République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Chadli Bendjedid
Faculté des Sciences et de la Technologie
Département d'Informatique

الجمهورية الجزائرية الديمقراطية الشعبية
وزارة التعليم العالي والبحث العلمي
جامعة الشاذلي بن جديد
كلية العلوم و التكنولوجيا
قسم الاعلام الالي

MEMOIRE

Présenté par

MR. BENDJEDID MOHAMED MEHDI

Pour l'obtention de diplôme de

MASTER

Filière : Informatique

Spécialité : Systèmes Informatiques Intelligents

Thème

SYSTEME DE RECOMMANDATION BASE SUR L'OPTIMISATION PAR ESSAIM DE PARTICULE

Soutenue le : octobre 2020 Devant le Jury composé de :

Qualité	Nom et Prénom	Grade
Président	Dr. Anguel Fouzia	MCB
Rapporteur	Dr. Gasmi Ibtissem	MCB
Examinateur	Dr. Zekri Meriem	MCB

Année Universitaire : 2019/2020

Remerciements

Ce travail est l'aboutissement d'un dur labeur et de beaucoup de sacrifices, nos remerciements vont d'abord au Créateur de l'univers qui nous a doté d'intelligence, courage et nous a maintenu en santé pour mener à bien cette année d'étude.

Je tiens aussi à adresser mes remerciements à ma famille, et plus précisément à mon oncle Brahim et ma sœur Fedia qui m'ont toujours soutenu et poussé à continuer mes études. Ce présent travail a pu voir le jour grâce à leur soutien.

J'exprime toute ma gratitude à ma directrice de mémoire madame Ibtissem Gasmi Docteur à l'université Chadli Bendjedid elTaref. Merci pour sa disponibilité indéfectible, son aide ainsi que ses remarques judicieuses qui m'ont permis de faire progresser ainsi que d'enrichir mon travail.

J'exprime ma profonde reconnaissance et mes chaleureux remerciements aux membres de jury qui m'ont fait l'honneur de bien vouloir évaluer et juger mon travail

Je souhaite faire part à toute ma famille, toutes mes chers copains ainsi que tous mes collègues.

Dédicace

Je dédie ce travail :

A mon père

Pour son grand amour, sa patience, son encouragement, son sens du devoir et ses sacrifices pour que je réussisse dans mes études.

A ma mère

Pour son affection, sa patience, son encouragement pendant les épreuves difficiles ainsi que ses prières qui m'apportent bonheur et réussite.

A ma sœur

Pour son soutien et ses encouragements.

A tous ceux qui m'aiment et à tous ceux que j'aime.

A tous ceux que je connais de près ou de loin.

Table des matières

Tables des figures

Tables des Tableaux :

Introduction Générale

Vu que les besoins des utilisateurs sont difficiles à traiter, d'une part parce qu'ils ne sont pas formulés explicitement et d'autre part parce qu'ils sont évolutifs, un grand nombre de systèmes de recommandation existe dans plusieurs domaines. En effet, aider les utilisateurs à découvrir et à choisir des ressources dans des grands espaces d'informations est un défi important qui reste toujours d'actualité. Les systèmes de recommandation constituent une solution à ce problème de surcharge d'informations. Le but principal de ces systèmes est de fournir à l'utilisateur des suggestions qui reflètent ses préférences personnelles. Ils permettent de réduire de manière considérable le temps que l'utilisateur met pour chercher les objets les plus intéressants pour lui, et aussi de trouver des objets qu'il est susceptible d'aimer. Les systèmes de recommandation ont été utilisés avec succès par des sites e-commerce comme Amazon, de streaming audio ou vidéo comme Netflix ou par des réseaux sociaux comme Facebook.

Un système de recommandation se focalise sur deux tâches. La première tâche est la prédiction : étant donné un utilisateur et un item, quelle serait la préférence de l'utilisateur pour cet item ? La deuxième tâche est la recommandation : étant donné un utilisateur, quelle liste ordonnée de n recommandations peut-on lui suggérer ? On parle alors de liste Top-n. Cette dernière n'est pas forcément la liste des n items avec les plus hautes valeurs de prédiction. La prédiction des notes n'est pas le seul critère utilisé pour produire une liste de recommandations. En effet, un algorithme de recommandation peut utiliser d'autres critères, tels que le contexte : dans plusieurs situations, l'utilité de certains items pour un utilisateur dépend de son âge ou de sa profession et parfois du temps (la saison, le soir, la nuit), etc. Une personne aime voir les films mais pas pendant la période des examens. De même, un quadragénaire peut ne pas apprécier une chanson qu'il a adorée pendant sa jeunesse. Par contre, un chercheur dans un domaine particulier sera forcément intéressé par les articles publiés dans les revues scientifiques de ce domaine.

Face à la surcharge d'information, le problème n'est plus sa disponibilité mais sa pertinence relativement à un contexte d'utilisation spécifique. C'est-à-dire comment délivrer à l'utilisateur la bonne information, au bon moment et au bon endroit. Ainsi, la recommandation traditionnelle d'items pour des usagers n'est pas suffisante, il faut donc l'adapter non seulement aux besoins et aux profils des utilisateurs mais aussi au contexte d'utilisation. L'intégration des informations contextuelles (telles que la localisation de

l'utilisateur, son âge, son sexe, la date et l'heure de l'évaluation, etc.) devient indispensable et peut jouer un rôle déterminant dans les systèmes de prédiction. C'est pourquoi des *systèmes sensibles au contexte* ont vu le jour ces dernières années.

Le travail présenté dans ce mémoire se situe dans le domaine des systèmes de recommandation, et plus précisément, les systèmes sensibles au contexte. Le modèle proposé prend en considération cinq informations contextuelles ainsi que leur degré d'importance. En effet, les paramètres contextuels ont des niveaux d'influences différents sur la pertinence de la prédiction, ce qui nécessite le calcul d'un poids représentant l'importance de chaque information contextuelle par rapport aux autres. Il est donc indispensable d'utiliser une technique bio-inspirée. La méthode d'optimisation choisie afin d'extraire automatiquement les poids à attribuer à chaque information contextuelle est l'optimisation par essaim de particules.

Ce mémoire est organisé en trois chapitres. Le premier chapitre est consacré à la présentation des concepts de base sur les systèmes de recommandation et les méthodes bio-inspirées.

Le deuxième chapitre est dédié à une étude conceptuelle détaillée du système de recommandation proposé.

Le dernier chapitre présente les outils et les langages utilisés pour le développement de notre modèle ainsi que les différents résultats obtenus.

Enfin, nous terminons ce mémoire par une conclusion générale et quelques perspectives.

Chapitre 1 : SYSTÈMES DE RECOMMANDATION ET MÉTHODES BIO-INSPIRÉES

I. INTRODUCTION

Au cours des dernières décennies, de nombreux efforts de recherches ont été consentis pour tenter d'utiliser les méthodes bio-inspirées afin de résoudre plusieurs limitations liées aux systèmes de recommandation, telles que l'amélioration de la qualité des recommandations. Les méthodes bio-inspirées s'inspirent de la nature pour résoudre des problèmes complexes dont le nombre de solutions possibles à tester est grand ou la résolution nécessite l'aspect aléatoire. La plupart des méthodes bio-inspirées se basent sur l'amélioration itérative. Elles sont adaptables à un grand nombre de problèmes sans modifications majeurs dans leurs algorithmes. Toutefois, elles n'offrent aucune garantie quant à la convergence vers l'optimum global.

II. SYSTÈME DE RECOMMANDATION

Avec la prolifération de la quantité d'information disponible sur le web, il est difficile à l'utilisateur de trouver ce qu'il cherche et ce qui peut l'intéresser. Ainsi, pour assister l'activité de recherche de l'utilisateur et l'orienter vers l'information qui lui convient, des systèmes de recommandation sont apparus [1]. Ces systèmes ont pour but d'améliorer l'interaction entre des services en ligne et des utilisateurs en suggérant des items qui correspondent aux goûts et aux attentes des usagers. Les items peuvent être des films, musiques, news, pages Web, livres, vidéos, images, etc. Les systèmes de recommandation veillent à accroître la satisfaction des utilisateurs et par conséquent leur fidélité. Ils sont particulièrement sollicités dans les applications de commerce électronique tels qu'Amazon, CNW, Netclic, eBay, etc. [2]. Ainsi, une multitude d'approches et de méthodes de recommandation qui s'appuient sur divers algorithmes ont été proposées. La plupart de ces méthodes proposent à l'utilisateur des items que les usagers les plus similaires ont préférés. D'autres lui suggèrent des ressources en lien avec celles qu'il a déjà apprécié.

II.1.Recommandation basée sur le filtrage collaboratif

Le filtrage collaboratif se base sur les appréciations données par un ensemble d'utilisateurs sur un ensemble d'items. Ces appréciations, traduites en valeurs numériques, peuvent être des notes, des comptes d'achats effectués, des nombres de visites, etc. En effet, cette méthode cherche à rapprocher l'utilisateur courant avec un ensemble d'utilisateurs existants. Ce type de recommandation est applicable à n'importe quel type de données puisque le contenu des items n'est pas considéré.

Malgré leur popularité croissante, les méthodes de filtrage collaboratif sont confrontées à de nombreuses limitations et problématiques en particulier, celles liées à la rareté des données. Si la matrice des votes est creuse, il est difficile d'identifier les voisins les plus proches et les recommandations ne seront pas pertinentes. De même, les algorithmes de filtrage collaboratif rencontrent des difficultés à sélectionner l'utilisateur qui sera intéressé par un nouvel item tant que ce dernier n'a pas été évalué par un certain nombre d'usagers. Cette limitation concerne aussi les utilisateurs ; il faut qu'ils fournissent un certain nombre de jugements pour que les recommandations obtenues soient satisfaisantes. Ce problème est connu sous le nom de démarrage à froid. Il existe deux types de Filtrage Collaboratif : Le filtrage basé mémoire et le filtrage basé modèle [3].

II.1.1. Algorithmes basés mémoire

Les algorithmes basés mémoire, appelés aussi les méthodes heuristiques, opèrent directement sur la matrice user×item pour effectuer la recommandation. La prédiction pour un utilisateur actif est estimée en utilisant ses votes et un ensemble de poids calculés à partir des jugements des utilisateurs ayant les mêmes préférences que lui. Ces techniques appliquent principalement des méthodes statistiques, telles que la méthode de K plus proches voisins, afin d'identifier des utilisateurs ayant des goûts similaires que l'utilisateur actif. Elles offrent l'avantage d'être réactifs en intégrant dynamiquement de nouveaux utilisateurs ou items. Cependant, la complexité du traitement augmente avec l'augmentation du nombre d'usagers et d'objets.

Les approches basées mémoire s'appuient sur le calcul des similarités à partir de la matrice user×item. Elles estiment les similarités entre les colonnes de la matrice des évaluations lorsqu'il s'agit d'un filtrage collaboratif basé item. Dans le cas du filtrage basé utilisateur, elles calculent les similarités entre les lignes de la matrice.

Le filtrage basé item cherche à rapprocher les items ayant été mesurés de la même façon par les mêmes utilisateurs. En effet, pour suggérer un item *i* à un utilisateur *u*, les évaluations de l'utilisateur u sur les objets similaires à l'item *i* sont utilisées. Chaque évaluation sera pondérée par un poids représentant un degré de similarité avec l'item *i* (équation I.1).

$$P_{u,i} = \overline{R}_i + \frac{\sum_{j=1}^{n} \left(R_{u,j} - \overline{R}_j\right) \times corr_{i,j}}{\sum_{j=1}^{n} corr_{i,j}} \qquad \text{(I.1)}$$

$P_{u,i}$ est la prédiction de l'utilisateur u pour l'item i.
$R_{u,j}$ est l'évaluation de l'utilisateur u sur l'item j.

$\overline{R}_i$ et $\overline{R}_j$ représentent la moyenne des évaluations de l'item *i* et *j*, respectivement.
corri,j représente le degré de similarité entre les items *i* et *j*.

Le filtrage basé utilisateur consiste à rapprocher les utilisateurs ayant les mêmes goûts. Pour prédire la pertinence d'un item *i* pour un utilisateur *u*, les évaluations des utilisateurs ayant les mêmes préférences que l'utilisateur *u* sont utilisées. Chaque évaluation sera pondérée par un poids représentant une mesure de similarité avec l'utilisateur *u* (équation I.2).

$$P_{u,i} = \overline{R}_u + \frac{\sum_{k=1}^{m} \left(R_{k,i} - \overline{R}_k\right) \times corr_{u,k}}{\sum_{k=1}^{m} corr_{u,k}} \qquad \text{(I.2)}$$

Pu,i est la prédiction de l'utilisateur u pour l'item i.
Rk,i est l'évaluation de l'utilisateur k sur l'item i.
$\overline{R}_u$ et $\overline{R}_k$ représentent la moyenne des évaluations fournies par les utilisateurs *u* et *k*, respectivement.
corru,k représente le degré de similarité entre les utilisateurs *u* et *k*.

II.1.2. Algorithmes basés modèle

L'approche basée modèle répond à la problématique de la complexité des calculs, de l'approche basée mémoire face à un nombre important d'utilisateurs et d'items. Cette méthode utilise la matrice des évaluations des utilisateurs pour élaborer un modèle qui sera utilisé pour la prédiction. Le modèle est construit en phase d'apprentissage à partir d'un corpus d'entrainement afin d'être utilisé en ligne pour calculer les recommandations. Malheureusement, cette méthode n'est pas assez dynamique et sa mise à jour est coûteuse en temps de calcul. Plusieurs techniques sont utilisées dans les systèmes de recommandation basés modèle, telles que les modèles probabilistes, les méthodes de clustering, les arbres de décision, les réseaux de neurones, la régression linéaire, etc. [4].

II.2. La recommandation basée sur le contenu (Content-based recommendation)

La recommandation basée sur le contenu peut être considérée comme un système de recherche d'information. Elle s'appuie sur un profil qui décrit les besoins de l'utilisateur du point de vue thématique, de façon analogue à une requête qui sera destinée à un système de recherche d'information. Les techniques basées sur le contenu sont fondées sur l'analyse des similarités de contenu entre les items précédemment consultés par les utilisateurs. Elles utilisent également les informations de retour fournis par l'utilisateur (feedback) pour mettre à jour son profil.

II.3. La recommandation basée sur l'utilité (Utility-based recommendation)

La recommandation basée sur l'utilité utilise une fonction d'utilité des items pour les utilisateurs afin de calculer les prédictions. Elle permet donc de suggérer à l'utilisateur actif les objets qui maximisent la fonction d'utilité. Elle modélise la préférence d'un utilisateur pour une ressource donnée par une valeur réel. En effet, la difficulté de cette méthode réside dans la définition de la fonction d'utilité [1].

II.4. La recommandation basée sur les données démographiques (Demographic recommendation)

Cette méthode se base sur les données démographiques des utilisateurs (le sexe, l'âge, le niveau social, la profession, la localisation, etc.) pour générer des recommandations [5]. Elle consiste à répartir les utilisateurs en plusieurs groupes en fonction de leurs informations démographiques. En effet, dès que l'usager commence à utiliser le système et les informations nécessaires sont obtenues, la recommandation démographique fournit des suggestions relativement satisfaisantes. Toutefois, il n'est pas toujours possible d'obtenir de telles informations à cause du respect de la vie privée des utilisateurs [6].

II.5. La recommandation basée sur la connaissance (Knowledge-based recommendation)

La recommandation basée sur la connaissance consiste à collecter des informations sur l'utilisateur pour pouvoir ensuite lui recommander des items en exploitant ces connaissances [7]. Elle permet également d'exprimer des liens entre les items. La recommandation à base des cas est un exemple de cette technique ainsi que la recommandation faite par un ami. Cette dernière se base sur des informations précises concernant l'utilisateur au lieu de ses préférences.

II.6.Système de recommandation hybride (Hybrid Recommender Systems)

Les techniques de filtrage hybride combinent deux ou plusieurs techniques de recommandation afin de surmonter leurs limitations et améliorer les performances. L'hybridation peut être effectuée de différentes manières. Nous pouvons par exemple construire un modèle général qui incorpore les caractéristiques de plusieurs techniques, ou bien intégrer certaines caractéristiques de l'une des méthodes de filtrage dans une autre méthode, ou encore implémenter séparément différentes techniques pour générer des recommandations candidates qui seront combinées par l'une des méthodes de combinaison (pondération, cascade, etc.). Burke présente sept techniques principales d'hybridation [1]:

- ***Pondération (Weighted):*** elle consiste à exécuter séparément des algorithmes de recommandation et à combiner les prédictions obtenues en utilisant une combinaison linéaire.
- ***Commutation (Switching) :*** elle consiste à déterminer dynamiquement la technique de recommandation la plus appropriée selon un critère donné. Dans ce cas, il faut définir les critères de commutation ou les cas où l'utilisation de chaque technique est recommandée.
- ***Technique mixte (Mixed):*** elle consiste à concaténer les recommandations issues des différentes techniques en prenant en considération les évaluations des utilisateurs et la description des items.
- ***Combinaison de caractéristiques (Features combination)*** : elle consiste à combiner des données provenant de différentes techniques.
- ***Cascade :*** elle consiste à instaurer une hiérarchie au sein des modèles, où chaque méthode en aval sert à raffiner la recommandation obtenue par la méthode précédente.
- ***Augmentation de caractéristiques (Feature augmentation):*** cette méthode est semblable à la cascade, elle consiste à utiliser l'une des techniques pour calculer les données qui sont ensuite utilisés par la deuxième méthode.
- ***Meta-niveau :*** la première méthode construit un modèle qui sera utilisé par la deuxième technique.

II.7.Système de recommandation sensibilité au contexte

La notion de sensibilité au contexte (Context-Aware) est apparue pour la première fois en 1994 pour désigner l'utilisation du contexte de l'application ou de l'utilisateur afin de personnaliser et d'adapter dynamiquement les comportements des applications [8]. Brown [9]considère qu'«un système sensible au contexte doit automatiquement extraire de l'information ou effectuer des actions en fonctions du contexte utilisateur détecté par les capteurs ». Selon Dey [10] la sensibilité au contexte est « la capacité d'un système à utiliser le contexte pour fournir des informations et/ou des services pertinents pour l'utilisateur, où la pertinence dépend de la tâche de l'utilisateur ». Xiaohang [11] définit une application sensible au contexte comme « un système capable d'acquérir, gérer, interpréter et répondre aux changements du contexte afin de fournir les services appropriés ». Chaari et al. [12] considèrent la sensibilité au contexte comme « la capacité de percevoir la situation de l'utilisateur en plusieurs aspects, et d'adapter en conséquence le comportement du système ».

III. CALCUL DE SIMILARITÉ

Le calcul des similarités est un élément déterminant dans le processus de prédiction. Il consiste à mesurer la similitude entre les variables descriptives des items ou des utilisateurs afin de sélectionner le meilleur voisinage sur lequel se base le calcul des prédictions.

L'identification du voisinage s'appuie généralement sur le calcul d'une distance entre deux vecteurs en utilisant la mesure la plus adaptée au type de descripteurs utilisés.

Plusieurs mesures de similarité statistiques ont été exploitées dans le cadre des systèmes de recommandation. Ces mesures se basent seulement sur les jugements communs des utilisateurs. Elles permettent de construire soit une matrice *item×item* (Tableau I.2), soit une matrice *user×user* (Tableau 1) suivant les lignes ou les colonnes. Les deux mesures les plus utilisées sont la similarité de Cosinus et la similarité de Pearson.

Tableau 1: Exemple de matrice de similarité item×item

	Item 1	***Item 2***	***Item 3***	***Item 4***	***Item 5***	***…….***
Item 1	-	*Sim(1,2)*	*Sim(1,3)*	*Sim(1,4)*	*Sim(1,5)*	…….
Item 2	*Sim(2,1)*	-	*Sim(2,3)*	*Sim(2,4)*	*Sim(2,5)*	…….
Item 3	*Sim(3,1)*	*Sim(3,2)*	-	*Sim(3,4)*	*Sim(3,5)*	…….
Item 4	*Sim(4,1)*	*Sim(4,2)*	*Sim(4,3)*	-	*Sim(4,5)*	
…….	…….	…….	…….	…….	-	…….

Tableau 2: Exemple de matrice de similarité user×user.

	User 1	***User 2***	***User 3***	***User 4***	***User 5***	***…….***
User 1	-	*Sim(1,2)*	*Sim(1,3)*	*Sim(1,4)*	*Sim(1,5)*	…….
User 2	*Sim(2,1)*	-	*Sim(2,3)*	*Sim(2,4)*	*Sim(2,5)*	…….
User 3	*Sim(3,1)*	*Sim(3,2)*	-	*Sim(3,4)*	*Sim(3,5)*	…….
User 4	*Sim(4,1)*	*Sim(4,2)*	*Sim(4,3)*	-	*Sim(4,5)*	
…….	…….	…….	…….	…….	-	…….

III.1. Distance euclidienne

La mesure de similarité euclidienne calcule la distance entre deux vecteurs X et Y de dimension n selon l'équation I.3. Plus cette distance est petite, plus les deux individus se ressemblent. Cette mesure est très sensible aux valeurs des descripteurs des vecteurs [13].

$$d(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \qquad \text{(I.3)}$$

III.2. Similarité de Cosinus

La similarité de Cosinus utilise la représentation vectorielle. Ainsi, chaque utilisateur (ou item) est représenté par un vecteur de votes. Si les deux clients (ou items) sont similaires, leurs vecteurs sont confondus ; sinon ils forment un angle dont le cosinus correspond au degré de similarité. Ce dernier est donc compris entre 0 et 1. Le cosinus de l'angle formé par les deux vecteurs i et j est calculé selon l'équation I.4. Les termes du dénominateur servent à normaliser les évaluations pour que les utilisateurs ayant évalué plusieurs items ne soient pas favorisés [14].

$$sim(i,j) = \cos(\vec{i},\vec{j}) = \frac{\vec{i}\cdot\vec{j}}{\left\|\vec{i}\right\|_2 \times \left\|\vec{j}\right\|_2} \qquad \text{(I.4)}$$

La similarité de cosinus ne prend pas en considération les différences individuelles entre les utilisateurs lorsqu'ils indiquent leurs préférences. En effet, certains utilisateurs sont très généreux dans leurs évaluations par rapport à d'autres. Ce problème est corrigé en prenant en compte les moyennes des notes de chaque utilisateur.

III.3. Coefficient de corrélation de Pearson

Le coefficient de corrélation de Pearson est une mesure de similarité célèbre en raison de sa performance dans le calcul des prédictions. Il est utilisé pour calculer les distances entre l'utilisateur (item) actif et les autres utilisateurs (items) [14]. Le coefficient de corrélation de Pearson entre les deux utilisateurs *u* et *v* (les deux items *i* et *j*) est calculé selon l'équation I.5 (l'équation I.6) :

$$sim(u,v) = \frac{\sum_{i\in I}(R_{u,i} - \overline{R}_u)(R_{v,i} - \overline{R}_v)}{\sqrt{\sum_{i\in I}\left(R_{u,i} - \overline{R}_u\right)^2}\sqrt{\sum_{i\in I}\left(R_{v,i} - \overline{R}_v\right)^2}} \qquad \text{(I.5)}$$

Ru,i et *Rv,i* représentent l'évaluation de l'utilisateur *u* et *v* respectivement sur l'item *i*.

$\overline{R}_u$ et $\overline{R}_v$ sont les moyennes des évaluations des utilisateurs *u* et *v*, respectivement.

$$sim(i,j) = corr_{i,j} = \frac{\sum_{u\in U}(R_{u,i} - \overline{R}_i)(R_{u,j} - \overline{R}_j)}{\sqrt{\sum_{u\in U}\left(R_{u,i} - \overline{R}_i\right)^2}\sqrt{\sum_{u\in U}\left(R_{u,j} - \overline{R}_j\right)^2}} \qquad \text{(I.6)}$$

Ru,j est l'évaluation de l'utilisateur u sur l'item j.
$\overline{R_i}$ et $\overline{R_j}$ représentent les moyennes des évaluations des items i et j, respectivement.

La valeur de similarité sim(u, v) (sim(i, j)) est entre -1 et 1. Si cette valeur est égale à -1 alors les deux utilisateurs (items) sont fortement opposés. Si la valeur est égale à 1, les deux utilisateurs (items) sont fortement semblables. Les deux utilisateurs (items) sont indépendants si la valeur est égale à 0.

III.4. Distance de Spearman

La distance de Spearman utilise les rangs plutôt que les évaluations. Elle consiste à trouver un coefficient de corrélation, non pas entre les votes constituant les deux vecteurs mais entre le classement des préférences. Par exemple, si un utilisateur a noté 10 items, l'objet préféré a une note de 10 et le moins préféré a une note de 1. Ainsi, la distance de Spearman consiste à calculer la distance de Pearson en utilisant ce classement. Cette mesure n'est pas très significative et ne doit être utilisée que lorsque les données ne peuvent pas être normalisées.

III.5. Coefficient de Jaccard

Le coefficient de Jaccard ne prend pas en considération les valeurs des votes. Il est généralement utilisé avec les données binaires. Il mesure la similarité entre deux vecteurs en fonction du nombre de leurs éléments communs [13]. Ce qui permet d'évaluer l'importance des appréciations partagées par rapport au nombre total d'évaluations des deux vecteurs. Ce coefficient correspond ainsi au rapport entre la cardinalité de l'intersection des descripteurs des deux vecteurs et le nombre total de leurs éléments (équation I.7). Plus les deux vecteurs ont des objets communs, plus ils sont corrélés. Les valeurs de l'indice de Jaccard sont comprises entre 0 et 1.

$$Sim_{Jaccard} = \frac{Nombre\ de\ composantes\ communes}{Nombre\ total\ de\ composantes} \qquad (I.7)$$

Huang [15] montre que le coefficient de Pearson, la similarité de cosinus et le coefficient de Jaccard donnent des résultats très proches. En outre, ils sont généralement plus performants que la distance euclidienne. Néanmoins, la distance euclidienne donne les meilleurs résultats lorsque les vecteurs sont de petites tailles [13].

III.6. Distance de Levenshtein

La distance de Levenshtein correspond au nombre d'actions nécessaires pour transformer un vecteur à un autre. Elle calcule le coût minimal pour aller d'un vecteur X à un vecteur Y en effectuant les opérations élémentaires de remplacements, de suppressions et d'insertions [13]. Si une substitution élémentaire est effectuée, son coût est égal à 1 ; sinon il est égal à 0. La mesure de Levenshtein est normalisée par le nombre de modifications dans le pire des cas (équation I.8). Une grande valeur de cette mesure signifie que la différence entre les deux vecteurs est grande.

$$Sim_{Lev}(x, y) = \frac{Nombre\ total\ de\ transformations\ pour\ aller\ de\ x\ à\ y}{Nombre\ total\ de\ transformation\ dans\ le\ pire\ des\ cas} \qquad (I.8)$$

IV. LES LIMITATIONS DES SYSTEMES DE RECOMMANDATION

Les systèmes de recommandation se basant sur les techniques précédemment expliquées ont certaines limites. Plusieurs approches ont été proposées dans la littérature pour pallier ces limites. Ci-dessous les problèmes les plus importants sont décrits.

IV.1. Démarrage à froid

Les systèmes de recommandations dépendent des évaluations des items par les utilisateurs. Ainsi, un nouvel item ne peut pas être recommandé tant qu'aucun utilisateur ne l'a évalué. Le démarrage à froid est le problème qui se produit dans les premières périodes de l'utilisation d'un système de recommandation, en raison d'un manque dans les données disponibles nécessaires pour la génération d'un processus de recommandation personnalisée de haute qualité. [16]

Le démarrage à froid pour un nouvel utilisateur est dû au fait qu'un nouvel utilisateur entre dans le système et qu'aucune donnée sur lui n'a été collectée parce qu'il n'a pas encore fourni des appréciations. Par conséquent, ses voisins ne peuvent pas être identifiés. Afin de résoudre ce problème, il faut un ensemble d'informations relatives à l'usager pour construire son profil. [17]

Le démarrage à froid pour un nouvel item dans les systèmes à base du filtrage collaboratif, les items sont représentés par les évaluations qu'ils ont eues à partir des utilisateurs du système. Donc, un nouvel item qui n'est pas connu pour les utilisateurs, et qui ne reçoit pas suffisamment d'évaluations, il ne sera pas recommandé malgré qu'il convient très bien aux préférences de certains utilisateurs. Une solution efficace pour ce problème est

de combiner le filtrage basé contenu avec le filtrage collaboratif, afin de calculer les corrélations entre items en se basant sur leur contenu. [16]

IV.2. Manque de données :

Ce problème illustre le fait que la matrice user×item possède souvent un petit nombre d'évaluations, ce qui rend le calcul des similarités difficile. Plusieurs approches ont été proposées afin d'alléger l'impact du manque des appréciations sur l'identification des voisins fiables.

Certains chercheurs proposent de remplir les évaluations manquantes de la matrice user×item par des prédictions. Ces dernières sont considérées comme des données réelles qui seront utilisées dans le calcul des similarités.

IV.3. Montée en charge

Souvent des algorithmes de recommandation qui ont une énorme base d'utilisateurs et d'items préfèrent avoir des recommandations moins précises avec un temps de calcul rapide. Plus le nombre d'utilisateurs et d'items augmente dans le système de recommandation, plus les calculs nécessaires à la recommandation deviennent très couteux. [18]

V. SYSTÈME DE RECOMMANDATION ET MÉTHODES BIO-INSPIRÉE

Depuis quelques années, le monde naturel est devenu une nouvelle source d'inspiration pour le développement des systèmes informatiques. Ainsi, la théorie de l'évolution naturelle a donné naissance aux algorithmes génétiques, le fonctionnement du cerveau humain a conduit au développement des réseaux de neurones artificiels, l'étude du comportement des animaux évoluant en essaim a permis d'inspirer les méthodes d'optimisation par essaims de particules, les modèles d'organisation naturels observés dans les sociétés des abeilles ont permis le développement des algorithme d'abeille, les phénomènes de recherche de nourriture chez les fourmis ont inspiré les algorithmes de colonies de fourmis, les mécanismes de l'immunologie ont conduit à l'élaboration des systèmes immunitaires artificiels, etc.

La plupart des méthodes bio-inspirées se basent presque sur les mêmes principes et ont les mêmes motivations. Elles nécessitent généralement la modélisation de l'espace des solutions dans lequel s'effectue la recherche de l'optimum, la définition d'une fonction

objective qui doit être minimisée ou maximisée selon le problème à traiter et la détermination des règles de déplacement pour créer de nouvelles solutions.

V.1. Optimisation par essaim de particules (Particle Swarm Optimization)

L'optimisation par essaim de particules est une méta-heuristique proposée par Kennedy et al. Comme une alternative à l'algorithme génétique [19]. Elle s'inspire des déplacements collectifs de certains animaux sociaux tels que les poissons qui se déplacent en bancs ou les oiseaux migrateurs. Cette méthode d'optimisation se base sur la collaboration d'individus peu intelligents afin de converger progressivement vers la solution optimale. En effet, chaque particule utilise sa propre expérience ainsi que l'expérience globale de l'essaim de particules pour décider de son prochain déplacement.

Figure 1: Volée d'Anser en formation en V

Un essaim de particules est une population de solutions candidates du problème à traiter. Chaque particule est caractérisée par une position dans l'espace de recherche et une vitesse de déplacement. Elle possède également une mémoire lui permettant de se souvenir de la meilleure solution qu'elle a visitée et de la meilleure solution trouvée par les autres particules.

L'algorithme d'optimisation par essaim de particules débute avec un ensemble de particules réparties aléatoirement dans l'espace de recherche et ayant des vitesses également aléatoires. A chaque itération, chaque particule ajuste sa position et sa vitesse. Elle évalue donc la qualité de sa position courante, et sauvegarde sa meilleure position et la valeur de la fonction de fitness à cette position. En outre, elle obtient la meilleure performance de toutes

les particules de l'essaim. En se basant sur ces informations, la particule modifie sa vitesse et se déplace en conséquence.

Le processus de déplacement de la particule s'appuie sur sa tendance à retourner vers sa meilleure position et sa tendance de se déplacer vers la meilleure position de son voisinage (figure 3). La nouvelle position est donc calculée en fonction de sa vitesse actuelle, sa meilleure position et la meilleure position de l'essaim [20].

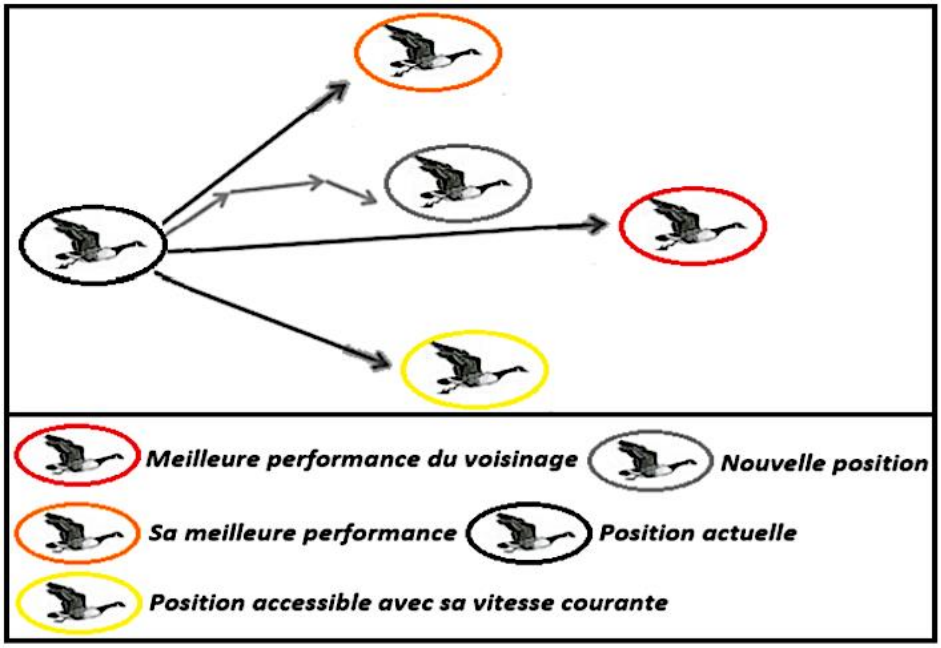


Figure 2: Déplacement d'une particule

V.1.1. Formalisation

On note *xi* le vecteur de position de la ième particule de l'essaim et *vi* son vecteur de vitesse. *xi* et *vi* sont des vecteurs à *d* composantes dont les *jème* composantes sont *xi,j* et *vi,j*, respectivement. On note également *pi* le vecteur de dimension *d* qui correspond à la meilleure position de la particule *i* ; et *pi,j* sa jème coordonnée. Enfin, on note *g* le vecteur de dimension *d* qui correspond à la meilleure position de l'essaim et *gj* sa jème coordonnée. Les déplacements des particules entre les itérations *t* et *t+1* sont calculés selon les équations 1.9 et 1.10 :

$$v_{i,j}(t+1) = w \times v_{i,j}(t) + c_1 \times r_1 \times \left(P_{i,j}(t) - x_{i,j}(t)\right) + c_2 \times r_2 \times \left(g_j(t) - x_{i,j}(t)\right) \qquad \text{(I.9)}$$

$$x_{i,j}(t+1) = x_{i,j}(t) + v_{i,j}(t+1) \qquad \text{(I.10)}$$

Avec $j = 1, ..., d$ $\quad i = 1, ..., N$
N est la taille de l'essaim.
w est le coefficient d'inertie.

c1 et *c2* sont les coefficients d'accélération.
r1 et *r2* sont des nombres aléatoires appartenant à l'intervalle (0,1).

L'algorithme d'optimisation par essaim de particules peut être résumé dans l'algorithme suivant :

Algorithme d' Optimisation par Essaim de Particules

Début

1. Initialiser aléatoirement les positions et les vitesses des particules.
2. Evaluer la position de chaque particule.
3. Pour chaque particule Faire sa meilleure position égale à sa position.
4. Calculer la meilleure position de l'essaim.
5. Tant qu'un critère d'arrêt n'est pas satisfait Faire
6. Pour chaque particule Faire
   - Mettre à jour sa position et sa vitesse.
   - Evaluer sa nouvelle position.
   - Mettre à jour sa meilleure position.
   - Mettre à jour la meilleure position de l'essaim.

Fin

L'optimisation par essaim de particules a l'avantage d'être robuste et facile à implémenter. Toutefois, elle présente certains inconvénients tels que la convergence prématurée et les problèmes du paramétrage. En effet, cette méthode nécessite une instanciation des différents paramètres qui sont souvent spécifiques à chaque problème. De plus, elle requiert souvent un temps de calcul important pour converger vers la solution optimale.

V.1.2. Application de l'optimisation par essaim de particules dans les systèmes de recommandation

Dans le cadre des systèmes de recommandation, plusieurs algorithmes à base d'essaim de particules sont développés afin d'attribuer des poids aux différents paramètres ou bien pour réaliser des classifications non-supervisées dans le but de trouver un bon voisinage. Ujjin et Bentley [21] réalisent un système de recommandation hybride qui permet d'apprendre les préférences des utilisateurs. L'algorithme proposé utilise une optimisation par essaim de particules afin de calculer les poids à attribuer aux différents attributs constituant le profil

utilisateur. Il permet ainsi de trouver le groupe des utilisateurs ayant des profils similaires à celui de l'utilisateur actif.

Dans le but de traiter le problème de manque de données et d'évolutivité, Bakshi et al. [22] conçoivent un système de recommandation à base d'un algorithme d'optimisation par essaim de particules afin de calculer, pour chaque prédiction, les poids du paramètre combinant la similarité locale et globale. De leur côté, Diaz-Aviles et al. [23] réalisent un système de recommandation qui s'appuie sur une optimisation par essaim de particules afin d'optimiser la combinaison des variables latentes extraites à partir de la factorisation de la matrice des votes. Le système proposé apprend une fonction linéaire permettant de classer les documents suivant l'intérêt de l'utilisateur.

D'autre part, Zhang et Fei [24] développent un algorithme de recommandation qui utilise un modèle d'optimisation multi-objective par essaim de particules basé sur *Pareto dominance* et la classification floue. L'algorithme proposé permet de résoudre le problème de la convergence prématuré des particules vers un optimum local. Au départ un essaim de particules est initialisé aléatoirement. A chaque itération, les performances des particules sont calculées. Si le fitness d'une particule ne change pas après un certain nombre d'itérations, la particule est retirée de l'essaim et un nouveau sous-essaim est créé. En outre, la classification floue est utilisée afin de créer dynamiquement les différentes classes de l'essaim. Ainsi, chaque classe de particules a son propre ensemble de leaders et évolue en utilisant l'optimisation par essaim de particules et le concept de dominance de Pareto. Le concept de migration est intégré afin de maintenir la diversité des sous-essaims et d'améliorer en conséquence la qualité des solutions trouvées.

Alam et al. [25] réalisent deux systèmes de recommandation qui s'appuient sur une classification hiérarchique intelligente à base d'optimisation par essaim de particules. Les algorithmes présentés regroupent les sessions des utilisateurs dans différents clusters, ce qui permet de réduire l'espace de recherche en le divisant en plusieurs sous-espaces. Les recommandations sont ensuite générées à l'utilisateur actif à partir de ces groupes. Dans[26], les auteurs utilisent des données implicites qui doivent être traitées et regroupées afin de fournir des recommandations en temps réel.

Abdelwahab et Nishida [27] conçoivent un algorithme de filtrage collaboratif hybride afin de traiter le problème de manque de données et d'évolutivité. Le système proposé s'appuie sur une indexation sémantique latente qui se base sur la décomposition en valeurs singulières. Les valeurs singulières optimales de la matrice sont identifiées en utilisant une

optimisation par essaim de particule. Dans un travail ultérieur, Abdelwahab et al. [28] développent un algorithme de filtrage collaboratif qui se base sur une optimisation par essaim de particules afin de regrouper les utilisateurs et les items dans des cluster en se basant non seulement sur les évaluations mais aussi sur les poids exprimant l'importance des utilisateurs et des items dans le processus de recommandation. D'autre part, les poids utilisés dans le calcul des similarités qui combine la similarité basée item et celle basée utilisateur sont également estimés avec un algorithme d'optimisation par essaim de particules.

Anandakumar et al. [29] proposent un système de recommandation qui traite le problème de manque de données et du démarrage à froid en combinant un algorithme génétique avec une méthode d'optimisation par essaim de particules. Les résultats obtenus montrent que le modèle proposé améliore la précision et diminue le taux d'erreur.

V.2. Les Algorithmes Génétiques (Genetic Algorithm)

L'algorithme génétique est une méthode stochastique fondée sur des mécanismes de la sélection naturelle et de la génétique. Il a été initialement développé par Holland et popularisé par Goldberg [30]. Cette méthode s'inspire de l'évolution des êtres vivants, élaborée par Charles Darwin, et la transpose à la recherche de solutions adaptées à un problème donné [31]. L'algorithme génétique fait partie des méthodes évolutionnaires permettant de résoudre, dans un temps raisonnable, des problèmes d'optimisation pour lesquels il n'existe pas de méthodes traditionnelles plus efficaces. C'est une technique robuste d'exploration d'espaces de recherche complexes et de taille importante. Il fait évoluer une population de solutions candidates à un problème donné dans le but de trouver une solution optimale. Ainsi, il nécessite seulement une fonction de codage et une fonction d'évaluation de la pertinence de chaque solution.

V.2.1. Application des algorithmes génétiques dans les systèmes de recommandation

Les systèmes de recommandation consistent à suggérer à l'utilisateur des informations correspondant à son profil ou en se servant de l'expérience des autres usagers. Beaucoup de recherches ont étudié la contribution des algorithmes génétiques à résoudre des problématiques liées à la sélection d'informations pertinentes dans les systèmes de recommandation et dans les systèmes de recherche d'information, de façon générale. En effet, les algorithmes génétiques se basent sur des mécanismes d'adaptation et d'évolution naturels. Ils ont la capacité d'explorer l'espace de recherche pour une éventuelle

recommandation en s'adaptant au profil et aux besoins des utilisateurs qui peuvent être imprécis. Cette capacité à explorer des espaces de recherche variés et complexes leur permet de :

- Fournir une représentation optimale des documents en adaptant leurs descripteurs aux profils des utilisateurs.
- Fournir une représentation optimale des requêtes en appliquant des techniques de reformulation de requêtes.
- Modéliser un processus de recherche adaptatif afin de fournir des items pertinents.
- Modéliser le profil de l'utilisateur.

Plusieurs chercheurs utilisent les algorithmes génétiques pour la classification non-supervisée des utilisateurs ou des items. L'idée de base consiste à regrouper dans le même cluster les utilisateurs ayant les mêmes goûts ou bien, les items qui portent sur les mêmes thèmes ou qui plaisent les mêmes personnes. En d'autres termes, nous associons une ou plusieurs classes à chaque utilisateur ou à chaque item. Ainsi, pour prédire la note qu'un usager donnera à un objet, l'algorithme utilise les avis des utilisateurs appartenant à la même classe. Le travail proposé par Hwang et Chang [32] effectue une classification non-supervisée des utilisateurs dans un système de recommandation à l'aide d'un algorithme génétique hybride afin de résoudre le problème d'évolutivité. L'auteur remplace l'opérateur de croisement par un algorithme de K-moyennes pour accélérer la convergence de l'algorithme.

Kanchana et Sujatha [33] développent un système de recommandation qui se base sur un algorithme de parcours en profondeur et un algorithme génétique afin de créer des profils de navigation et d'affecter l'utilisateur actif à l'un de ces profils. Le processus de recommandation consiste donc à suggérer les navigations les plus susceptibles d'être effectuées par l'utilisateur.

Afin d'améliorer la qualité des prédictions, Ujjin et Bentley [34], Fong et al. [35] et Bobadilla et al. [36] présentent des systèmes de filtrage collaboratif qui utilisent un algorithme génétique pour déterminer les poids des informations utiles dans le calcul des similarités. Dans [35], le profil de l'utilisateur pour chaque film est représenté par un chromosome de 12 gènes. Il contient le vote de l'item, les informations démographiques de l'utilisateur (sexe, âge, profession) ainsi que des informations sur ses préférences (film préféré, acteur préféré, langue préférée, etc.). Après avoir sélectionné le voisinage de

l'utilisateur actif, la prédiction est obtenue en utilisant la distance euclidienne et les poids calculés via l'algorithme génétique. Le système conçu par Bobadilla et al. [36] utilise un codage binaire ce qui nécessite une opération de conversion du binaire au réel et vice versa.

Navgaran et al. [37] utilisent une méthode de factorisation de la matrice des votes qui s'appuie sur un algorithme génétique afin de traiter l'augmentation du temps de calcul dans les systèmes à base de filtrage collaboratif. Chaque chromosome est représenté par une matrice dont le nombre de lignes égale au nombre de valeur latente et le nombre de colonnes égale au nombre d'utilisateurs plus le nombre d'items.

Al-Shamri et Bhardwaj [30] réalisent un système de recommandation basé sur une approche génétique floue afin de traiter le problème de manque de données et de l'évolutivité. La similitude entre les préférences des utilisateurs est calculée en utilisant une fonction de distance floue. Ainsi, les poids à attribuer aux différentes caractéristiques décrivant le profil utilisateur sont calculés en utilisant un algorithme génétique.

Dao et al. [31] proposent un système de recommandation sensible au contexte à base d'algorithme génétique. Le modèle proposé fournit des publicités de localisation en fonction du contexte et des préférences indiquées par les utilisateurs. Le mécanisme de filtrage collaboratif utilise la notion de similitude entre les contextes. Ainsi, les valeurs de similarité optimales entre les différents contextes sont calculées en utilisant un algorithme génétique.

V.3. Algorithme de colonie de fourmis (Ant System)

L'algorithme de colonies de fourmis a été proposé pour la première fois pour résoudre le problème du voyageur de commerce [38]. Il s'appuie sur le comportement collectif des fourmis lors de la recherche de nourriture appelée aussi fourragement. Au départ, les fourmis se déplacent aléatoirement de leur fourmilière à la recherche d'une source de nourriture. Elles laissent sur leurs chemins une substance chimique volatile appelée phéromone afin d'attirer l'attention de leurs congénères. En effet, les fourmis qui prennent le trajet de plus courte distance arrivent plus rapidement à la fourmilière. Par conséquent, elles déposent une quantité importante de phéromones sur ce chemin. Cela augmente l'importance du trajet et la probabilité qu'il soit choisi par d'autres fourmis de la colonie [39]. Ainsi, grâce à l'évaporation naturelle de la phéromone, les chemins les plus longs disparaissent progressivement. Ceci amène toutes les fourmis à suivre le chemin le plus court. Ce type de communication indirecte permet le partage d'information entre les fourmis. Il constitue donc la mémoire collective de la colonie.

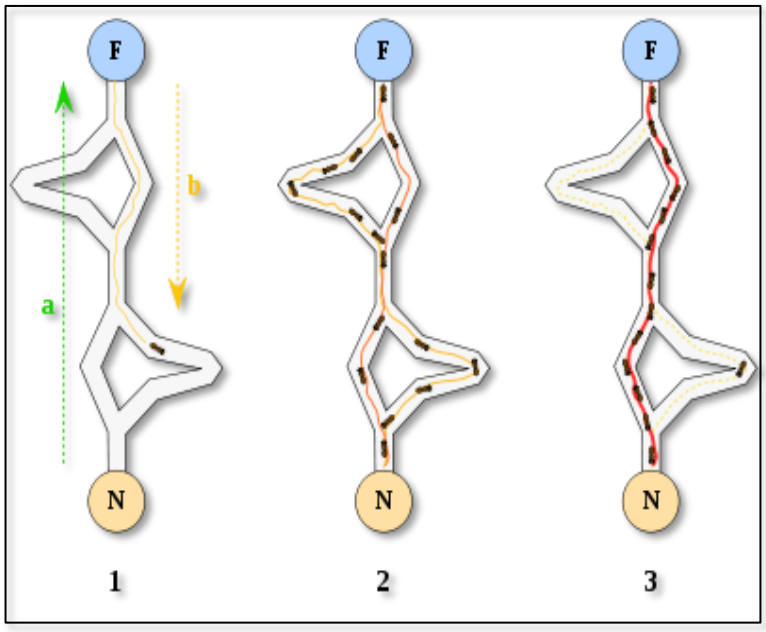


Figure 3: colonie des fourmis

Les problèmes à base de colonie de fourmis sont généralement modélisés par un graphe dont chaque arête représente un chemin et son poids correspond à la quantité de phéromone déposée sur ce chemin [40]. L'algorithme de colonie de fourmis fait évoluer une population d'agents selon un modèle stochastique. Chaque fourmi artificielle représente une solution au problème traité. Elle modélise certaines caractéristiques de la fourmi réelle et peut être enrichie par des comportements que ne possèdent pas les fourmis réelles pour la rendre plus efficace dans la résolution des problèmes. Une phéromone est une valeur associée à une solution trouvée. Cette valeur est proportionnelle à la qualité de la solution. Contrairement aux algorithmes génétiques, la recherche d'une solution optimale à base d'algorithme de colonie de fourmis s'effectue à partir de la totalité de la population et non pas à travers quelques meilleurs individus. Ainsi, les fourmis coopèrent ensemble pour construire progressivement des solutions. A chaque itération, chaque fourmi choisit de façon probabiliste un sommet parmi les nœuds adjacents en se basant sur la mémoire collective de la colonie. Afin de ne pas revenir sur ses pas, la fourmi tient à jour une liste *Tabou* qui mémorise les sites qu'elle a déjà visités. La quantité de phéromone d'une piste est modifiée après le passage d'une fourmi. Les différentes étapes de l'algorithme de colonie de fourmis sont résumées dans l'algorithme suivant :

Algorithme de colonie de fourmis artificielles

Début

1. Créer aléatoirement une population de *m* fourmis.
2. Evaluer les *m* fourmis.
3. **Tant qu'**un critère d'arrêt n'est pas satisfait **faire**
    1. **Pour** chaque fourmi **faire**
    - Construire le trajet de la fourmi.
    - Déposer des phéromones sur le trajet de la fourmi.
    2. Evaluer les *m* fourmis.
    3. Evaporer les pistes de phéromones.
4. Retourner la ou les meilleures solutions.

Fin

L'algorithme de colonie de fourmis a subi de nombreuses modifications afin d'améliorer ses performances ou de l'adapter à des problèmes particuliers. Ant System & élitisme est l'une des premières variantes. Dans cet algorithme, la fourmi qui a effectué le trajet le plus court, appelée fourmi élitiste, dépose une quantité de phéromone plus grande sur les arcs appartenant au meilleur chemin afin d'accroître la probabilité des autres fourmis d'explorer la solution la plus prometteuse. L'algorithme Ant Colony System est une autre variante dont les déplacements des fourmis se basent sur une règle proportionnelle pseudo-aléatoire. Ainsi, en plus de la mise à jour de la quantité de phéromone à la fin de chaque cycle d'une fourmi, la phéromone sur les arcs constituant le meilleur chemin est renforcée par la meilleure fourmi à la fin de la construction d'une solution. L'algorithme Max-Min a introduit d'autres particularités. En effet, les quantités des phéromones sont bornées par une valeur inférieure et supérieure. En outre, la mise à jour de la quantité de phéromone est inversement proportionnelle à sa valeur et n'est autorisée que par la fourmi ayant trouvé la meilleure solution.

Les colonies de fourmis artificielles sont des méthodes robustes qui s'adaptent rapidement à des applications dynamiques. Néanmoins, elles sont capables de résoudre uniquement les problèmes à base de graphes. Ils peuvent également se bloquer dans un minimum local. Ainsi, le temps d'exécution est parfois long surtout lorsqu'un nœud possède trop d'enfants.

V.3.1. Application de l'algorithme de colonie de fourmis dans les systèmes de recommandation

Dans le cadre des systèmes de recommandation, les algorithmes de colonie de fourmis artificielles sont utilisés pour des tâches de classification et d'optimisation. Salehi et al. développent un système de recommandation à base de colonie de fourmis artificielles pour le commerce électronique. Dans le système proposé, les produits disponibles sont représentés sous forme d'un graphe. Chaque nœud correspond à un produit à recommander et comporte des informations sur ce produit. Une arête existe entre deux nœuds si leur similarité dépasse un certain seuil ou s'il existe une relation entre les deux produits telle que *IS-A, Part_of*, etc. Chaque arête est étiquetée par le degré de similarité entre les deux nœuds. Lorsqu'un client entre dans le système, une fourmi contenant son historique d'achat est créée automatiquement. Quand le client sélectionne un produit, son agent s'installe sur le nœud correspondant à cet item et se déplace dans le graphe à l'aide de l'historique de cet utilisateur pour lui suggérer les produits qui l'intéressent.

Caicedo-Castro et Duarte-Amaya [41] conçoivent un système de recommandation de composants logiciels réutilisables à base d'algorithme de colonie de fourmis pour aider les ingénieurs et les programmeurs à récupérer des extraits de code source utiles (potentiellement écrits dans n'importe quel langage de programmation) pour la mise en œuvre de nouveaux logiciels. La méthode proposée traite le problème de démarrage à froid et celui de la rareté des données. Elle s'appuie sur la répétition et la similitude entre les requêtes effectuées par les programmateurs. En effet, les requêtes précédemment effectuées sont représentées par un graphe orienté dont les poids des arcs sont calculés en fonction de la similarité textuelle, la corrélation et la confiance entre les sommets. Des extraits de code source sont suggérés aux programmeurs en fonction des déplacements des fourmis dans le graphe.

Les colonies de fourmis sont également utilisées pour la recherche des règles. Paranjape-Voditel et Thakare [42] proposent un système de recommandation basé sur les règles d'association. En effet, les auteurs expriment l'extraction des règles d'association comme une recherche de plus court chemin. Ils utilisent donc un algorithme de colonies de fourmis pour extraire ces règles.

Afin de traiter le problème de manque de données, Nadi et al. [43] réalisent un système de recommandation qui combine le filtrage collaboratif et le filtrage basé sur le contenu. Le système proposé intègre la logique floue dans l'algorithme de colonie de fourmis afin de

fournir des recommandations qui prennent en considération l'incertitude dans les préférences des utilisateurs.

Bedi et al. [40], Bellaachia et Alathel [39] et Kaleroun et Batra [44] proposent des systèmes de recommandation à base de confiance afin de traiter le problème de manque de données. En effet, un graphe de confiance pour chaque utilisateur est créé à partir de la matrice des évaluations. Dans le processus de recommandation, le meilleur voisinage de l'utilisateur actif est sélectionné en utilisant une optimisation à base de colonie de fourmis.

Bedi et al.[40] développent un système de recommandation qui se base sur une optimisation par colonie de fourmis afin de regrouper les items dans des clusters. Ainsi, les voisins les plus proches peuvent être choisis parmi les groupes les plus similaires.

VI. CONCLUSION

Dans ce chapitre, nous avons présenté les systèmes de recommandation, les principales techniques de recommandation et leurs limitations. Nous avons également exposé des méthodes bio-inspirées et leurs applications dans les systèmes de recommandation ainsi que les différents travaux de recherche dans ce domaine. Le chapitre suivant présente notre système de recommandation à base d'optimisation par essaim de particules. Le modèle proposé permet l'intégration des informations contextuelles sur les utilisateurs et les items afin d'améliorer la qualité des prédictions.

Chapitre 2 : CONCEPTION DU SYSTEME PROPOSE

I. MOTIVATION ET OBJECTIF :

Les préférences des utilisateurs évoluent selon le contexte il est donc nécessaire de traiter ces changements afin d'adapter les réponses des systèmes de recommandation aux préférences et au contexte des usagers. La dimension temporelle est une information contextuelle qui peut avoir une grande influence sur l'activité de recherche de l'utilisateur. Ainsi dans plusieurs situations, l'utilité de certains items pour un usager dépend étroitement de sa profession et parfois des personnes avec qui l'objet sera consommé. Par exemple, la recommandation d'un film diffère selon que la personne projette de le voir avec ses amis, ses parents ou ses enfants. En outre, la recommandation des livres est fortement dépendante de la profession et des passions de l'utilisateur. De plus, la suggestion des habits est fortement liée à la saison, au sexe du client et même à son âge. D'autre part, le degré d'influence de l'information contextuelle sur les préférences des utilisateurs diffère d'un paramètre contextuel à un autre.

Selon Adomavicius et Tuzhilin malgré un nombre considérable de recherches faites sur les SR, la plupart des approches se focalisent sur la recommandation des items les plus pertinents pour les utilisateurs sans prendre en compte les informations contextuelles (exemple : le temps, la localisation ou la compagnie d'autres personnes). Ils ont montré que les informations contextuelles pertinentes ont des influences importantes sur un SR. Il est donc important d'étudier les SR sensibles aux contextes. [45]

La notion de contexte est intéressante pour les SR. Selon Dey un contexte est n'importe quelle information qui peut caractériser la situation d'une entité (personne, localisation, produit, etc.). [46]

D'autres chercheurs définissent le contexte comme l'identité de l'utilisateur, ressources de l'environnement proche, localisation de l'utilisateur et période temporelle d'exécution de l'interaction. Selon Berry et Linoof, les contextes sont définis comme des évènements qui caractérisent les phases de la vie d'un client et qui peuvent influencer ses préférences, son statut et sa valeur pour une entreprise. Des études comportementales en marketing ont montré que la prise de décision des clients dépend des contextes dans lesquels ils se trouvent [47].En effet, selon les contextes comme la localisation, les saisons, l'humeur, etc. le même client peut choisir différents produits. Plusieurs recherches ont été menées dans différents

domaines pour évaluer l'impact des contextes dans les SR. Ces recherches ont été faites sur les contextes observables (localisation, compagnie, période, etc.) (et les contextes non observables (identité d'un membre d'une famille) [47]

Adomavicius et Tuzhilin ont présenté un SR avec une méthode multidimensionnelle. Des contextes sont ajoutés à la fonction d'évaluation de dimension deux (R : User x Item→ Rating). Ainsi on obtient une fonction multidimensionnelle (R : User x Item x Contexte →Rating) qui inclut les informations contextuelles dans la prédiction des préférences des utilisateurs. Pour implémenter la méthode multidimensionnelle et tester sa performance, des données sur des films (notes) et des données contextuelles (localisation, période, compagnie) ont été collectées.

Le système décrit dans ce chapitre est basé sur une approche de filtrage collaboratif, construisant des profils d'utilisateurs, puis utilisant un algorithme pour trouver des profils similaires à l'utilisateur actuel. Les données sélectionnées de ces profils sont ensuite utilisées pour élaborer des recommandations. Parce que les profils contiennent de nombreux attributs, dont beaucoup ont des données rares ou incomplètes, la tâche de trouver des similitudes appropriées est souvent difficile.

Pour surmonter ces problèmes, les systèmes actuels (tels que MovieLens) utilisent des modèles stochastiques et heuristiques pour accélérer et améliorer la qualité de la correspondance des profils. Ces travaux poussent ces idées un peu plus loin, en appliquant un algorithme d'optimisation de l'essaim de particules au problème de l'adéquation des profils.

Ces données contextuelles ne sont pas disponibles sur la collection de référence MovieLens (movielens.umn.edu) généralement utilisé pour évaluer les SR, ni sur les autres données publiques. Par conséquent, un site internet spécifique a été créé et il a été demandé à des utilisateurs d'évaluer les films qu'ils ont vus ainsi que les informations contextuelles pertinentes. Les résultats montrent empiriquement une amélioration de la prédiction des films des systèmes sensibles aux contextes par rapport aux systèmes qui ne les incluent pas. **[**48]

II. Architecture générale du système proposé :

L'architecture générale du système de recommandation proposé est illustrée par la figure 4 :

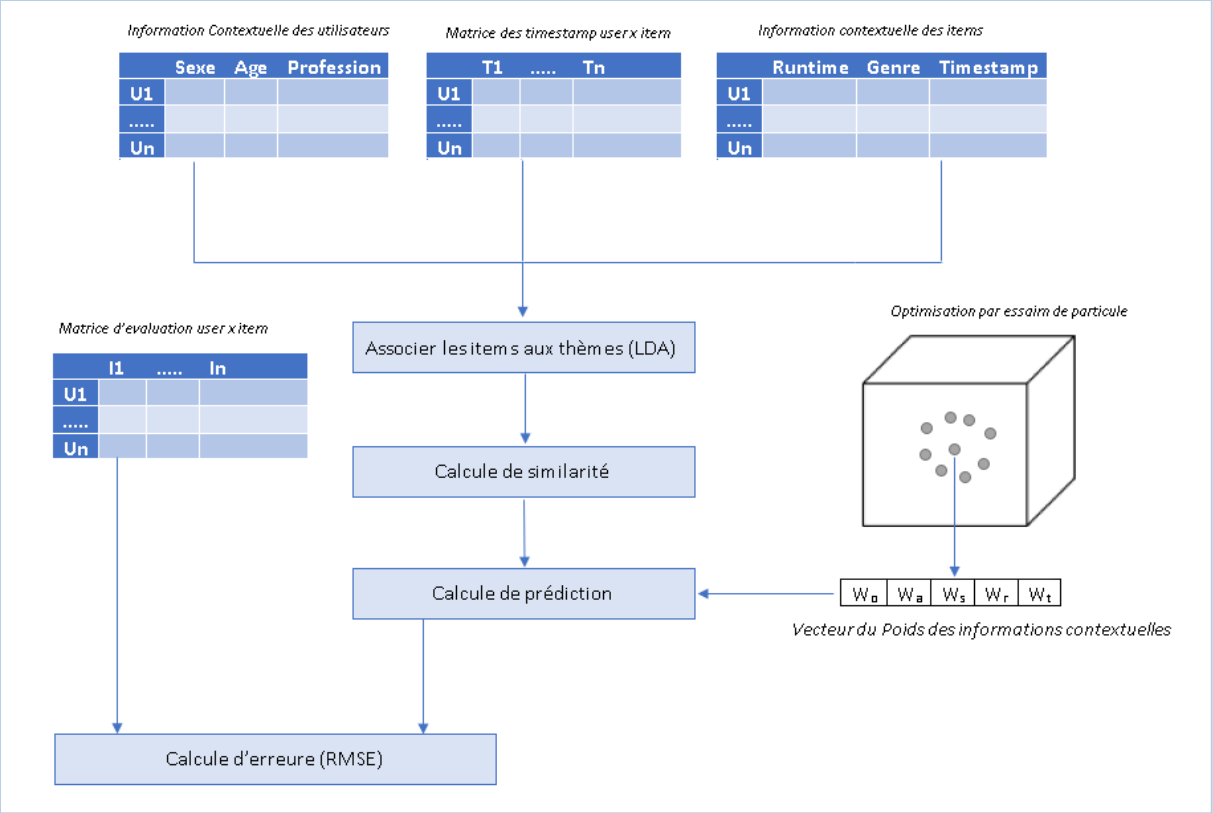


Figure 4: Architecture générale du système proposé

III. Algorithmes utilisés :

Dans cette section nous présentons les différents algorithmes que nous avons utilisé dans le cadre de ce travail.

III.1. Filtrage collaboratif traditionnel :

Le filtrage collaboratif basé item considère la totalité des évaluations disponibles lors du calcul des recommandations. La prédiction de l'évaluation de l'utilisateur *u* pour un film *j* est fondée sur les estimations des items similaires de *i*. Ainsi, Chaque item possède sa propre distribution à partir de l'estimation obtenue. Pour mesurer la similarité entre deux items, différentes mesures peuvent être utilisées. Nous avons choisi d'utiliser le coefficient de corrélation de Pearson :

$$sim(u,v) = \frac{\sum_{i \in I}(R_{u,i} - \overline{R}_u)(R_{v,i} - \overline{R}_v)}{\sqrt{\sum_{i \in I}\left(R_{u,i} - \overline{R}_u\right)^2}\sqrt{\sum_{i \in I}\left(R_{v,i} - \overline{R}_v\right)^2}} \quad \text{(II.1)}$$

Ru,i et *Rv,i* représentent l'évaluation de l'utilisateur *u* et *v* respectivement sur l'item *i*.

$\overline{R}_u$ et $\overline{R}_v$ sont les moyennes des évaluations des utilisateurs *u* et *v*, respectivement.

$$sim(i,j) = corr_{i,j} = \frac{\sum_{u \in U}(R_{u,i} - \overline{R}_i)(R_{u,j} - \overline{R}_j)}{\sqrt{\sum_{u \in U}\left(R_{u,i} - \overline{R}_i\right)^2}\sqrt{\sum_{u \in U}\left(R_{u,j} - \overline{R}_j\right)^2}} \quad \text{(II.2)}$$

Ru,j est l'évaluation de l'utilisateur u sur l'item j.

$\overline{R}_i$ et $\overline{R}_j$ représentent les moyennes des évaluations des items *i* et *j*, respectivement.

Supposons que nous ayons l'historique des préférences des utilisateurs vus comme une matrice M, qui est la matrice d'évaluation employée dans le filtrage collaboratif. Nous regardons ensuite l'ensemble des items que l'utilisateur courant a déjà consulté et déterminons la similarité avec les autres items que l'utilisateur courant n'a pas encore vus en utilisant la matrice de similarité de l'étape précédente.

En effectuant cela, les notes des items pour l'utilisateur courant peuvent être obtenue et servira à indiquer le degré de préférence de l'utilisateur courant pour les nouveaux items non consultés. La prédiction des préférences P (u, i) pour un item i, pour l'utilisateur u, est calculée par cette formule :

$$Pu,i = Ri + \frac{\sum_{j=1}^{n}(Ru,j - Rj) \times sim(i,j)}{\sum_{j=1}^{n} sim(i,j)} \quad \text{(II.3)}$$

$\overline{R}_i$: la moyenne des évaluations de l'item i
$\overline{R}_j$: la moyenne des évaluations de l'item j
$\overline{R}_{u,j}$:la note donnée par l'utilisateur « u » à l'item « j »
Sim(i,j) : le degré de similarité entre l'item i et j

III.2. LDA (Allocation de Dirichlet latente) :

L'allocation de Dirichlet latente est un modèle thématique génératif probabiliste **.**Il se base sur l'intuition que les documents sont composés de plusieurs thèmes (et non pas de mots), où un thème est une distribution multinomiale sur un vocabulaire fixé W.

Le but de LDA est ainsi de découvrir les thèmes présents au sein d'une collection de documents. Les documents de la collection sont modélisés comme des ensembles de K thèmes qui sont eux-mêmes des distributions multinomiales sur W. La distribution thématique φk d'un thème k est générée par une loi de Dirichlet avec un paramètre β, tandis que la distribution θd d'un document d est générée par une loi de Dirichlet avec un paramètre α. [49]

III.3. Filtrage collaboratif basé sur LDA :

LDA a été appliqué dans l'analyse de documents **,** la catégorisation, le regroupement de documents [50]et l'ordonnancement de systèmes de recherche d'information [51]. LDA a été introduit dans les SR afin d'analyser le contexte dans les méthodes basées sur le contenu [52]. Dans ce travail, nous proposons une autre utilisation du modèle LDA. Le résultat du modèle LDA est intégré dans un système de filtrage collaboratif pour trouver la similarité entre les items consultés par un utilisateur courant.

Nous présentons chaque item par un ensemble de thèmes. Notre approche recommande alors des items pour lesquels les distributions de thèmes sont similaires à l'item courant.

Ainsi, cet article propose un SR sensible aux contextes, il s'agit d'un modèle de recommandation hybride qui combine la méthode de modélisation de thèmes basée sur LDA et la méthode de filtrage collaboratif.

III.3.1. Modèle LDA pour la représentation de thèmes :

La première étape implémente le modèle LDA à partir des descriptions des items que les utilisateurs ont consultés. LDA est utilisé pour extraire la structure sémantique cachée dans les descriptions des items que les utilisateurs ont consultés, la distribution des mots sur les thèmes latents et le mélange des distributions des thèmes latents. Cela consiste à estimer la distribution de thèmes latents pour chaque item et la distribution de mots pour chaque thème. Ces distributions vont permettre d'identifier la sémantique de l'espace de thèmes latents en les rapportant aux mots et aux items. Pour améliorer les résultats de LDA, il faut faire un apprentissage basé sur un filtrage collaboratif pour régler les paramètres de LDA :

- ✓ L : nombres de thèmes.
- ✓ Eta : la distribution des mots par thème
- ✓ Alpha : la distribution des thèmes par item

III.4. Nouvel algorithme de filtrage sensible au contexte :

Dans le cadre de ce travail, un algorithme de filtrage collaboratif basé item est proposé afin de prendre en considération l'évolution des préférences des utilisateurs selon le contexte. Cet algorithme utilise une fonction de pondération qui intègre cinq paramètres qui seront utilisés sous forme de vecteur pour décrire le contexte : l'âge de l'utilisateur, son sexe et sa profession, le runtime de l'item et le temps d'évaluation.

- Le sexe est représenté par deux valeurs *M* pour male et *F* pour femelle.
- La profession est un attribut représenté par les valeurs suivantes : "*administrator*», «*artist*", " *doctor*", "*educator*", "*engineer*", "*healthcare*", "*homemaker*", "*entertainment*", "*executive*", "*lawyer*", "*librarian*", "*marketing*", "*none*", "*other*", "*programmer*", "*retired*", "*salesman*", "*scientist*", "*student*", "*technician*" , "*writer*". Dans le cadre de ce mémoire la profession est représentée respectivement par des valeurs entre 1 et 21.
- Le runtime des items est représenté par des valeurs entre 0 et 390.
- Le temps correspond au temps d'évaluation de l'item.

D'autre part, il est évident que des caractéristiques différentes ont des niveaux d'influences différents sur les préférences de l'utilisateur. On qualifie l'influence d'une information contextuelle par la notion de "*poids*". Le poids d'un paramètre contextuel représente le degré d'importance de cette information dans le calcul des prédictions. Le problème consiste donc à déterminer les degrés d'importance des cinq éléments contextuels.

Il existe une multitude de méthodes qui peuvent être appliquées pour déterminer les poids des caractéristiques. Dans le cadre de cette thèse, un algorithme d'essaim de particule afin de calculer automatiquement le poids à attribuer à chaque information contextuelle. En effet, la sortie de ce module est un vecteur *W* de poids qui représente l'influence de chaque information contextuelle sur le calcul des prédictions. Le calcul de ces poids se déroule selon les étapes suivantes :

III.4.1. Calcul de poids de chaque paramètre contextuel :

Afin de calculer le poids de chaque paramètre contextuel, le système proposé utilise un algorithme d'optimisation par essaim de particules.

III.4.1.1.1. Génération de la population initiale :

La population initiale est constituée de *N* particules de type réel générés aléatoirement. Chaque particule contient cinq valeurs représentant les poids des cinq informations contextuelles. Ces valeurs appartiennent à l'intervalle (0,1) et leur somme est égale à 1, ($\sum_{i=1}^{5} W_i = 1$). Chaque particule peut être représenté par la figure 5.

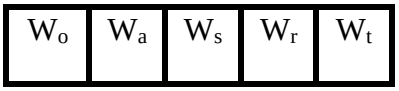


Figure 5: Schéma représentant la structure d'une particule

Wo représente le poids attribué à la profession (occupation) de l'utilisateur.
Wa représente le poids attribué à l'âge de l'utilisateur.
Ws représente le poids attribué au sexe de l'utilisateur.
Wr représente le poids attribué à la durée de l'item.
Wt représente le poids attribué au temps de l'évaluation.

a) Traitement de données utilisées par l'algorithme PSO :

L'objectif de cette étape est de normaliser les valeurs des paramètres contextuelles c'est-à-dire les rendre entre 0 et 1 d'où on a utilisé les fonctions ci-dessous :

i. Normaliser l'âge :

$$f(age) = \frac{age - ageMin}{ageMAx - AgeMin} \quad \text{(II.4)}$$

ii. Normaliser l'occupation :

D'abord, dans la base de données on a 21 occupations, on affecte un numéro entre (1-21) à chaque occupation pour pouvoir les traiter :

$$f(occupation) = \frac{occupation - occupationMin}{occupationMAx - occupationMin} \quad \text{(II.5)}$$

iii. Normaliser le sexe :

On a introduit des valeurs numériques au sexe :

- 0.5 pour les hommes
- 1 pour les femmes

iv. Normaliser le runtime :

$$f(runtime) = \frac{runtime - runtimeMin}{runtimeMAx - runtimeMin} \quad \text{(II.6)}$$

v. Normaliser le timestamp :

$$f(timestamp) = e^{-t} \qquad \text{(II.7)}$$

Où :

$$t = \frac{timestamp - timestamp\ précédant\ de\ l'utilisateur}{la\ moyenne\ des\ ecart\ temporels\ entre\ deux\ evaluations\ successives\ de\ l'utilisateur}$$

b) Définir la fonction objective de l'algorithme PSO :

L'objectif de cette étape est de définir la fonction qu'on doit minimiser :

$$fitness = RMSE = \sqrt{\frac{\sum_{h=1}^{nb}(Ph - reel\ h)^2}{nb}} \qquad \text{(II.8)}$$

Ph : la note prédite par le système pour un thème donné

Réel h : la note donnée par l'utilisateur à un thème

Nb : nombre de thème évalué

c) Calcule de prédiction :

L'objectif de cette étape est de prédire les préférences de l'utilisateur courant aux items non consultés en utilisant les résultats obtenus par l'algorithme PSO, la prédiction sera donc calculée selon la formule suivante :

$$Pu,i = Ri + \frac{\sum_{j=1}^{n}(Ru,j - \overline{R}_j) \times sim(i,j) \times f(u,i)}{\sum_{j=1}^{n} sim(i,j) \times f(u,i)} \qquad \text{(II.9)}$$

Où :

$f(u,i)$ = *Wa* × *f(age)* + *Wo* × *f(occupation)*+*Ws* × *f(sexe)* + *Wr* × *f(runtime)* + *Wt* × *f(timestamp)*

Où :

Age : l'Age de l'utilisateur courant.
Occupation : la valeur qui correspond à l'occupation de l'utilisateur courant
Sexe : la valeur qui correspond au sexe de l'utilisateur courant
Runtime : la durée du film courant
Timestamp : la valeur qui correspond au temps que l'utilisateur courant a vu le film

III.4.1.1.2. Le déplacement des particules :

La population est renouvelée à chaque génération en se basant sur deux fonctions : la première est utilisée pour changer la vitesse de la particule (formule 1.9) et la deuxième pour changer la position de cette dernière (formule1.10).

III.4.1.1.3. Vérification de la condition d'arrêt :

La condition d'arrêt est fixée par un certain nombre d'itération, si ce dernier est atteint, alors la meilleure solution sera extraite et l'algorithme affiche le vecteur des poids optimal, sinon une nouvelle itération sera effectuée.

III.4.1.1.4. Paramètres de l'algorithme PSO :

- Topologie de voisinage
- Variable C1
- Variable C2
- Variable 'w'
- Nombre d'itération
- Nombre de particule

IV. Etape d'implémentation :

Le système mis en œuvre est un algorithme de filtrage collaboratif sensible au contexte. Il permet de prendre en considération cinq paramètres contextuels avec leur degré d'importance afin de fournir à l'utilisateur des informations qui s'adaptent à ses préférences et son contexte de recherche.

IV.1. Outils logiciel de l'algorithme PSO :

L'algorithme décrit dans ce chapitre se base sur un algorithme d'optimisation par essaim de particule pour le calcul des poids à affecter à chaque information contextuelle. L'utilisation d'un outil logiciel générique et flexible est essentielle à la réalisation d'une application afin de rendre son développement beaucoup moins coûteux.

A cet effet, il existe de nombreuses librairies disponibles sur le *Web.* Dans le cadre de ce manuscrit, l'algorithme est implémenté en utilisant l'API Pyswarm qui est développée autour de la plate-forme *python.* Ce choix est justifié par le fait que cette bibliothèque est totalement libre, facile à utiliser et offre les services nécessaires pour l'implémentation de l'algorithme proposé.

IV.1.1. Structure de données :

Il existe trois couches principales dans l'API principale de Pyswarm :

- Optimiseurs : inclut toutes les implémentations standard de la plupart des algorithmes d'intelligence en essaim
- Base : API de base sur laquelle étaient basées la plupart des implémentations d'Optimizer. Chaque module de base est conçu en fonction du problème qu'ils tentent de résoudre : simple continu, discret, multiobjectif, contraint, etc.
- Backend : API qui expose les opérations courantes pour tout algorithme d'essaimage comme l'initialisation d'essaimage , la recherche du voisin le plus proche, etc.

La classe Swarm : ***pyswarms.backend.swarms.Swarm*** agit comme une classe de données qui conserve tous les attributs nécessaires dans une implémentation d'essaim donnée. On l'initialise en fournissant les matrices de position et de vitesse initiales. Pour l'itération en cours, on peut obtenir les informations suivantes de la classe :

Classe `pyswarms.backend.swarms.Swarm`(*position:numpy.ndarray , velocity: numpy.ndarray , n_particles: int = NOTHING , dimensions: int = NOTHING , options: dict = {} , pbest_pos: numpy.ndarray = NOTHING , best_pos: numpy.ndarray = array (( ) , dtype = float64) , pbest_cost: numpy.ndarray = array (() , dtype = float64) , best_cost: float = inf , current_cost: numpy.ndarray = array (() , dtype = float64)*)

1. **Position :** numpy.ndarray - matrice de position à un pas de temps donné(n_particles, dimensions)
2. **Velocity :** numpy.ndarray - matrice de vitesse à un pas de temps donné de la forme(n_particles, dimensions)
3. **n_particles :** int - nombre de particules dans un essaim.
4. **Dimensions :** int - nombre de dimensions dans un essaim.
5. **Options :** dict - un dictionnaire contenant les paramètres de la technique d'optimisation spécifique {'c1', 'c2', 'w'}
6. **pbest_pos :** numpy.ndarray - meilleures positions personnelles de chaque particule de forme. La valeur par défaut n'est Aucune(n_particles, dimensions)
7. **best_pos :** numpy.ndarray - meilleure position trouvée par l'essaim de forme pour la topologie et pour les autres topologies(dimensions,)pyswarms.backend.topology.Star(dimensions, particles)

8. **pbest_cost :** numpy.ndarray - meilleurs coûts personnels de chaque particule de forme(n_particles)
9. **best_cost :** float - meilleur coût trouvé par l'essaim, la valeur par défaut est numpy.inf
10. **current_cost :** numpy.ndarray - le coût actuel trouvé par l'essaim de forme(n_particles, dimensions)

Classe optimizer : Effectue l'optimisation pour évaluer la fonction objective pour un certain nombre d'itérations.

Paramètres :

objective_func (fonction) : fonction objective à évaluer

iters (int) : nombre d'itérations

n_processes (int) : nombre de processus à utiliser pour l'évaluation des particules parallèles. La valeur par défaut est None sans parallélisation

kwargs (dict) : arguments pour la fonction objective

Classe de topologie *pyswarms.backend.base.topology* : héberge toutes les opérations que vous pouvez utiliser sur les attributs Swarm. Une topologie met en œuvre trois méthodes régissant le comportement des essaims :

compute_gbest: calcule la meilleure particule (à la fois le coût et la position) en fonction d'une instance swarm.

compute_position: calcule la position suivante de l'essaim en fonction de sa position actuelle.

compute_velocity: calcule la vitesse de l'essaim en fonction de ses attributs.

IV.1.2. Processus d'exécution de l'algorithme PSO :

Le processus d'exécution d'un algorithme d'optimisation par essaim de particules est résumé dans la figure 6 :

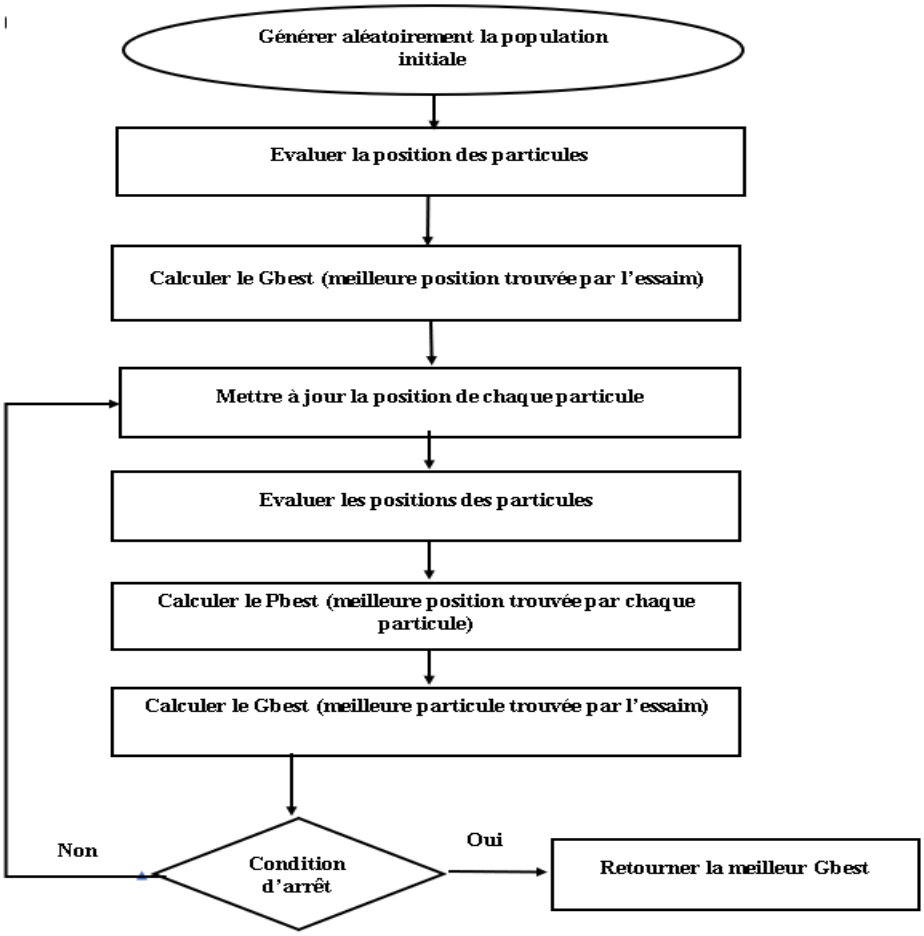


Figure 6: processus d'exécution de l'algorithme PSO

IV.2. Corpus d'évaluation utilisé :

Pour l'évaluation de l'algorithme proposé, le corpus *MovieLens* est utilisé pour faire les différents tests. Pour évaluer notre méthode, nous avons utilisé la méthode de validation utilisant 80% des données pour l'entrainement du modèle et 20% de données pour le test. Pour un utilisateur dans les données tests, nous tirons aléatoirement un item supposé être en cours de consultation. Le système entrainer propose ensuite des films similaires à cet item.

Si un item recommandé est effectivement noté positivement par l'utilisateur dans la collection, la recommandation est considérée comme pertinente. Chaque item est représenté par son identifiant, la durée de film, un code de Wikipédia et un résumé .

Les âges des clients se distribuent entre 7 à 73 ans. Plus de 50 % des individus ont un âge entre 26 et 45 ans et 23% d'entre eux sont entre 16 et 25 ans. Il y a également 21 métiers dont les plus représentatifs sont les étudiants (196 utilisateurs), les autres (105

utilisateurs), les professeurs (95 utilisateurs) et les agents de l'administration (79 utilisateurs). De plus 670 des utilisateurs sont des hommes et 273 sont des femmes.

IV.2.1. Préparation de données :

Le pré-traitement du texte comprend les étapes suivantes :

- ✓ Récupération du corpus par scraping ou en téléchargeant des fichiers textes, par exemple.
- ✓ La tokenization, qui désigne le découpage en mots des différents documents qui constituent le corpus.
- ✓ La normalisation et la construction du dictionnaire qui permet de ne pas prendre en compte des détails importants au niveau local (ponctuation, majuscules, conjugaison, etc.).

a) Récupération du corpus :

La première étape est la récupération du texte. Il existe plusieurs manières de récupérer du texte par exemple depuis une base de données en utilisant la bibliothèque « Pandas ».

b) La tokenization :

On veut dans un premier temps étudier le vocabulaire utilisé dans chaque description de films. On va utiliser la fonction « preprocess » qui va décomposer les vers en tableaux de mots afin de pouvoir effectuer des opérations dessus.

Le fait d'essayer d'harmoniser les tokens est un processus nommé « **normalisation »**.

b. La normalisation :

i. Supprimer les stopwords :

La première manipulation souvent effectuée dans le traitement du texte est la suppression de ce qu'on appelle en anglais les *stopwords*. Ce sont les mots très courants dans la langue étudiée ("et", "à", "le"... en français) qui n'apportent pas de valeur informative pour la compréhension du "sens" d'un document et corpus. Ils sont très fréquents et ralentissent notre travail : nous souhaitons donc les supprimer.

Il existe dans la librairie NLTK une liste par défaut des stopwords dans plusieurs langues, notamment le français. Mais nous allons faire ceci d'une autre manière : on va supprimer les mots les plus fréquents du corpus et considérer qu'ils font partie du vocabulaire commun et n'apportent aucune information. Ensuite on supprimera aussi les stopwords fournis par NLTK.

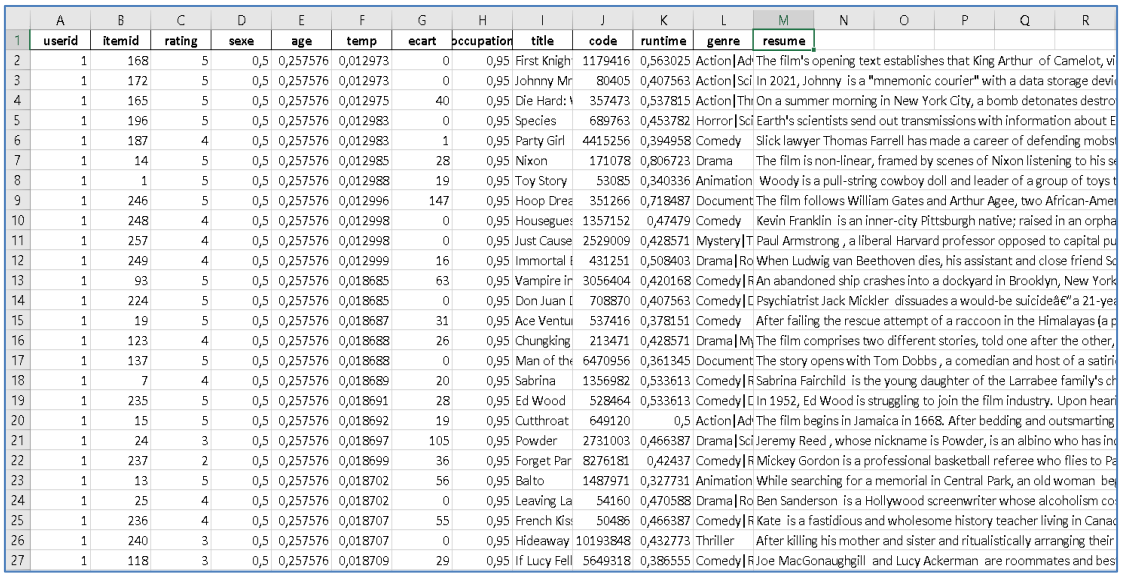
ii. Lemmatisation ou racinisation :

Le processus de « lemmatisation » consiste à représenter les mots sous leur forme canonique. Par exemple pour un verbe, ce sera son infinitif. Pour un nom, son masculin singulier. L'idée étant encore une fois de ne conserver que le sens des mots utilisés dans le corpus.

Il existe un autre processus qui exerce une fonction similaire qui s'appelle la racinisation (ou *stemming* en anglais). Cela consiste à ne conserver que la racine des mots étudiés. L'idée étant de supprimer les suffixes, préfixes des mots afin de ne conserver que leur origine. C'est un procédé plus simple que la lemmatisation et plus rapide à effectuer puisqu'on tronque les mots essentiellement contrairement à la lemmatisation qui nécessite d'utiliser un dictionnaire.

Dans notre cas, on applique une racinisation parce qu'il n'existe pas de fonction de lemmatisation de corpus anglais dans NLTK.

Afin de simplifier l'évaluation des algorithmes implémentés, toutes les données nécessaires sont rangées dans un seul fichier au format (.xlsx) nommé *BDDFinal.xlsx*. Les données sont triées dans un ordre croissant selon les *user id* puis les *timestamps*. 20% des données sont utilisées pour le test de l'algorithme implémenté et les données restantes sont utilisés afin de remplir les différentes matrices.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	userid	itemid	rating	sexe	age	temp	ecart	occupation	title	code	runtime	genre	resume					
2	1	168	5	0,5	0,257576	0,012973	0	0,95	First Knigh	1179416	0,563025	Action|Ad	The film's opening text establishes that King Arthur of Camelot, vi					
3	1	172	5	0,5	0,257576	0,012973	0	0,95	Johnny Mr	80405	0,407563	Action|Sci	In 2021, Johnny is a "mnemonic courier" with a data storage devi					
4	1	165	5	0,5	0,257576	0,012975	40	0,95	Die Hard: \	357473	0,537815	Action|Th	On a summer morning in New York City, a bomb detonates destro					
5	1	196	5	0,5	0,257576	0,012983	0	0,95	Species	689763	0,453782	Horror|Sci	Earth's scientists send out transmissions with information about E					
6	1	187	4	0,5	0,257576	0,012983	1	0,95	Party Girl	4415256	0,394958	Comedy	Slick lawyer Thomas Farrell has made a career of defending mobs					
7	1	14	5	0,5	0,257576	0,012985	28	0,95	Nixon	171078	0,806723	Drama	The film is non-linear, framed by scenes of Nixon listening to his s					
8	1	1	5	0,5	0,257576	0,012988	19	0,95	Toy Story	53085	0,340336	Animation	Woody is a pull-string cowboy doll and leader of a group of toys t					
9	1	246	5	0,5	0,257576	0,012996	147	0,95	Hoop Drea	351266	0,718487	Document	The film follows William Gates and Arthur Agee, two African-Amer					
10	1	248	4	0,5	0,257576	0,012998	0	0,95	Housegues	1357152	0,47479	Comedy	Kevin Franklin is an inner-city Pittsburgh native; raised in an orpha					
11	1	257	4	0,5	0,257576	0,012998	0	0,95	Just Cause	2529009	0,428571	Mystery|T	Paul Armstrong , a liberal Harvard professor opposed to capital pu					
12	1	249	4	0,5	0,257576	0,012999	16	0,95	Immortal E	431251	0,508403	Drama|Ro	When Ludwig van Beethoven dies, his assistant and close friend Sc					
13	1	93	5	0,5	0,257576	0,018685	63	0,95	Vampire in	3056404	0,420168	Comedy|R	An abandoned ship crashes into a dockyard in Brooklyn, New York					
14	1	224	5	0,5	0,257576	0,018685	0	0,95	Don Juan D	708870	0,407563	Comedy|D	Psychiatrist Jack Mickler dissuades a would-be suicideâ€"a 21-yea					
15	1	19	5	0,5	0,257576	0,018687	31	0,95	Ace Ventur	537416	0,378151	Comedy	After failing the rescue attempt of a raccoon in the Himalayas (a p					
16	1	123	4	0,5	0,257576	0,018688	26	0,95	Chungking	213471	0,428571	Drama|My	The film comprises two different stories, told one after the other,					
17	1	137	5	0,5	0,257576	0,018688	0	0,95	Man of the	6470956	0,361345	Document	The story opens with Tom Dobbs , a comedian and host of a satiri					
18	1	7	4	0,5	0,257576	0,018689	20	0,95	Sabrina	1356982	0,533613	Comedy|R	Sabrina Fairchild is the young daughter of the Larrabee family's ch					
19	1	235	5	0,5	0,257576	0,018691	28	0,95	Ed Wood	528464	0,533613	Comedy|D	In 1952, Ed Wood is struggling to join the film industry. Upon hear					
20	1	15	5	0,5	0,257576	0,018692	19	0,95	Cutthroat	649120	0,5	Action|Ad	The film begins in Jamaica in 1668. After bedding and outsmarting					
21	1	24	3	0,5	0,257576	0,018697	105	0,95	Powder	2731003	0,466387	Drama|Sci	Jeremy Reed , whose nickname is Powder, is an albino who has inc					
22	1	237	2	0,5	0,257576	0,018699	36	0,95	Forget Par	8276181	0,42437	Comedy|R	Mickey Gordon is a professional basketball referee who flies to Pa					
23	1	13	5	0,5	0,257576	0,018702	56	0,95	Balto	1487971	0,327731	Animation	While searching for a memorial in Central Park, an old woman be					
24	1	25	4	0,5	0,257576	0,018702	0	0,95	Leaving La	54160	0,470588	Drama|Ro	Ben Sanderson is a Hollywood screenwriter whose alcoholism co					
25	1	236	4	0,5	0,257576	0,018707	55	0,95	French Kis	50486	0,466387	Comedy|R	Kate is a fastidious and wholesome history teacher living in Canad					
26	1	240	3	0,5	0,257576	0,018707	0	0,95	Hideaway	10193848	0,432773	Thriller	After killing his mother and sister and ritualistically arranging their					
27	1	118	3	0,5	0,257576	0,018709	29	0,95	If Lucy Fell	5649318	0,386555	Comedy|R	Joe MacGonaughgill and Lucy Ackerman are roommates and bes					

Figure 7 : Données utilisées par de l'algorithme proposé

IV.2.2. Métrique d'évaluation utilisé :.

La métrique utilisée pour l'évaluation de l'algorithme proposé est la racine de l'erreur moyenne quadratique (Mean Squard Error : RMSE) calculer selon l'équation (I.6)

IV.3. Méthodologie d'implémentation :

Etape 1 : Remplissage du fichier BDDFInal.xlsx à partir de 3 autres fichiers pour obtenir la base de données nécessaire au travail.

Etape 2 : Division de données.

On doit trier les données selon userid et timestamp en suite on les devise en deux :

80% pour apprentissage et 20% pour test.

```
apprentissage,test = train_test_split(bdd , test_size = 0.2)
```

Etape 3 : traitement des données.

Cette procédure de LDA utilise les packages Python suivants :

- NLTK (http://www.nltk.org/install.html), une boîte à outils en langage naturel pour Python. Un package utile pour tout traitement du langage naturel.
- stop_words (https://pypi.python.org/pypi/stop-words) , un package Python contenant des mots vides (https://pypi.python.org/pypi/stop-words) .

- gensim (https://radimrehurek.com/gensim/install.html) , un package de modélisation de sujet contenant notre modèle LDA.

```
stemmer = PorterStemmer()
#fonction de lemmatisation
def lemmatize_stemming(text):
    return stemmer.stem(WordNetLemmatizer().lemmatize(text, pos='v'))
```

```
def preprocess(text):
    result = []
    for token in gensim.utils.simple_preprocess(text):
        if token not in gensim.parsing.preprocessing.STOPWORDS and len(token) > 3:
            result.append(lemmatize_stemming(token))
    return result
```

```
doc_filtre = datatext['dictionnaire'].map(preprocess)
```

Etape 4 : Création du dictionnaire de données.

```
doc_filtre = datatext['dictionnaire'].map(preprocess)
```

```
doc_filtre
```

```
0      [stori, anim, children, comedi, woodi, pull, s...
1      [jumanji, adventur, children, fantasi, boy, bu...
2      [grumpier, comedi, romanc, lifelong, feud, joh...
3      [wait, exhal, comedi, drama, wait, exhal, stor...
4      [father, bride, comedi, georg, bank, accept, r...
                             ...
789    [star, map, drama, allegor, tale, concern, exi...
790    [comedi, howard, brackett, like, english, lite...
791    [confidenti, crime, film, noir, mysteri, thril...
792    [soul, food, drama, soul, food, tell, eye, yea...
793    [wishmast, horror, narrat, angu, scrimm, word,...
```

Etape 5 : Exécution de LDA :

```
bow_corpus = [dictionary.doc2bow(doc) for doc in doc_filtre]
```

```
lda_model = gensim.models.LdaMulticore(bow_corpus,
                                       num_topics=60,
                                       id2word=dictionary,
                                       passes=2,
                                       alpha=0.5,
                                       eta=0.2)
```

Etape 6 : Création de la matrice d'item × topic :

index											
topic	0	1	2	3	4	5	6	7	8	9	...
value											
0	0.000000	0.000000	0.000000	0.000000	0.237781	0.000000	0.000000	0.000000	0.000000	0.00000	...
1	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000	...
2	0.000000	0.000000	0.000000	0.000000	0.033800	0.000000	0.000000	0.000000	0.000000	0.00000	...
3	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000	...
4	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000	...
5	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000	...
6	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000	...
7	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000	...
8	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000	...

Etape 7 : Création de la matrice de similarité :

	topic	0	1	2	3	4	5	6	7	8	9	...
	topic											
	0	1.000000	0.196672	0.002670	-0.022639	0.017295	-0.001516	0.027561	0.597376	-0.001315	-0.022340	...
	1	0.196672	1.000000	-0.005486	0.033465	0.018458	-0.019174	-0.016516	0.047354	-0.019023	-0.017815	...
	2	0.002670	-0.005486	1.000000	0.007508	0.381735	0.996864	-0.017803	0.799916	0.997528	-0.017161	...
	3	-0.022639	0.033465	0.007508	1.000000	0.041521	0.006308	0.019397	-0.005853	0.006474	-0.018073	...
	4	0.017295	0.018458	0.381735	0.041521	1.000000	0.385619	-0.037481	0.315109	0.387459	-0.038703	...
index	...	...	...	...	...	...	...	...	...	...	...	...

Etape 8 : on a implémenté plusieurs fonctions pour calculer les prédictions, par exemple :

```
def getRuij(itemid,userid):
    filtre=apprentissage['item id']==itemid
    apprenFiltre=apprentissage[filtre]
    filtre=apprenFiltre['user id']==userid
    apprenFiltre=apprenFiltre[filtre]
    if apprenFiltre.empty:
        return 0
    else:
        row=apprenFiltre.iloc[0]
        return(row[2])
```

Etape 9 : Fonction de prédiction et la fonction d'erreur RMSE :

```
from math import sqrt
i=0
while i<5:
#len(test):

    Ri=get_moyenne_rating_item(itemid)
    itemindex=getindexfromitemid(itemid)
    similarlist=get_similar_list(itemindex)
    j=0
    somme1=0
    somme2=0
    RMSE=0
    while j<len(similarlist):
        indexj=(get_index_list(similarlist,j))
        itemidj=getitemidfromindex(indexj)
        Rj=get_moyenne_rating_item(itemidj)
        Ruij=getRuij(itemidj,userid)


        #************************************************************
        simij=similarlist[j]
        #simij=similarList[j]*f(u,i)
        nb=(Ruij-Rj)*simij
        #nb=(Ruij-Rj)*simij*f(u,i)
        somme1=somme1+nb
        somme2=somme2+simij
        #************************************************************


        j=j+1
    Pui=Ri+(somme1/somme2)
    RMSEi=Pui-rating
    RMSE=RMSE+(RMSEi**2)
    print(Pui)
    print('   ')

    i=i+1
RMSE=sqrt(RMSE/5)
#RMSE=sqrt(RMSE/len(test))
print('RMSE : ')
print(RMSE)
```

Etape 10 : l'implémentation de l'algorithme d'essaim de particule.

```
# instatiate the optimizer
x_max = 0.09
x_min = 0.1
bounds = (x_min, x_max)
options = {'c1': 0.5, 'c2': 0.3, 'w': 0.9}
iteration = 10000
#initialize the particles
particles = []
for i in range(swarms):
    p = Particle()
    p.params = array([random() for i in range(dimensions)])
    p.fitness = 10.0
    p.v = 0.0
    particles.append(p)
optimizer = GlobalBestPSO(RMSE, n_particles=50, dimensions=5, options=options, bounds=bounds)
```

V. Conclusion :

Dans ce chapitre, on a discuté de la méthode de travail et des algorithmes. On a proposé un schéma global du système de recommandation souhaité dont chaque étape est détaillée d'une façon théorique rien que pour montrer les fonctions, les formules et les outils de développement utilisés. La pratique était basée sur la théorie qu'on a déjà expliquée. Dans le chapitre de suite, on va tester la qualité du système pour obtenir les meilleurs résultats.

Chapitre 3 : EXPERIMENTATION ET RESULTAT

I. Introduction :

Les différents algorithmes utilisés dans le cadre de ce travail sont établis et testé sur l'ensemble de données MovieLens afin de prouver leur efficacité en préconisant des listes de recommandation pertinentes et diversifiées.

II. Caractéristique de PC :

L'algorithme décrit dans ce chapitre est développé sur un Pc équipé d'un Intel Core i5, avec une vitesse de 2.5 Ghz et 8 Go de RAM. Il est développé en langage python sous l'environnement de développement Jupyter.

III. Environnement de travail :

Cette section présente les outils nécessaires pour le développement du système proposé :

- **Python :** Python est un langage de programmation interprété, multiparadigme et multiplateformes. Il favorise la programmation impérative structurée, fonctionnelle et orientée objet. Il est doté d'un typage dynamique fort, d'une gestion automatique de la mémoire par ramasse-miettes et d'un système de gestion d'exceptions. Le langage Python est placé sous une licence libre proche de la licence BSD et fonctionne sur la plupart des plates-formes informatiques, des smartphones aux ordinateurs centraux, de Windows à Unix avec notamment GNU/Linux en passant par macOS, ou encore Android, iOS, et peut aussi être traduit en Java ou .NET. Il est conçu pour optimiser la productivité des programmeurs en offrant des outils de haut niveau et une syntaxe simple à utiliser.

- **Jupyter :** Jupyter est une application web utilisée pour programmer dans plus de 40 langages de programmation, dont Python, Julia, Ruby, R, ou encore Scala. Jupyter permet de réaliser des calepins ou notebooks, c'est-à-dire des programmes contenant à la fois du texte en markdown et du code en Julia, Python, R... Ces calepins sont utilisés en science des données pour explorer et analyser des données.

- **Microsoft excel** : un logiciel tableur de la suite bureautique Microsoft office développé et distribué par l'éditeur Microsoft. La version la plus récente est Excel 2019. Il est destiné à fonctionner sur les plates-formes Microsoft Windows, Mac OS X, Android ou Linux (moyennant l'utilisation de Wine). Le logiciel Excel intègre des fonctions de calcul numérique, de représentation graphique, d'analyse de

données (notamment de tableau croisé dynamique) et de programmation, laquelle utilise les macros écrites dans le langage VBA (Visual Basic for Applications) qui est commun aux autres logiciels de Microsoft Office.

IV. Base de données :

Nous avons utilisé dans nos expérimentations la base de données MovieLens afin d'évaluer notre algorithme. L'ensemble de données MovieLens est composé de 943 utilisateurs et 1682 items avec une échelle d'évaluation comprise entre [1,5], où chaque utilisateur dispose de plus de 20 votes.

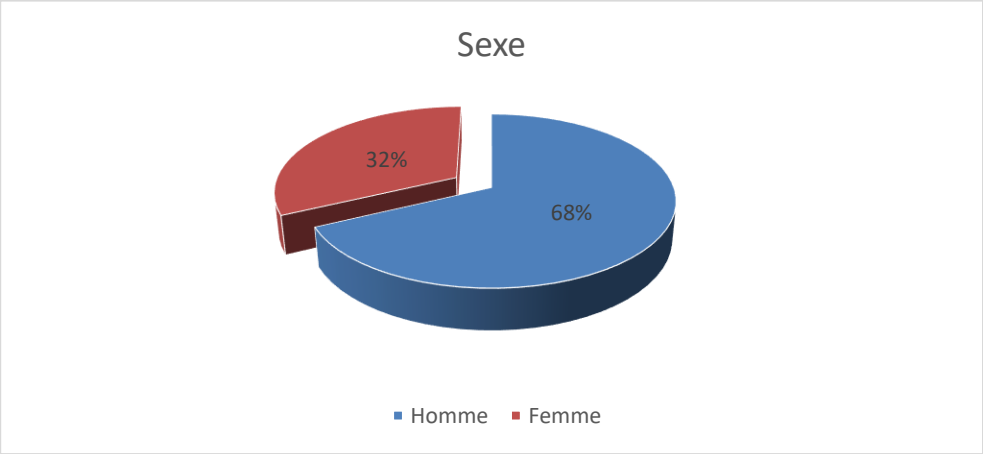


Figure 8 : secteur de sexe de la BDD utilisée

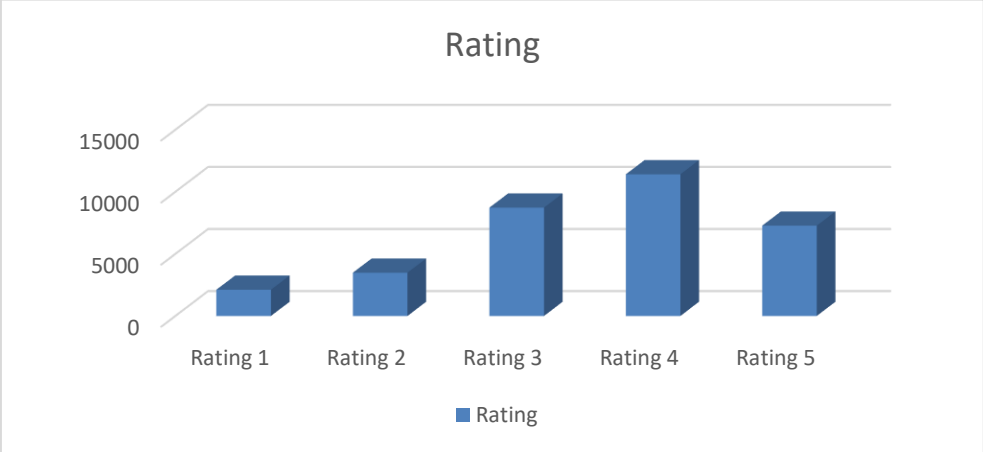


Figure 9 : histogramme de rating des utilisateurs

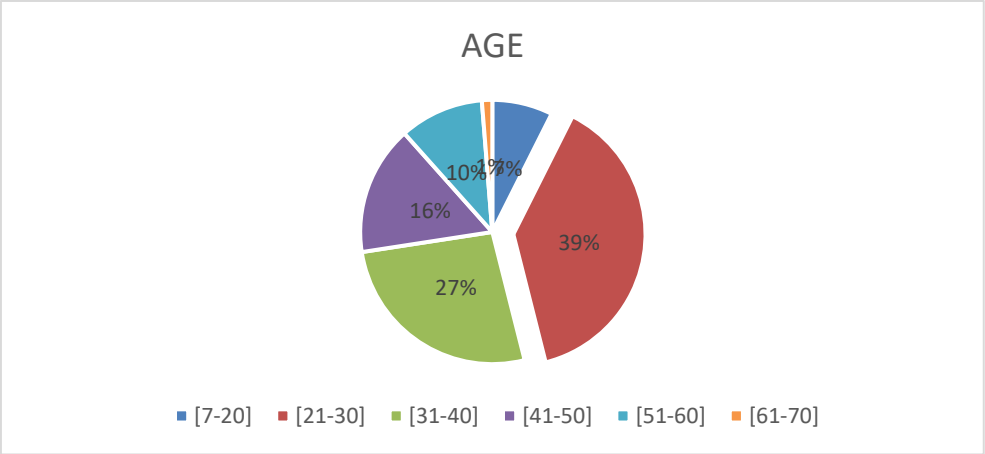


Figure 10 : Secteur d'âges des utilisateurs

IV.1. Objective des expérimentations :

Nous menons une série d'expériences pour examiner l'efficacité et la faisabilité des différentes propositions présentées dans le chapitre précédent. Ces expériences sont faites pour trouver les meilleures valeurs de chaque variable proposée pour cela on doit faire plusieurs expérimentations pour avoir une bonne conclusion. En particulier, nous abordons les questions suivantes :

- Es-ce-que le filtrage collaboratif est fiable ?
- Quelle sont les meilleurs paramètres pour trouver un meilleur résultat ?
- Es-ce-que l'ajout de PSO peut influencer sur les résultats obtenus ?
- Quelles sont les meilleurs paramètres de PSO ?

IV.1.1.Expérimentation 1 :

Les figures de cette expérimentation représentent les variations de RMSE en fonction de la similarité ; donnant des valeurs constantes pour : num_topics = 50, passes = 10, eta = 0.1

Les différentes valeurs de RMSE sont représentées sur l'axe des ordonnés tandis que l'axe des abscisses est réservé pour la similarité ou 0.1 est représenté par 1.

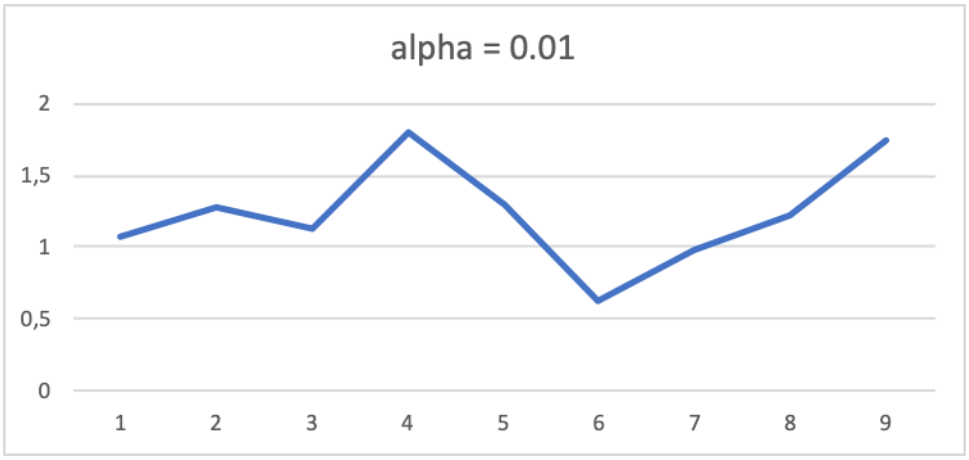


Figure 11 : courbe de RMSE en fonction de similarité avec alpha = 0.01

Tout d'abord on peut constater que les valeurs de RMSE ont faiblement augmenté jusqu'à ce qu'elle atteigne une valeur maximale 1.804413143 pour une similarité =0.4. Puis, d'après les valeurs de RMSE on note un recul important de ces dernières jusqu'à ce qu'elle atteigne sa valeur minimale 0.618353762 où similarité =0.6. Enfin, en examinant attentivement les chiffres on s'aperçoit que les valeurs de RMSE sont nettement redressées.

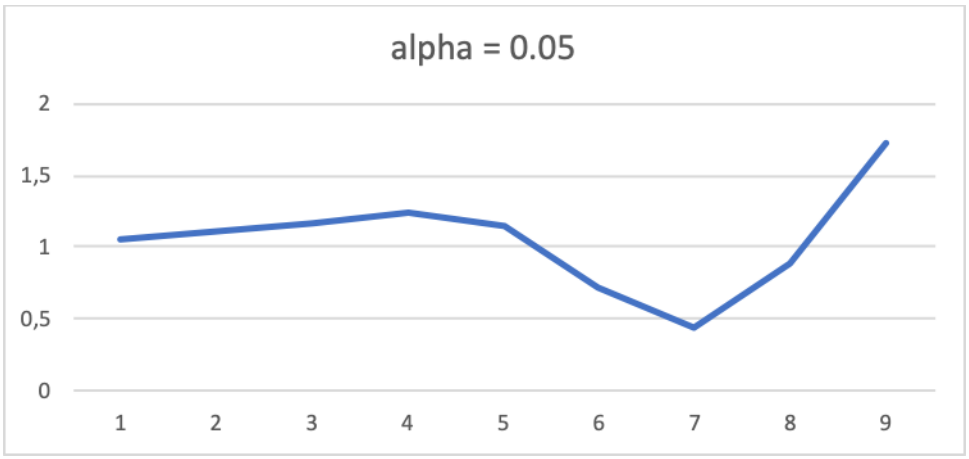


Figure 12 : courbe de RMSE en fonction de similarité avec alpha = 0.05

Tout d'abord, les chiffres montrent une faible augmentation de RMSE en fonction de similarité. Puis, d'après ces chiffres on observe une baisse des valeurs de RMSE jusqu'à ce qu'elle atteigne une valeur minimale = 0.441332242 avec une similarité 0.7. Enfin, on observe une croissance rapide des valeurs de RMSE en fonction de similarité jusqu'à ce qu'elle atteigne un maximum 1.729664347 pour une similarité de 0.9.

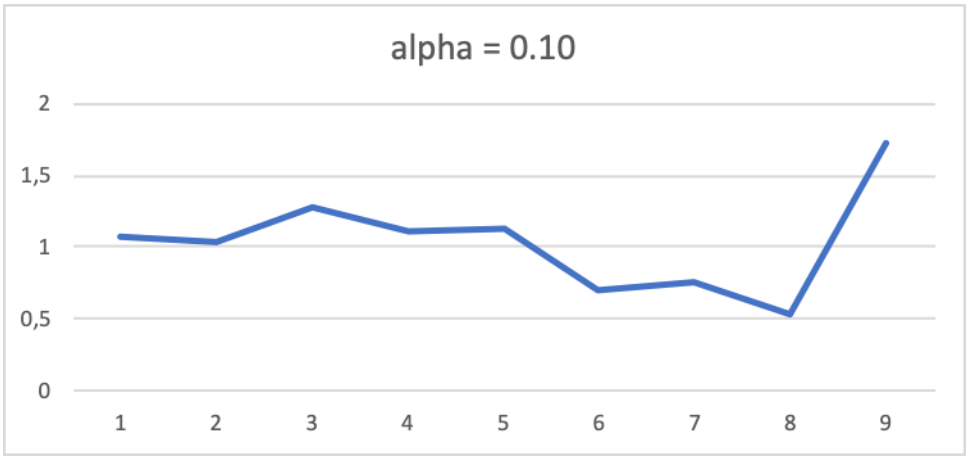


Figure 13 : courbe de RMSE en fonction de similarité avec alpha = 0.1

Tout d'abord la courbe montre une petite amélioration des valeurs de RMSE avec l'augmentation de la similarité. Ensuite, les chiffres nous montrent une dégradation des valeurs de RMSE jusqu'à ce qu'elle atteigne une valeur minimale égale à 0.534719501 Pour une similarité de 0.8. Enfin, en examinant attentivement les chiffres on s'aperçoit que les valeurs de RMSE sont sensiblement progressées pour atteindre un maximum qui est représenté par la valeur 1.719642754 avec une similarité 0.9.

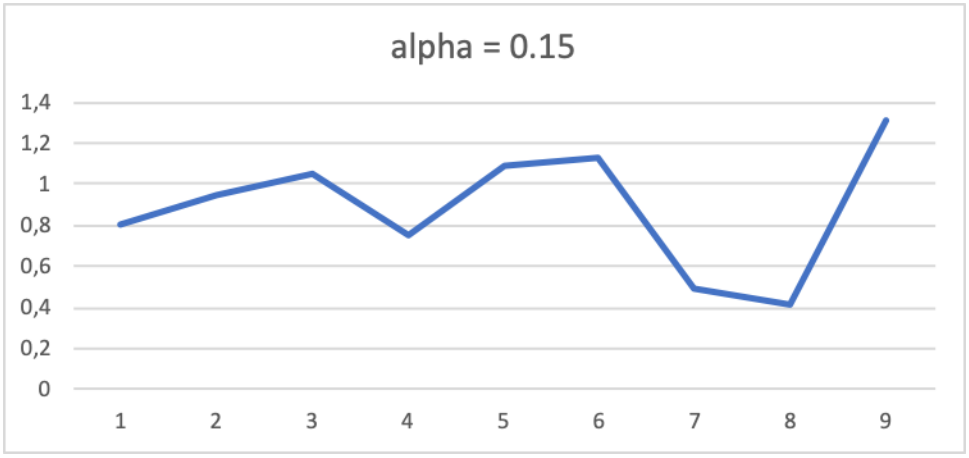


Figure 14 : courbe de RMSE en fonction de similarité avec alpha = 0.15

Tout d'abord, la courbe est oscillante et les valeurs de RMSE sont très proches les unes des autres. Puis, on observe une régression importante des valeurs de RMSE jusqu'à atteigne sa valeur minimale 0.420460539 et une similarité égale à 0.8. Enfin, en examinant le reste de la courbe on s'aperçoit que les valeurs de RMSE ont fortement augmenté pour atteindre un maximum 1.304113.

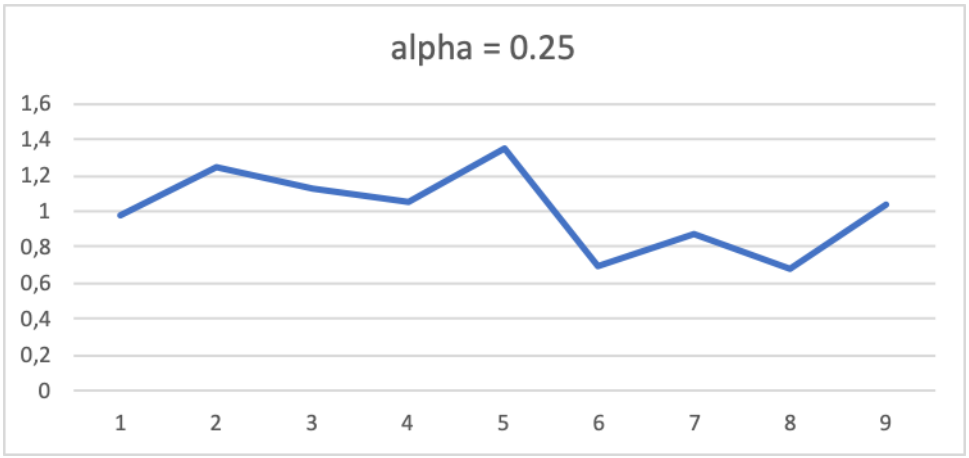


Figure 15 : courbe de RMSE en fonction de similarité avec alpha = 0.25

Tout d'abord, la courbe est instable et les valeurs de RMSE sont proches les unes des autres avec un maximum 1.34355908 pour une similarité de 0.5. Puis, on note un déclin des valeurs de RMSE jusqu'à atteigne sa valeur minimale 0.692918788 pour une similarité de 0.6, ensuite une petite augmentation de la courbe suivie d'une baisse des valeurs de RMSE pour atteindre une deuxième fois pratiquement la même valeur minimale précédente mais pour une similarité 0.8. Enfin, en analysant dans le détail le graphique on s'aperçoit que les valeurs de RMSE haussent.

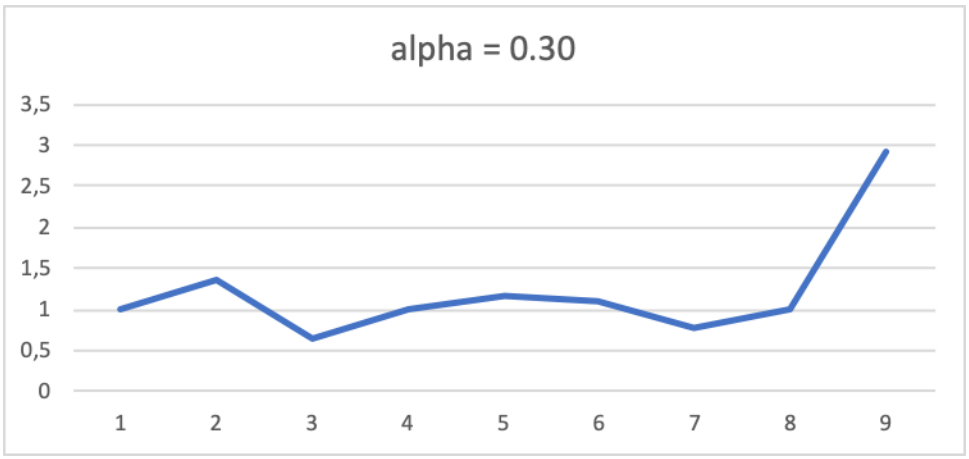


Figure 16 : courbe de RMSE en fonction de similarité avec alpha = 0.3

Tout d'abord, les chiffres montrent une faible augmentation de RMSE en fonction de similarité. Puis, d'après ces chiffres on observe une baisse des valeurs de RMSE jusqu'à ce qu'elle atteigne une valeur minimale = 0.6509594 avec une similarité de 0.3. Enfin, on observe progression des valeurs de RMSE en fonction de similarité jusqu'à ce qu'elle atteigne un maximum 2.912382 pour une similarité de 0.9.

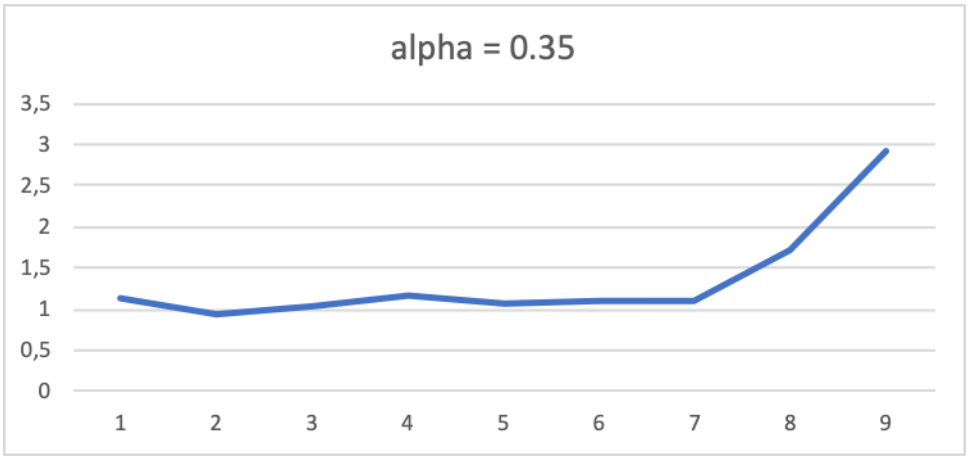


Figure 17: courbe de RMSE en fonction de similarité avec alpha = 0.35

Tout d'abord on peut constater que les valeurs de RMSE sont stables. Puis, d'après les valeurs de RMSE on note une croissance rapide de ces dernières à partir de la similarité 0.2 jusqu'à ce qu'elle atteigne sa valeur maximale 2.9187169 où similarité =0.9.

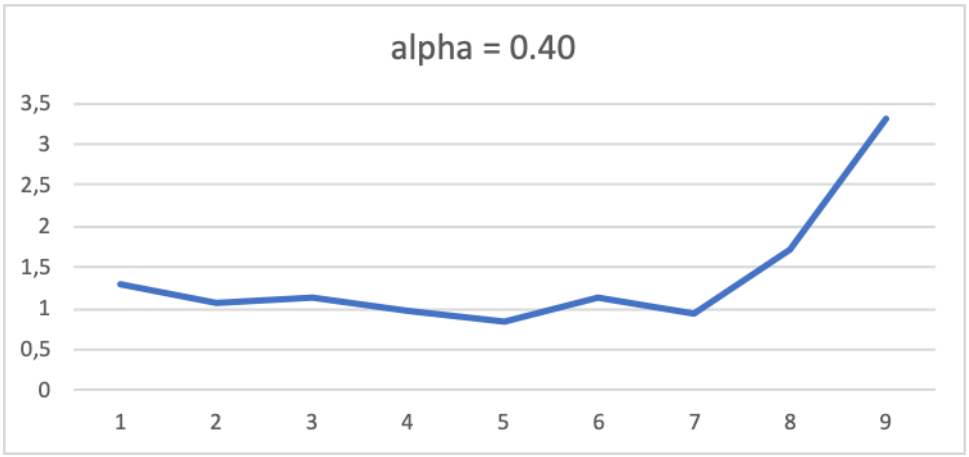


Figure 18 : courbe de RMSE en fonction de similarité avec alpha = 0.4

Tout d'abord, la courbe est stable et les valeurs de RMSE sont proches les unes des autres. Puis, on note un faible recul des valeurs de RMSE jusqu'à atteigne sa valeur minimale 0.836682 pour une similarité de 0.5, ensuite une petite amélioration de la courbe. Enfin, on remarque une croissance rapide de la courbe à partir de la similarité 0.7 pour atteindre son maximum où RMSE =3.293693 pour une similarité 0.9

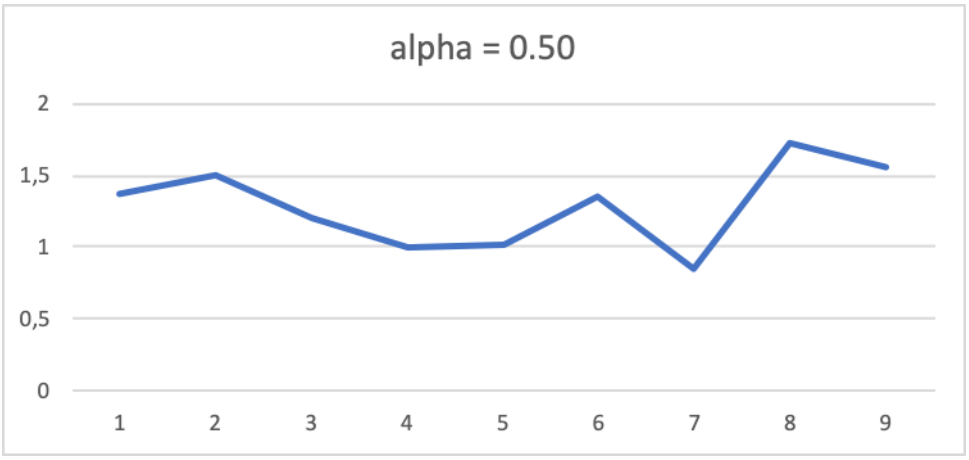


Figure 19: courbe de RMSE en fonction de similarité avec alpha = 0.5

En examinant attentivement les chiffres les valeurs de RMSE sont instables. Cette variation des valeurs est responsable de cette courbe oscillante.

Ensuite, d'après les valeurs de RMSE on peut extraire deux valeurs marginales, la plus petite 0.849198 pour une similarité 0.7 et la plus grande 1.715752 pour une similarité de 0.8**.**

IV.1.2.Expérimentation 2 :

Les figures de cette expérimentation représentent les variations de RMSE en fonction du seuil de similarité variations de RMSE en fonction de la similarité ; donnant des valeurs constantes pour : num_topics = 50, passes = 10, alpha= 0.1

Les différentes valeurs de RMSE sont représentées sur l'axe des ordonnés tandis que l'axe des abscisses est réservé pour la similarité ou 0.1 est représenté par 1.

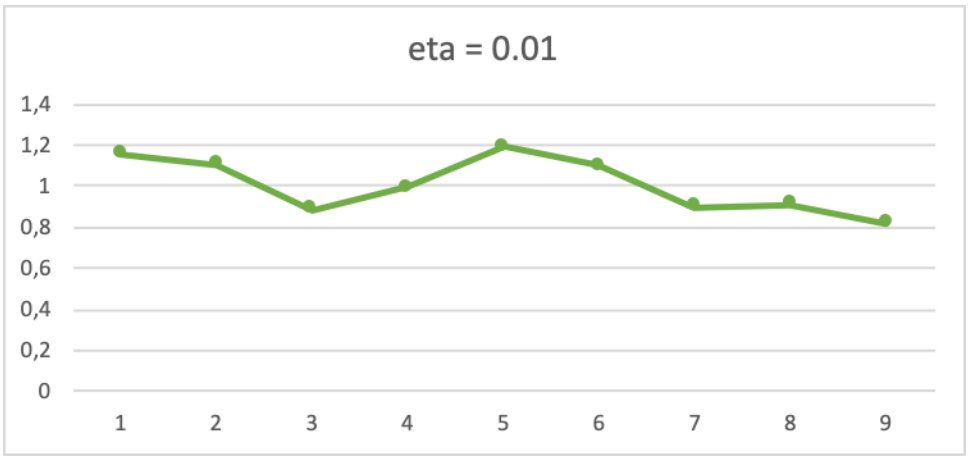


Figure 10: courbe de RMSE en fonction de similarité avec eta = 0.01

En premier lieu, les chiffres montrent une diminution de RMSE en fonction de similarité. Puis, on observe une augmentation des valeurs de RMSE jusqu'à ce qu'elle atteigne une valeur maximale = 1.190267 avec une similarité de 0.5.

Enfin, on remarque une diminution progressive des valeurs de RMSE jusqu'à ce qu'elle atteigne la valeur la plus petite 0.817842 pour une similarité de 0.9.

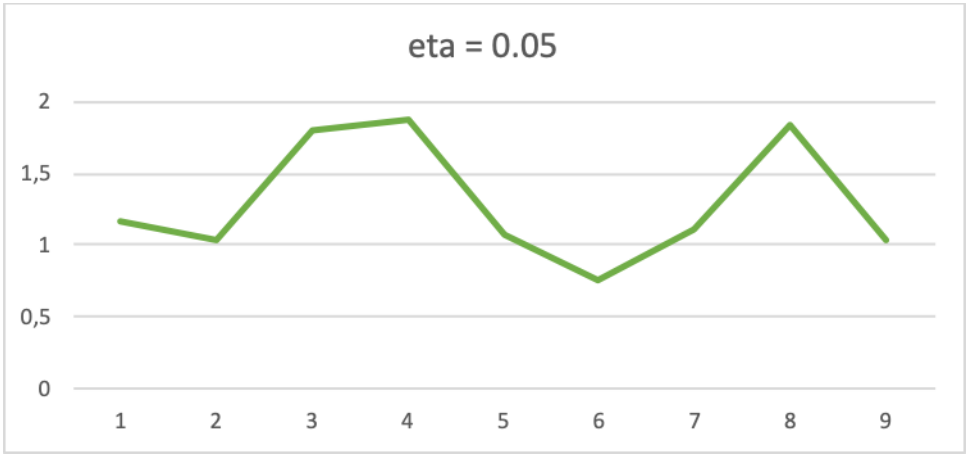


Figure 11: courbe de RMSE en fonction de similarité avec eta = 0.05

Tout d'abord, la courbe est croissante et les valeurs de RMSE augmentent pour atteindre un maximum 1.8752522 pour une similarité de 0.5. Puis, on note une décroissance rapide des valeurs de RMSE jusqu'à atteigne sa valeur minimale0.7543786 pour une similarité de 0.6, ensuite une augmentation progressive de la courbe. Enfin, en analysant dans le détail le graphique on s'aperçoit que les valeurs de RMSE reculent.

Figure 22: courbe de RMSE en fonction de similarité avec eta = 0.1

Tout d'abord, la courbe est stable et les valeurs de RMSE sont proches les unes des autres. Puis, on note un faible recul des valeurs de RMSE jusqu'à atteigne sa valeur minimale 0.8449208 pour une similarité de 0.4, ensuite une petite amélioration de la courbe suivie

d’une faible diminution. Enfin, on remarque une croissance rapide de la courbe à partir de la similarité 0.8 pour atteindre son maximum où RMSE =1.90695709 pour une similarité 0.9.

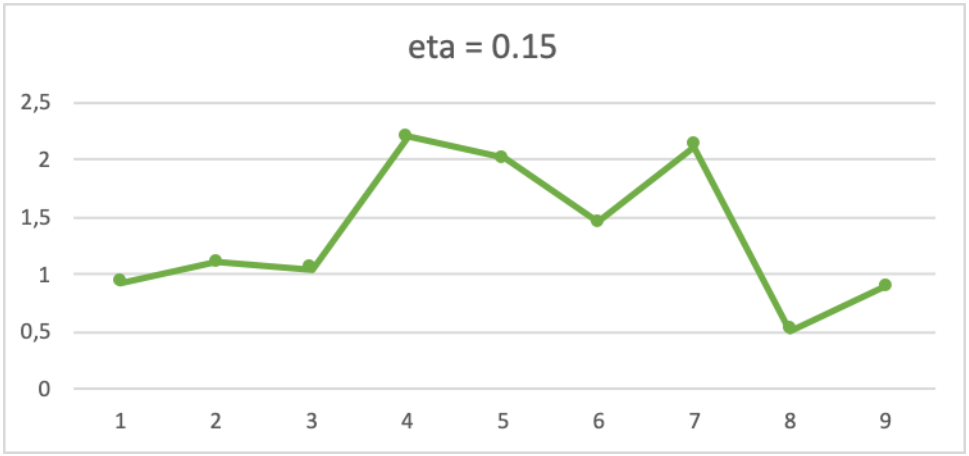


Figure 23: courbe de RMSE en fonction de similarité avec eta = 0.15

Tout d’abord on peut constater que les valeurs de RMSE augmentent progressivement jusqu’à ce qu’elle atteigne une valeur maximale2.192326 pour une similarité = 0.4.

Puis, on note une décroissance des valeurs de RMSE, ensuite une croissance de la courbe. Enfin, en analysant le graphique on s’aperçoit que les valeurs de RMSE reculent jusqu’à ce qu’elles atteignent un minimum = 0.51089920 pour une similarité 0.8.

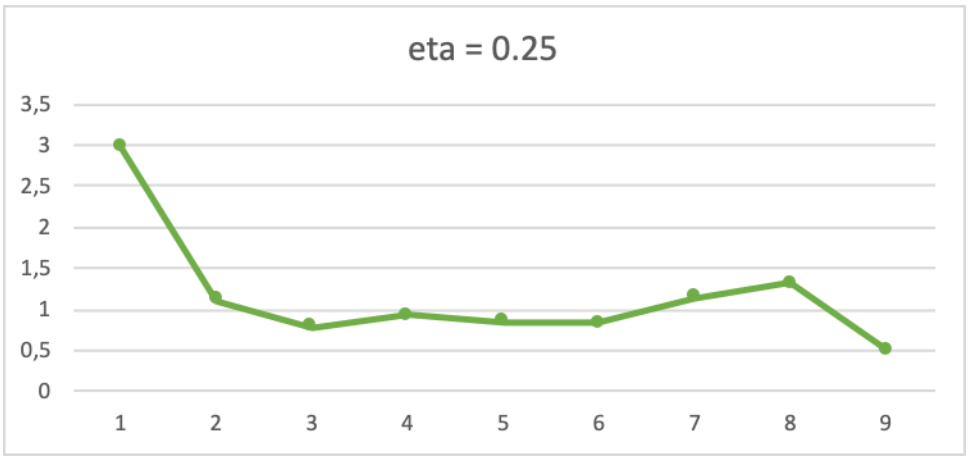


Figure 24: courbe de RMSE en fonction de similarité avec eta = 0.25

Tout d’abord la courbe est décroissante, elle montre une diminution importante des valeurs de RMSE en fonction de la similarité. Ensuite, on observe que la courbe est devenue stable et les valeurs de RMSE sont proches les unes des autres avec une petite amélioration de ces dernières.

Enfin, en examinant attentivement les chiffres on s'aperçoit que les valeurs de RMSE ont diminuées pour atteindre un minimum qui est représenté par la valeur 0.50493073 avec une similarité de 0.9.

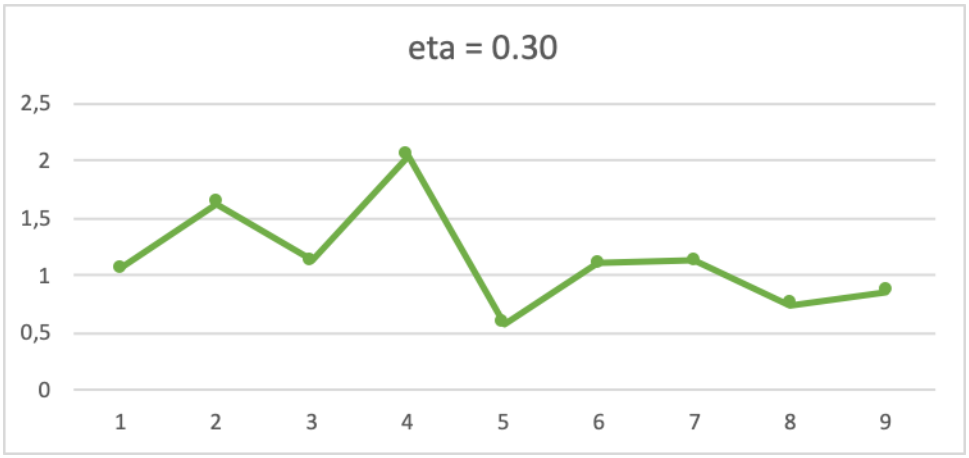


Figure 25: courbe de RMSE en fonction de similarité avec eta = 0.30

Tout d'abord, la courbe est oscillante et les valeurs de RMSE se varient entre diminution et élévation jusqu'à ce qu'elles atteignent sa valeur minimale 0.58334147, ensuite à partir de cette valeur de similarité la courbe commence à se stabiliser et les valeurs de RMSE se rapprochent les unes des autres.

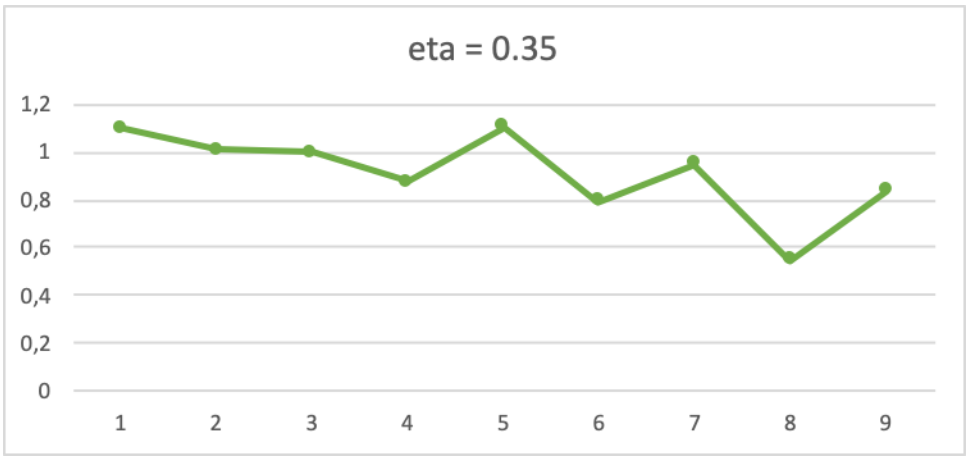


Figure 26: courbe de RMSE en fonction de similarité avec eta = 0.35

Tout d'abord, la courbe décroissante, elle montre une diminution des valeurs de RMSE en fonction de la similarité.

Puis, on remarque que la courbe est devenue oscillante et les valeurs de RMSE se varient entre diminution et élévation, à partir de cette valeur de cette variation on peut extraire le minimum qui est 0.5467857 et un maximum 1.10189007.

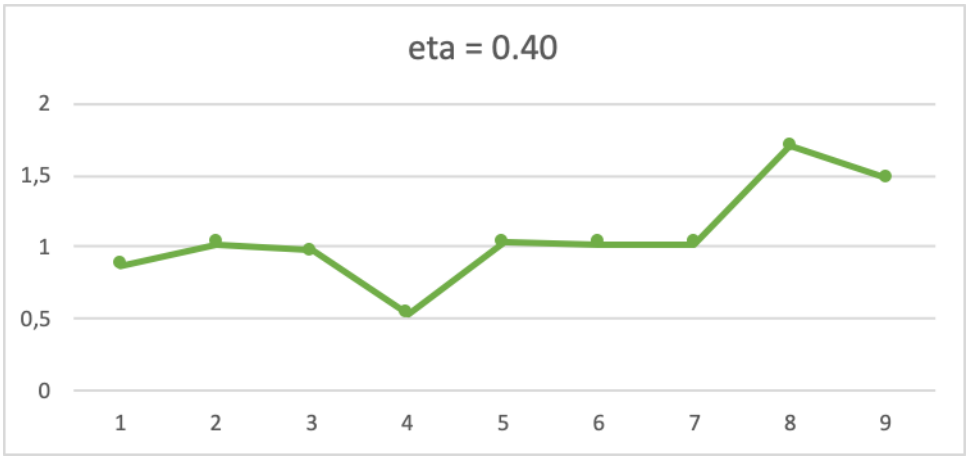


Figure 27: courbe de RMSE en fonction de similarité avec eta = 0.4

Tout d'abord, la courbe est stable et les valeurs de RMSE sont proches les unes des autres.

Puis, on note un déclin des valeurs de RMSE = 0.5369905, suivi d'une augmentation de la courbe qui va atteindre sa valeur maximale 1.6987633 pour une similarité de 0.8. Enfin, en analysant dans le détail le graphique on s'aperçoit que les valeurs de RMSE reculent légèrement.

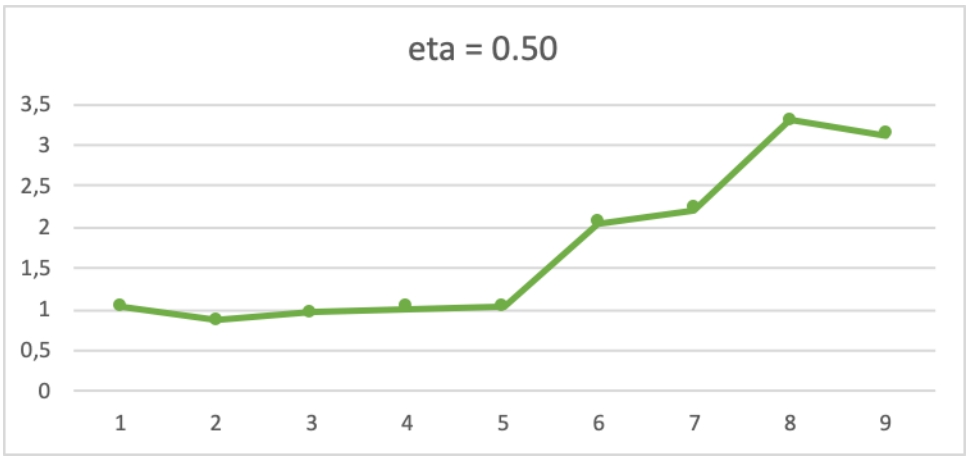


Figure 28: courbe de RMSE en fonction de similarité avec eta = 0.5

Tout d'abord, on observe que la courbe est stable et d'après les valeurs de RMSE en fonction de similarité on peut noter que la plus petite valeur est :0.8716974. Puis, d'après ces chiffres on observe une augmentation des valeurs de RMSE jusqu'à ce qu'elle atteigne une valeur maximale =3.293693 avec une similarité de 0.8.

IV.1.3.Expérimentation 3 :

Tableau 3: Valeurs de similarité par rapport au schema

Num séries	Ic 1	2	3	4	5	6	7	8	9
Valeur similarité	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9

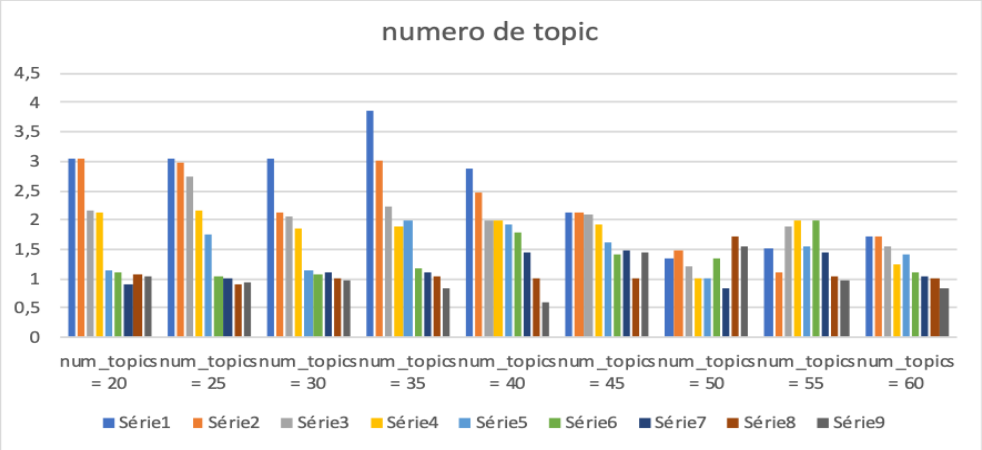


Figure 29: Histogramme de RMSE en fonction de numéro de topic

Commençant tout d'abord par le groupe où num_topics = 20 ; en observant les barres on remarque que :

- La valeur de RMSE pour une similarité égale à 0.2 est la plus élevée.
- Elle est suivie de la similarité 0.1 Pour RMSE égale à 3.03046.
- À l'inverse pour une similarité égale à 0.7 RMSE=0.918236 est la plus petite.
- Aussi plus la similarité augmente ; on remarque une diminution de RMSE.

Puis dans le groupe où num_topics = 25, une lecture attentive des chiffres révèle :

- Une dégradation de RMSE en fonction de la similarité.
- La plus grande valeur de RMSE est 3.052825 représentée par la similarité0.1.
- À l'inverse pour une similarité égale à0.8, RMSE 0.888920 est la plus petite.
- Aussi plus la similarité augmente ; on remarque une diminution de RMSE.

Ensuite dans le groupe où num_topics = 30 ; en observant les barres on note :

- La valeur de RMSE pour une similarité égale à 0.1 est la plus élevée.
- On note une diminution de RMSE.
- Suivie d'une stabilité des valeurs de RMSE qui sont très proches les unes des autres.

- La plus petite valeur de RMSE qu'elle peut atteindre est 0.97888 pour une similarité 0.9.

Puis dans le groupe où num_topics = 35 ; en observant les barres on peut extraire :

- La valeur de RMSE pour une similarité égale à 0.1 est la plus élevée.
- Elle est suivie d'une diminution des valeurs de RMSE.
- À l'inverse pour une similarité égale à 0.9 RMSE 0.850344 est la plus petite.

Après, pour le groupe où num_topics = 40 ; en observant les barres on remarque que :

- La valeur de RMSE pour une similarité égale à 0.1 est la plus élevée.
- Elle est suivie d'une diminution des valeurs de RMSE jusqu'à ce qu'elle atteigne un minimum égal à 0.59433 pour une similarité= 0.9.
- Aussi on peut noter que les valeurs de RMSE sont très proches.

Puis pour le groupe où num_topics = 45 ; en observant les barres on remarque que :

- La plus grande valeur de RMSE est 2.143256 pour une similarité de 0.1.
- À l'inverse pour une similarité égale à 0.9 RMSE = 0.99732 est la plus petite.
- L'allure générale montre une stabilité du graphique.

Ensuite le groupe où num_topics =50 ; on peut noter :

- Une instabilité des valeurs de RMSE et les valeurs de cette dernière sont proches les unes des autres.
- La plus grande valeur de RMSE est 1.715752 pour une similarité de 0.9
- À l'inverse la plus petite valeur est 0.8491

Puis pour le groupe où num_topics = 55 ; en observant les barres on note :

- Un balancement des valeurs de RMSE.
- La plus grande valeur est 1.98452 pour une similarité = 0.6
- Cependant la similarité 0.9 donne RMSE = 0.97123 représente la plus petite valeur.

Enfin dans le groupe où num_topics = 60 ; d'après ces barres on observe :

- Une diminution progressive des valeurs de RMSE en fonction de la similarité.
- La plus petite valeur atteinte est 0.85168 pour une similarité de 0.9.
- À l'inverse le maximum de RMSE est 1.72683 pour une similarité égale à 0.1.

IV.1.4.Expérimentation 4 :

Tableau 4: échantillons des tests avec C1 variable

itération>=	C1 = 0.01	C1 = 0.05	C1 = 0.10	C1 = 0.20	C1 = 0.30	C1 = 0.40	C1 = 0.50
10	0,7013581	0,71681516	0,7945686	0,8491513	0,81681688	0,96817468	1,01681381
75	0,6618168	0,5268516	0,51681357	0,76916814	0,62165135	0,83165165	0,89637125
200	0,51518256	0,5056331	0,51023202	0,61123459	0,53518133	0,71616516	0,70235615

Après le réglage des paramètres de la méthodes LDA, on passe l'apprentissage de l'algorithme d'optimisation par essaim de particules.

Le tableau suivant représente la variation de RMSE de fonction du nombre d'itération et en modifiant C1, dans cette expérimentation Num_topic = 50, passes = 10, alpha =0.15, eta= 0.1, C2= 0.1, W= 0.1, Nb Particule= 50.

On remarque une diminution de l'erreur sur toutes les valeurs de C1 en fonction de nombre d'itération.

IV.1.5.Expérimentation 5 :

Dans cette expérimentation je vais vous présenter un tableau de variations de RMSE en fonction de nombre d'itération d'algorithme et en modifiant aussi la valeur du paramètre C2, donnant des valeurs constantes pour : num_topics = 50, passes = 10, alpha =0.15, eta = 0.1, similarité = 0.8 , C1= 0.1, W=0.1 , NB particul =50.

Tableau 5: échantillons des tests avec C2 variable

Iteration>=	C2 = 0.01	C2 = 0.05	C2 = 0.10	C2 = 0.20	C2 = 0.30	C2 = 0.40	C2 = 0.50
10	1,01126515	1,31265131	1,00238468	1,03889195	0,99813153	1,03816531	0,97621637
75	0,72135135	0,79685135	0,71365165	0,63225364	0,70984265	0,76153894	0,81223543
200	0,5065135	0,50011651	0,52361657	0,5016153	0,51165132	0,50036847	0,64784465

On remarque une diminution de l'erreur sur toutes les valeurs de C2 en fonction de nombre d'itération.

IV.1.6.Expérimentation 6 :

Le tableau suivant représente la variation de RMSE en fonction du nombre d'itération et en modifiant les valeurs du paramètre *W*, dans cette expérimentation : num_topics = 50, passes = 10, alpha =0.15, eta = 0.1, similarité = 0.8, C1= 0.1, C2=0.1, NB particul =50.

Tableau 6 : échantillons des tests avec W variable

iteration>=	W = 0.01	W = 0.05	W = 0.10	W = 0.20	W = 0.30	W = 0.40	W = 0.50
10	1,06545356	1,24549835	1,2384681	1,389195	0,99598531	1,816531	0,9813668
75	0,72384651	0,75168965	0,78915133	0,68161321	0,69181651	0,73012005	0,64932513
200	0,52816486	0,51516843	0,52819561	0,53185165	0,50894621	0,57065465	0,51846533

On remarque une diminution de l'erreur sur toutes les valeurs de W en fonction de nombre d'itération.

IV.1.7.Expérimentation 7 :

Le tableau suivant représente la variation de RMSE en fonction du nombre d'itération et en modifiant le nombre de particules, dans cette expérimentation : num_topics = 50, passes = 10, alpha =0.15, eta = 0.1, similarité = 0.8, C1= 0.1, C2=0.1, W=0.1.

Tableau 7 : échantillons des tests avec num part variable

Iteration>=	NB part = 10	NB part = 25	NB part =50	NB part =75	NB part = 100	NB part = 150	NB part = 200
10	0,69967412	0,59181682	0,63651532	0,64915123	0,59018515	0,59545614	0,51635435
75	0,51281853	0,50216065	0,50016511	0,50094351	0,50139788	0,50039158	0,41003695
200	0,43486651	0,4619526	0,47165117	0,43183514	0,43161652	0,4005351	0,36079336

D'abord on remarque que les valeurs sont faiblement augmentées par rapport à la meilleure valeur des expérimentations précédentes en suite en consultant le tableau en aperçoit que les résultats RMSE diminuent en fonction de nombre d'itération et nombre de particules utilisé.

V. Conclusion :

Selon les différentes expérimentations effectuées dans le cadre de ce travail on peut conclure que le filtrage collaboratif basé item est une méthode efficace.

Cependant, elle peut être améliorée en la combinant avec d'autres approches. En outre, l'utilisation de la méthode LDA et de l'optimisation par essaim de particules nécessite le réglage de plusieurs paramètres. Ainsi, le bon choix des paramètres peut améliorer les résultats.

Finalement, la combinaison du filtrage collaboratif basé item avec la méthode LDA et l'algorithme d'optimisation par essaim de particules améliore considérablement la qualité des prédictions.

Conclusion et perspectives

Les systèmes de recommandation ont reconnu une grande importance dans les différentes applications web pour aider les utilisateurs à sélectionner les items et produits qui correspondent à leurs besoins. L'objectif de notre travail, est de proposer un système de recommandation de films où le calcul des prédictions se base d'une part sur l'application de la méthode LDA et l'intégration de cinq informations contextuelles, en l'occurrence le temps d'évaluation et la durée du film ainsi que l'âge, le sexe et la profession de l'utilisateur. D'autre part, il utilise le degré d'importance de chaque paramètre contextuel calculé en se basant sur l'optimisation par essaim de particules.

Ce travail a montré comment la combinaison de la méthode LDA et l'optimisation par essaim de particules peut être utilisé pour améliorer le filtrage collaboratif basé item. Cependant, le temps d'apprentissage de ces deux méthodes entrave l'amélioration des résultats.

La performance des méthodes appliquées dans les systèmes de recommandation dépend d'un ensemble de données à l'autre et d'un domaine à l'autre, ce qui nous mène à envisager l'application du modèle proposé dans d'autres domaines utilisant les systèmes de recommandation comme la recommandation dans les réseaux sociaux pour en tester leur validité.

Références

[1] Burke R., “Hybrid Recommender Systems: Survey and Experiments”, User modeling and user-adapted interaction, Vol. 12, N°. 4, pp. 331 - 370, 2002.

[2] Candillier L., Jack K., Fessant F., Meyer F., “State-of-the-art recommender systems”, Collaborative and Social Information Retrieval and Access: Techniques for Improved User Modeling, 22 pages, 2009.

[3] Goldberg K., Roeder T., Gupta D., Perkins C., “Eigentaste: A Constant Time Collaborative Filtering Algorithm”, Information Retrieval, Vol. 4, N°. 2, pp. 133 - 151, 2001.

[4] Liu H., Liang Z., “Implicit Rating Model in M-Commerce Recommendation System”, In: Proceedings of the international conference on Computational Intelligence and Software Engineering, pp.1 - 4, 2009.

[5] Pazzani M. J., “A framework for collaborative, content-based and demographic filtering”, Artificial Intelligence Review, Vol.13, N°. 5, pp. 393 - 408, 1999.

[6] Nguyen A.T., Denos N., Berrut C., “Modèle d’espaces de communautés basé sur la théorie des ensembles d’approximation dans un système de filtrage hybride”, Conférence en Recherche Information et Applications, pp. 303 - 314, 2006.

[7] Trousse B., “Evaluation of the Prediction Capability of a User Behaviour Mining Approach for Adaptive Web Sites”, In: Proceedings of the 6th RIAO Conference - Content-Based Multimedia Information Access, 2000.

[8] Brown P. J., “Triggering information by context”, personnal Technologies, Vol. 2, N°.1, pp.19 - 29, 1998.

[9] Schilit B. N., Adams N. I., Want R., “Context-Aware Computing Applications”, In: Proceedings of the Workshop on Mobile Computing Systems and Applications, IEEE Computer Society, Santa Cruz, CA, pp. 85 - 90, 1994.

[10] Dey A. K., “Understanding and using context”, Personal and ubiquitous computing, Vol. 5, pp. 4 - 7, 2001.

[11] Xiaohang W., “The context gateway: a pervasive computing infrastructure for context aware services”, Technical report, National university of Singapore, 2003.

[12] Chaari T., Laforest F., “Génération et adaptation automatiques des interfaces utilisateurs pour des environnements multi-terminaux”, Revue Ingénierie des Systèmes d'Information, N°. spécial Systèmes d'information pervasifs, Vol. 2, 2004.

[13] Ding Y., Li X., Orlowska M. E., “Recency-based collaborative filtering”, In: Proceedings of the 17th Australasian Database Conference, Australian Computer Society, Vol. 49, pp. 99 - 107, 2006.

[14] Negre E., “Comparaison de textes : Quelques approches”, Rapports techniques, Université Paris-Dauphine, 2013.

[15] Huang A., “Similarity Measures for Text Document Clustering”, In: Proceedings of the Computer Science Research Student Conference, New Zealand, pp. 49 - 56, 2008.

[16] Majda , MAATALLAH. Une Technique Hybride pour les Systèmes de Recommandation . thèse présentée en vue de l’obtention du diplôme de Doctorat 3ème Cycle en Informatique. 2016.

[17] Middleton S. E., Alani H., Shadbolt N. R., De Roure C., (2002). Exploiting Synergy Between Ontologies and Recommender Systems. Proc. Of the 11th Inter. WWW Conf., Workshop on the Semantic Web.

[18] Mohammed, TADLAOUI. Système de recommandation de ressources pédagogiques fondé sur les liens sociaux : formalisation et évaluation. THESE de DOCTORAT DE L'UNIVERSITE DE LYON opérée au sein de L'INSA Lyon. Lyon, 2018.

[19] Sobecki J., Szczepanski L., "Wiki-News Interface Agent Based on AIS Methods", In: Proceedings of the First KES International Symposium on Agent and Multi-Agent Systems: Technologies and Applications, Wroclaw, Poland, pp. 258 - 266, 2007.

[20] Clerc M., Siarry P., "Une nouvelle métaheuristique pour l'optimisation difficile: la méthode des essaims particulaires", J3eA, Vol. 3, 2004.

[21] Ujjin S., Bentley P. J. "Particle swarm optimization recommender system", In: Proceedings of the IEEE Swarm Intelligence Symposium, pp. 124 - 131, 2003.

[22] Bakshi S., Jagadev A. K., Dehuri S., Wang G., "Enhancing scalability and accuracy of recommendation systems using unsupervised learning and particle swarm optimization", Applied Soft Computing journal, Vol. 15, pp. 21 - 29, 2014.

[23] Diaz-Aviles E., Georgescu M., Nejdl W., "Swarming to rank for recommender systems", In: Proceedings of RecSys'12 of the 6th ACM Conference on Recommender Systems, pp. 229 - 232, 2012.

[24] Zhang H., You F., "A Novel Fuzzy Clustering Recommendation Algorithm Based on PSO", Cybernetics and Information Technologies, Vol. 14, N°. 5, pp. 108 - 117, 2014.

[25] Alam S., Dobbie G., Riddle P., "Towards Recommender System Using Particle Swarm Optimization Based Web Usage Clustering", New Frontiers in Applied Data Mining, Vol. 7104, pp. 316 - 326, 2012.

[26] Alam S., Dobbie G., Riddle P., Koh Y. S., “Hierarchical PSO Clustering Based Recommender System”, In: Proceedings of the IEEE World Congress on Computational Intelligence, Brisbane, Australia, 2012.

[27] Abdelwahab A., Nishida M., “An Efficient Collaborative Filtering Algorithm using SVD-free Latent Semantic Indexing and Particle Swarm Optimization”, In: Proceedings of the International Conference on Natural Language Processing and Knowledge Engineering, pp. 1 - 4, 2009.

[28] Abdelwahab A., Sekiya H., Matsuba I., Horiuchi Y., Kuroiwa S., “Feature optimization approach for improving the collaborative filtering performance using particle swarm optimization”, Journal of Computational Information Systems, Vol. 8, N°. 1, pp. 435 - 450, 2012.

[29] Anandakumar K., Rathipriya K., Bharathi A., “A Web Based Recommendation System for Personal Learning Environments Using Hybrid Collaborative Filtering Approach”, International Journal of Innovative Research in Science, Engineering and Technology, Vol. 3, N°. 9, 2014.

[30] Al-Shamri M. Y. H., Bharadwaj K. K., “Fuzzy-genetic approach to recommender systems based on a novel hybrid user model”, Expert Systems With Applications, Vol. 35, N°. 3, pp. 386 -1399, 2008.

[31] Dao T. H., Jeong S. R., Ahn H., “A novel recommendation model of location-based advertising: Context-Aware Collaborative Filtering using GA approach”, Expert Systems with Applications, Vol. 39, N°. 3, pp. 3731 - 3739, 2012.

[32] Hwang C., Chang T., “Genetic K-Means Collaborative Filtering for Multi-Criteria Recommendation”, Journal of Computational Information Systems, Vol. 8, N°. 1, pp. 293 - 303, 2012.

[33] Kanchana J. S., Sujatha S., “Recommendation Engine Formation Using Depth First Search and Genetic Approach”, Journal of Computer Science, Vol. 11, N°. 1, pp. 188 - 194, 2015.

[34] Ujjin S., Bentley P. J., “Learning User Preferences Using Evolution”, In: Proceedings of the 4th Asia-Pacific Conference on simulated Evolution and Learning, Singapore. 2002.

[35] Fong S., Ho Y., Hang Y., “Using Genetic Algorithm for Hybrid Modes of Collaborative Filtering in Online Recommenders”, In: Proceedings of the eighth International Conference on Hybrid Intelligent Systems, pp. 174 - 179, 2008.

[36] Bobadilla J., Ortega F., Hernando A., Alcalá J., “Improving collaborative filtering recommender system results and performance using genetic algorithms”, Knowledge-Based Systems, Vol. 24, N°. 8, pp. 1310 - 1316, 2011.

[37] Navgaran D. Z., Moradi P., Akhlaghian F., “Evolutionary based matrix factorization method for collaborative filtering systems”, In: Proceedings of the 21st Iranian Conference on Electrical Engineering, pp. 1 - 5, 2013.

[38] Salehi M., Fathi A., Abdali-Mohammadi F., “ANTSREC: A Semantic Recommender System Based on Ant Colony Meta-Heuristic in Electronic Commerce”, International Journal of Advanced Science and Technology, Vol. 56, 2013.

[39] Bellaachia A., Alathel D., “Trust-Based Ant Recommender (T-BAR)”, In: Proceedings of the 6th IEEE International Conference on Intelligent Systems, pp. 130 - 135, 2012.

[40] Bedi P., Sharma R., Kaur H., “Recommender system based on collaborative behavior of ants”, Journal of Artificial Intelligence, Vol. 2, N°. 2, pp. 40 - 55, 2009.

[41] Caicedo-Castro I., Duarte-Amaya H., “CodeRAnts: A recommendation method based on collaborative searching and ant colonies, applied to reusing of open source code”, Ingeniería E Investigación, Vol. 34, N°. 1, pp. 72 - 78, 2014.

[42] Paranjape-Voditel P., Thakare A., “Use of Sampling and Ant Colony Optimization for predicting support in Recommender System”, In: Proceedings of the Science and Information Conference, London, pp. 7 - 9, 2013.

[43] Nadi S., Saraee M., Bagheri A., Jazi M. D., “FARS: fuzzy ant based recommender system for web users”, International Journal of Computer Science Issues, Vol. 8, N°. 1, pp. 203 - 209, 2011.

[44] Kaleroun A., Shalini B., “Collaborating trust and item-prediction with ant colony for recommendation”, In: Proceedings of the Seventh International Conference on Contemporary Computing, pp. 334 - 339, 2014.

[45] Adomavicius, G., Tuzhilin, A. Contextaware recommender systems. In Recommender systemshandbook, pages 217-253. Springer

[46] Dey A. K., “Understanding and using context”, Personal and ubiquitous computing, Vol. 5, pp. 4 - 7, 2001.

[47] Borras, J., Moreno, A., Valls, A. Intelligent tourism recommender systems : A survey. Expert Systems with Applications, 41(16) :7370-7389.

[48] Palmisano, C., Tuzhilin, A., Gorgoglione, M. Using context to improve predictive modeling of customers in personalization applications. IEEE transactions on knowledge and data engineering, 20(11) :1535-1549.

[49] Griffths, T. L. & Steyvers, M. Finding scientific topics. Proceedings of the National academy of Sciences, 101(suppl 1) :5228-5235.

[50] Wei, X., Croft, W. B. Lda-based document models for ad-hoc retrieval. In Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval, pages 178-185. ACM.

[51] Ionescu, R. T., Chifu, A. G., Mothe, J. DeShaTo: describing the shape of cumulative topic distributions to rank retrieval systems without relevance judgments. In International Symposium on String Processing and Information Retrieval (pp. 75-82). Springer International Publishing.

[52] Yu, K., Zhang, B., Zhu, H., Cao, H., Tian, J. Towards personalizedcontext-aware recommendation by mining context logs through topic models. In Pacific-Asia Conference on Knowledge Discovery and Data Mining, pages 431-443. Springer

Resumé:

Le travail présenté dans ce manuscrit se situe dans le domaine des systèmes de recommandation qui est devenu une méthodologie dominante dans la majorité des applications Web. L'objectif de notre travail est de présenter les systèmes de recommandation, les principales techniques de recommandation et leurs limitations. Nous avons également exposé des méthodes bio-inspirées et leurs applications dans les systèmes de recommandation ainsi que les différents travaux de recherche dans ce domaine. On a présenté notre système de recommandation à base d'optimisation par essaim de particules sur une base de données MovieLens. Le modèle proposé permet l'intégration des informations contextuelles sur les utilisateurs et les items afin d'améliorer la qualité des prédictions. La première étape implémente le modèle LDA à partir des descriptions des items que les utilisateurs ont consultés. Ce travail a montré comment la combinaison de la méthode LDA et l'optimisation par essaim de particules peut être utilisé pour améliorer le filtrage collaboratif basé item.

Mots clés :

Système de recommandation, Filtrage collaboratif, essaim de particules, LDA, python.

Abstract

The work presented in this manuscript is in the area of recommender systems which has become a dominant methodology in the majority of web applications. The objective of our work is to present recommendation systems, the main recommendation techniques and their limitations. We also exposed bio-inspired methods and their applications in recommender systems as well as the various research works in this field. We presented our recommendation system based on particle swarm optimization on a MovieLens database. The proposed model allows the integration of contextual information on users and items in order to improve the quality of predictions. The first step implements the LDA model from the descriptions of the items that users have viewed. This work has shown how the combination of LDA method and particle swarm optimization can be used to improve item-based collaborative filtering.

Keywords :

Recommendation system, Collaborative filtering, particle swarm, LDA, python.

ملخص:

العمل المقدم في هذه المخطوطة هو في مجال أنظمة التوصية التي أصبحت منهجية سائدة في غالبية تطبيقات الويب. الهدف من عملنا هو تقديم أنظمة التوصية وتقنيات التوصية الرئيسية وقيودها. لقد كشفنا أيضًا عن الأساليب المستوحاة من الحيوية وتطبيقاتها في أنظمة التوصية بالإضافة إلى الأعمال البحثية المختلفة في هذا المجال. قدمنا نظام التوصية الخاص بنا بناءً على تحسين سرب الجسيمات في قاعدة بيانات .MovieLens يسمح النموذج المقترح بدمج المعلومات السياقية حول المستخدمين والعناصر من أجل تحسين جودة التنبؤات. تنفذ الخطوة الأولى نموذج LDA من أوصاف العناصر التي شاهدها المستخدمون. أظهر هذا العمل كيف يمكن استخدام مزيج من طريقة LDA وتحسين سرب الجسيمات لتحسين التصفية التعاونية القائمة على العناصر.

الكلمات الدالة:

نظام التوصية ، الترشيح التعاوني ، سرب الجسيمات ، LDA، Python.