a software interface with four tabs: "Data processing", "Filtering Data", "Test without Clustering", and "Clustering and Optimization". The "Clustering and Optimization" tab is selected. Inside this tab, there is a "Load Inputs" section with three text input fields and three buttons: "Load Dataset", "Distance Matrix", and "Test File". Below this is a "Threshold" section with a text input field and two radio buttons labeled "user/user" and "item/item". At the bottom is a "Results" section with two "TextLabel" placeholders and a "test" button. A status bar at the very bottom shows "Running CF user/user..."

FIGURE 4.5 – Capture d'écran de l'onglet *Classification and optimisation*

4.4 Expérimentation

4.4.1 Dataset

1) Présentation de *MovieLens*

GroupLens Research (un laboratoire de recherche de l'université du Minnesota) a mis à disposition un ensemble de données d'évaluation à partir du site Web MovieLens collectés sur différentes périodes. On retrouve des jeux de données allant de 100 000 évaluations jusqu'à 20 millions, ce dernier est plus destinées à la recherche ou de grand moyens matériels sont souvent nécessaires. Étant limité par le temps et par la puissance de calcul de nos machines, nous opterons donc dans un premier temps pour des dataset de 100 000 évaluations afin d'effectuer nos tests.

Architecture du dataset

Le dataset *100K* de *MovieLens* se présente sous la structure suivante :

- Un fichier *u.data* : de 100 000 lignes, contenant l'ensemble des évaluations (l'id de l'utilisateur, l'id de l'item, l'évaluation (de 1 à 5), la date d'évaluation).
- Un fichier *u.genre* : ou on peut retrouver les différent catégories des films présents sur la plate-forme.
- Un fichier *u.item* : ou sont indiqués tous les films avec l'ensemble des informations qui leurs sont relatives : date de sortie, genre etc ...
- Un fichier *u.occupation* : indiquant les différentes professions des utilisateurs.
- Un fichier *u.user* : contenant les informations relatives à chaque utilisateur : age, sexe, profession etc ...
- Un fichier *u.base* : contient 80% des données *u.data*, ce fichier est utilisé pour effectuer l'apprentissage (classification et construction de modèle).
- Un fichier *u.test* : contient 20% des données *u.data*, ce fichier est utilisé pour effectuer les testes (prédiction et calcule d'erreur).

Exploitation du dataset

Le processus de test et d'évaluation d'un algorithme nécessite deux ensembles de données : un ensemble d'apprentissage et un ensemble de test. En effet, l'ensemble de données du fichier

u.data contient en tout 100 000 évaluations, nous prendrons donc *u.base* qui contient 80 000 évaluations pour l'apprentissage et *u.test* qui contient 20 000 évaluations pour la prédiction et le calcul des métriques.

2) Présentation de *RED*

Epinions.com était un site général d'évaluation des consommateurs créé en 1999. Epinions a été acquis par Shopping.com (connu sous le nom de DealTime au moment de l'acquisition) en 2003, qui a ensuite été acquis par eBay en 2005. Chez Epinions, les visiteurs pouvaient lire nouvelles et anciennes critiques sur une variété d'articles pour les aider à effectuer leurs achats. Le 25 mars 2014, toutes les fonctionnalités de la communauté, ainsi que les fonctionnalités de soumission et de modification des avis, ont été désactivées. Par la suite, en mai 2018, le site a été complètement fermé et les URL du domaine epinions.com sont redirigées vers Shopping.com.

RED : (*Rich Epinions Dataset for Recommender Systems*) est un jeu de données extrait d'Epinions et enrichi. Il contient des critiques d'utilisateurs sur les éléments, les valeurs de confiance entre les utilisateurs, la catégorie d'éléments, la hiérarchie des catégories et l'expertise des utilisateurs sur les catégories. Cet ensemble de données peut être utilisé pour évaluer divers systèmes de recommandation.

Architecture du dataset

Le dataset est une base de données relationnelle avec les tables suivantes :

- Utilisateur : nom (pseudo et URL du profil), location, rang (peut être nul) et les visites de profil.
- Item : nom, catégorie et URL du profil.
- Catégorie : nom, catégorie parent, url de description, lignage (chemin dans l'arborescence des catégories) et profondeur (dans l'arborescence des catégories).
- Critique : une critique associe un utilisateur à un élément, elle contient la note, entre 1 et 5, la note de la revue (moyenne de toutes les notes associées à cette critique) et la date de révision.
- Expertise : les utilisateurs experts dans une catégorie apparaissent ici avec l'expertise (responsable de la catégorie, examinateur principal, conseiller) associée à la catégorie considérée.
- Trust : Web de confiance, c'est-à-dire une valeur de confiance (-1 ou 1) d'un utilisateur

à un autre, seules les valeurs de confiance positives apparaissent dans le jeu de données.

- Similarité : la similarité entre tous les couples d'utilisateurs avec la corrélation de coefficient de Pearson.

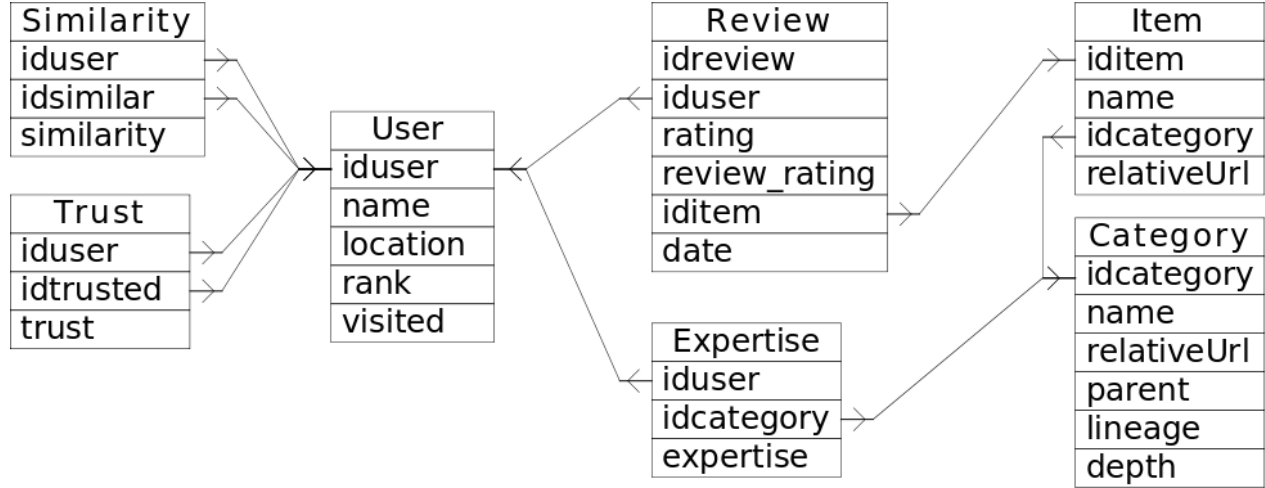


FIGURE 4.6 – Architecture de la base de données RED

Exploitation du dataset

Nous avons pris un échantillon du dataset RED, qui est formé de 100K ratings de 149 utilisateurs sur 5000 items, et nous avons divisé cette échantillon en deux ensemble de données, un ensemble d'apprentissage environ 80% de l'échantillon et un ensemble de 20% pour les testes.

4.4.2 Métriques d'évaluation

Nous nous baserons sur deux métriques afin d'évaluer notre travail : MAE et RMSE cité dans le chapitre précédant (voir formule 3.10), qui sont des métriques permettant de calculer la différence entre deux variables continues. En effet, ces deux métriques restent les plus utilisées dans le domaine et les plus pertinentes quant à évaluer la qualité d'un système de recommandation.

4.4.3 Résultats des évaluations

Dans cette partie, nous réaliserons un ensemble de tests portant sur les performances de notre approche et nous comparerons nos résultats avec les résultats des approches existantes.

A) Évaluations avec MovieLens-100K

1) Filtrage collaboratif

FC sans classification

Dans cette phase d'expérimentations, nous allons évaluer la qualité de la recommandation lorsque celle-ci est basée sur un filtrage collaboratif standard (user-user, item-item, SVD). Nous ferons des expérimentations par rapport à la corrélation de Pearson, étant donné que celle-ci reste l'une des plus connues et des plus utilisées. Nous avons évalué les algorithmes suivants :

- FC item-item avec un prétraitement la moyenne des évaluations des utilisateur (FC-item-item-UA).
- FC item-item avec un prétraitement la moyenne des évaluations des items (FC-item-item-IA).
- FC item-item avec un prétraitement la méthode SVD (FC-item-item-SVD).
- FC item-item sans prétraitement (FC-item-item).
- FC user-user avec un prétraitement la moyenne des évaluations des utilisateur (FC-user-user-UA).
- FC user-user avec un prétraitement la moyenne des évaluations des items (FC-user-user-IA).
- FC user-user avec un prétraitement la méthode SVD (FC-user-user-SVD).
- FC user-user sans prétraitement (FC-user-user).

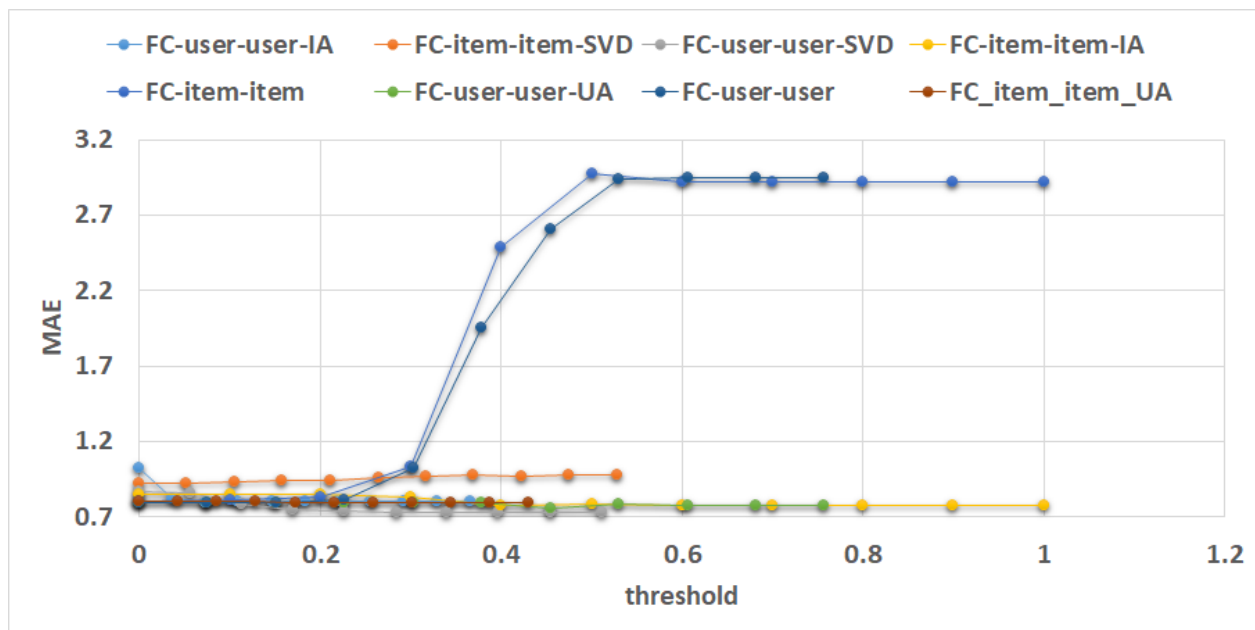


FIGURE 4.7 – Évaluation MAE du FC standard en fonction du seuil

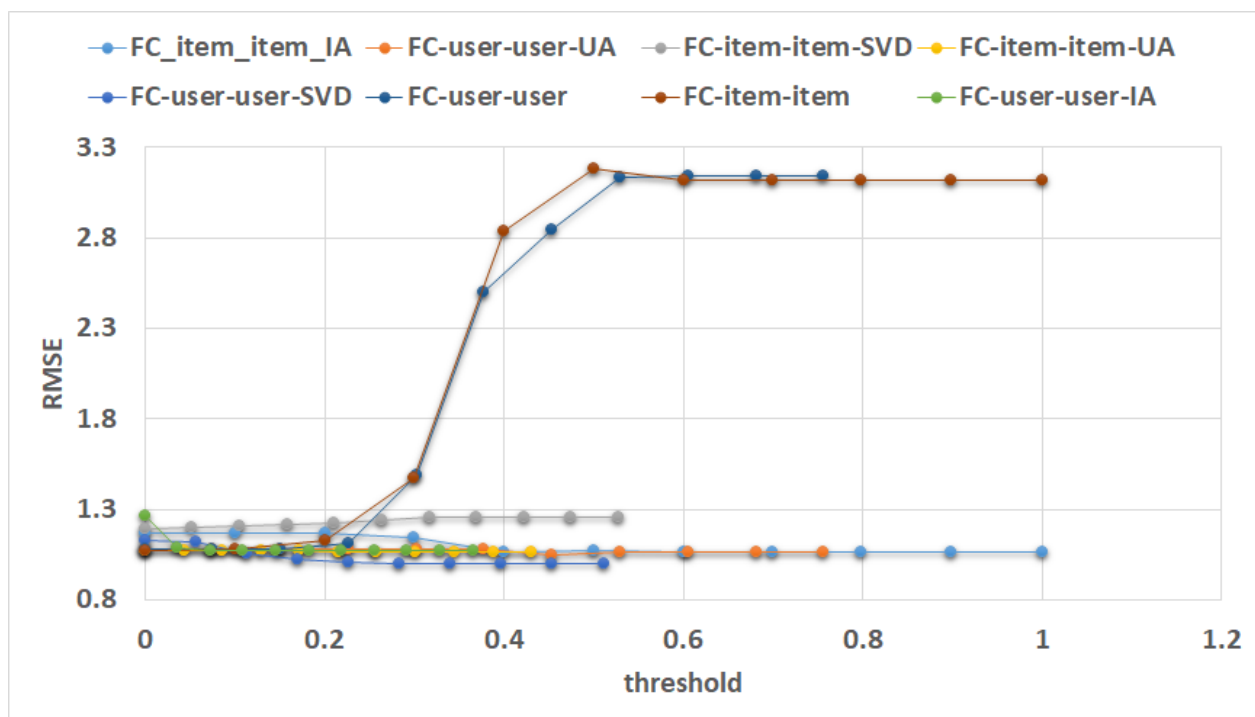


FIGURE 4.8 – Évaluation RMSE du FC standard en fonction du seuil

| | | | | | | | |
|--------------------|--------------|-------|-------|---------------------|-------|-------|-------|
| threshold | 0.258 | 0.345 | 0.301 | threshold | 0.073 | 0.109 | 0.256 |
| MAE (FC-item-UA) | 0.796 | 0.797 | 0.798 | MAE (FC-user-IA) | 0.798 | 0.799 | 0.8 |
| RMSE (FC-item-UA) | 1.067 | 1.067 | 1.068 | RMSE (FC-user-IA) | 1.069 | 1.07 | 1.07 |
| threshold | 0.284 | 0.227 | 0.17 | threshold | 0.4 | 0.6 | 0.5 |
| MAE (FC-user-SVD) | 0.73 | 0.736 | 0.75 | MAE (FC-item-IA) | 0.775 | 0.78 | 0.782 |
| RMSE (FC-user-SVD) | 1.001 | 1.007 | 1.023 | RMSE (FC-item-IA) | 1.064 | 1.066 | 1.07 |
| threshold | 0.454 | 0.606 | 0.53 | threshold | 0.151 | 0.227 | 0.303 |
| MAE (FC-user-ua) | 0.761 | 0.78 | 0.781 | MAE (FC-user-user) | 0.795 | 0.81 | 1.022 |
| RMSE (FC-user-ua) | 1.047 | 1.066 | 1.066 | RMSE (FC-user-user) | 1.079 | 1.111 | 1.492 |
| threshold | 0 | 0.105 | 0.158 | threshold | 0 | 0.1 | 0.2 |
| MAE (FC-item-SVD) | 0.923 | 0.93 | 0.94 | MAE (FC-item-item) | 0.804 | 0.808 | 0.829 |
| RMSE (FC-item-SVD) | 1.197 | 1.207 | 1.217 | RMSE (FC-item) | 1.076 | 1.085 | 1.133 |

TABLE 4.1 – Récapitulatif des trois meilleurs résultats (MAE et RMSE) pour chaque FC sans classification

Discussion des résultats

En analysant les deux figures ci-dessus, il est clair qu’au delà d’un certain seuil, environ 0.5, la qualité de la prédiction stagne pour les différent types de FC.

On remarque que lorsque le dataset n’est pas prétraité les résultats sont de qualité médiocre, ce qui confirme l’importance du prétraitement du dataset, les meilleures valeurs de MAE et RMSE sont au alentour de respectivement 0.80 et 1.2 dans le cas de FC user-user et FC item-item.

Les deux filtrages collaboratif user-user avec comme prétraitement la moyennes des items (FC-user-user-IA) et FC item-item avec prétraitement la moyenne des users (FC-item-item-UA), donnent des résultat presque similaire environ 0.79 pour le meilleur MAE et 1.06 pour RMSE.

L’utilisation de la technique de SVD pour le prétraitement et l’application d’un FC item-item par la suite (FC-item-item-SVD), donne de très mauvais résultats, donc cette technique

est à exclure pour la suite des évaluations avec classification.

Le filtrage qui a donné les meilleures évaluations est FC user-user (FC-user-user-SVD) avec comme prétraitement SVD, et la meilleure valeur de MAE est égale à 0.73 et RMSE égale à 1.001, pour un seuil de 0.284, donc nous allons opter pour ce filtrage dans la suite des évaluations.

FC Avec classification

Pour cette approche, nous ferons varier la valeur de k dans les algorithmes de FC avec classification, et essayer de déduire laquelle de ses valeurs nous permettra d'atteindre les meilleurs résultats. Nous avons évalué les algorithmes suivants :

- FC user-user basé K-medoids (FC-user-user-Kmedoids).
- FC user-user basé K-medoids et BSO (FC-user-user-Kmedoids-BSO).
- FC user-user basé K-KNN (FC-user-user-KNN).

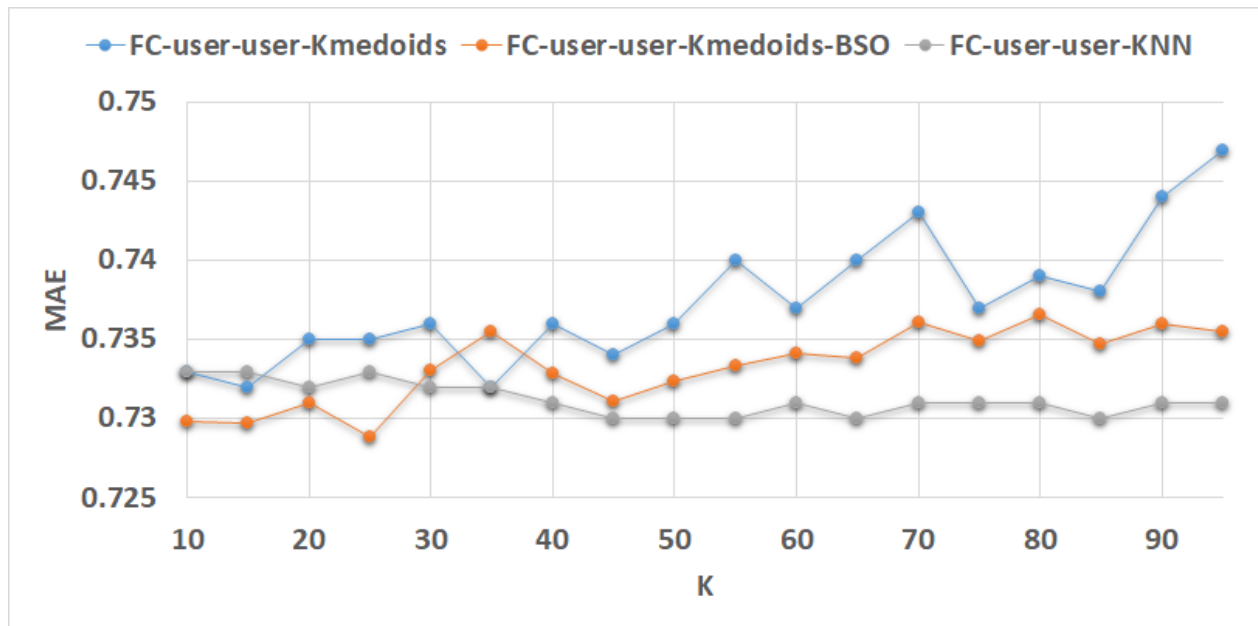


FIGURE 4.9 – Évaluation MAE du FC avec classification en fonction du seuil

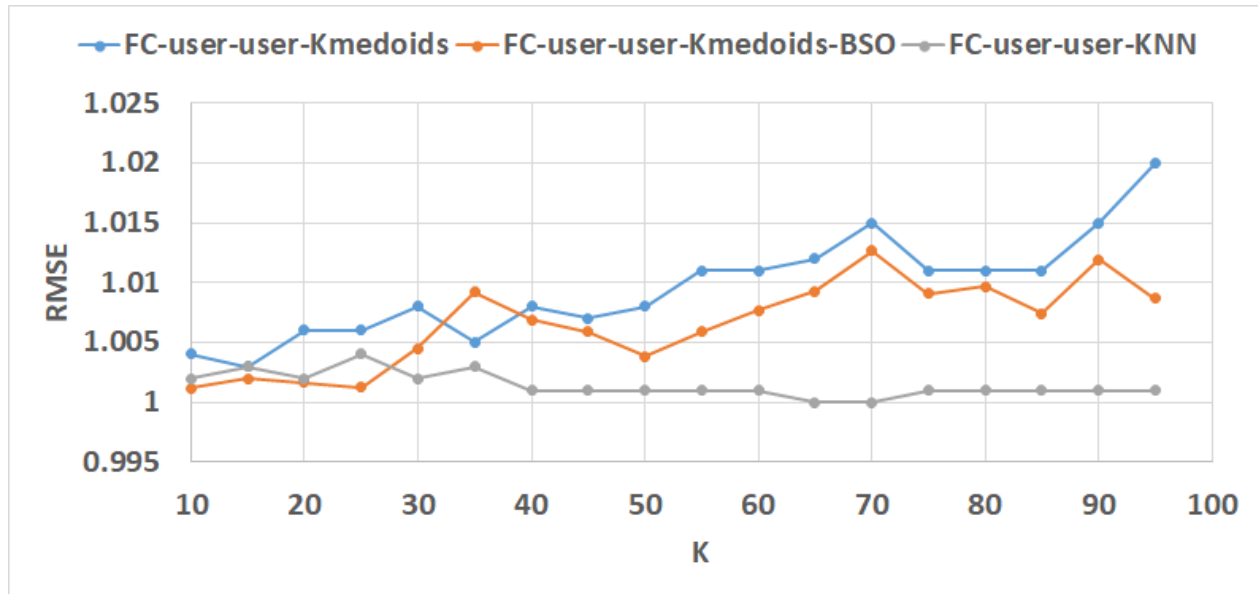


FIGURE 4.10 – Évaluation RMSE du FC avec classification en fonction du seuil

| | | | |
|-----------------------|--------------|-------|-------|
| threshold | 0.284 | 0.227 | 0.17 |
| MAE (FC-user-SVD) | 0.73 | 0.736 | 0.75 |
| RMSE (FC-user-SVD) | 1.001 | 1.007 | 1.023 |
| K | 10 | 15 | 45 |
| MAE(FC-kmedoids) | 0.733 | 0.732 | 0.734 |
| RMSE(FC-svd-kmedoids) | 1.004 | 1.003 | 1.007 |
| K | 25 | 15 | 20 |
| MAE(FC-kmedoids-bso) | 0.728 | 0.729 | 0.73 |
| RMSE(FC-kmedoids-bso) | 1.001 | 1.001 | 1.001 |
| K | 50 | 40 | 30 |
| MAE(FC-KNN) | 0.73 | 0.731 | 0.732 |
| RMSE(FC-KNN) | 1.001 | 1.001 | 1.002 |

TABLE 4.2 – Récapitulatif des trois meilleurs résultats (MAE et RMSE) pour l’algorithme de FC avec classification

Discussion des résultats

Comme nous pouvons le constater sur les figures 4.9 et 4.10 ci-dessus, la qualité de la prédiction évolue selon le nombre de voisins choisis. En prenant plus de 85 voisins la qualité semble diminuer pour les trois algorithmes.

L’ajout du clustering au FC avec comme prétraitement SVD ne semble pas améliorer les résultats car la meilleure valeur de MAE obtenue est 0.733 pour FC basé Kmedoids alors que

FC avec SVD elle était égale à 0.73, on suppose que le choix aléatoire des medoids est la cause de la non amélioration des évaluations, de ce fait une optimisation du choix des medoids est nécessaire pour voir l'apport de classification.

Nous pouvons constater que le FC basé Kmedoids-BSO améliore les résultats de FC basé K-medoids et FC standard, avec MAE égale à 0.728, ce qui confirme l'apport de l'optimisation sur le clustering et donc on peut dire que le clustering peut améliorer les résultats d'un FC standard si le jeu de teste est d'un volume plus important.

En ce qui concerne le FC basé K-NN, il n'améliore pas la qualité des évaluations du FC standard mais donne de résultats plus au moins stable avec MAE ne dépassant pas les 0.74 et RMSE les 1.005.

2) Filtrage sémantique standard

Dans cette phase d'expérimentations, nous allons évaluer la qualité de la recommandation lorsque celle-ci est basée sur un filtrage sémantique. Nous ferons des expérimentations en utilisant la formule de distance d'intérêt entre utilisateur, et la distance de Jaccard entre les items car le dataset MovieLense ne possède pas une représentation sous forme d'ontologie des items. Le prétraitement des valeurs manquantes est effectué avec la méthode SVD.

FSem sans classification

Nous avons évalué l'algorithme FSem user-user.

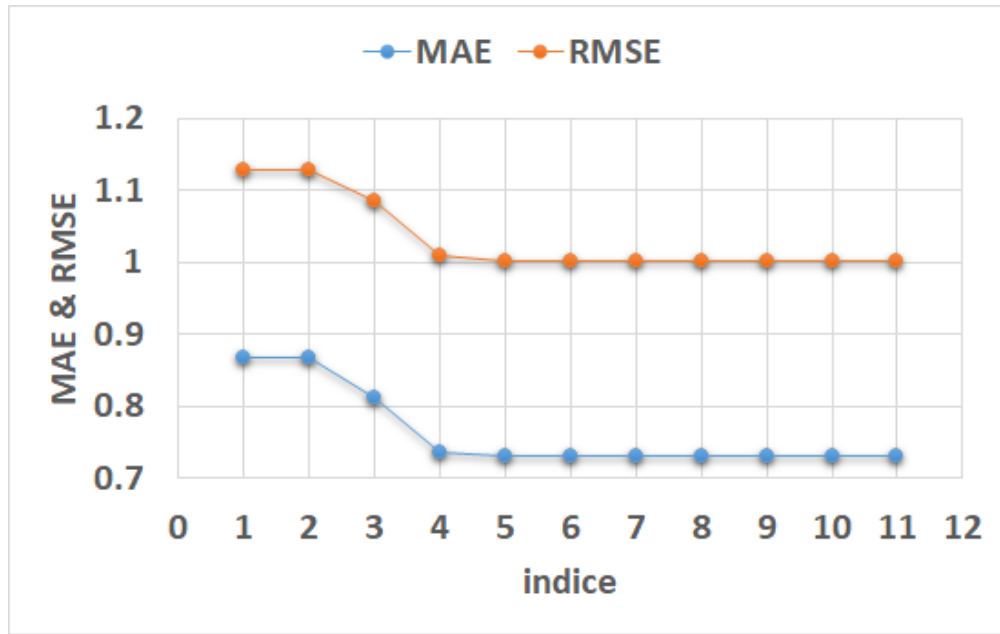


FIGURE 4.11 – Évaluation MAE et RMSE du FSem en fonction du seuil de similarité

Afin de ne pas encombré le graphe ci-dessus (4.11) nous avons indexé les onze valeurs des seuils du FSem et nous les avons mis dans le tableau suivant :

| indice | threshold |
|--------|----------------------|
| 1 | 0.66666666666666 564 |
| 2 | 0.66666666666666 592 |
| 3 | 0.66666666666666 62 |
| 4 | 0.66666666666666 647 |
| 5 | 0.66666666666666 675 |
| 6 | 0.66666666666666 703 |
| 7 | 0.66666666666666 731 |
| 8 | 0.66666666666666 758 |
| 9 | 0.66666666666666 786 |
| 10 | 0.66666666666666 814 |
| 11 | 0.66666666666666 842 |

TABLE 4.3 – Les valeurs de seuils correspondant aux indices du graphes 4.11

| indice | 5 | 6 | 9 |
|-----------------------|---------------|--------|--------|
| MAE (FSem-user-user) | 0.7297 | 0.7306 | 0.7303 |
| RMSE (FSem-user-user) | 1.0012 | 1.0022 | 1.0020 |

TABLE 4.4 – Récapitulatif des trois meilleurs résultats (MAE et RMSE) pour l’algorithme de FSem sans classification

Discussion des résultats

En analysant la figure 4.11 ci-dessus, il est clair qu'au delà d'un certain seuil, plus précisément le seuil possédant l'indice 6, la qualité de la prédiction stagne au alentour de 0.73 pour MAE et 1.002 pour RMSE.

On constate que la meilleure valeur du seuil à choisir dans le FSem standard est le seuil possédant l'indice 5, qui a donné MAE égale à 0.729 et RMSE égale à 1.001.

FSem avec classification

Pour cette approche, nous ferons varier la valeur de K dans les algorithmes de FSem avec classification, et essayer de déduire laquelle de ses valeurs nous permettra d'atteindre les meilleurs résultats. Nous avons évalué les algorithmes suivant :

- FSem user-user basé K-medoids (FSem-user-user-Kmedoids).
- FSem user-user basé K-medoids et BSO (FSem-user-user-Kmedoids-BSO).
- FSem user-user basé K-NN (FSem-user-user-KNN).

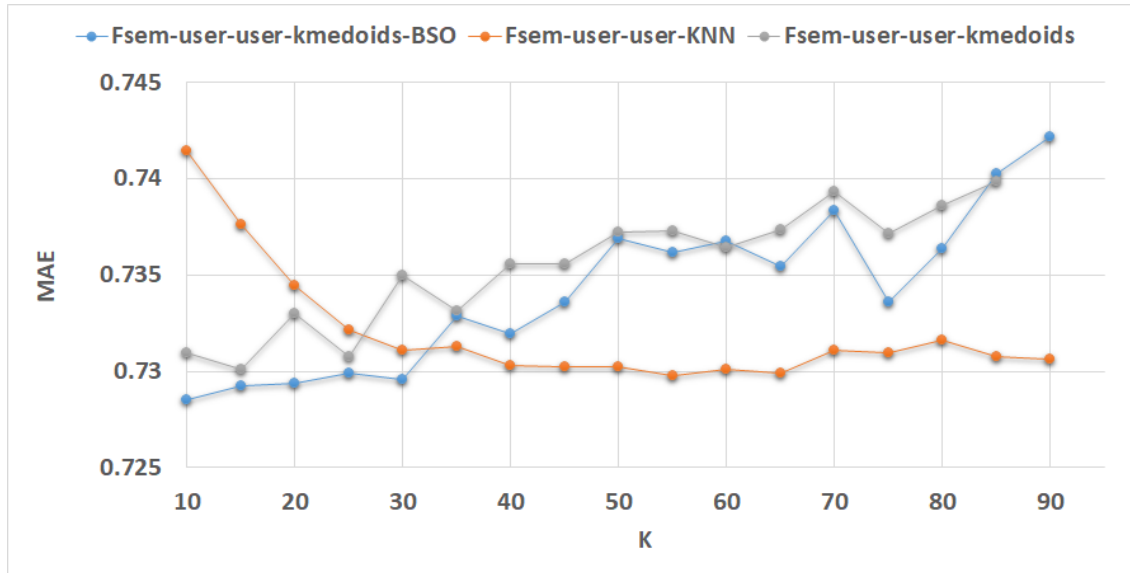


FIGURE 4.12 – Évaluation MAE avec les algorithmes de FSem basés classification

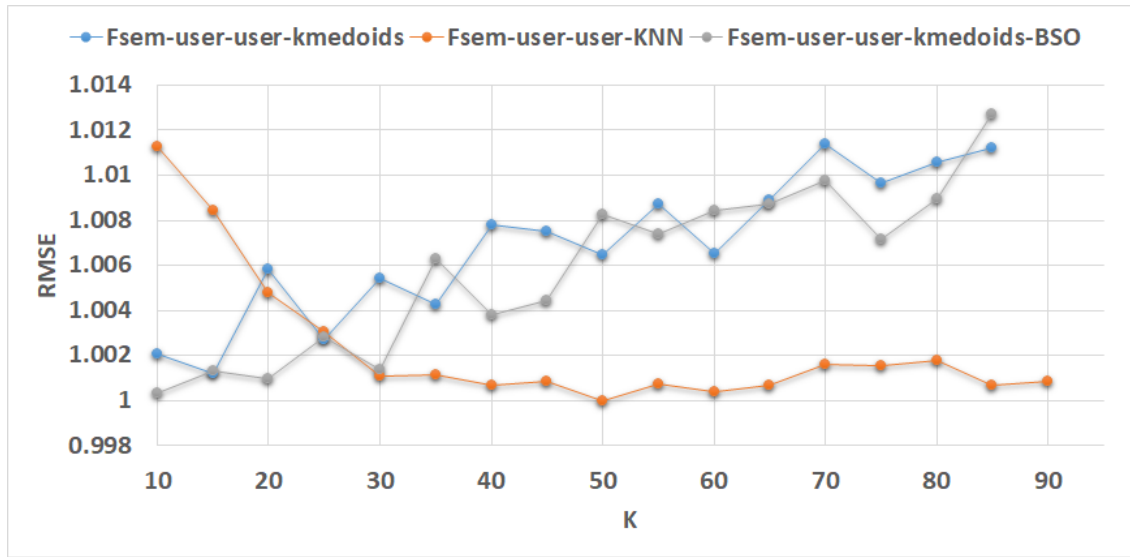


FIGURE 4.13 – Évaluation RMSE avec les algorithmes de FSem basés classification

| | | | |
|------------------------------------|---------------|--------|--------|
| indice | 5 | 6 | 9 |
| MAE (FSem-user-user) | 0.7297 | 0.7306 | 0.7303 |
| RMSE (FSem-user-user) | 1.0012 | 1.0022 | 1.0020 |
| k | 10 | 15 | 20 |
| MAE (FSem-user-user-kmedoids-bso) | 0.728 | 0.7292 | 0.7293 |
| RMSE (FSem-user-user-kmedoids-bso) | 1.002 | 1.0012 | 1.005 |
| k | 55 | 65 | 60 |
| MAE (Fsem-user-user-KNN) | 0.7297 | 0.7299 | 0.7301 |
| RMSE (Fsem-user-user-KNN) | 1.0007 | 1.0006 | 1.0003 |
| k | 15 | 25 | 10 |
| MAE (FSem-user-user-kmedoid) | 0.7301 | 0.7307 | 0.7309 |
| RMSE (FSem-user-user-kmedoid) | 1.0012 | 1.0027 | 1.002 |

TABLE 4.5 – Récapitulatif des trois meilleurs résultats (MAE et RMSE) pour l’algorithme de FSem avec classification

Discussion des résultats

Comme nous pouvons le constater sur les figures 4.12 et 4.13 ci-dessus, la qualité de la prédiction évolue selon le nombre de voisins choisis. En prenant plus de 85 voisins la qualité semble diminuer pour les trois algorithmes.

L'ajout du clustering au FSem avec comme prétraitement SVD ne semble pas améliorer les résultats car la meilleure valeur de MAE obtenu est 0.730 pour FSem basé Kmedoids, alors que le FSem standard avait donné MAE égale à 0.729, on suppose que le choix aléatoire des medoids est la cause de la non amélioration des évaluations, de ce fait une optimisation du choix des medoids est nécessaire pour voir l'apport de classification.

Après l'ajout de l'optimisation BSO au FSem basé K-medoids, cela a permis d'améliorer MAE de 0.001, cependant FSem standard reste mieux que FSem basé clustering optimisé.

Le FSem basé K-NN, ne semble pas apporter des améliorations au FSem standard car les deux possèdent des évaluations plus ou moins égaux.

3) Filtrage hybride

Dans cette phase d'expérimentations, nous allons évaluer la qualité de la recommandation lorsque celle-ci est basée sur un filtrage hybride entre FC et FSem à savoir FHyb pondéré (FHyb-alpha), FHyb collaboratif basé sémantique (FHyb-FC-Sem) et FHyb sémantique basé collaboratif (FHyb-Sem-FC).

FHyb sans classification

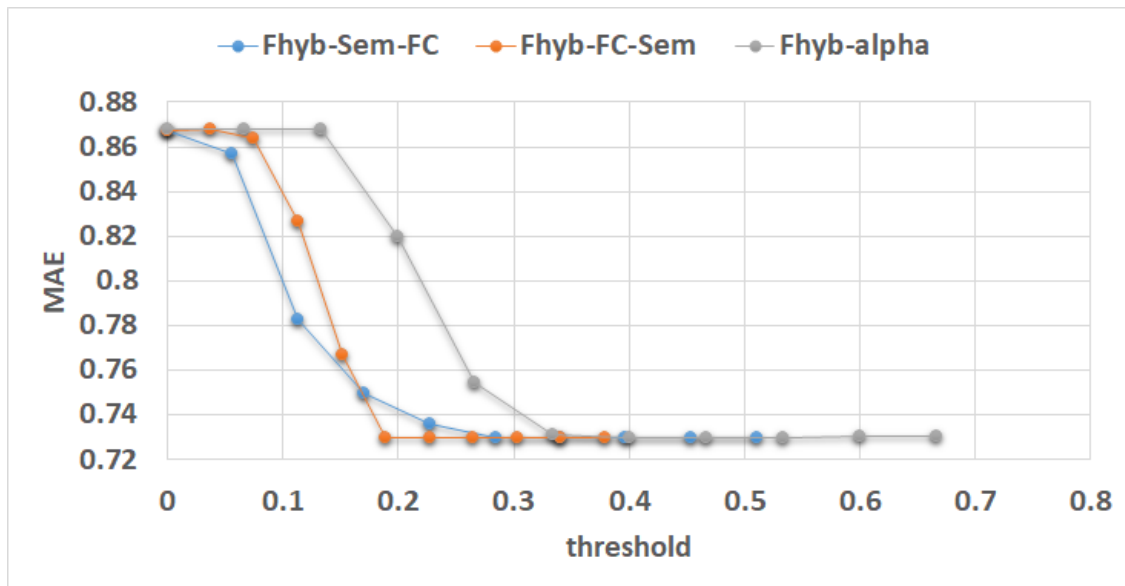


FIGURE 4.14 – Évaluation MAE de l'hybridation sans classification des algorithmes FC et FSem

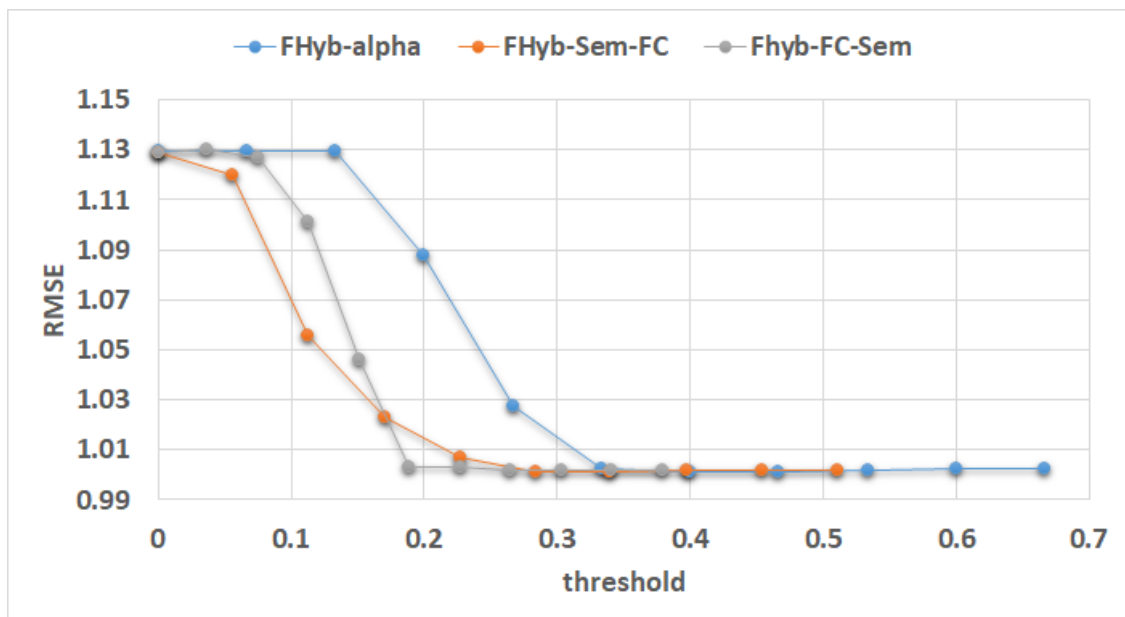


FIGURE 4.15 – Évaluation RMSE de l'hybridation sans classification des algorithmes FC et FSem

Le tableau suivant présente les meilleures valeurs d'Alpha :

| threshold | Best Alpha for threshold |
|-------------|--------------------------|
| 0 | 1 |
| 0.066666667 | 1 |
| 0.133333333 | 1 |
| 0.2 | 8 |
| 0.266666667 | 8 |
| 0.333333333 | 8 |
| 0.4 | 7 |
| 0.466666667 | 5 |
| 0.533333333 | 5 |
| 0.6 | 5 |
| 0.666666667 | 5 |

TABLE 4.6 – Tableau des differents seuils combiné à Alpha du FHyb pondéré

Le tableau suivant 4.7 décrit les trois meilleures valeurs obtenues pour chaque algorithme en termes de MAE et RMSE.

| | | | |
|--------------------|---------------|--------|--------------|
| threshold | 0.189 | 0.151 | 0.113 |
| MAE (FHyb-FC-Sem) | 0.73 | 0.767 | 0.827 |
| RMSE (FHyb-FC-Sem) | 1.003 | 1.046 | 1.101 |
| threshold | 0.284 | 0.227 | 0.17 |
| MAE (FHyb-Sem-FC) | 0.73 | 0.736 | 0.75 |
| RMSE (FHyb-Sem-FC) | 1.001 | 1.007 | 1.023 |
| threshold | 0.466 | 0.533 | 0.6 |
| Alpha | 0.5 | 0.5 | 0.5 |
| MAE (FHyb-alpha) | 0.7295 | 0.73 | 0.7301 |
| RMSE (FHyb-alpha) | 1.0014 | 1.0018 | 1.002 |

TABLE 4.7 – Récapitulatif des trois meilleurs résultats (MAE et RMSE) pour le FHyb pondéré

Discussion des résultats

Nous pouvons déduire de ces résultats que les trois hybridations donnent des résultats approximativement proches avec comme meilleure valeur de MAE égale à 0.73 et RMSE égale à 1. Cependant, le FHyb pondéré a donné la meilleure valeur de MAE qui est égale à 0.729 avec la valeur Alpha égale à 5.

FHyb avec classification

Pour cette approche, nous ferons varier la valeur du nombre de clusters K dans les algorithmes suivants :

- FHyb pondéré K-NN (FHyb-alpha).
- FHyb pondéré K-medoids (FHyb-alpha-Kmedoids).
- FHyb collaboratif basé sémantique K-NN (FHyb-FC-Sem-KNN).
- FHyb collaboratif basé sémantique K-medoids (FHyb-FC-Sem-Kmedoids).
- FHyb collaboratif basé sémantique K-medoids et BSO (FHyb-FC-Sem-Kmedoids-BSO).
- FHyb sémantique basé collaboratif K-NN (FHyb-Sem-FC-KNN).
- FHyb sémantique basé collaboratif K-medoids (FHyb-Sem-FC-Kmedoids).
- FHyb sémantique basé collaboratif K-medoids et BSO (FHyb-Sem-FC-Kmedoids-BSO).
- FHyb Multivues K-NN (FHyb-Multiview-KNN).

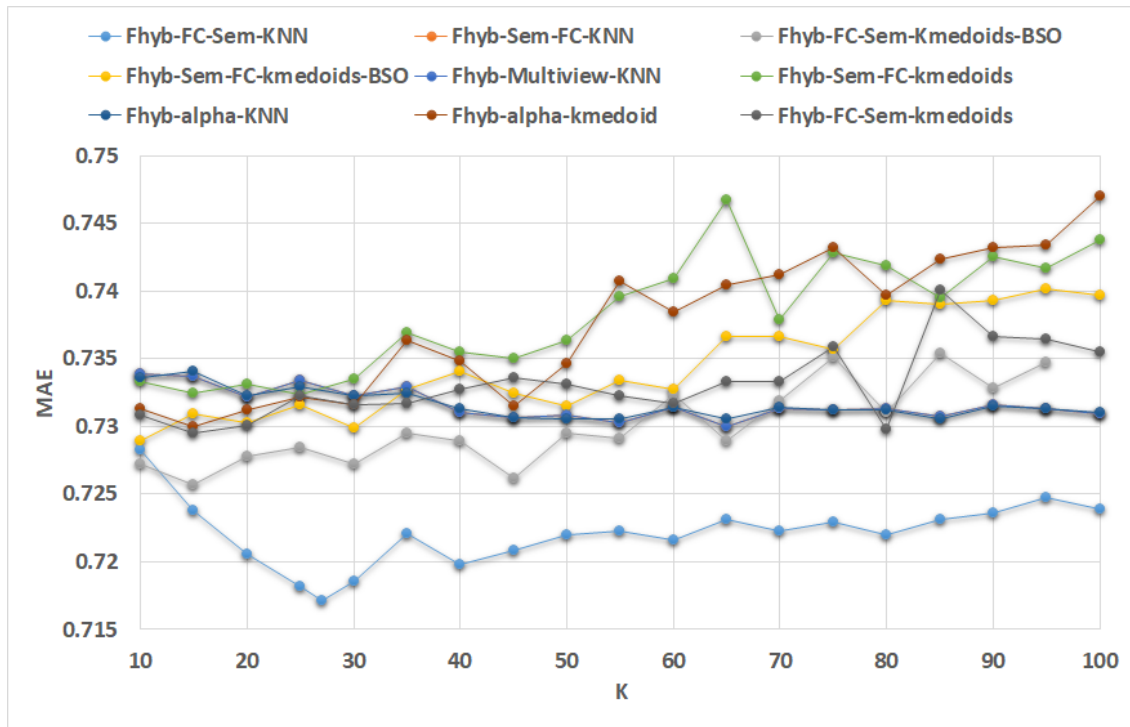


FIGURE 4.16 – Évaluation MAE des algorithmes hybrides basés classification

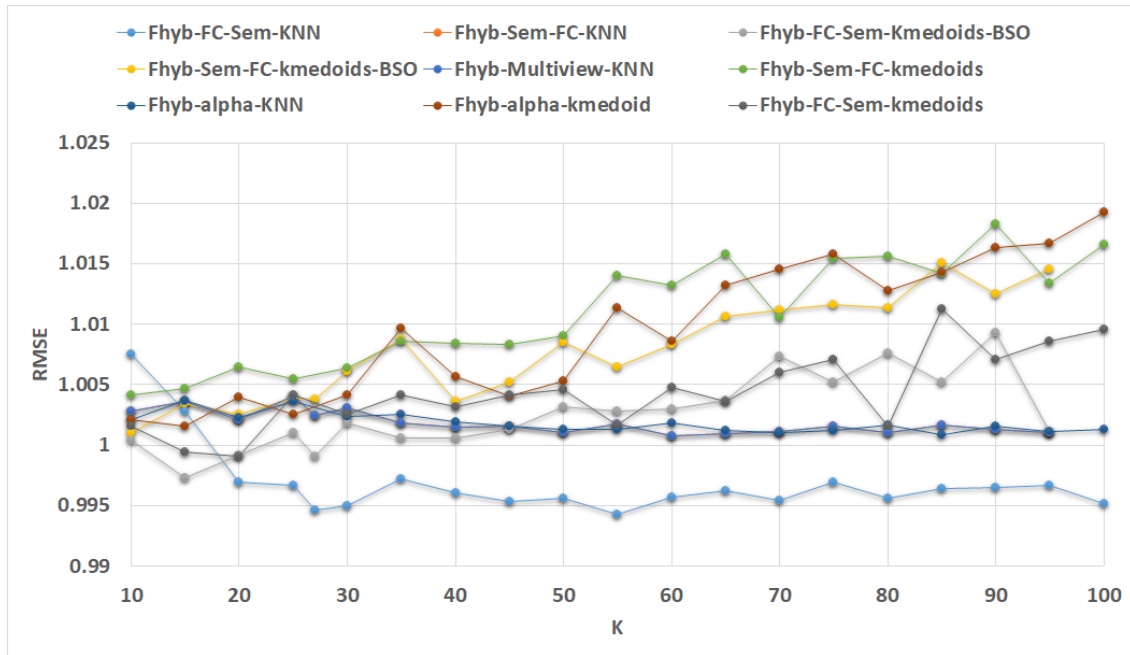


FIGURE 4.17 – Évaluation RMSE des algorithmes hybrides basés classification

Le tableau suivant décrit les trois meilleures valeurs obtenues pour chaque algorithme en termes de MAE et RMSE.

| | | | |
|--------------------------------|---------------|--------|---------|
| k | 27 | 25 | 30 |
| MAE(FHyb-FC-Sem-KNN) | 0.7171 | 0.7181 | 0.7185 |
| RMSE(FHyb-FC-Sem-KNN) | 0.9946 | 0.9967 | 0.9950 |
| k | 15 | 45 | 30 |
| MAE(FHyb-Sem-FC-KNN) | 0.7256 | 0.7261 | 0.7272 |
| RMSE(FHyb-Sem-FC-KNN) | 1.003 | 1.0015 | 1.0030 |
| k | 65 | 55 | 45 |
| MAE(FHyb-FC-Sem-Kmedoids-BSO) | 0.73001 | 0.7302 | 0.73064 |
| RMSE(FHyb-FC-Sem-Kmedoids-BSO) | 1.0029 | 1.0029 | 1.0013 |
| k | 10 | 15 | 20 |
| MAE(FHyb-Sem-FC-Kmedoids-BSO) | 0.7289 | 0.7309 | 0.7302 |
| RMSE(FHyb-Sem-FC-Kmedoids-BSO) | 1.001 | 1.0034 | 1.0025 |
| k | 65 | 55 | 45 |
| MAE(multiview_knn_svd) | 0.7300 | 0.7302 | 0.7306 |
| RMSE(multiview_knn_svd) | 1.0009 | 1.0017 | 1.0015 |
| k | 15 | 80 | 20 |
| MAE(FHyb-Multiview-KNN) | 0.7294 | 0.7298 | 0.730 |
| RMSE(FHyb-Multiview-KNN) | 0.9994 | 1.0015 | 0.9990 |
| k | 25 | 15 | 20 |
| MAE(FHyb-Sem-FC-Kmedoids) | 0.7323 | 0.7324 | 0.733 |
| RMSE(FHyb-Sem-FC-Kmedoids) | 1.005 | 1.004 | 1.0065 |
| k | 140 | 120 | 135 |
| alpha | 0.1 | 0.4 | 0.6 |
| MAE(FHyb-alpha-KNN) | 0.7287 | 0.7289 | 0.7289 |
| RMSE(FHyb-alpha-KNN) | 0.9991 | 0.9991 | 0.9994 |
| k | 15 | 20 | 10 |
| alpha | 0.1 | 0.2 | 0.3 |
| MAE(FHyb-alpha-Kmedoids) | 0.7300 | 0.731 | 0.7312 |
| RMSE(FHyb-alpha-Kmedoids) | 1.0015 | 1.0039 | 1.0021 |

TABLE 4.8 – Récapitulatif des trois meilleurs résultats (MAE et RMSE) pour les différents FHyb avec classification

Discussion des résultats

FHyb pondéré

Pour ce filtrage l'ajout du clustering n'a pas amélioré les performances de la prédiction, car le FHyb pondéré donne comme meilleur résultat 0.729 pour MAE et 1.001 pour RMSE, après l'ajout du clustering, MAE est passée à 0.730 et RMSE est restée presque la même. Cependant l'ajout de la classification K-NN, a amélioré les performances de MAE qui est devenu 0.7287 et de RMSE qui est passé à 0.99.

FHyb sémantique basé collaboratif

Après avoir incorporer le clustering K-medoids à ce filtrage, nous constatons que les évaluations ne se sont pas améliorées, car sans clustering MAE été égale à 0.73 et RMSE à 1.00, après l'ajout du clustering MAE est passée à 0.732 et RMSE est restée la même, donc on suppose que le choix des medoids d'une façon optimisée peut montrer l'apport du clustering, après avoir ajouté l'optimisation BSO, RMSE n'a pas changé mais MAE s'est améliorée et est passée à 0.728, donc le clustering optimisé est efficace sur ce filtrage. Après l'ajout de la classification avec K-NN à ce filtrage, nous remarquons que MAE s'est améliorée et est devenu égale à 0.725, pour RMSE pas de changement à noter, donc la classification pour ce filtrage améliore la qualité de MAE seulement.

FHyb collaboratif basé sémantique

Ce filtrage sans classification a donnée comme meilleur résultat pour MAE=0.73 et RMSE=1.00, après l'ajout du clustering, MAE c'est améliorée et est passé à 0.729 et RMSE à 0.999 mais l'ajout de l'optimisation ne semble pas améliorer les performances de ces deux évaluations car elles ont passée respectivement à 0.73 et 1.00. Cependant l'ajout de la classification semble être la meilleure option à choisir pour ce filtrage car elle a fait passé MAE à 0.717 et RMSE à 0.994, donc elle est plus efficace que le clustering et le clustering optimisé.

FHyb Multivues K-NN

Ce filtrage utilise que la classification K-NN pour hybrider les deux vue sémantique et collaboratif, nous pouvons constaté qu'il a donné une valeur de 0.73 pour MAE et une valeur de 1.00 pour RMSE, donc pour cette base de donnée cette approche ne semble pas meilleure que les filtres hybrides standard proposé ou avec classification car nous avons eu des performances mieux que celle-ci. Nous supposons que cette approche serait meilleure testée sur un autre dataset, de ce fait par la suite nous allons testé cette approche sur un autre jeu de données.

Comparaison des résultats avec d'autre méthodes utilisée sur MovieLens

Pour voir si nos résultats sont de bonne qualité, nous avons décidé de les comparés à des méthodes déjà utilisé sur le dataset MovieLens 100k, les résultats peuvent être consultés sur le site [43]. Le tableau suivant est un récapitulatif de nos meilleurs résultats pour chaque

approches développées :

| | MAE | RMSE |
|-----------------------------------|-------|-------|
| FC standard | 0.73 | 1.001 |
| FC basé k-medoids | 0.733 | 1.004 |
| FC basé k-medoids et BSO | 0.728 | 1.001 |
| FC basé K-NN | 0.73 | 1.001 |
| FSem standard | 0.729 | 1.002 |
| FSem k-medoids | 0.730 | 1.001 |
| FSem k-medoids et BSO | 0.728 | 1.002 |
| FSem K-NN | 0.729 | 1.000 |
| FHyb pondéré | 0.729 | 1.001 |
| FHyb pondéré k-medoids | 0.730 | 1.001 |
| FHyb pondéré K-NN | 0.728 | 0.991 |
| FHyb Sem basé FC | 0.73 | 1.001 |
| FHyb Sem basé FC k-medoids | 0.732 | 1.005 |
| FHyb Sem basé FC k-medoids et BSO | 0.728 | 1.001 |
| FHyb Sem basé FC K-NN | 0.725 | 1.003 |
| FHyb FC basé Sem | 0.730 | 1.003 |
| FHyb FC basé Sem k-medoids | 0.729 | 0.999 |
| FHyb FC basé Sem k-medoids et BSO | 0.730 | 1.002 |
| FHyb FC basé Sem K-NN | 0.717 | 0.994 |
| FHyb mutivues K-NN | 0.730 | 1.000 |

TABLE 4.9 – Récapitulatif des meilleurs résultats (MAE et RMSE) pour tous les filtrages conçus

Le tableau suivant montre les résultats obtenu à partir du site [43]. Les deux méthodes utilisés sont SVDplusplus qui est une amélioration de la méthode SVD et UserKNNPearson qui est un filtrage collaboratif avec la corrélation de Pearson comme distance entre utilisateur et une classification des voisins des ces derniers avec l'algorithme K-NN.

| | MAE | RMSE |
|-----------------------|-------|---------|
| UserKNNPearson | 0.728 | 0.929 |
| SVDPlusPlus version 1 | 0.718 | 0.913 |
| SVDPlusPlus version 2 | 0.713 | 0.90829 |

TABLE 4.10 – Les meilleurs résultats (MAE et RMSE) trouver à partir de [43] pour le dataset MovieLens 100k

Nous pouvons conclure que nos résultats sont dans les normes comparé avec ceux déjà développé ou part avant, de plus nous nous rapprochons avec notre méthode FHyb FC basé Sem et K-NN du résultat de SVDPlusPlus version 2 du tableau . Sachant que la méthode de

SVDPlusPlus prend beaucoup de temps pour s'exécutée, il serait possible d'opter pour notre approche qui est FHyb FC basé Sem et K-NN, qui s'exécutera plus rapidement que SVD-PlusPlus, de plus notre approche est simple à implémenter et donne des résultats proches de la méthode SVDPlusPlus.

B) Evaluations avec RED

Afin de confirmer les résultats obtenus de nos approches sur le dataset MovieLens, nous avons décidé de refaire les testes sur un autre dataset qui est RED. Nous avons choisi de refaire les testes que sur les approches qui ont donné de bons résultats, car nous somme limité par le temps et la puissance de calcul de nos machines.

Évaluations

Dans cette partie, nous réaliserons un ensemble de tests portant sur les performances de notre approche et nous comparerons nos résultats avec les résultats des approches existantes.

1) Filtrage collaboratif avec classification et classification optimisée

Dans cette phase d'expérimentations, nous allons évaluer la qualité de la recommandation lorsque celle-ci est basée sur un FC basé K-medoids, FC basé K-medoids et BSO et FC basé K-NN. Nous ferons des expérimentations par rapport à la corrélation de Pearson, étant donné que celle-ci reste l'une des plus connues et des plus utilisées, et comme prétraitement du dataset nous avons opté pour la méthode SVD vu son efficacité d'après les testes précédent.

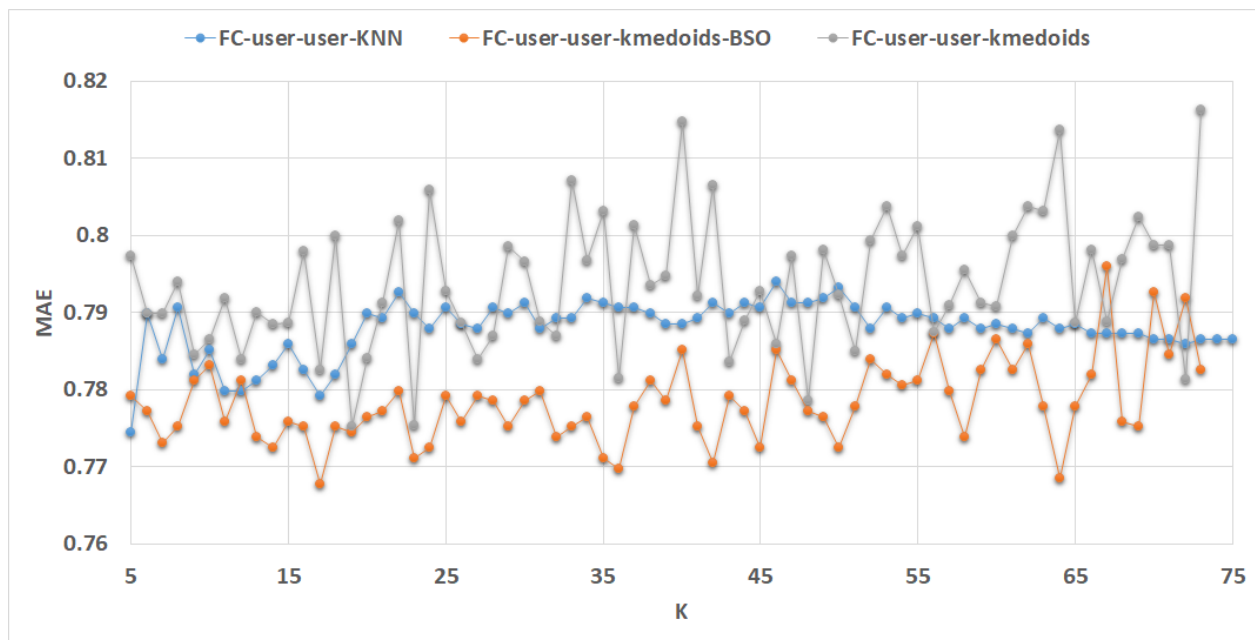


FIGURE 4.18 – Évaluation MAE du FC avec classification sur RED

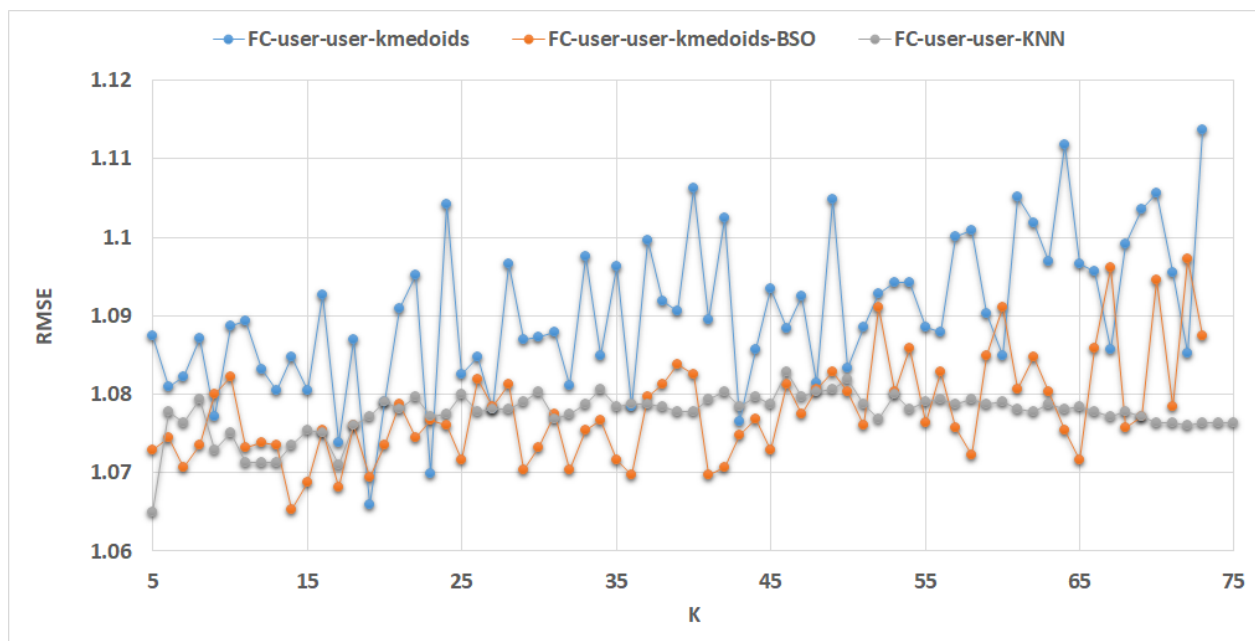


FIGURE 4.19 – Évaluation RMSE du FC avec classification sur RED

Le tableau suivant décrit les trois meilleures valeurs obtenues en termes de MAE et RMSE.

| | | | |
|---------------------------------|---------------|---------|---------|
| k | 5 | 17 | 11 |
| MAE(FC-user-user-KNN) | 0.7744 | 0.779 | 0.7798 |
| RMSE(FC-user-user-KNN) | 1.066 | 1.0709 | 1.0712 |
| k | 17 | 36 | 64 |
| MAE(FC-user-user-Kmedoids-BSO) | 0.7677 | 0.7697 | 0.7684 |
| RMSE(FC-user-user-Kmedoids-BSO) | 1.060 | 1.0697 | 1.07534 |
| k | 19 | 23 | 48 |
| MAE(FC-user-user-Kmedoids) | 0.7751 | 0.7753 | 0.7785 |
| RMSE(FC-user-user-Kmedoids) | 1.0659 | 1.06989 | 1.0814 |

TABLE 4.11 – Récapitulatif des trois meilleurs résultats (MAE et RMSE) pour le FC avec classification du dataset RED

Discussion des résultats

D’après le tableau récapitulatif et les graphes ci-dessus, nous constatons que l’approche qui donne les meilleurs résultats est le FC basé K-medoids et BSO avec MAE égale à 0.767 et RMSE égale à 1.060, ce qui confirme l’apport de l’optimisation sur le clustering.

2) Filtrage sémantique avec classification et classification optimisée

Dans cette partie, nous allons évaluer la qualité de la recommandation lorsque celle-ci est basée sur un FSem basé K-medoids, FSem basé K-medoids et BSO et FSem basé K-NN. Nous ferons des expérimentations en utilisant la formule de distance d’intérêt entre utilisateur, et la distance de Wu and Palmer entre les items car le dataset RED possède une représentation sous forme d’ontologie des items. Le prétraitement des valeurs manquantes est effectué avec la méthode SVD.

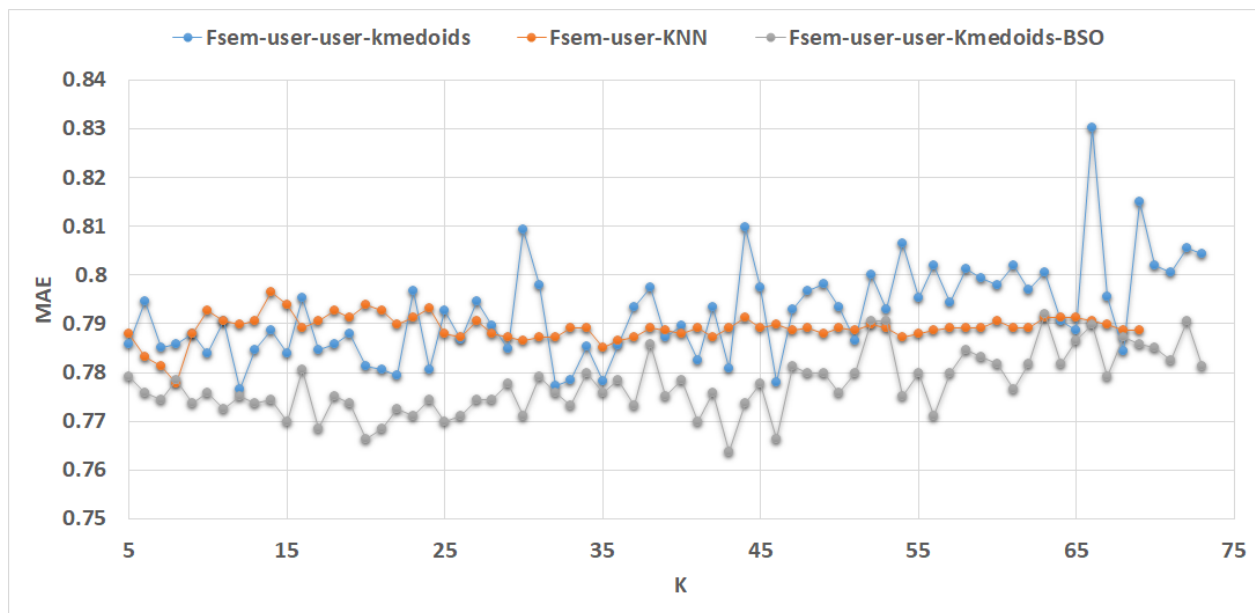


FIGURE 4.20 – Évaluation MAE du FSem avec classification sur RED

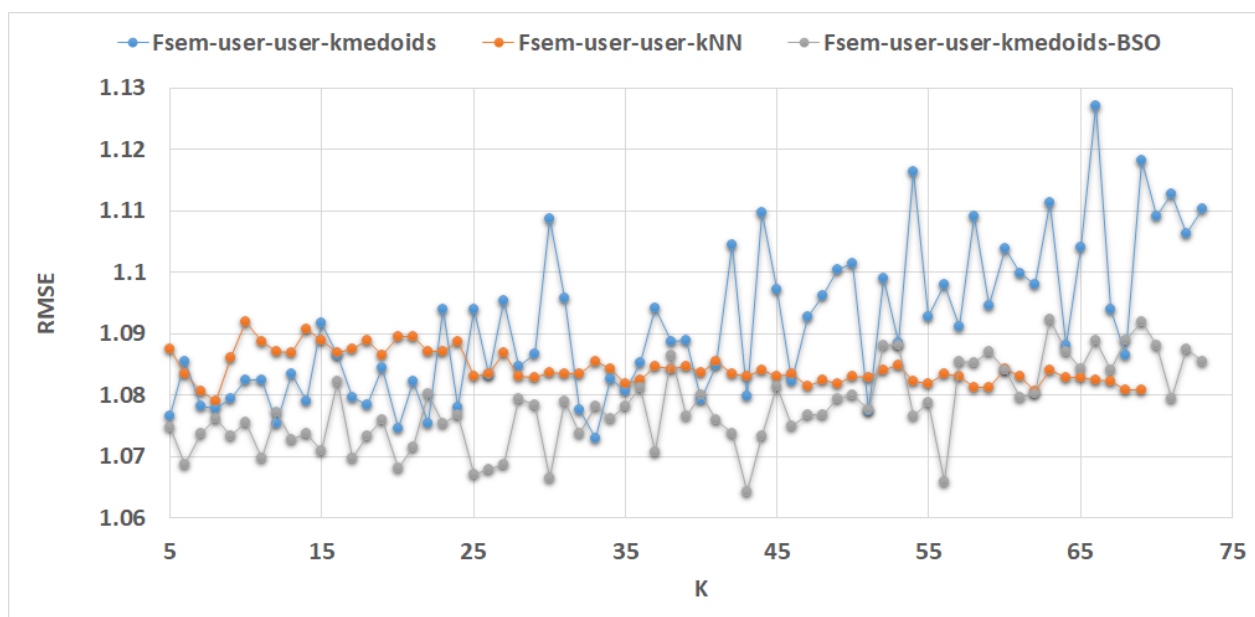


FIGURE 4.21 – Évaluation RMSE du FSem avec classification sur RED

La tableau suivant décrit les trois meilleures valeurs en termes de MAE et RMSE.

| | | | |
|-----------------------------------|---------------|--------|--------|
| k | 12 | 32 | 46 |
| MAE(FSem-user-user-Kmedoids) | 0.7765 | 0.7771 | 0.7779 |
| RMSE(FSem-user-user-Kmedoids) | 1.0753 | 1.0775 | 1.0823 |
| k | 8 | 7 | 6 |
| MAE(FSem-user-user-KNN) | 0.7778 | 0.7812 | 0.7832 |
| RMSE(FSem-user-user-KNN) | 1.0790 | 1.0806 | 1.0834 |
| k | 43 | 20 | 46 |
| MAE(FSem-user-user-Kmedoids-BSO) | 0.7637 | 0.7664 | 0.7664 |
| RMSE(FSem-user-user-Kmedoids-BSO) | 1.064 | 1.0681 | 1.0750 |

TABLE 4.12 – Récapitulatif des trois meilleurs résultats (MAE et RMSE) pour le FSem avec classification du dataset RED

Discussion des résultats

D’après le tableau récapitulatif et les graphes ci-dessus, nous constatons que l’approche qui donne les meilleurs résultats est le FSem basé K-medoids et BSO avec MAE égale à 0.763 et RMSE égale à 1.064, ce qui confirme l’apport de l’optimisation sur le clustering.

3) Filtrage hybride avec classification et classification optimisée

Dans cette partie, nous allons évaluer la qualité de la recommandation lorsque celle-ci est basée sur un FHyb basé K-medoids, FHyb basé K-medoids et BSO et FHyb basé K-NN. Nous ferons des expérimentations par rapport à la corrélation de Pearson, étant donné que celle-ci reste l’une des plus connues et des plus utilisées, et comme prétraitement du dataset nous avons opté pour la méthode SVD vu son efficacité d’après les tests précédents.

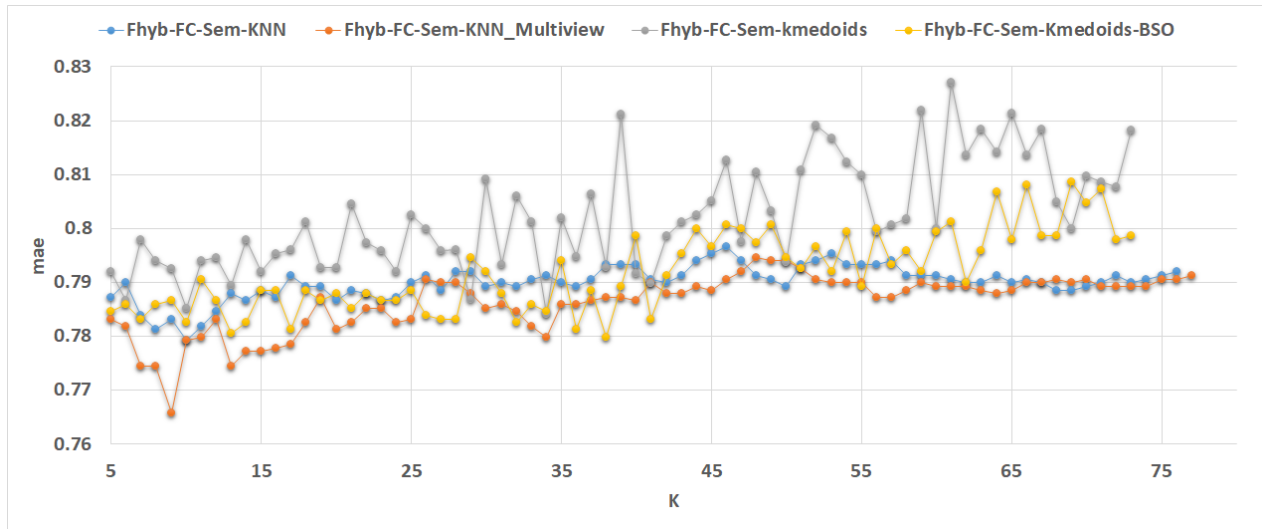


FIGURE 4.22 – Évaluation MAE du filtrage hybride avec classification sur RED

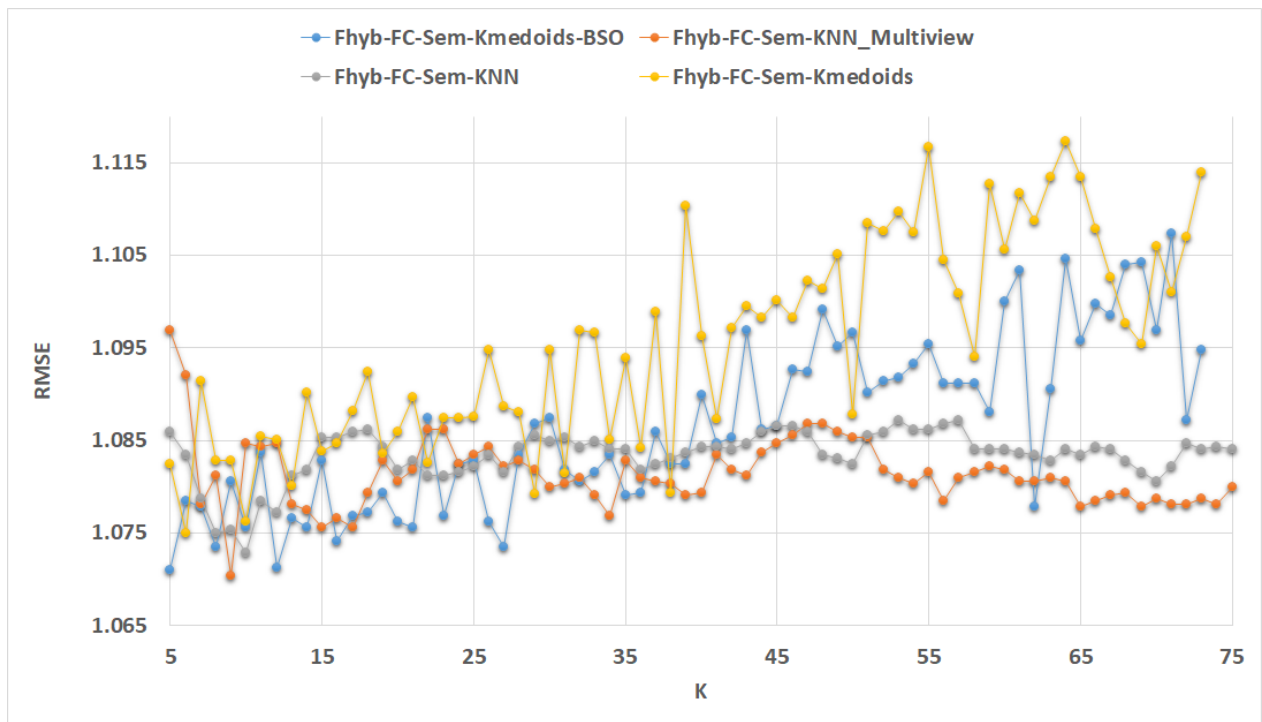


FIGURE 4.23 – Évaluation RMSE du filtrage hybride avec classification sur RED

Le tableau suivant décrit les trois meilleurs valeurs obtenues pour chaque algorithme en termes de MAE et RMSE.

| | | | |
|--------------------------------|---------------|---------|---------|
| k | 10 | 8 | 11 |
| MAE(FHyb-FC-Sem-KNN) | 0.7791 | 0.7812 | 0.7818 |
| RMSE(FHyb-FC-Sem-KNN) | 1.0728 | 1.07503 | 1.07846 |
| k | 9 | 7 | 8 |
| MAE(FHyb-Multiview) | 0.7657 | 0.7744 | 0.7744 |
| RMSE(FHyb-Multiview) | 1.0703 | 1.0781 | 1.0812 |
| k | 34 | 10 | 6 |
| MAE(FHyb-FC-Sem-Kmedoids) | 0.784 | 0.7852 | 0.7865 |
| RMSE(FHyb-FC-Sem-Kmedoids) | 1.0850 | 1.0762 | 1.07503 |
| k | 38 | 13 | 17 |
| MAE(FHyb-FC-Sem-Kmedoids-BSO) | 0.7798 | 0.78053 | 0.7812 |
| RMSE(FHyb-FC-Sem-Kmedoids-BSO) | 1.0825 | 1.0765 | 1.0769 |

TABLE 4.13 – Récapitulatif des trois meilleurs résultats (MAE et RMSE) pour le FHyb avec classification du dataset RED

Discussion des résultats

D’après le tableau récapitulatif et les graphes ci-dessus, nous constatons que l’approche qui donne les meilleurs résultats dans le cas d’un filtrage hybride est le FHyb multivues K-NN avec MAE égale à 0.765 et RMSE égale à 1.070, c’est l’approche inspiré du travail de [30].

Comparaison des résultats avec d’autre méthodes utilisée sur RED

Pour voir si nos résultats sont de bonne qualité et dans les normes, nous avons décidé de les comparés à des méthodes déjà utilisé sur le dataset RED, les résultats peuvent être consultés sur le site [44]. Le tableau suivant est un récapitulatif de nos meilleurs résultats pour chaque approches développées :

| | MAE | RMSE |
|--|-------|-------|
| FC basé K-NN | 0.774 | 1.065 |
| FC basé k-medoids | 0.775 | 1.065 |
| FC basé k-medoids et BSO | 0.767 | 1.060 |
| FSem basé K-NN | 0.778 | 1.079 |
| FSem basé k-medoids | 0.776 | 1.075 |
| FSem basé k-medoids et BSO | 0.763 | 1.064 |
| FHyb collaboratif basé sémantique K-NN | 0.779 | 1.072 |
| FHyb collaboratif basé sémantique et k-medoids | 0.784 | 1.085 |
| FHyb collaboratif basé sémantique et k-medoids-BSO | 0.779 | 1.082 |
| FHyb multivues | 0.765 | 1.070 |

TABLE 4.14 – Récapitulatif des meilleurs résultats (MAE et RMSE) pour les différents filtrage conçu et testés sur RED

Sur le site [44], le meilleur résultat de MAE est de 0.804, et 1.047 pour RMSE avec la méthode SVD. En comparant nos résultats avec celui du site, nous pouvons dire que nous avons amélioré nettement MAE avec notre approche qui est FSem basé K-medoids et BSO avec MAE égale à 0.763, pour RMSE il n'y a pas une grande différence entre notre approche et SVD. Par conséquent nous avons prouvé l'efficacité de notre méthode comparé aux méthodes déjà appliqué sur le dataset RED.

4.5 Récapitulatif des différents résultats des évaluations

Dans cette partie nous allons récapituler les différents résultats obtenus pour chaque filtrage réalisé, en termes d'hybridation, de classification et de classification optimisée sur les deux dataset utilisés.

1) Filtrage collaboratif

Pour le filtrage collaboratif standard, le clustering optimisé avec l'algorithme K-medoids et BSO a amélioré les résultats du FC standard. En utilisant la classification avec K-NN sur le FC standard, cela n'a pas apporté d'amélioration. Donc pour le FC le meilleur filtrage est le FC basé K-medoids et BSO, il a pu donner des résultats très proches à ceux des algorithmes déjà appliqués sur MovieLens, d'environ 0.015 de différence pour MAE et de 0.092 pour RMSE.

2) Filtrage sémantique

Le FSem standard n'a pas pu être amélioré ni par la classification avec K-NN et ni par le clustering optimisé. Mais ses résultats restent dans les normes comparé aux résultats données par d'autres méthodes appliquées sur le dataset MovieLens, avec MAE égale à 0.7297 et RMSE égale à 1.0012. En appliquant le FSem basé K-medoids, le FSem basé K-medoids-BSO et le FSem basé K-NN sur le dataset RED, nous avons obtenus de très bons résultats comparé à ceux du site [44].

3) Filtrage hybride

L'approche que nous avons conçue et qui a donné les meilleures performances sur le dataset MovieLens est le FHyb collaboratif basé sémantique et K-NN, ce résultat est très proche des meilleures performances existantes de ce dataset (voir les deux tableaux 4.10 et 4.9).

Comparaison entre les résultats de MovieLens et RED

En rajoutant la classification et classification optimisée aux trois types de filtres et après les avoir testés sur le dataset RED, nous avons obtenu de très bons résultats comparé à ceux du site [44], et nous pouvons ainsi conclure et dire qu'il y'a un apport lors de l'ajout de la classification et la classification optimisée aux différents types de filtres conçus, car cela nous a permis de nous rapprocher des meilleurs résultats déjà existants dans le cas du dataset MovieLens, et de surpasser les performances des algorithmes déjà existants en utilisant le dataset RED.

4.6 Conclusion

Dans ce chapitre que nous clôturons, nous avons pu tester différents algorithmes de recommandation que nous avons réalisé sur les datasets MovieLens 100k et RED. En effet, ces expérimentations nous ont permis de faire des comparaisons entre les différents filtres conçus, et de conclure en disant que les filtres conçus se sont montrés très efficaces sur le dataset RED. Nous avons pu nous rapprocher des résultats déjà existants sur le dataset MovieLens avec notre approche (FHyb collaboratif basé sémantique et K-NN) qui est simple et plus rapide que SVDPlusPlus.

Annexe

Évaluation de MovieLens

Filtrage collaboratif

| | | | | | | | | | | | |
|-----------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| threshold | 0.0 | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.799 | 0.899 | 1.0 |
| mae | 0.804 | 0.808 | 0.829 | 1.032 | 2.492 | 2.981 | 2.921 | 2.921 | 2.921 | 2.921 | 2.921 |
| rmse | 1.076 | 1.085 | 1.133 | 1.476 | 2.837 | 3.178 | 3.119 | 3.119 | 3.119 | 3.119 | 3.119 |

TABLE 2 – Résultats de MAE et RMSE pour l'algorithme de FC item-item

| | | | | | | | | | | | |
|-----------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| threshold | 0.0 | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.799 | 0.899 | 1.0 |
| mae | 0.852 | 0.853 | 0.851 | 0.831 | 0.775 | 0.782 | 0.78 | 0.78 | 0.78 | 0.78 | 0.78 |
| rmse | 1.169 | 1.169 | 1.167 | 1.146 | 1.064 | 1.07 | 1.066 | 1.066 | 1.066 | 1.066 | 1.066 |

TABLE 3 – Résultats de MAE et RMSE pour l'algorithme de FC item-item-IA

| | | | | | | | | | | | |
|-----------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| threshold | 0.0 | 0.052 | 0.105 | 0.158 | 0.211 | 0.264 | 0.317 | 0.369 | 0.422 | 0.475 | 0.528 |
| mae | 0.923 | 0.923 | 0.93 | 0.94 | 0.942 | 0.959 | 0.967 | 0.974 | 0.973 | 0.974 | 0.974 |
| rmse | 1.197 | 1.198 | 1.207 | 1.217 | 1.225 | 1.245 | 1.254 | 1.259 | 1.26 | 1.259 | 1.259 |

TABLE 4 – Résultats de MAE et RMSE pour l'algorithme de FC item-item-SVD

| | | | | | | | | | | | |
|-----------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| threshold | 0.0 | 0.043 | 0.086 | 0.129 | 0.172 | 0.215 | 0.258 | 0.301 | 0.345 | 0.388 | 0.431 |
| mae | 0.804 | 0.803 | 0.802 | 0.8 | 0.798 | 0.793 | 0.796 | 0.798 | 0.797 | 0.797 | 0.797 |
| rmse | 1.076 | 1.074 | 1.073 | 1.071 | 1.07 | 1.065 | 1.067 | 1.068 | 1.067 | 1.067 | 1.067 |

TABLE 5 – Résultats de MAE et RMSE pour l'algorithme de FC item-item-UA

| | | | | | | | | | | | |
|-----------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| threshold | 0.0 | 0.075 | 0.151 | 0.227 | 0.303 | 0.378 | 0.454 | 0.53 | 0.606 | 0.681 | 0.757 |
| mae | 0.795 | 0.795 | 0.795 | 0.81 | 1.022 | 1.957 | 2.612 | 2.941 | 2.948 | 2.948 | 2.948 |
| rmse | 1.079 | 1.079 | 1.079 | 1.111 | 1.492 | 2.504 | 2.842 | 3.135 | 3.143 | 3.142 | 3.142 |

TABLE 6 – Résultats de MAE et RMSE pour l'algorithme de FC user-user

| | | | | | | | | | | | |
|-----------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| threshold | 0.0 | 0.036 | 0.073 | 0.109 | 0.146 | 0.183 | 0.219 | 0.256 | 0.292 | 0.329 | 0.366 |
| mae | 1.028 | 0.821 | 0.798 | 0.799 | 0.801 | 0.801 | 0.801 | 0.8 | 0.8 | 0.8 | 0.8 |
| rmse | 1.265 | 1.093 | 1.069 | 1.07 | 1.071 | 1.072 | 1.072 | 1.07 | 1.07 | 1.07 | 1.07 |

TABLE 7 – Résultats de MAE et RMSE pour l'algorithme de FC user-user-IA

| | | | | | | | | | | | |
|-----------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| threshold | 0 | 0.075 | 0.151 | 0.227 | 0.303 | 0.378 | 0.454 | 0.53 | 0.606 | 0.681 | 0.757 |
| mae | 0.795 | 0.795 | 0.795 | 0.795 | 0.796 | 0.795 | 0.761 | 0.781 | 0.78 | 0.78 | 0.78 |
| rmse | 1.079 | 1.079 | 1.079 | 1.079 | 1.081 | 1.08 | 1.047 | 1.066 | 1.066 | 1.066 | 1.066 |

TABLE 8 – Résultats de MAE et RMSE pour l'algorithme de FC user-user-UA

| | | | | | | | | | | |
|-----------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| threshold | 0.0 | 0.056 | 0.113 | 0.17 | 0.227 | 0.284 | 0.34 | 0.397 | 0.454 | 0.511 |
| mae | 0.867 | 0.857 | 0.783 | 0.75 | 0.736 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 |
| rmse | 1.129 | 1.12 | 1.056 | 1.023 | 1.007 | 1.001 | 1.001 | 1.002 | 1.002 | 1.002 |

TABLE 9 – Résultats de MAE et RMSE pour l'algorithme de FC user-user-SVD

| | | | | | | | | | | | | |
|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| K | 10 | 15 | 20 | 25 | 30 | 35 | 40 | 45 | 50 | 55 | 60 | 65 |
| mae | 0.733 | 0.732 | 0.735 | 0.735 | 0.736 | 0.732 | 0.736 | 0.734 | 0.736 | 0.74 | 0.737 | 0.74 |
| rmse | 1.004 | 1.003 | 1.006 | 1.006 | 1.008 | 1.005 | 1.008 | 1.007 | 1.008 | 1.011 | 1.011 | 1.012 |
| K | 70 | 75 | 80 | 85 | 90 | 95 | 100 | 105 | 110 | 115 | 120 | 125 |
| mae | 0.743 | 0.737 | 0.739 | 0.738 | 0.744 | 0.747 | 0.745 | 0.746 | 0.747 | 0.74 | 0.751 | 0.747 |
| rmse | 1.015 | 1.011 | 1.011 | 1.011 | 1.015 | 1.02 | 1.017 | 1.017 | 1.02 | 1.015 | 1.024 | 1.019 |
| K | 130 | 135 | 140 | 145 | 150 | 155 | 160 | 165 | 170 | 175 | 180 | 185 |
| mae | 0.753 | 0.744 | 0.756 | 0.747 | 0.752 | 0.755 | 0.746 | 0.745 | 0.76 | 0.754 | 0.753 | 0.759 |
| rmse | 1.025 | 1.019 | 1.028 | 1.021 | 1.027 | 1.028 | 1.021 | 1.021 | 1.036 | 1.026 | 1.027 | 1.033 |
| K | 190 | 195 | 200 | 205 | 210 | 215 | 220 | 225 | 230 | 235 | 240 | 245 |
| mae | 0.754 | 0.761 | 0.76 | 0.752 | 0.76 | 0.755 | 0.755 | 0.76 | 0.766 | 0.76 | 0.76 | 0.762 |
| rmse | 1.028 | 1.033 | 1.037 | 1.026 | 1.034 | 1.027 | 1.03 | 1.03 | 1.038 | 1.039 | 1.032 | 1.036 |
| K | 250 | 255 | 260 | 265 | 270 | 275 | 280 | 285 | 290 | 295 | 300 | 305 |
| mae | 0.764 | 0.769 | 0.763 | 0.754 | 0.766 | 0.77 | 0.771 | 0.776 | 0.766 | 0.764 | 0.766 | 0.775 |
| rmse | 1.037 | 1.042 | 1.042 | 1.03 | 1.04 | 1.039 | 1.044 | 1.053 | 1.041 | 1.039 | 1.043 | 1.051 |
| K | 310 | 315 | 320 | 325 | 330 | 335 | 340 | 345 | 350 | 355 | 360 | 365 |
| mae | 0.766 | 0.774 | 0.768 | 0.778 | 0.775 | 0.77 | 0.767 | 0.776 | 0.772 | 0.78 | 0.777 | 0.78 |
| rmse | 1.041 | 1.048 | 1.048 | 1.049 | 1.052 | 1.043 | 1.041 | 1.053 | 1.047 | 1.056 | 1.052 | 1.056 |
| K | 370 | 375 | 380 | 385 | 390 | 395 | 400 | 405 | 410 | 415 | 420 | 425 |
| mae | 0.778 | 0.78 | 0.774 | 0.778 | 0.778 | 0.782 | 0.781 | 0.786 | 0.779 | 0.78 | 0.777 | 0.79 |
| rmse | 1.051 | 1.052 | 1.052 | 1.053 | 1.058 | 1.051 | 1.052 | 1.059 | 1.054 | 1.054 | 1.051 | 1.067 |
| K | 430 | 435 | 440 | 445 | 450 | 455 | 460 | 465 | 470 | 475 | 480 | 485 |
| mae | 0.785 | 0.786 | 0.784 | 0.786 | 0.787 | 0.791 | 0.784 | 0.792 | 0.784 | 0.792 | 0.79 | 0.79 |
| rmse | 1.063 | 1.063 | 1.057 | 1.058 | 1.056 | 1.06 | 1.057 | 1.064 | 1.06 | 1.069 | 1.066 | 1.062 |
| K | 490 | 495 | 500 | 505 | 510 | 515 | 520 | 525 | 530 | 535 | 540 | 545 |
| mae | 0.797 | 0.788 | 0.789 | 0.791 | 0.795 | 0.798 | 0.794 | 0.808 | 0.792 | 0.79 | 0.808 | 0.791 |
| rmse | 1.07 | 1.061 | 1.066 | 1.063 | 1.067 | 1.072 | 1.062 | 1.076 | 1.069 | 1.064 | 1.081 | 1.06 |
| K | 550 | 555 | 560 | 565 | 570 | 575 | 580 | 585 | 590 | 595 | 600 | 605 |
| mae | 0.797 | 0.792 | 0.806 | 0.795 | 0.8 | 0.818 | 0.798 | 0.8 | 0.79 | 0.793 | 0.799 | 0.805 |
| rmse | 1.074 | 1.067 | 1.078 | 1.068 | 1.07 | 1.094 | 1.07 | 1.073 | 1.06 | 1.064 | 1.074 | 1.079 |
| K | 610 | 615 | 620 | 625 | 630 | 635 | 640 | 645 | 650 | 655 | 660 | 665 |
| mae | 0.811 | 0.798 | 0.811 | 0.811 | 0.807 | 0.806 | 0.798 | 0.812 | 0.807 | 0.807 | 0.812 | 0.811 |
| rmse | 1.08 | 1.066 | 1.081 | 1.089 | 1.079 | 1.078 | 1.07 | 1.083 | 1.084 | 1.081 | 1.081 | 1.085 |
| K | 670 | 675 | 680 | 685 | 690 | 695 | | | | | | |
| mae | 0.811 | 0.823 | 0.816 | 0.808 | 0.806 | 0.816 | | | | | | |
| rmse | 1.081 | 1.091 | 1.09 | 1.076 | 1.086 | 1.087 | | | | | | |

TABLE 10 – Résultats de MAE et RMSE pour l'algorithme de FC user-user-Kmedoids

| | | | | | | | | | | | | |
|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| K | 10 | 15 | 20 | 25 | 30 | 35 | 40 | 45 | 50 | 55 | 60 | 65 |
| mae | 0.733 | 0.733 | 0.732 | 0.733 | 0.732 | 0.732 | 0.731 | 0.73 | 0.73 | 0.73 | 0.731 | 0.73 |
| rmse | 1.002 | 1.003 | 1.002 | 1.004 | 1.002 | 1.003 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1 |
| K | 70 | 75 | 80 | 85 | 90 | 95 | 100 | 105 | 110 | 115 | 120 | 125 |
| mae | 0.731 | 0.731 | 0.731 | 0.73 | 0.731 | 0.731 | 0.73 | 0.731 | 0.73 | 0.73 | 0.728 | 0.729 |
| rmse | 1 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1 | 0.999 | 0.999 |
| K | 130 | 135 | 140 | 145 | 150 | 155 | 160 | 165 | 170 | 175 | 180 | 185 |
| mae | 0.729 | 0.728 | 0.729 | 0.729 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 |
| rmse | 0.999 | 0.999 | 0.999 | 1 | 1 | 1 | 1.001 | 1 | 1 | 1.001 | 1.001 | 1.001 |
| K | 190 | 195 | 200 | 205 | 210 | 215 | 220 | 225 | 230 | 235 | 240 | 245 |
| mae | 0.73 | 0.73 | 0.731 | 0.73 | 0.731 | 0.73 | 0.73 | 0.731 | 0.731 | 0.73 | 0.73 | 0.731 |
| rmse | 1 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 |
| K | 250 | 255 | 260 | 265 | 270 | 275 | 280 | 285 | 290 | 295 | 300 | 305 |
| mae | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.729 | 0.73 |
| rmse | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 |
| K | 310 | 315 | 320 | 325 | 330 | 335 | 340 | 345 | 350 | 355 | 360 | 365 |
| mae | 0.73 | 0.729 | 0.73 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 |
| rmse | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1 | 1 | 1.001 | 1 | 1 | 1.001 |
| K | 370 | 375 | 380 | 385 | 390 | 395 | 400 | 405 | 410 | 415 | 420 | 425 |
| mae | 0.729 | 0.729 | 0.73 | 0.73 | 0.73 | 0.73 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 |
| rmse | 1 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1 | 1 | 1.001 | 1 | 1.001 |
| K | 430 | 435 | 440 | 445 | 450 | 455 | 460 | 465 | 470 | 475 | 480 | 485 |
| mae | 0.729 | 0.729 | 0.729 | 0.73 | 0.73 | 0.729 | 0.73 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 |
| rmse | 1.001 | 1 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1 | 1 | 1 | 1 | 1 |
| K | 490 | 495 | 500 | 505 | 510 | 515 | 520 | 525 | 530 | 535 | 540 | 545 |
| mae | 0.73 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.73 | 0.729 | 0.729 | 0.729 | 0.729 |
| rmse | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0.999 | 0.999 |
| K | 550 | 555 | 560 | 565 | 570 | 575 | 580 | 585 | 590 | 595 | 600 | 605 |
| mae | 0.729 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.73 |
| rmse | 1 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1 | 1 | 1.001 | 1.001 | 1.001 |
| K | 610 | 615 | 620 | 625 | 630 | 635 | 640 | 645 | 650 | 655 | 660 | 665 |
| mae | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.731 | 0.731 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 |
| rmse | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.002 | 1.002 | 1.001 | 1.002 | 1.001 | 1.001 | 1.001 |
| K | 670 | 675 | 680 | 685 | 690 | 695 | | | | | | |
| mae | 0.73 | 0.729 | 0.73 | 0.729 | 0.729 | 0.73 | | | | | | |
| rmse | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | | | | | | |

TABLE 11 – Résultats de MAE et RMSE pour l’algorithme de FC user-user-KNN

Filtrage sémantique

| | | | | | | | | | | | |
|-----------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| threshold | 0.666 | 0.666 | 0.666 | 0.666 | 0.666 | 0.666 | 0.666 | 0.666 | 0.666 | 0.666 | 0.666 |
| mae | 0.867 | 0.867 | 0.811 | 0.736 | 0.729 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 |
| rmse | 1.129 | 1.128 | 1.085 | 1.009 | 1.001 | 1.002 | 1.002 | 1.002 | 1.002 | 1.002 | 1.002 |

TABLE 12 – Résultats de MAE et RMSE pour l’algorithme de FSem user-user

| | | | | | | | | | | | | |
|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| K | 10 | 15 | 20 | 25 | 30 | 35 | 40 | 45 | 50 | 55 | 60 | 65 |
| mae | 0.73 | 0.73 | 0.732 | 0.73 | 0.734 | 0.733 | 0.735 | 0.735 | 0.737 | 0.737 | 0.736 | 0.737 |
| rmse | 1.002 | 1.001 | 1.005 | 1.002 | 1.005 | 1.004 | 1.007 | 1.007 | 1.006 | 1.008 | 1.006 | 1.008 |
| K | 70 | 75 | 80 | 85 | 90 | 95 | 100 | 105 | 110 | 115 | 120 | 125 |
| mae | 0.739 | 0.737 | 0.738 | 0.739 | 0.739 | 0.745 | 0.748 | 0.745 | 0.747 | 0.746 | 0.744 | 0.747 |
| rmse | 1.011 | 1.009 | 1.01 | 1.011 | 1.011 | 1.017 | 1.016 | 1.018 | 1.017 | 1.018 | 1.015 | 1.018 |
| K | 130 | 135 | 140 | 145 | 150 | 155 | 160 | 165 | 170 | 175 | 180 | 185 |
| mae | 0.743 | 0.751 | 0.752 | 0.749 | 0.751 | 0.753 | 0.755 | 0.752 | 0.757 | 0.753 | 0.758 | 0.763 |
| rmse | 1.014 | 1.022 | 1.024 | 1.019 | 1.023 | 1.026 | 1.024 | 1.024 | 1.028 | 1.026 | 1.027 | 1.034 |
| K | 190 | 195 | 200 | 205 | 210 | 215 | 220 | 225 | 230 | 235 | 240 | 245 |
| mae | 0.753 | 0.753 | 0.757 | 0.765 | 0.764 | 0.754 | 0.759 | 0.759 | 0.76 | 0.769 | 0.752 | 0.776 |
| rmse | 1.024 | 1.023 | 1.028 | 1.037 | 1.034 | 1.026 | 1.029 | 1.03 | 1.034 | 1.041 | 1.025 | 1.044 |
| K | 250 | 255 | 260 | 265 | 270 | 275 | 280 | 285 | 290 | 295 | 300 | 305 |
| mae | 0.761 | 0.77 | 0.772 | 0.77 | 0.763 | 0.768 | 0.762 | 0.765 | 0.771 | 0.771 | 0.768 | 0.773 |
| rmse | 1.03 | 1.04 | 1.044 | 1.039 | 1.033 | 1.035 | 1.032 | 1.036 | 1.043 | 1.042 | 1.04 | 1.044 |
| K | 310 | 315 | 320 | 325 | 330 | 335 | 340 | 345 | 350 | 355 | 360 | 365 |
| mae | 0.771 | 0.775 | 0.768 | 0.778 | 0.77 | 0.778 | 0.774 | 0.775 | 0.78 | 0.788 | 0.781 | 0.776 |
| rmse | 1.043 | 1.049 | 1.037 | 1.049 | 1.041 | 1.051 | 1.044 | 1.042 | 1.051 | 1.058 | 1.051 | 1.043 |
| K | 370 | 375 | 380 | 385 | 390 | 395 | 400 | 405 | 410 | 415 | 420 | 425 |
| mae | 0.778 | 0.78 | 0.783 | 0.79 | 0.783 | 0.783 | 0.78 | 0.784 | 0.779 | 0.78 | 0.788 | 0.796 |
| rmse | 1.045 | 1.051 | 1.052 | 1.06 | 1.056 | 1.051 | 1.05 | 1.052 | 1.048 | 1.049 | 1.055 | 1.063 |
| K | 430 | 435 | 440 | 445 | 450 | 455 | 460 | 465 | 470 | 475 | 480 | 485 |
| mae | 0.786 | 0.797 | 0.788 | 0.779 | 0.795 | 0.796 | 0.789 | 0.786 | 0.784 | 0.788 | 0.801 | 0.792 |
| rmse | 1.057 | 1.069 | 1.059 | 1.047 | 1.062 | 1.07 | 1.059 | 1.057 | 1.056 | 1.059 | 1.073 | 1.06 |
| K | 490 | 495 | 500 | 505 | 510 | 515 | 520 | 525 | 530 | 535 | 540 | 545 |
| mae | 0.791 | 0.801 | 0.797 | 0.794 | 0.793 | 0.81 | 0.794 | 0.803 | 0.802 | 0.796 | 0.807 | 0.798 |
| rmse | 1.064 | 1.069 | 1.063 | 1.059 | 1.061 | 1.078 | 1.066 | 1.072 | 1.066 | 1.066 | 1.076 | 1.066 |
| K | 550 | 555 | 560 | 565 | 570 | 575 | 580 | 585 | 590 | 595 | 600 | 605 |
| mae | 0.81 | 0.795 | 0.798 | 0.806 | 0.805 | 0.795 | 0.8 | 0.798 | 0.809 | 0.807 | 0.808 | 0.806 |
| rmse | 1.079 | 1.063 | 1.062 | 1.073 | 1.071 | 1.068 | 1.069 | 1.067 | 1.076 | 1.073 | 1.078 | 1.076 |
| K | 610 | 615 | 620 | 625 | 630 | 635 | 640 | 645 | 650 | 655 | 660 | 665 |
| mae | 0.818 | 0.803 | 0.805 | 0.807 | 0.8 | 0.818 | 0.816 | 0.805 | 0.809 | 0.805 | 0.812 | 0.812 |
| rmse | 1.085 | 1.073 | 1.074 | 1.074 | 1.069 | 1.085 | 1.08 | 1.074 | 1.076 | 1.072 | 1.084 | 1.077 |
| K | 670 | 675 | 680 | 685 | 690 | 695 | | | | | | |
| mae | 0.81 | 0.81 | 0.809 | 0.812 | 0.815 | 0.81 | | | | | | |
| rmse | 1.075 | 1.076 | 1.081 | 1.08 | 1.081 | 1.078 | | | | | | |

TABLE 13 – Résultats de MAE et RMSE pour l’algorithme de FSem user-user-Kmedoids

Filtrage hybride

| | | | | | | | | | | | |
|-----------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| threshold | 0 | 0.066 | 0.133 | 0.2 | 0.266 | 0.333 | 0.4 | 0.466 | 0.533 | 0.6 | 0.666 |
| alpha | 1 | 1 | 1 | 8 | 8 | 8 | 7 | 5 | 5 | 5 | 5 |
| mae | 0.867 | 0.867 | 0.867 | 0.819 | 0.754 | 0.73 | 0.73 | 0.729 | 0.73 | 0.73 | 0.73 |
| rmse | 1.129 | 1.129 | 1.129 | 1.088 | 1.027 | 1.002 | 1.001 | 1.001 | 1.001 | 1.002 | 1.002 |

TABLE 14 – Résultats de MAE et RMSE pour l'algorithme de FHyb-alpha

| | | | | | | | | | | | | |
|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| K | 10 | 15 | 20 | 25 | 30 | 35 | 40 | 45 | 50 | 55 | 60 | 65 |
| mae | 3 | 1 | 2 | 1 | 6 | 6 | 1 | 1 | 7 | 1 | 2 | 1 |
| rmse | 0.731 | 0.73 | 0.731 | 0.732 | 0.731 | 0.736 | 0.734 | 0.731 | 0.734 | 0.74 | 0.738 | 0.74 |
| K | 70 | 75 | 80 | 85 | 90 | 95 | 100 | 105 | 110 | 115 | 120 | 125 |
| mae | 1 | 8 | 7 | 6 | 8 | 3 | 6 | 1 | 4 | 1 | 1 | 4 |
| rmse | 0.741 | 0.743 | 0.739 | 0.742 | 0.743 | 0.743 | 0.746 | 0.742 | 0.742 | 0.747 | 0.747 | 0.749 |
| K | 130 | 135 | 140 | 145 | 150 | 155 | 160 | 165 | 170 | 175 | 180 | 185 |
| mae | 3 | 3 | 5 | 3 | 2 | 8 | 1 | 3 | 5 | 8 | 8 | 3 |
| rmse | 0.749 | 0.747 | 0.742 | 0.753 | 0.75 | 0.745 | 0.747 | 0.75 | 0.758 | 0.748 | 0.752 | 0.752 |
| K | 190 | 195 | 200 | 205 | 210 | 215 | 220 | 225 | 230 | 235 | 240 | 245 |
| mae | 8 | 3 | 3 | 3 | 2 | 3 | 8 | 1 | 3 | 1 | 1 | 3 |
| rmse | 0.758 | 0.761 | 0.757 | 0.75 | 0.762 | 0.757 | 0.759 | 0.76 | 0.758 | 0.764 | 0.77 | 0.764 |
| K | 250 | 255 | 260 | 265 | 270 | 275 | 280 | 285 | 290 | 295 | 300 | 305 |
| mae | 8 | 5 | 6 | 5 | 1 | 1 | 2 | 5 | 5 | 8 | 1 | 7 |
| rmse | 0.765 | 0.767 | 0.766 | 0.761 | 0.766 | 0.769 | 0.771 | 0.763 | 0.765 | 0.77 | 0.763 | 0.774 |
| K | 310 | 315 | 320 | 325 | 330 | 335 | 340 | 345 | 350 | 355 | 360 | 365 |
| mae | 2 | 4 | 5 | 7 | 2 | 1 | 1 | 8 | 6 | 6 | 4 | 3 |
| rmse | 0.768 | 0.765 | 0.776 | 0.767 | 0.766 | 0.772 | 0.773 | 0.777 | 0.772 | 0.784 | 0.778 | 0.779 |
| K | 370 | 375 | 380 | 385 | 390 | 395 | 400 | 405 | 410 | 415 | 420 | 425 |
| mae | 6 | 1 | 8 | 4 | 3 | 8 | 1 | 2 | 4 | 7 | 1 | 3 |
| rmse | 0.777 | 0.78 | 0.785 | 0.781 | 0.772 | 0.78 | 0.789 | 0.782 | 0.786 | 0.778 | 0.787 | 0.785 |
| K | 430 | 435 | 440 | 445 | 450 | 455 | 460 | 465 | 470 | 475 | 480 | 485 |
| mae | 3 | 2 | 8 | 1 | 1 | 8 | 6 | 2 | 6 | 1 | 7 | 3 |
| rmse | 0.776 | 0.78 | 0.788 | 0.793 | 0.786 | 0.793 | 0.788 | 0.799 | 0.794 | 0.792 | 0.789 | 0.79 |
| K | 490 | 495 | 500 | 505 | 510 | 515 | 520 | 525 | 530 | 535 | 540 | 545 |
| mae | 3 | 3 | 7 | 1 | 1 | 1 | 1 | 1 | 6 | 2 | 4 | 3 |
| rmse | 0.792 | 0.792 | 0.796 | 0.79 | 0.787 | 0.797 | 0.792 | 0.799 | 0.789 | 0.796 | 0.794 | 0.791 |
| K | 550 | 555 | 560 | 565 | 570 | 575 | 580 | 585 | 590 | 595 | 600 | 605 |
| mae | 3 | 2 | 1 | 1 | 3 | 2 | 4 | 1 | 3 | 3 | 6 | 3 |
| rmse | 0.812 | 0.794 | 0.802 | 0.799 | 0.793 | 0.8 | 0.801 | 0.808 | 0.803 | 0.8 | 0.8 | 0.796 |
| K | 610 | 615 | 620 | 625 | 630 | 635 | 640 | 645 | 650 | 655 | 660 | 665 |
| mae | 7 | 1 | 8 | 8 | 7 | 3 | 2 | 3 | 3 | 1 | 7 | 7 |
| rmse | 0.816 | 0.8 | 0.806 | 0.8 | 0.809 | 0.806 | 0.813 | 0.813 | 0.804 | 0.811 | 0.803 | 0.805 |
| K | 670 | 675 | 680 | 685 | 690 | 695 | | | | | | |
| mae | 3 | 1 | 8 | 1 | 2 | 1 | | | | | | |
| rmse | 0.805 | 0.807 | 0.812 | 0.804 | 0.817 | 0.814 | | | | | | |

TABLE 15 – Résultats de MAE et RMSE pour l’algorithme de FHyb alpha-Kmedoids

| | | | | | | | | | | | | |
|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| K | 10 | 15 | 20 | 25 | 30 | 35 | 40 | 45 | 50 | 55 | 60 | 65 |
| mae | 1 | 3 | 1 | 8 | 1 | 1 | 7 | 7 | 1 | 6 | 6 | 8 |
| rmse | 0.733 | 0.734 | 0.732 | 0.732 | 0.732 | 0.732 | 0.731 | 0.73 | 0.73 | 0.73 | 0.731 | 0.73 |
| K | 70 | 75 | 80 | 85 | 90 | 95 | 100 | 105 | 110 | 115 | 120 | 125 |
| mae | 2 | 1 | 3 | 1 | 5 | 8 | 4 | 1 | 4 | 1 | 4 | 1 |
| rmse | 0.731 | 0.731 | 0.731 | 0.73 | 0.731 | 0.731 | 0.731 | 0.73 | 0.73 | 0.729 | 0.728 | 0.73 |
| K | 130 | 135 | 140 | 145 | 150 | 155 | 160 | 165 | 170 | 175 | 180 | 185 |
| mae | 1 | 6 | 1 | 1 | 1 | 8 | 8 | 5 | 1 | 8 | 7 | 6 |
| rmse | 0.729 | 0.728 | 0.728 | 0.729 | 0.729 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 |
| K | 190 | 195 | 200 | 205 | 210 | 215 | 220 | 225 | 230 | 235 | 240 | 245 |
| mae | 1 | 1 | 1 | 1 | 8 | 1 | 1 | 1 | 2 | 6 | 1 | 1 |
| rmse | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 |
| K | 250 | 255 | 260 | 265 | 270 | 275 | 280 | 285 | 290 | 295 | 300 | 305 |
| mae | 1 | 7 | 8 | 1 | 3 | 3 | 1 | 1 | 1 | 1 | 5 | 6 |
| rmse | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.729 | 0.729 | 0.73 | 0.729 | 0.73 |
| K | 310 | 315 | 320 | 325 | 330 | 335 | 340 | 345 | 350 | 355 | 360 | 365 |
| mae | 8 | 1 | 8 | 1 | 1 | 1 | 6 | 4 | 7 | 6 | 7 | 8 |
| rmse | 0.73 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.73 | 0.729 | 0.729 | 0.729 |
| K | 370 | 375 | 380 | 385 | 390 | 395 | 400 | 405 | 410 | 415 | 420 | 425 |
| mae | 8 | 3 | 1 | 1 | 7 | 1 | 1 | 8 | 1 | 1 | 8 | 2 |
| rmse | 0.729 | 0.729 | 0.729 | 0.729 | 0.73 | 0.73 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 |
| K | 430 | 435 | 440 | 445 | 450 | 455 | 460 | 465 | 470 | 475 | 480 | 485 |
| mae | 1 | 8 | 8 | 1 | 1 | 1 | 5 | 1 | 1 | 3 | 7 | 6 |
| rmse | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.73 | 0.729 | 0.729 | 0.729 | 0.73 |
| K | 490 | 495 | 500 | 505 | 510 | 515 | 520 | 525 | 530 | 535 | 540 | 545 |
| mae | 1 | 1 | 4 | 8 | 5 | 1 | 6 | 1 | 1 | 6 | 8 | 8 |
| rmse | 0.73 | 0.73 | 0.729 | 0.729 | 0.73 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 |
| K | 550 | 555 | 560 | 565 | 570 | 575 | 580 | 585 | 590 | 595 | 600 | 605 |
| mae | 6 | 6 | 1 | 1 | 6 | 4 | 1 | 1 | 1 | 5 | 1 | 6 |
| rmse | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.73 |
| K | 610 | 615 | 620 | 625 | 630 | 635 | 640 | 645 | 650 | 655 | 660 | 665 |
| mae | 1 | 1 | 3 | 1 | 8 | 2 | 2 | 1 | 7 | 1 | 6 | 4 |
| rmse | 0.729 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.729 | 0.729 |
| K | 670 | 675 | 680 | 685 | 690 | 695 | | | | | | |
| mae | 1 | 1 | 4 | 8 | 2 | 1 | | | | | | |
| rmse | 0.73 | 0.73 | 0.73 | 0.729 | 0.729 | 0.729 | | | | | | |

TABLE 16 – Résultats de MAE et RMSE pour l’algorithme de FHyb alpha-KNN

| | | | | | | | | | | | |
|-----------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| threshold | 0 | 0.037 | 0.075 | 0.113 | 0.151 | 0.189 | 0.227 | 0.265 | 0.303 | 0.341 | 0.379 |
| mae | 0.867 | 0.868 | 0.864 | 0.827 | 0.767 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 |
| rmse | 1.129 | 1.13 | 1.127 | 1.101 | 1.046 | 1.003 | 1.003 | 1.002 | 1.002 | 1.002 | 1.002 |

TABLE 17 – Résultats de MAE et RMSE pour l’algorithme de FHyb FC basé Sem

| | | | | | | | | | | | | |
|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| K | 10 | 15 | 20 | 25 | 30 | 35 | 40 | 45 | 50 | 55 | 60 | 65 |
| mae | 0.73 | 0.729 | 0.73 | 0.732 | 0.731 | 0.731 | 0.732 | 0.733 | 0.733 | 0.732 | 0.731 | 0.733 |
| rmse | 1.001 | 0.999 | 0.999 | 1.004 | 1.002 | 1.004 | 1.003 | 1.004 | 1.004 | 1.001 | 1.004 | 1.003 |
| K | 70 | 75 | 80 | 85 | 90 | 95 | 100 | 105 | 110 | 115 | 120 | 125 |
| mae | 0.733 | 0.735 | 0.729 | 0.74 | 0.736 | 0.736 | 0.735 | 0.741 | 0.741 | 0.741 | 0.737 | 0.741 |
| rmse | 1.006 | 1.007 | 1.001 | 1.011 | 1.007 | 1.008 | 1.009 | 1.016 | 1.014 | 1.016 | 1.012 | 1.011 |
| K | 130 | 135 | 140 | 145 | 150 | 155 | 160 | 165 | 170 | 175 | 180 | 185 |
| mae | 0.744 | 0.739 | 0.745 | 0.746 | 0.741 | 0.748 | 0.742 | 0.75 | 0.747 | 0.742 | 0.746 | 0.745 |
| rmse | 1.014 | 1.011 | 1.016 | 1.018 | 1.012 | 1.021 | 1.018 | 1.021 | 1.019 | 1.019 | 1.018 | 1.022 |
| K | 190 | 195 | 200 | 205 | 210 | 215 | 220 | 225 | 230 | 235 | 240 | 245 |
| mae | 0.746 | 0.75 | 0.75 | 0.747 | 0.747 | 0.761 | 0.747 | 0.759 | 0.75 | 0.749 | 0.753 | 0.761 |
| rmse | 1.019 | 1.026 | 1.025 | 1.023 | 1.025 | 1.033 | 1.024 | 1.03 | 1.026 | 1.026 | 1.027 | 1.036 |
| K | 250 | 255 | 260 | 265 | 270 | 275 | 280 | 285 | 290 | 295 | 300 | 305 |
| mae | 0.756 | 0.766 | 0.761 | 0.753 | 0.764 | 0.757 | 0.776 | 0.765 | 0.756 | 0.76 | 0.762 | 0.766 |
| rmse | 1.031 | 1.04 | 1.036 | 1.027 | 1.036 | 1.035 | 1.05 | 1.038 | 1.031 | 1.035 | 1.036 | 1.043 |
| K | 310 | 315 | 320 | 325 | 330 | 335 | 340 | 345 | 350 | 355 | 360 | 365 |
| mae | 0.764 | 0.766 | 0.761 | 0.775 | 0.767 | 0.775 | 0.767 | 0.773 | 0.774 | 0.768 | 0.772 | 0.777 |
| rmse | 1.038 | 1.042 | 1.038 | 1.051 | 1.04 | 1.048 | 1.046 | 1.052 | 1.051 | 1.041 | 1.048 | 1.053 |
| K | 370 | 375 | 380 | 385 | 390 | 395 | 400 | 405 | 410 | 415 | 420 | 425 |
| mae | 0.777 | 0.777 | 0.767 | 0.771 | 0.781 | 0.787 | 0.776 | 0.778 | 0.776 | 0.769 | 0.78 | 0.778 |
| rmse | 1.049 | 1.055 | 1.045 | 1.047 | 1.054 | 1.06 | 1.05 | 1.051 | 1.053 | 1.046 | 1.059 | 1.053 |
| K | 430 | 435 | 440 | 445 | 450 | 455 | 460 | 465 | 470 | 475 | 480 | 485 |
| mae | 0.781 | 0.78 | 0.778 | 0.784 | 0.783 | 0.779 | 0.793 | 0.791 | 0.795 | 0.785 | 0.79 | 0.79 |
| rmse | 1.059 | 1.06 | 1.054 | 1.063 | 1.056 | 1.053 | 1.068 | 1.069 | 1.067 | 1.063 | 1.064 | 1.067 |
| K | 490 | 495 | 500 | 505 | 510 | 515 | 520 | 525 | 530 | 535 | 540 | 545 |
| mae | 0.791 | 0.79 | 0.794 | 0.786 | 0.793 | 0.809 | 0.797 | 0.799 | 0.79 | 0.794 | 0.795 | 0.795 |
| rmse | 1.065 | 1.068 | 1.065 | 1.062 | 1.067 | 1.087 | 1.07 | 1.075 | 1.068 | 1.069 | 1.07 | 1.068 |
| K | 550 | 555 | 560 | 565 | 570 | 575 | 580 | 585 | 590 | 595 | 600 | 605 |
| mae | 0.798 | 0.798 | 0.785 | 0.801 | 0.788 | 0.791 | 0.797 | 0.79 | 0.801 | 0.806 | 0.797 | 0.797 |
| rmse | 1.077 | 1.073 | 1.058 | 1.079 | 1.066 | 1.069 | 1.067 | 1.064 | 1.076 | 1.08 | 1.076 | 1.073 |
| K | 610 | 615 | 620 | 625 | 630 | 635 | 640 | 645 | 650 | 655 | 660 | 665 |
| mae | 0.808 | 0.805 | 0.803 | 0.803 | 0.816 | 0.806 | 0.798 | 0.812 | 0.805 | 0.801 | 0.804 | 0.803 |
| rmse | 1.082 | 1.079 | 1.076 | 1.078 | 1.089 | 1.079 | 1.08 | 1.084 | 1.088 | 1.075 | 1.082 | 1.074 |
| K | 670 | 675 | 680 | 685 | 690 | 695 | | | | | | |
| mae | 0.813 | 0.806 | 0.807 | 0.811 | 0.799 | 0.817 | | | | | | |
| rmse | 1.089 | 1.077 | 1.081 | 1.082 | 1.079 | 1.091 | | | | | | |

TABLE 18 – Résultats de MAE et RMSE pour l'algorithme de FHyb FC basé Sem et Kme-doids

| | | | | | | | | | | | | |
|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| K | 10 | 15 | 20 | 25 | 30 | 35 | 40 | 45 | 50 | 55 | 60 | 65 |
| mae | 0.728 | 0.723 | 0.72 | 0.718 | 0.718 | 0.722 | 0.719 | 0.72 | 0.721 | 0.722 | 0.721 | 0.723 |
| rmse | 1.007 | 1.002 | 0.996 | 0.996 | 0.995 | 0.997 | 0.996 | 0.995 | 0.995 | 0.994 | 0.995 | 0.996 |
| K | 70 | 75 | 80 | 85 | 90 | 95 | 100 | 105 | 110 | 115 | 120 | 125 |
| mae | 0.722 | 0.722 | 0.721 | 0.723 | 0.723 | 0.724 | 0.723 | 0.724 | 0.725 | 0.723 | 0.725 | 0.725 |
| rmse | 0.995 | 0.996 | 0.995 | 0.996 | 0.996 | 0.996 | 0.995 | 0.997 | 0.997 | 0.996 | 0.997 | 0.998 |
| K | 130 | 135 | 140 | 145 | 150 | 155 | 160 | 165 | 170 | 175 | 180 | 185 |
| mae | 0.725 | 0.725 | 0.725 | 0.726 | 0.724 | 0.724 | 0.724 | 0.724 | 0.724 | 0.724 | 0.724 | 0.723 |
| rmse | 0.998 | 0.997 | 0.997 | 0.998 | 0.997 | 0.997 | 0.997 | 0.997 | 0.997 | 0.997 | 0.996 | 0.996 |
| K | 190 | 195 | 200 | 205 | 210 | 215 | 220 | 225 | 230 | 235 | 240 | 245 |
| mae | 0.724 | 0.723 | 0.723 | 0.723 | 0.723 | 0.723 | 0.722 | 0.723 | 0.723 | 0.723 | 0.723 | 0.723 |
| rmse | 0.997 | 0.996 | 0.995 | 0.996 | 0.996 | 0.996 | 0.996 | 0.996 | 0.996 | 0.996 | 0.997 | 0.996 |
| K | 250 | 255 | 260 | 265 | 270 | 275 | 280 | 285 | 290 | 295 | 300 | 305 |
| mae | 0.723 | 0.724 | 0.725 | 0.725 | 0.725 | 0.725 | 0.725 | 0.724 | 0.725 | 0.725 | 0.726 | 0.726 |
| rmse | 0.996 | 0.997 | 0.998 | 0.997 | 0.997 | 0.998 | 0.998 | 0.997 | 0.998 | 0.998 | 0.998 | 0.999 |
| K | 310 | 315 | 320 | 325 | 330 | 335 | 340 | 345 | 350 | 355 | 360 | 365 |
| mae | 0.726 | 0.727 | 0.727 | 0.727 | 0.727 | 0.728 | 0.728 | 0.727 | 0.728 | 0.727 | 0.728 | 0.727 |
| rmse | 0.999 | 1 | 0.999 | 0.999 | 0.999 | 1 | 1 | 0.999 | 1 | 1 | 1 | 1 |
| K | 370 | 375 | 380 | 385 | 390 | 395 | 400 | 405 | 410 | 415 | 420 | 425 |
| mae | 0.727 | 0.728 | 0.728 | 0.727 | 0.727 | 0.728 | 0.727 | 0.728 | 0.728 | 0.727 | 0.727 | 0.727 |
| rmse | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| K | 430 | 435 | 440 | 445 | 450 | 455 | 460 | 465 | 470 | 475 | 480 | 485 |
| mae | 0.727 | 0.728 | 0.728 | 0.729 | 0.729 | 0.728 | 0.728 | 0.728 | 0.729 | 0.729 | 0.729 | 0.729 |
| rmse | 1 | 1 | 1 | 1.001 | 1.002 | 1.001 | 1.001 | 1.001 | 1.002 | 1.001 | 1.001 | 1.001 |
| K | 490 | 495 | 500 | 505 | 510 | 515 | 520 | 525 | 530 | 535 | 540 | 545 |
| mae | 0.729 | 0.73 | 0.73 | 0.73 | 0.729 | 0.73 | 0.73 | 0.73 | 0.73 | 0.729 | 0.729 | 0.729 |
| rmse | 1.001 | 1.002 | 1.002 | 1.002 | 1.002 | 1.002 | 1.002 | 1.002 | 1.002 | 1.002 | 1.002 | 1.002 |
| K | 550 | 555 | 560 | 565 | 570 | 575 | 580 | 585 | 590 | 595 | 600 | 605 |
| mae | 0.73 | 0.729 | 0.73 | 0.73 | 0.73 | 0.729 | 0.73 | 0.729 | 0.73 | 0.73 | 0.729 | 0.729 |
| rmse | 1.002 | 1.002 | 1.002 | 1.002 | 1.002 | 1.002 | 1.002 | 1.002 | 1.002 | 1.002 | 1.002 | 1.002 |
| K | 610 | 615 | 620 | 625 | 630 | 635 | 640 | 645 | 650 | 655 | 660 | 665 |
| mae | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.73 | 0.73 |
| rmse | 1.002 | 1.001 | 1.002 | 1.002 | 1.002 | 1.002 | 1.001 | 1.002 | 1.002 | 1.002 | 1.002 | 1.002 |
| K | 670 | 675 | 680 | 685 | 690 | 695 | | | | | | |
| mae | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | | | | | | |
| rmse | 1.002 | 1.002 | 1.002 | 1.002 | 1.001 | 1.002 | | | | | | |

TABLE 19 – Résultats de MAE et RMSE pour l’algorithme de FHyb FC basé Sem et K-NN

| | | | | | | | | | | |
|-----------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| threshold | 0 | 0.056 | 0.113 | 0.17 | 0.227 | 0.284 | 0.34 | 0.397 | 0.454 | 0.511 |
| mae | 0.867 | 0.857 | 0.783 | 0.75 | 0.736 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 |
| rmse | 1.129 | 1.12 | 1.056 | 1.023 | 1.007 | 1.001 | 1.001 | 1.002 | 1.002 | 1.002 |

TABLE 20 – Résultats de MAE et RMSE pour l’algorithme de FHyb Sem basé FC

| | | | | | | | | | | | | |
|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| K | 10 | 15 | 20 | 25 | 30 | 35 | 40 | 45 | 50 | 55 | 60 | 65 |
| mae | 0.733 | 0.732 | 0.733 | 0.732 | 0.733 | 0.736 | 0.735 | 0.734 | 0.736 | 0.739 | 0.74 | 0.746 |
| rmse | 1.004 | 1.004 | 1.006 | 1.005 | 1.006 | 1.008 | 1.008 | 1.008 | 1.009 | 1.014 | 1.013 | 1.015 |
| K | 70 | 75 | 80 | 85 | 90 | 95 | 100 | 105 | 110 | 115 | 120 | 125 |
| mae | 0.737 | 0.742 | 0.741 | 0.739 | 0.742 | 0.741 | 0.743 | 0.749 | 0.749 | 0.745 | 0.745 | 0.745 |
| rmse | 1.01 | 1.015 | 1.015 | 1.014 | 1.018 | 1.013 | 1.016 | 1.023 | 1.021 | 1.021 | 1.015 | 1.019 |
| K | 130 | 135 | 140 | 145 | 150 | 155 | 160 | 165 | 170 | 175 | 180 | 185 |
| mae | 0.748 | 0.745 | 0.749 | 0.746 | 0.752 | 0.753 | 0.749 | 0.758 | 0.756 | 0.756 | 0.755 | 0.754 |
| rmse | 1.02 | 1.019 | 1.022 | 1.022 | 1.027 | 1.024 | 1.024 | 1.032 | 1.034 | 1.032 | 1.028 | 1.032 |
| K | 190 | 195 | 200 | 205 | 210 | 215 | 220 | 225 | 230 | 235 | 240 | 245 |
| mae | 0.759 | 0.761 | 0.754 | 0.763 | 0.753 | 0.761 | 0.759 | 0.756 | 0.761 | 0.761 | 0.768 | 0.761 |
| rmse | 1.033 | 1.035 | 1.028 | 1.039 | 1.028 | 1.033 | 1.035 | 1.032 | 1.034 | 1.038 | 1.046 | 1.037 |
| K | 250 | 255 | 260 | 265 | 270 | 275 | 280 | 285 | 290 | 295 | 300 | 305 |
| mae | 0.767 | 0.763 | 0.758 | 0.761 | 0.771 | 0.768 | 0.767 | 0.768 | 0.773 | 0.771 | 0.775 | 0.772 |
| rmse | 1.045 | 1.038 | 1.035 | 1.033 | 1.044 | 1.042 | 1.042 | 1.037 | 1.044 | 1.044 | 1.049 | 1.043 |
| K | 310 | 315 | 320 | 325 | 330 | 335 | 340 | 345 | 350 | 355 | 360 | 365 |
| mae | 0.778 | 0.769 | 0.77 | 0.772 | 0.781 | 0.776 | 0.775 | 0.776 | 0.776 | 0.774 | 0.777 | 0.779 |
| rmse | 1.049 | 1.046 | 1.04 | 1.041 | 1.054 | 1.051 | 1.047 | 1.053 | 1.051 | 1.047 | 1.052 | 1.053 |
| K | 370 | 375 | 380 | 385 | 390 | 395 | 400 | 405 | 410 | 415 | 420 | 425 |
| mae | 0.787 | 0.784 | 0.784 | 0.781 | 0.782 | 0.781 | 0.778 | 0.787 | 0.787 | 0.784 | 0.8 | 0.778 |
| rmse | 1.06 | 1.058 | 1.052 | 1.054 | 1.056 | 1.053 | 1.053 | 1.06 | 1.064 | 1.059 | 1.072 | 1.052 |
| K | 430 | 435 | 440 | 445 | 450 | 455 | 460 | 465 | 470 | 475 | 480 | 485 |
| mae | 0.774 | 0.788 | 0.789 | 0.789 | 0.791 | 0.797 | 0.793 | 0.797 | 0.788 | 0.787 | 0.792 | 0.794 |
| rmse | 1.051 | 1.06 | 1.066 | 1.062 | 1.065 | 1.069 | 1.066 | 1.068 | 1.058 | 1.062 | 1.067 | 1.068 |
| K | 490 | 495 | 500 | 505 | 510 | 515 | 520 | 525 | 530 | 535 | 540 | 545 |
| mae | 0.791 | 0.792 | 0.807 | 0.805 | 0.796 | 0.797 | 0.793 | 0.806 | 0.8 | 0.794 | 0.798 | 0.802 |
| rmse | 1.062 | 1.063 | 1.08 | 1.08 | 1.069 | 1.071 | 1.066 | 1.079 | 1.073 | 1.064 | 1.07 | 1.075 |
| K | 550 | 555 | 560 | 565 | 570 | 575 | 580 | 585 | 590 | 595 | 600 | 605 |
| mae | 0.797 | 0.794 | 0.811 | 0.8 | 0.798 | 0.796 | 0.803 | 0.812 | 0.792 | 0.801 | 0.805 | 0.808 |
| rmse | 1.069 | 1.067 | 1.082 | 1.073 | 1.072 | 1.073 | 1.073 | 1.083 | 1.062 | 1.07 | 1.077 | 1.081 |
| K | 610 | 615 | 620 | 625 | 630 | 635 | 640 | 645 | 650 | 655 | 660 | 665 |
| mae | 0.81 | 0.803 | 0.798 | 0.807 | 0.814 | 0.8 | 0.797 | 0.812 | 0.802 | 0.804 | 0.816 | 0.813 |
| rmse | 1.076 | 1.078 | 1.071 | 1.081 | 1.088 | 1.072 | 1.069 | 1.087 | 1.074 | 1.075 | 1.09 | 1.084 |
| K | 670 | 675 | 680 | 685 | 690 | 695 | | | | | | |
| mae | 0.818 | 0.818 | 0.812 | 0.818 | 0.82 | 0.822 | | | | | | |
| rmse | 1.089 | 1.086 | 1.085 | 1.091 | 1.089 | 1.093 | | | | | | |

TABLE 21 – Résultats de MAE et RMSE pour l’algorithme de FHyb Sem basé FC et K-medoids

| | | | | | | | | | | | | |
|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| K | 10 | 15 | 20 | 25 | 30 | 35 | 40 | 45 | 50 | 55 | 60 | 65 |
| mae | 0.733 | 0.733 | 0.732 | 0.733 | 0.732 | 0.732 | 0.731 | 0.73 | 0.73 | 0.73 | 0.731 | 0.73 |
| rmse | 1.002 | 1.003 | 1.002 | 1.004 | 1.002 | 1.003 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1 |
| K | 70 | 75 | 80 | 85 | 90 | 95 | 100 | 105 | 110 | 115 | 120 | 125 |
| mae | 0.731 | 0.731 | 0.731 | 0.73 | 0.731 | 0.731 | 0.73 | 0.731 | 0.73 | 0.73 | 0.728 | 0.729 |
| rmse | 1 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1 | 0.999 | 0.999 |
| K | 130 | 135 | 140 | 145 | 150 | 155 | 160 | 165 | 170 | 175 | 180 | 185 |
| mae | 0.729 | 0.728 | 0.729 | 0.729 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 |
| rmse | 0.999 | 0.999 | 0.999 | 1 | 1 | 1 | 1.001 | 1 | 1 | 1.001 | 1.001 | 1.001 |
| K | 190 | 195 | 200 | 205 | 210 | 215 | 220 | 225 | 230 | 235 | 240 | 245 |
| mae | 0.73 | 0.73 | 0.731 | 0.73 | 0.731 | 0.73 | 0.73 | 0.731 | 0.731 | 0.73 | 0.73 | 0.731 |
| rmse | 1 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 |
| K | 250 | 255 | 260 | 265 | 270 | 275 | 280 | 285 | 290 | 295 | 300 | 305 |
| mae | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.729 | 0.73 |
| rmse | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 |
| K | 310 | 315 | 320 | 325 | 330 | 335 | 340 | 345 | 350 | 355 | 360 | 365 |
| mae | 0.73 | 0.729 | 0.73 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 |
| rmse | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1 | 1 | 1.001 | 1 | 1 | 1.001 |
| K | 370 | 375 | 380 | 385 | 390 | 395 | 400 | 405 | 410 | 415 | 420 | 425 |
| mae | 0.729 | 0.729 | 0.73 | 0.73 | 0.73 | 0.73 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 |
| rmse | 1 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1 | 1 | 1.001 | 1 | 1.001 |
| K | 430 | 435 | 440 | 445 | 450 | 455 | 460 | 465 | 470 | 475 | 480 | 485 |
| mae | 0.729 | 0.729 | 0.729 | 0.73 | 0.73 | 0.729 | 0.73 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 |
| rmse | 1.001 | 1 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1 | 1 | 1 | 1 | 1 |
| K | 490 | 495 | 500 | 505 | 510 | 515 | 520 | 525 | 530 | 535 | 540 | 545 |
| mae | 0.73 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.73 | 0.729 | 0.729 | 0.729 | 0.729 |
| rmse | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0.999 | 0.999 |
| K | 550 | 555 | 560 | 565 | 570 | 575 | 580 | 585 | 590 | 595 | 600 | 605 |
| mae | 0.729 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.729 | 0.729 | 0.729 | 0.729 | 0.729 | 0.73 |
| rmse | 1 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1 | 1 | 1.001 | 1.001 | 1.001 |
| K | 610 | 615 | 620 | 625 | 630 | 635 | 640 | 645 | 650 | 655 | 660 | 665 |
| mae | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.731 | 0.731 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 |
| rmse | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.002 | 1.002 | 1.001 | 1.002 | 1.001 | 1.001 | 1.001 |
| K | 670 | 675 | 680 | 685 | 690 | 695 | | | | | | |
| mae | 0.73 | 0.729 | 0.73 | 0.729 | 0.729 | 0.73 | | | | | | |
| rmse | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | 1.001 | | | | | | |

TABLE 22 – Résultats de MAE et RMSE pour l’algorithme de FHyb Sem basé FC et K-NN

| | | | | | | | | | | | | |
|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| K | 10 | 15 | 20 | 25 | 30 | 35 | 40 | 45 | 50 | 55 | 60 | 65 |
| mae | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 |
| rmse | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| K | 70 | 75 | 80 | 85 | 90 | 95 | 100 | 105 | 110 | 115 | 120 | 125 |
| mae | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 |
| rmse | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| K | 130 | 135 | 140 | 145 | 150 | 155 | 160 | 165 | 170 | 175 | 180 | 185 |
| mae | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 |
| rmse | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| K | 190 | 195 | 200 | 205 | 210 | 215 | 220 | 225 | 230 | 235 | 240 | 245 |
| mae | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 |
| rmse | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| K | 250 | 255 | 260 | 265 | 270 | 275 | 280 | 285 | 290 | 295 | 300 | 305 |
| mae | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 |
| rmse | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| K | 310 | 315 | 320 | 325 | 330 | 335 | 340 | 345 | 350 | 355 | 360 | 365 |
| mae | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 |
| rmse | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| K | 370 | 375 | 380 | 385 | 390 | 395 | 400 | 405 | 410 | 415 | 420 | 425 |
| mae | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 |
| rmse | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| K | 430 | 435 | 440 | 445 | 450 | 455 | 460 | 465 | 470 | 475 | 480 | 485 |
| mae | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 |
| rmse | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| K | 490 | 495 | 500 | 505 | 510 | 515 | 520 | 525 | 530 | 535 | 540 | 545 |
| mae | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 |
| rmse | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| K | 550 | 555 | 560 | 565 | 570 | 575 | 580 | 585 | 590 | 595 | 600 | 605 |
| mae | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 |
| rmse | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| K | 610 | 615 | 620 | 625 | 630 | 635 | 640 | 645 | 650 | 655 | 660 | 665 |
| mae | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 |
| rmse | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| K | 670 | 675 | 680 | 685 | 690 | 695 | | | | | | |
| mae | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | 0.731 | | | | | | |
| rmse | 1 | 1 | 1 | 1 | 1 | 1 | | | | | | |

TABLE 23 – Résultats de MAE et RMSE pour l’algorithme de FHyb multivues K-NN

Évaluation de RED

Filtrage collaboratif

| | | | | | | | | | | | | |
|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| K | 5.0 | 6.0 | 7.0 | 8.0 | 9.0 | 10.0 | 11.0 | 12.0 | 13.0 | 14.0 | 15.0 | 16.0 |
| mae | 0.797 | 0.789 | 0.789 | 0.793 | 0.784 | 0.786 | 0.791 | 0.783 | 0.79 | 0.788 | 0.788 | 0.797 |
| rmse | 1.087 | 1.08 | 1.082 | 1.087 | 1.077 | 1.088 | 1.089 | 1.083 | 1.08 | 1.084 | 1.08 | 1.092 |
| K | 17.0 | 18.0 | 19.0 | 20.0 | 21.0 | 22.0 | 23.0 | 24.0 | 25.0 | 26.0 | 27.0 | 28.0 |
| mae | 0.782 | 0.8 | 0.775 | 0.784 | 0.791 | 0.801 | 0.775 | 0.805 | 0.792 | 0.788 | 0.783 | 0.787 |
| rmse | 1.073 | 1.086 | 1.065 | 1.078 | 1.09 | 1.095 | 1.069 | 1.104 | 1.082 | 1.084 | 1.078 | 1.096 |
| K | 29.0 | 30.0 | 31.0 | 32.0 | 33.0 | 34.0 | 35.0 | 36.0 | 37.0 | 38.0 | 39.0 | 40.0 |
| mae | 0.798 | 0.796 | 0.788 | 0.786 | 0.807 | 0.796 | 0.803 | 0.781 | 0.801 | 0.793 | 0.794 | 0.814 |
| rmse | 1.086 | 1.087 | 1.087 | 1.081 | 1.097 | 1.084 | 1.096 | 1.078 | 1.099 | 1.091 | 1.09 | 1.106 |
| K | 41.0 | 42.0 | 43.0 | 44.0 | 45.0 | 46.0 | 47.0 | 48.0 | 49.0 | 50.0 | 51.0 | 52.0 |
| mae | 0.792 | 0.806 | 0.783 | 0.788 | 0.792 | 0.785 | 0.797 | 0.778 | 0.798 | 0.792 | 0.785 | 0.799 |
| rmse | 1.089 | 1.102 | 1.076 | 1.085 | 1.093 | 1.088 | 1.092 | 1.081 | 1.104 | 1.083 | 1.088 | 1.092 |
| K | 53.0 | 54.0 | 55.0 | 56.0 | 57.0 | 58.0 | 59.0 | 60.0 | 61.0 | 62.0 | 63.0 | 64.0 |
| mae | 0.803 | 0.797 | 0.801 | 0.787 | 0.791 | 0.795 | 0.791 | 0.79 | 0.8 | 0.803 | 0.803 | 0.813 |
| rmse | 1.094 | 1.094 | 1.088 | 1.087 | 1.1 | 1.1 | 1.09 | 1.084 | 1.105 | 1.101 | 1.096 | 1.111 |
| K | 65.0 | 66.0 | 67.0 | 68.0 | 69.0 | 70.0 | 71.0 | 72.0 | 73.0 | | | |
| mae | 0.788 | 0.798 | 0.788 | 0.796 | 0.802 | 0.798 | 0.798 | 0.781 | 0.816 | | | |
| rmse | 1.096 | 1.095 | 1.085 | 1.099 | 1.103 | 1.105 | 1.095 | 1.085 | 1.113 | | | |

TABLE 24 – Résultats de MAE et RMSE pour l’algorithme FC basé K-medoids sur le dataset RED

| | | | | | | | | | | | | |
|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| K | 5.0 | 6.0 | 7.0 | 8.0 | 9.0 | 10.0 | 11.0 | 12.0 | 13.0 | 14.0 | 15.0 | 16.0 |
| mae | 0.774 | 0.789 | 0.783 | 0.79 | 0.781 | 0.785 | 0.779 | 0.779 | 0.781 | 0.783 | 0.785 | 0.782 |
| rmse | 1.065 | 1.077 | 1.076 | 1.079 | 1.072 | 1.075 | 1.071 | 1.071 | 1.071 | 1.073 | 1.075 | 1.075 |
| K | 17.0 | 18.0 | 19.0 | 20.0 | 21.0 | 22.0 | 23.0 | 24.0 | 25.0 | 26.0 | 27.0 | 28.0 |
| mae | 0.779 | 0.781 | 0.785 | 0.789 | 0.789 | 0.792 | 0.789 | 0.787 | 0.79 | 0.788 | 0.787 | 0.79 |
| rmse | 1.07 | 1.075 | 1.077 | 1.079 | 1.078 | 1.079 | 1.077 | 1.077 | 1.08 | 1.077 | 1.078 | 1.078 |
| K | 29.0 | 30.0 | 31.0 | 32.0 | 33.0 | 34.0 | 35.0 | 36.0 | 37.0 | 38.0 | 39.0 | 40.0 |
| mae | 0.789 | 0.791 | 0.787 | 0.789 | 0.789 | 0.791 | 0.791 | 0.79 | 0.79 | 0.789 | 0.788 | 0.788 |
| rmse | 1.079 | 1.08 | 1.076 | 1.077 | 1.078 | 1.08 | 1.078 | 1.078 | 1.078 | 1.078 | 1.077 | 1.077 |
| K | 41.0 | 42.0 | 43.0 | 44.0 | 45.0 | 46.0 | 47.0 | 48.0 | 49.0 | 50.0 | 51.0 | 52.0 |
| mae | 0.789 | 0.791 | 0.789 | 0.791 | 0.79 | 0.793 | 0.791 | 0.791 | 0.791 | 0.793 | 0.79 | 0.787 |
| rmse | 1.079 | 1.08 | 1.078 | 1.079 | 1.078 | 1.082 | 1.079 | 1.08 | 1.08 | 1.081 | 1.078 | 1.076 |
| K | 53.0 | 54.0 | 55.0 | 56.0 | 57.0 | 58.0 | 59.0 | 60.0 | 61.0 | 62.0 | 63.0 | 64.0 |
| mae | 0.79 | 0.789 | 0.789 | 0.789 | 0.787 | 0.789 | 0.787 | 0.788 | 0.787 | 0.787 | 0.789 | 0.787 |
| rmse | 1.08 | 1.078 | 1.079 | 1.079 | 1.078 | 1.079 | 1.078 | 1.079 | 1.078 | 1.077 | 1.078 | 1.078 |
| K | 65.0 | 66.0 | 67.0 | 68.0 | 69.0 | 70.0 | 71.0 | 72.0 | 73.0 | 74.0 | 75.0 | 76.0 |
| mae | 0.788 | 0.787 | 0.787 | 0.787 | 0.787 | 0.786 | 0.786 | 0.785 | 0.786 | 0.786 | 0.786 | 0.787 |
| rmse | 1.078 | 1.077 | 1.077 | 1.077 | 1.077 | 1.076 | 1.076 | 1.075 | 1.076 | 1.076 | 1.076 | 1.076 |
| K | 77.0 | 78.0 | 79.0 | 80.0 | 81.0 | 82.0 | 83.0 | 84.0 | 85.0 | 86.0 | 87.0 | 88.0 |
| mae | 0.789 | 0.789 | 0.789 | 0.79 | 0.79 | 0.791 | 0.789 | 0.789 | 0.789 | 0.789 | 0.789 | 0.791 |
| rmse | 1.078 | 1.078 | 1.079 | 1.078 | 1.078 | 1.079 | 1.078 | 1.078 | 1.079 | 1.078 | 1.079 | 1.081 |
| K | 89.0 | 90.0 | 91.0 | 92.0 | 93.0 | 94.0 | 95.0 | 96.0 | 97.0 | 98.0 | 99.0 | 100.0 |
| mae | 0.79 | 0.791 | 0.791 | 0.791 | 0.79 | 0.79 | 0.791 | 0.791 | 0.791 | 0.79 | 0.79 | 0.79 |
| rmse | 1.081 | 1.081 | 1.081 | 1.081 | 1.08 | 1.08 | 1.08 | 1.08 | 1.08 | 1.079 | 1.079 | 1.078 |
| K | 101.0 | 102.0 | 103.0 | 104.0 | 105.0 | 106.0 | 107.0 | 108.0 | 109.0 | 110.0 | 111.0 | 112.0 |
| mae | 0.791 | 0.791 | 0.791 | 0.79 | 0.792 | 0.791 | 0.792 | 0.792 | 0.793 | 0.791 | 0.791 | 0.791 |
| rmse | 1.08 | 1.08 | 1.081 | 1.08 | 1.08 | 1.08 | 1.081 | 1.082 | 1.083 | 1.081 | 1.08 | 1.081 |
| K | 113.0 | 114.0 | 115.0 | 116.0 | 117.0 | 118.0 | 119.0 | 120.0 | 121.0 | 122.0 | 123.0 | 124.0 |
| mae | 0.792 | 0.791 | 0.791 | 0.791 | 0.791 | 0.791 | 0.791 | 0.791 | 0.79 | 0.791 | 0.789 | 0.789 |
| rmse | 1.081 | 1.08 | 1.08 | 1.08 | 1.081 | 1.08 | 1.08 | 1.081 | 1.081 | 1.081 | 1.078 | 1.078 |
| K | 125.0 | 126.0 | 127.0 | 128.0 | 129.0 | 130.0 | 131.0 | 132.0 | 133.0 | 134.0 | 135.0 | 136.0 |
| mae | 0.79 | 0.787 | 0.788 | 0.789 | 0.789 | 0.789 | 0.789 | 0.789 | 0.789 | 0.789 | 0.789 | 0.789 |
| rmse | 1.08 | 1.076 | 1.078 | 1.078 | 1.078 | 1.077 | 1.077 | 1.079 | 1.079 | 1.078 | 1.078 | 1.078 |
| K | 137.0 | 138.0 | 139.0 | 140.0 | 141.0 | 142.0 | 143.0 | 144.0 | 145.0 | 146.0 | 147.0 | 148.0 |
| mae | 0.79 | 0.79 | 0.791 | 0.79 | 0.79 | 0.792 | 0.791 | 0.791 | 0.791 | 0.791 | 0.791 | 0.791 |
| rmse | 1.079 | 1.08 | 1.079 | 1.08 | 1.08 | 1.08 | 1.08 | 1.08 | 1.08 | 1.081 | 1.084 | 1.083 |

TABLE 25 – Résultats de MAE et RMSE pour l’algorithme FC basé K-NN sur le dataset RED

| | | | | | | | | | | | | |
|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| K | 5.0 | 6.0 | 7.0 | 8.0 | 9.0 | 10.0 | 11.0 | 12.0 | 13.0 | 14.0 | 15.0 | 16.0 |
| mae | 0.779 | 0.777 | 0.773 | 0.775 | 0.781 | 0.783 | 0.775 | 0.781 | 0.773 | 0.772 | 0.775 | 0.775 |
| rmse | 1.072 | 1.074 | 1.07 | 1.073 | 1.08 | 1.082 | 1.073 | 1.073 | 1.073 | 1.065 | 1.068 | 1.075 |
| K | 17.0 | 18.0 | 19.0 | 20.0 | 21.0 | 22.0 | 23.0 | 24.0 | 25.0 | 26.0 | 27.0 | 28.0 |
| mae | 0.767 | 0.775 | 0.774 | 0.776 | 0.777 | 0.779 | 0.771 | 0.772 | 0.779 | 0.775 | 0.779 | 0.778 |
| rmse | 1.068 | 1.075 | 1.069 | 1.073 | 1.078 | 1.074 | 1.076 | 1.075 | 1.071 | 1.081 | 1.078 | 1.081 |
| K | 29.0 | 30.0 | 31.0 | 32.0 | 33.0 | 34.0 | 35.0 | 36.0 | 37.0 | 38.0 | 39.0 | 40.0 |
| mae | 0.775 | 0.778 | 0.779 | 0.773 | 0.775 | 0.776 | 0.771 | 0.769 | 0.777 | 0.781 | 0.778 | 0.785 |
| rmse | 1.07 | 1.073 | 1.077 | 1.07 | 1.075 | 1.076 | 1.071 | 1.069 | 1.079 | 1.081 | 1.083 | 1.082 |
| K | 41.0 | 42.0 | 43.0 | 44.0 | 45.0 | 46.0 | 47.0 | 48.0 | 49.0 | 50.0 | 51.0 | 52.0 |
| mae | 0.775 | 0.77 | 0.779 | 0.777 | 0.772 | 0.785 | 0.781 | 0.777 | 0.776 | 0.772 | 0.777 | 0.783 |
| rmse | 1.069 | 1.07 | 1.074 | 1.076 | 1.072 | 1.081 | 1.077 | 1.08 | 1.082 | 1.08 | 1.075 | 1.091 |
| K | 53.0 | 54.0 | 55.0 | 56.0 | 57.0 | 58.0 | 59.0 | 60.0 | 61.0 | 62.0 | 63.0 | 64.0 |
| mae | 0.781 | 0.78 | 0.781 | 0.787 | 0.779 | 0.773 | 0.782 | 0.786 | 0.782 | 0.785 | 0.777 | 0.768 |
| rmse | 1.08 | 1.085 | 1.076 | 1.082 | 1.075 | 1.072 | 1.084 | 1.091 | 1.08 | 1.084 | 1.08 | 1.075 |
| K | 65.0 | 66.0 | 67.0 | 68.0 | 69.0 | 70.0 | 71.0 | 72.0 | 73.0 | | | |
| mae | 0.777 | 0.781 | 0.795 | 0.775 | 0.775 | 0.792 | 0.784 | 0.791 | 0.782 | | | |
| rmse | 1.071 | 1.085 | 1.096 | 1.075 | 1.077 | 1.094 | 1.078 | 1.097 | 1.087 | | | |

TABLE 26 – Résultats de MAE et RMSE pour l’algorithme FC basé K-medoids et BSO sur le dataset RED

Filtrage sémantique

| | | | | | | | | | | | | |
|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| K | k | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| mae | mae | 0.785 | 0.794 | 0.785 | 0.785 | 0.787 | 0.783 | 0.79 | 0.776 | 0.784 | 0.788 | 0.784 |
| rmse | rmse | 1.076 | 1.085 | 1.078 | 1.077 | 1.079 | 1.082 | 1.082 | 1.075 | 1.083 | 1.079 | 1.091 |
| K | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 |
| mae | 0.795 | 0.784 | 0.785 | 0.787 | 0.781 | 0.78 | 0.779 | 0.796 | 0.78 | 0.792 | 0.786 | 0.794 |
| rmse | 1.086 | 1.079 | 1.078 | 1.084 | 1.074 | 1.082 | 1.075 | 1.093 | 1.077 | 1.093 | 1.083 | 1.095 |
| K | 28 | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 |
| mae | 0.789 | 0.784 | 0.809 | 0.798 | 0.777 | 0.778 | 0.785 | 0.778 | 0.785 | 0.793 | 0.797 | 0.787 |
| rmse | 1.084 | 1.086 | 1.108 | 1.095 | 1.077 | 1.072 | 1.082 | 1.08 | 1.085 | 1.094 | 1.088 | 1.088 |
| K | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 50 | 51 |
| mae | 0.789 | 0.782 | 0.793 | 0.78 | 0.809 | 0.797 | 0.777 | 0.792 | 0.796 | 0.798 | 0.793 | 0.786 |
| rmse | 1.079 | 1.084 | 1.104 | 1.079 | 1.109 | 1.097 | 1.082 | 1.092 | 1.096 | 1.1 | 1.101 | 1.077 |
| K | 52 | 53 | 54 | 55 | 56 | 57 | 58 | 59 | 60 | 61 | 62 | 63 |
| mae | 0.8 | 0.792 | 0.806 | 0.795 | 0.801 | 0.794 | 0.801 | 0.799 | 0.798 | 0.801 | 0.796 | 0.8 |
| rmse | 1.098 | 1.088 | 1.116 | 1.092 | 1.098 | 1.091 | 1.109 | 1.094 | 1.103 | 1.099 | 1.097 | 1.111 |
| K | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | 73 | | |
| mae | 0.79 | 0.788 | 0.83 | 0.795 | 0.784 | 0.815 | 0.801 | 0.8 | 0.805 | 0.804 | | |
| rmse | 1.088 | 1.104 | 1.127 | 1.094 | 1.086 | 1.118 | 1.109 | 1.112 | 1.106 | 1.11 | | |

TABLE 27 – Résultats de MAE et RMSE pour l’algorithme FSem basé K-medoids sur le dataset RED

| | | | | | | | | | | | | |
|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| K | 5.0 | 6.0 | 7.0 | 8.0 | 9.0 | 10.0 | 11.0 | 12.0 | 13.0 | 14.0 | 15.0 | 16.0 |
| mae | 0.779 | 0.775 | 0.774 | 0.778 | 0.773 | 0.775 | 0.772 | 0.775 | 0.773 | 0.774 | 0.769 | 0.78 |
| rmse | 1.074 | 1.068 | 1.073 | 1.076 | 1.073 | 1.075 | 1.069 | 1.077 | 1.072 | 1.073 | 1.07 | 1.082 |
| K | 17.0 | 18.0 | 19.0 | 20.0 | 21.0 | 22.0 | 23.0 | 24.0 | 25.0 | 26.0 | 27.0 | 28.0 |
| mae | 0.768 | 0.775 | 0.773 | 0.766 | 0.768 | 0.772 | 0.771 | 0.774 | 0.769 | 0.771 | 0.774 | 0.774 |
| rmse | 1.069 | 1.073 | 1.075 | 1.068 | 1.071 | 1.08 | 1.075 | 1.076 | 1.067 | 1.067 | 1.068 | 1.079 |
| K | 29.0 | 30.0 | 31.0 | 32.0 | 33.0 | 34.0 | 35.0 | 36.0 | 37.0 | 38.0 | 39.0 | 40.0 |
| mae | 0.777 | 0.771 | 0.779 | 0.775 | 0.773 | 0.779 | 0.775 | 0.778 | 0.773 | 0.785 | 0.775 | 0.778 |
| rmse | 1.078 | 1.066 | 1.079 | 1.073 | 1.078 | 1.076 | 1.078 | 1.081 | 1.07 | 1.086 | 1.076 | 1.08 |
| K | 41.0 | 42.0 | 43.0 | 44.0 | 45.0 | 46.0 | 47.0 | 48.0 | 49.0 | 50.0 | 51.0 | 52.0 |
| mae | 0.769 | 0.775 | 0.763 | 0.773 | 0.777 | 0.766 | 0.781 | 0.779 | 0.779 | 0.775 | 0.779 | 0.79 |
| rmse | 1.075 | 1.073 | 1.064 | 1.073 | 1.081 | 1.075 | 1.076 | 1.076 | 1.079 | 1.08 | 1.077 | 1.088 |
| K | 53.0 | 54.0 | 55.0 | 56.0 | 57.0 | 58.0 | 59.0 | 60.0 | 61.0 | 62.0 | 63.0 | 64.0 |
| mae | 0.79 | 0.775 | 0.779 | 0.771 | 0.779 | 0.784 | 0.783 | 0.781 | 0.776 | 0.781 | 0.791 | 0.781 |
| rmse | 1.088 | 1.076 | 1.078 | 1.065 | 1.085 | 1.085 | 1.087 | 1.084 | 1.079 | 1.08 | 1.092 | 1.087 |
| K | 65.0 | 66.0 | 67.0 | 68.0 | 69.0 | 70.0 | 71.0 | 72.0 | 73.0 | | | |
| mae | 0.786 | 0.789 | 0.779 | 0.787 | 0.785 | 0.785 | 0.782 | 0.79 | 0.781 | | | |
| rmse | 1.084 | 1.088 | 1.084 | 1.088 | 1.092 | 1.088 | 1.079 | 1.087 | 1.085 | | | |

TABLE 28 – Résultats de MAE et RMSE pour l’algorithme FSem basé K-medoids et BSO sur le dataset RED

| | | | | | | | | | | | | |
|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| K | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| mae | 0.787 | 0.783 | 0.781 | 0.777 | 0.787 | 0.792 | 0.79 | 0.789 | 0.79 | 0.796 | 0.793 | 0.789 |
| rmse | 1.087 | 1.083 | 1.08 | 1.079 | 1.086 | 1.092 | 1.088 | 1.087 | 1.086 | 1.09 | 1.088 | 1.086 |
| K | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 |
| mae | 0.79 | 0.792 | 0.791 | 0.793 | 0.792 | 0.789 | 0.791 | 0.793 | 0.787 | 0.787 | 0.79 | 0.787 |
| rmse | 1.087 | 1.088 | 1.086 | 1.089 | 1.089 | 1.087 | 1.087 | 1.088 | 1.083 | 1.083 | 1.086 | 1.083 |
| K | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 |
| mae | 0.787 | 0.786 | 0.787 | 0.787 | 0.789 | 0.789 | 0.785 | 0.786 | 0.787 | 0.789 | 0.788 | 0.787 |
| rmse | 1.082 | 1.083 | 1.083 | 1.083 | 1.085 | 1.084 | 1.081 | 1.082 | 1.084 | 1.084 | 1.084 | 1.083 |
| K | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 50 | 51 | 52 |
| mae | 0.789 | 0.787 | 0.789 | 0.791 | 0.789 | 0.789 | 0.788 | 0.789 | 0.787 | 0.789 | 0.788 | 0.789 |
| rmse | 1.085 | 1.083 | 1.083 | 1.084 | 1.083 | 1.083 | 1.081 | 1.082 | 1.081 | 1.083 | 1.082 | 1.084 |
| K | 53 | 54 | 55 | 56 | 57 | 58 | 59 | 60 | 61 | 62 | 63 | 64 |
| mae | 0.789 | 0.787 | 0.787 | 0.788 | 0.789 | 0.789 | 0.789 | 0.79 | 0.789 | 0.789 | 0.791 | 0.791 |
| rmse | 1.084 | 1.082 | 1.081 | 1.083 | 1.083 | 1.081 | 1.081 | 1.084 | 1.083 | 1.08 | 1.084 | 1.082 |
| K | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | 73 | 74 | 75 | 76 |
| mae | 0.791 | 0.79 | 0.789 | 0.788 | 0.788 | 0.789 | 0.787 | 0.787 | 0.787 | 0.789 | 0.789 | 0.787 |
| rmse | 1.082 | 1.082 | 1.082 | 1.08 | 1.08 | 1.081 | 1.081 | 1.082 | 1.082 | 1.084 | 1.084 | 1.081 |
| K | 77 | 78 | 79 | 80 | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 |
| mae | 0.786 | 0.787 | 0.788 | 0.789 | 0.789 | 0.788 | 0.789 | 0.788 | 0.788 | 0.787 | 0.787 | 0.786 |
| rmse | 1.079 | 1.082 | 1.082 | 1.082 | 1.082 | 1.082 | 1.082 | 1.08 | 1.082 | 1.081 | 1.081 | 1.081 |
| K | 89 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 | 98 | 99 | 100 |
| mae | 0.787 | 0.788 | 0.789 | 0.788 | 0.788 | 0.79 | 0.789 | 0.789 | 0.789 | 0.789 | 0.789 | 0.79 |
| rmse | 1.081 | 1.082 | 1.083 | 1.082 | 1.082 | 1.084 | 1.084 | 1.084 | 1.084 | 1.083 | 1.083 | 1.084 |
| K | 101 | 102 | 103 | 104 | 105 | 106 | 107 | 108 | 109 | 110 | 111 | 112 |
| mae | 0.789 | 0.791 | 0.791 | 0.791 | 0.791 | 0.791 | 0.792 | 0.791 | 0.789 | 0.791 | 0.791 | 0.791 |
| rmse | 1.083 | 1.084 | 1.084 | 1.084 | 1.084 | 1.083 | 1.084 | 1.083 | 1.082 | 1.082 | 1.083 | 1.083 |
| K | 113 | 114 | 115 | 116 | 117 | 118 | 119 | 120 | 121 | 122 | 123 | 124 |
| mae | 0.791 | 0.791 | 0.79 | 0.79 | 0.788 | 0.787 | 0.788 | 0.787 | 0.788 | 0.79 | 0.791 | 0.791 |
| rmse | 1.083 | 1.082 | 1.082 | 1.082 | 1.08 | 1.08 | 1.08 | 1.08 | 1.08 | 1.081 | 1.083 | 1.083 |
| K | 125 | 126 | 127 | 128 | 129 | 130 | 131 | 132 | 133 | 134 | 135 | 136 |
| mae | 0.791 | 0.79 | 0.79 | 0.791 | 0.79 | 0.79 | 0.789 | 0.789 | 0.789 | 0.789 | 0.789 | 0.79 |
| rmse | 1.083 | 1.083 | 1.083 | 1.085 | 1.083 | 1.084 | 1.084 | 1.082 | 1.082 | 1.082 | 1.082 | 1.082 |
| K | 137 | 138 | 139 | 140 | 141 | 142 | 143 | 144 | 145 | 146 | 147 | 148 |
| mae | 0.79 | 0.791 | 0.791 | 0.79 | 0.789 | 0.79 | 0.79 | 0.791 | 0.791 | 0.791 | 0.791 | 0.79 |
| rmse | 1.081 | 1.083 | 1.082 | 1.08 | 1.079 | 1.08 | 1.08 | 1.08 | 1.08 | 1.08 | 1.08 | 1.08 |

TABLE 29 – Résultats de MAE et RMSE pour l’algorithme FSem basé K-NN sur le dataset RED

Filtrage hybride

| | | | | | | | | | | | | |
|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| K | 5.0 | 6.0 | 7.0 | 8.0 | 9.0 | 10.0 | 11.0 | 12.0 | 13.0 | 14.0 | 15.0 | 16.0 |
| mae | 0.791 | 0.786 | 0.797 | 0.793 | 0.792 | 0.785 | 0.794 | 0.794 | 0.789 | 0.797 | 0.792 | 0.795 |
| rmse | 1.082 | 1.075 | 1.091 | 1.082 | 1.082 | 1.076 | 1.085 | 1.085 | 1.08 | 1.09 | 1.083 | 1.084 |
| K | 17.0 | 18.0 | 19.0 | 20.0 | 21.0 | 22.0 | 23.0 | 24.0 | 25.0 | 26.0 | 27.0 | 28.0 |
| mae | 0.796 | 0.801 | 0.792 | 0.792 | 0.804 | 0.797 | 0.795 | 0.791 | 0.802 | 0.8 | 0.795 | 0.795 |
| rmse | 1.088 | 1.092 | 1.083 | 1.085 | 1.089 | 1.082 | 1.087 | 1.087 | 1.087 | 1.094 | 1.088 | 1.088 |
| K | 29.0 | 30.0 | 31.0 | 32.0 | 33.0 | 34.0 | 35.0 | 36.0 | 37.0 | 38.0 | 39.0 | 40.0 |
| mae | 0.786 | 0.809 | 0.793 | 0.805 | 0.801 | 0.784 | 0.801 | 0.794 | 0.806 | 0.792 | 0.821 | 0.791 |
| rmse | 1.079 | 1.094 | 1.081 | 1.096 | 1.096 | 1.085 | 1.093 | 1.084 | 1.098 | 1.079 | 1.11 | 1.096 |
| K | 41.0 | 42.0 | 43.0 | 44.0 | 45.0 | 46.0 | 47.0 | 48.0 | 49.0 | 50.0 | 51.0 | 52.0 |
| mae | 0.789 | 0.798 | 0.801 | 0.802 | 0.805 | 0.812 | 0.797 | 0.81 | 0.803 | 0.793 | 0.81 | 0.819 |
| rmse | 1.087 | 1.097 | 1.099 | 1.098 | 1.1 | 1.098 | 1.102 | 1.101 | 1.105 | 1.087 | 1.108 | 1.107 |
| K | 53.0 | 54.0 | 55.0 | 56.0 | 57.0 | 58.0 | 59.0 | 60.0 | 61.0 | 62.0 | 63.0 | 64.0 |
| mae | 0.816 | 0.812 | 0.809 | 0.799 | 0.8 | 0.801 | 0.821 | 0.8 | 0.827 | 0.813 | 0.818 | 0.814 |
| rmse | 1.109 | 1.107 | 1.116 | 1.104 | 1.1 | 1.094 | 1.112 | 1.105 | 1.111 | 1.108 | 1.113 | 1.117 |
| K | 65.0 | 66.0 | 67.0 | 68.0 | 69.0 | 70.0 | 71.0 | 72.0 | 73.0 | | | |
| mae | 0.821 | 0.813 | 0.818 | 0.804 | 0.8 | 0.809 | 0.808 | 0.807 | 0.818 | | | |
| rmse | 1.113 | 1.107 | 1.102 | 1.097 | 1.095 | 1.105 | 1.101 | 1.107 | 1.113 | | | |

TABLE 30 – Résultats de MAE et RMSE pour l’algorithme FHyb FC basé Sem et K-medoids sur le dataset RED

| | | | | | | | | | | | | |
|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| K | 5.0 | 6.0 | 7.0 | 8.0 | 9.0 | 10.0 | 11.0 | 12.0 | 13.0 | 14.0 | 15.0 | 16.0 |
| mae | 0.784 | 0.785 | 0.783 | 0.785 | 0.786 | 0.782 | 0.79 | 0.786 | 0.78 | 0.782 | 0.788 | 0.788 |
| rmse | 1.07 | 1.078 | 1.077 | 1.073 | 1.08 | 1.075 | 1.083 | 1.071 | 1.076 | 1.075 | 1.082 | 1.074 |
| K | 17.0 | 18.0 | 19.0 | 20.0 | 21.0 | 22.0 | 23.0 | 24.0 | 25.0 | 26.0 | 27.0 | 28.0 |
| mae | 0.781 | 0.788 | 0.786 | 0.787 | 0.785 | 0.787 | 0.786 | 0.786 | 0.788 | 0.783 | 0.783 | 0.783 |
| rmse | 1.076 | 1.077 | 1.079 | 1.076 | 1.075 | 1.087 | 1.076 | 1.082 | 1.082 | 1.076 | 1.073 | 1.083 |
| K | 29.0 | 30.0 | 31.0 | 32.0 | 33.0 | 34.0 | 35.0 | 36.0 | 37.0 | 38.0 | 39.0 | 40.0 |
| mae | 0.794 | 0.791 | 0.787 | 0.782 | 0.785 | 0.784 | 0.793 | 0.781 | 0.788 | 0.779 | 0.789 | 0.798 |
| rmse | 1.086 | 1.087 | 1.081 | 1.08 | 1.081 | 1.083 | 1.079 | 1.079 | 1.085 | 1.082 | 1.082 | 1.089 |
| K | 41.0 | 42.0 | 43.0 | 44.0 | 45.0 | 46.0 | 47.0 | 48.0 | 49.0 | 50.0 | 51.0 | 52.0 |
| mae | 0.783 | 0.791 | 0.795 | 0.8 | 0.796 | 0.8 | 0.8 | 0.797 | 0.8 | 0.794 | 0.792 | 0.796 |
| rmse | 1.084 | 1.085 | 1.096 | 1.086 | 1.086 | 1.092 | 1.092 | 1.099 | 1.095 | 1.096 | 1.09 | 1.091 |
| K | 53.0 | 54.0 | 55.0 | 56.0 | 57.0 | 58.0 | 59.0 | 60.0 | 61.0 | 62.0 | 63.0 | 64.0 |
| mae | 0.791 | 0.799 | 0.789 | 0.8 | 0.793 | 0.795 | 0.791 | 0.799 | 0.801 | 0.789 | 0.795 | 0.806 |
| rmse | 1.091 | 1.093 | 1.095 | 1.091 | 1.091 | 1.091 | 1.088 | 1.1 | 1.103 | 1.077 | 1.09 | 1.104 |
| K | 65.0 | 66.0 | 67.0 | 68.0 | 69.0 | 70.0 | 71.0 | 72.0 | 73.0 | | | |
| mae | 0.797 | 0.808 | 0.798 | 0.798 | 0.808 | 0.804 | 0.807 | 0.797 | 0.798 | | | |
| rmse | 1.095 | 1.099 | 1.098 | 1.103 | 1.104 | 1.096 | 1.107 | 1.087 | 1.094 | | | |

TABLE 31 – Résultats de MAE et RMSE pour l’algorithme FHyb FC basé Sem et K-medoids-BSO sur le dataset RED

| | | | | | | | | | | | | |
|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| K | 5.0 | 6.0 | 7.0 | 8.0 | 9.0 | 10.0 | 11.0 | 12.0 | 13.0 | 14.0 | 15.0 | 16.0 |
| mae | 0.787 | 0.789 | 0.783 | 0.781 | 0.783 | 0.779 | 0.781 | 0.784 | 0.787 | 0.786 | 0.788 | 0.787 |
| rmse | 1.085 | 1.083 | 1.078 | 1.075 | 1.075 | 1.072 | 1.078 | 1.077 | 1.081 | 1.081 | 1.085 | 1.085 |
| K | 17.0 | 18.0 | 19.0 | 20.0 | 21.0 | 22.0 | 23.0 | 24.0 | 25.0 | 26.0 | 27.0 | 28.0 |
| mae | 0.791 | 0.789 | 0.789 | 0.786 | 0.788 | 0.787 | 0.786 | 0.787 | 0.789 | 0.791 | 0.788 | 0.791 |
| rmse | 1.085 | 1.086 | 1.084 | 1.081 | 1.082 | 1.081 | 1.081 | 1.081 | 1.082 | 1.083 | 1.081 | 1.084 |
| K | 29.0 | 30.0 | 31.0 | 32.0 | 33.0 | 34.0 | 35.0 | 36.0 | 37.0 | 38.0 | 39.0 | 40.0 |
| mae | 0.791 | 0.789 | 0.789 | 0.789 | 0.79 | 0.791 | 0.789 | 0.789 | 0.79 | 0.793 | 0.793 | 0.793 |
| rmse | 1.085 | 1.084 | 1.085 | 1.084 | 1.084 | 1.084 | 1.084 | 1.081 | 1.082 | 1.083 | 1.083 | 1.084 |
| K | 41.0 | 42.0 | 43.0 | 44.0 | 45.0 | 46.0 | 47.0 | 48.0 | 49.0 | 50.0 | 51.0 | 52.0 |
| mae | 0.79 | 0.789 | 0.791 | 0.793 | 0.795 | 0.796 | 0.793 | 0.791 | 0.79 | 0.789 | 0.793 | 0.793 |
| rmse | 1.084 | 1.084 | 1.084 | 1.085 | 1.086 | 1.086 | 1.085 | 1.083 | 1.083 | 1.082 | 1.085 | 1.085 |
| K | 53.0 | 54.0 | 55.0 | 56.0 | 57.0 | 58.0 | 59.0 | 60.0 | 61.0 | 62.0 | 63.0 | 64.0 |
| mae | 0.795 | 0.793 | 0.793 | 0.793 | 0.793 | 0.791 | 0.791 | 0.791 | 0.79 | 0.789 | 0.789 | 0.791 |
| rmse | 1.087 | 1.086 | 1.086 | 1.086 | 1.087 | 1.084 | 1.084 | 1.084 | 1.083 | 1.083 | 1.082 | 1.084 |
| K | 65.0 | 66.0 | 67.0 | 68.0 | 69.0 | 70.0 | 71.0 | 72.0 | 73.0 | 74.0 | 75.0 | 76.0 |
| mae | 0.789 | 0.79 | 0.789 | 0.788 | 0.788 | 0.789 | 0.789 | 0.791 | 0.789 | 0.79 | 0.791 | 0.791 |
| rmse | 1.083 | 1.084 | 1.084 | 1.082 | 1.081 | 1.08 | 1.082 | 1.084 | 1.084 | 1.084 | 1.084 | 1.084 |
| K | 77.0 | 78.0 | 79.0 | 80.0 | 81.0 | 82.0 | 83.0 | 84.0 | 85.0 | 86.0 | 87.0 | 88.0 |
| mae | 0.792 | 0.793 | 0.792 | 0.791 | 0.793 | 0.793 | 0.792 | 0.79 | 0.791 | 0.791 | 0.791 | 0.791 |
| rmse | 1.084 | 1.085 | 1.085 | 1.083 | 1.084 | 1.084 | 1.083 | 1.081 | 1.082 | 1.082 | 1.082 | 1.081 |
| K | 89.0 | 90.0 | 91.0 | 92.0 | 93.0 | 94.0 | 95.0 | 96.0 | 97.0 | 98.0 | 99.0 | 100.0 |
| mae | 0.789 | 0.791 | 0.791 | 0.791 | 0.791 | 0.791 | 0.79 | 0.792 | 0.791 | 0.791 | 0.793 | 0.792 |
| rmse | 1.079 | 1.08 | 1.08 | 1.081 | 1.081 | 1.08 | 1.08 | 1.081 | 1.08 | 1.08 | 1.083 | 1.081 |
| K | 101.0 | 102.0 | 103.0 | 104.0 | 105.0 | 106.0 | 107.0 | 108.0 | 109.0 | 110.0 | 111.0 | 112.0 |
| mae | 0.79 | 0.791 | 0.793 | 0.793 | 0.793 | 0.794 | 0.795 | 0.794 | 0.795 | 0.793 | 0.793 | 0.793 |
| rmse | 1.08 | 1.08 | 1.081 | 1.081 | 1.082 | 1.081 | 1.082 | 1.081 | 1.082 | 1.081 | 1.082 | 1.082 |
| K | 113.0 | 114.0 | 115.0 | 116.0 | 117.0 | 118.0 | 119.0 | 120.0 | 121.0 | 122.0 | 123.0 | 124.0 |
| mae | 0.793 | 0.794 | 0.793 | 0.792 | 0.793 | 0.793 | 0.793 | 0.793 | 0.792 | 0.793 | 0.793 | 0.793 |
| rmse | 1.082 | 1.083 | 1.082 | 1.081 | 1.082 | 1.082 | 1.082 | 1.082 | 1.081 | 1.084 | 1.084 | 1.084 |
| K | 125.0 | 126.0 | 127.0 | 128.0 | 129.0 | 130.0 | 131.0 | 132.0 | 133.0 | 134.0 | 135.0 | 136.0 |
| mae | 0.793 | 0.793 | 0.793 | 0.793 | 0.793 | 0.792 | 0.792 | 0.793 | 0.793 | 0.792 | 0.792 | 0.793 |
| rmse | 1.084 | 1.084 | 1.083 | 1.083 | 1.084 | 1.082 | 1.081 | 1.083 | 1.083 | 1.081 | 1.081 | 1.083 |
| K | 137.0 | 138.0 | 139.0 | 140.0 | 141.0 | 142.0 | 143.0 | 144.0 | 145.0 | 146.0 | 147.0 | 148.0 |
| mae | 0.793 | 0.793 | 0.792 | 0.791 | 0.791 | 0.791 | 0.791 | 0.792 | 0.793 | 0.792 | 0.792 | 0.791 |
| rmse | 1.083 | 1.083 | 1.082 | 1.081 | 1.082 | 1.081 | 1.081 | 1.082 | 1.083 | 1.082 | 1.082 | 1.082 |

TABLE 32 – Résultats de MAE et RMSE pour l’algorithme FHyb FC basé Sem et K-NN sur le dataset RED

| | | | | | | | | | | | | |
|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| K | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| mae | 0.783 | 0.781 | 0.774 | 0.774 | 0.765 | 0.779 | 0.779 | 0.783 | 0.774 | 0.777 | 0.777 | 0.777 |
| rmse | 1.096 | 1.092 | 1.078 | 1.081 | 1.07 | 1.084 | 1.084 | 1.084 | 1.078 | 1.077 | 1.075 | 1.076 |
| K | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 |
| mae | 0.778 | 0.782 | 0.787 | 0.781 | 0.782 | 0.785 | 0.785 | 0.782 | 0.783 | 0.79 | 0.789 | 0.789 |
| rmse | 1.075 | 1.079 | 1.082 | 1.08 | 1.081 | 1.086 | 1.086 | 1.082 | 1.083 | 1.084 | 1.082 | 1.082 |
| K | 29 | 30 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 |
| mae | 0.787 | 0.785 | 0.785 | 0.784 | 0.781 | 0.779 | 0.785 | 0.785 | 0.786 | 0.787 | 0.787 | 0.786 |
| rmse | 1.081 | 1.08 | 1.08 | 1.08 | 1.079 | 1.076 | 1.082 | 1.08 | 1.08 | 1.08 | 1.079 | 1.079 |
| K | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 50 | 51 | 52 |
| mae | 0.789 | 0.787 | 0.787 | 0.789 | 0.788 | 0.79 | 0.791 | 0.794 | 0.793 | 0.793 | 0.792 | 0.79 |
| rmse | 1.083 | 1.081 | 1.081 | 1.083 | 1.084 | 1.085 | 1.086 | 1.086 | 1.085 | 1.085 | 1.085 | 1.081 |
| K | 53 | 54 | 55 | 56 | 57 | 58 | 59 | 60 | 61 | 62 | 63 | 64 |
| mae | 0.789 | 0.789 | 0.789 | 0.787 | 0.787 | 0.788 | 0.789 | 0.789 | 0.789 | 0.789 | 0.788 | 0.787 |
| rmse | 1.08 | 1.08 | 1.081 | 1.078 | 1.08 | 1.081 | 1.082 | 1.081 | 1.08 | 1.08 | 1.08 | 1.08 |
| K | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | 73 | 74 | 75 | 76 |
| mae | 0.788 | 0.789 | 0.789 | 0.79 | 0.789 | 0.79 | 0.789 | 0.789 | 0.789 | 0.789 | 0.79 | 0.79 |
| rmse | 1.077 | 1.078 | 1.079 | 1.079 | 1.077 | 1.078 | 1.078 | 1.078 | 1.078 | 1.078 | 1.08 | 1.078 |
| K | 77 | 78 | 79 | 80 | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 |
| mae | 0.791 | 0.79 | 0.789 | 0.789 | 0.789 | 0.791 | 0.791 | 0.791 | 0.791 | 0.791 | 0.792 | 0.791 |
| rmse | 1.079 | 1.078 | 1.076 | 1.077 | 1.078 | 1.08 | 1.082 | 1.079 | 1.08 | 1.079 | 1.081 | 1.08 |
| K | 89 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 | 98 | 99 | 100 |
| mae | 0.79 | 0.789 | 0.791 | 0.792 | 0.791 | 0.791 | 0.792 | 0.791 | 0.793 | 0.793 | 0.791 | 0.794 |
| rmse | 1.08 | 1.079 | 1.08 | 1.08 | 1.081 | 1.081 | 1.081 | 1.08 | 1.082 | 1.08 | 1.078 | 1.083 |
| K | 101 | 102 | 103 | 104 | 105 | 106 | 107 | 108 | 109 | 110 | 111 | 112 |
| mae | 0.791 | 0.791 | 0.791 | 0.791 | 0.791 | 0.791 | 0.792 | 0.792 | 0.792 | 0.79 | 0.791 | 0.791 |
| rmse | 1.08 | 1.08 | 1.078 | 1.079 | 1.078 | 1.078 | 1.08 | 1.079 | 1.079 | 1.078 | 1.079 | 1.079 |
| K | 113 | 114 | 115 | 116 | 117 | 118 | 119 | 120 | 121 | 122 | 123 | 124 |
| mae | 0.791 | 0.791 | 0.791 | 0.791 | 0.79 | 0.791 | 0.791 | 0.79 | 0.791 | 0.79 | 0.789 | 0.789 |
| rmse | 1.079 | 1.079 | 1.079 | 1.079 | 1.078 | 1.079 | 1.079 | 1.078 | 1.079 | 1.078 | 1.078 | 1.078 |
| K | 125 | 126 | 127 | 128 | 129 | 130 | 131 | 132 | 133 | 134 | 135 | 136 |
| mae | 0.789 | 0.789 | 0.789 | 0.789 | 0.789 | 0.788 | 0.787 | 0.787 | 0.788 | 0.788 | 0.787 | 0.787 |
| rmse | 1.078 | 1.078 | 1.078 | 1.078 | 1.078 | 1.076 | 1.077 | 1.077 | 1.077 | 1.077 | 1.077 | 1.077 |
| K | 137 | 138 | 139 | 140 | 141 | 142 | 143 | 144 | 145 | 146 | 147 | 148 |
| mae | 0.787 | 0.787 | 0.785 | 0.786 | 0.786 | 0.787 | 0.787 | 0.785 | 0.785 | 0.785 | 0.785 | 0.785 |
| rmse | 1.077 | 1.077 | 1.075 | 1.076 | 1.076 | 1.076 | 1.076 | 1.075 | 1.075 | 1.075 | 1.075 | 1.075 |

TABLE 33 – Résultats de MAE et RMSE pour l’algorithme FHyb multivues K-NN sur le dataset RED

Bibliographie

- [1] Haydar, Charif Alchiekh. Les systèmes de recommandation à base de confiance. Diss. Université de Lorraine, 2014.
- [2] Benouaret, Idir. Un système de recommandation contextuel et composite pour la visite personnalisée de sites culturels. Diss. Université de Technologie de Compiègne, 2017.
- [3] CHARU, C. AGGARWAL. Recommender Systems : The Textbook. Springer, 2018.
- [4] Balabanović, Marko, and Yoav Shoham. "Fab : content-based, collaborative recommendation." Communications of the ACM 40.3 (1997) : 66-72.
- [5] Rao, K. Nageswara. "Application domain and functional classification of recommender systems—a survey." DESIDOC Journal of Library & Information Technology 28.3 (2008) : 17-35.
- [6] Les systèmes de recommandation : une catégorisation. [en ligne], <https://interstices.info/les-systemes-de-recommandation-categorisation/> Page consultée le 11/06/2019.
- [7] Recommender systems-How they works and their impacts. [en ligne], <http://findoutyourfavorite.blogspot.com/2012/04/content-based-filtering.html> Page consultée le 11/06/2019.
- [8] Sulieman, Dalia. Towards semantic-social recommender systems. Diss. Cergy-Pontoise, 2014.

- [9] Esslimani, Ilham, Armelle Brun, and Anne Boyer. "Behavioral similarities for collaborative recommendations," 2008.
- [10] Berrut, Catherine, and Nathalie Denos. "Filtrage collaboratif." 2003.
- [11] Belloui, Amokrane. "Lusage Des Concepts Du Web Smantique Dans Le Filtrage Dinformation Collaboratif." Institut National dInformatique Alger, 2008.
- [12] Burke, Robin. "Hybrid recommender systems : Survey and experiments." User modeling and user-adapted interaction 12.4 (2002) : 331-370.
- [13] Gruber, Thomas R. "A translation approach to portable ontology specifications." Knowledge acquisition 5.2 (1993) : 199-220.
- [14] Petrakis, Euripides GM, et al. "X-similarity : Computing semantic similarity between concepts from different ontologies." Journal of Digital Information Management 4.4, 2006.
- [15] Rada, Roy, et al. "Development and application of a metric on semantic nets." IEEE transactions on systems, man, and cybernetics 19.1 (1989) : 17-30.
- [16] Resnik, Philip. "Using information content to evaluate semantic similarity in a taxonomy." arXiv preprint cmp-lg/9511007, 1995.
- [17] Seco, Nuno, Tony Veale, and Jer Hayes. "An intrinsic information content metric for semantic similarity in WordNet." Ecai. Vol. 16. 2004.
- [18] Lin, Dekang. "An information-theoretic definition of similarity." Icml. Vol. 98. No. 1998. 1998.
- [19] Jiang, Jay J., and David W. Conrath. "Semantic similarity based on corpus statistics and lexical taxonomy." arXiv preprint cmp-lg/9709008, 1997.
- [20] Bouzeghoub, Mokrane, and Dimitre Kostadinov. "Personnalisation de l'information : aperçu de l'état de l'art et définition d'un modèle flexible de profils." CORIA 5, 2005 : 201-218.
- [21] Brusilovsky, Peter. "Adaptive educational hypermedia." International PEG Conference. Vol. 10. 2001.

- [22] Les algorithmes de recommandation By Mathieu [en ligne], <https://www.podcastscience.fm/dossiers/2012/04/25/les-algorithmes-de-recommandation/>, Page consultée le 11/06/2019
- [23] Bouneffouf, Djallel. DRARS, a dynamic risk-aware recommender system. Diss. 2013.
- [24] Salton, Gerard, Anita Wong, and Chung-Shu Yang. "A vector space model for automatic indexing." *Communications of the ACM* 18.11, 1975 : 613-620.
- [25] Picot-Clément, Romain, Cécile Bothorel, and Philippe Lenca. "Towards Intention, Contextual and Social Based Recommender System." *Advanced Approaches to Intelligent Information and Database Systems*. Springer, Cham, 2014. 3-13.
- [26] Daoud, Mariam, et al. "Construction des profils utilisateurs à base d'une ontologie pour une recherche d'information personnalisée." *francophone en Recherche d'Information et Applications (CORIA 2008)*. 2008.
- [27] Zemirli, Nesrine, and Shrooq Alsenan. "PERSO-Retailer : Toward a Web Content Management System Based on a Personalized Marketing Recommender System for Retailers." *2015 International Conference on Cloud Computing (ICCC)*. IEEE, 2015.
- [28] Sieg, Ahu, Bamshad Mobasher, and Robin Burke. "Web search personalization with ontological user profiles." *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*. ACM, 2007.
- [29] Difference between Taxonomy and Ontology [en ligne], <http://www.differencebetween.info/difference-between-taxonomy-and-ontology>, Page consultée le 11/06/2019
- [30] Guo, Guibing, Jie Zhang, and Neil Yorke-Smith. "Leveraging multiviews of trust and similarity to enhance clustering-based recommender systems." *Knowledge-Based Systems* 74, 2015 : 14-27.
- [31] Introduction to Recommender System. Part 1 (Collaborative Filtering, Singular Value Decomposition) [en ligne], <https://hackernoon.com/introduction-to-recommender-system-part-1>, Page consultée le 11/06/2019.

- [32] Ujjin, Supiya, and Peter J. Bentley. "Particle swarm optimization recommender system." Proceedings of the 2003 IEEE Swarm Intelligence Symposium. SIS'03 (Cat. No. 03EX706). IEEE, 2003.
- [33] Introduction à l'algorithme K Nearst Neighbors (K-NN) [en ligne], <https://mrmint.fr/introduction-k-nearest-neighbors>, Page consultée le 11/06/2019.
- [34] Han, Jiawei, Jian Pei, and Micheline Kamber. Data mining : concepts and techniques. Elsevier, 2011.
- [35] Drias, Habiba, Souhila Sadeg, and Safa Yahi. "Cooperative bees swarm for solving the maximum weighted satisfiability problem." International Work-Conference on Artificial Neural Networks. Springer, Berlin, Heidelberg, 2005.
- [36] Bellahmer Meziane. Recommandation par filtrage collaboratif basé sur un clustering optimisé, mémoire de fin d'étude. 2018.
- [37] Shambour, Qusai, and Jie Lu. "A hybrid multi-criteria semantic-enhanced collaborative filtering approach for personalized recommendations." Proceedings of the 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology-Volume 01. IEEE Computer Society, 2011.
- [38] Wu, Zhibiao, and Martha Palmer. "Verbs semantics and lexical selection." Proceedings of the 32nd annual meeting on Association for Computational Linguistics. Association for Computational Linguistics, 1994.
- [39] Shet, K. C., and U. Dinesh Acharya. "A new similarity measure for taxonomy based on edge counting." arXiv preprint arXiv :1211.4709, 2012.
- [40] What is Python?[en ligne], <https://www.python.org/doc/essays/blurb/>, Page consultée le 11/06/2019.
- [41] What is Linux?[en ligne], <https://www.computerhope.com/jargon/l/linux.htm>, Page consultée le 11/06/2019.
- [42] Visual Studio Code[en ligne], <https://code.visualstudio.com/docs>, Page consultée le 11/06/2019.

- [43] MyMediaLite : Example Experiments[en ligne], <http://www.mymedialite.net/examples/datasets.html>, Page consultée le 11/06/2019.
- [44] LibRec Examples on Real Data Sets[en ligne], <https://librec.net/release/v1.3/example.html>, Page consultée le 11/06/2019.

Résumé

Le présent projet de fin d'études de master propose une approche de recommandation hybride, qui combine le Filtrage collaboratif (FC) et le filtrage sémantique (Fsem) selon différentes hybridations. Notre approche considère plusieurs méthodes du FC (basé utilisateur, AVG user, AVG Item, SVD) ; différentes mesures de similarité pour le Fsem (Jacquard, Wu & Palmer) ; et différentes hybridations (Filtrage pondéré, FC basé sémantique et Fsem basé collaboratif). De plus, afin d'améliorer les performances de nos algorithmes nous avons utilisé des techniques de classification, supervisé (KNN) et non supervisé (Kmedoids) avec les différents algorithmes. Pour avoir le meilleur partitionnement, nous avons intégré l'algorithme d'optimisation BSO avec l'algorithme de clustering BSO. Finalement, nous avons proposé une hybridation multi-vues basée sur KNN afin d'améliorer cette classification, en utilisant les deux vues collaborative et sémantique. Les résultats des évaluations en utilisant deux bases différentes : la base MovieLens et la base RED ont montré que les algorithmes de filtres basés sur la classification et la classification optimisée ont donné de bonnes performances sur la base RED, et ont donné des résultats très proches à ceux déjà existantes sur la base MovieLens.

Abstract

This master thesis project proposes a hybrid recommendation approach, which combines Collaborative Filtering (FC) and semantic filtering (Fsem) according to different hybridizations. Our approach considers several FC methods (user based, AVG user, AVG Item, SVD) ; different measures of similarity for the Fsem (Jacquard, Wu & Palmer) ; and different hybridizations (weighted filtering, semantic-based FC and collaborative-based Fsem). Moreover, in order to improve the performances of our algorithms we used classification, supervised (KNN) and unsupervised (Kmedoids) techniques with the previously mentioned algorithms. To have the best partitioning, we integrated the Bee Swarm Optimization (BSO) algorithm with the different implemented clustering algorithms. Finally, we proposed a KNN-based multi-view hybridization to improve this classification, using both collaborative and semantic views. The results of the evaluations using the two different databases : the MovieLens database and the RED database have shown that the classification and optimized classification filtering algorithms have given good performances on the RED database, and have given results very similar to those already existing on the database MovieLens.