TABLE DES MATIERES

INTRODUCTION GENERALE 1

CHAPITRE 1 : Les Systèmes de Recommandation 6
1.1. Introduction 7
1.2. Définition et fonctionnement des Systèmes de Recommandation 7
1.3. Historique des Systèmes de Recommandation 8
1.4. Filtrage d'information Vs. Rechechre d'Information 9
1.5. Concepts de base, Notation, et Notions liées 10
1.5.1. Les entités Utilisateur et Items 11
1.5.2. Evaluation (Note ou Vote) 11
1.5.3. Matrice d'évaluation Utilisateur-Item 12
1.5.4. Notion de communauté 12
1.5.5. Notion du profil 13
1.5.6. La prédiction 14
1.5.7. La recommandation 15
1.5.8. La personnalisation 15
1.6. La Formalisation des Systèmes de Recommandation 15
1.6.1. Formalisation du problème de la recommandation 15
1.6.2. La formalisation des méthodes collaboratives 16
1.6.2.1. Les algorithmes basés mémoire (memory-based algorithms) 16
1.6.2.2. Les algorithmes basés modèle (model-based algorithms) 17
1.6.3. La formalisation des méthodes basées sur le contenu 18
1.7. Les techniques de recommandation 18
1.7.1. Le Filtrage basé sur le Contenu 18
1.7.1.1. Représentation des items (profil Item) 19
1.7.1.2. Méthodes pour l'apprentissage du profil de l'utilisateur 21
1.7.1.3. Avantages 27
1.7.1.4. Limites 27
1.7.1.5. Exemple d'un système de recommandation basé sur le contenu 28
1.7.2. Le Filtrage Collaboratif 29
1.7.2.1. La méthode basée modèle 29
1.7.2.2. La méthode basée mémoire (basée voisinage) 37
1.7.2.3. Avantages des méthodes collaboratives 47
1.7.2.4. Limites des méthodes collaboratives 48
1.7.2.5. Exemples de systèmes de recommandation basés sur le FC 49
1.7.3. Le Filtrage démographique 50
1.7.3.1. Exemple d'un système de recommandation basé sur le filtrage démographique 50
1.7.4. Le Filtrage basé utilité 50
1.7.5. Le Filtrage basé connaissance 51
1.7.5.1. Avantages 52
1.7.5.2. Limites 52
1.7.5.3. Exemple de systèmes de recommandation basés connaissance 52
1.7.6. Les systèmes de recommandation hybrides 53
1.7.6.1. L'hybride pondéré (Weighted) 54
1.7.6.2. L'hybride par commutation (Switching) 54
1.7.6.3. L'hybride mixte (Mixed) 55
1.7.6.4. L'hybride par combinaison de caractéristiques (Feature combination) 55
1.7.6.5. L'hybride en cascade 56
1.7.6.6. L'hybride par augmentation de caractéristiques (Feature augmentation) 56

1.7.6.7. L'hybride de Méta-niveau (Meta-level) 57
1.7.6.8. Des exemples de systèmes de recommandation hybrides 57
1.8. Les problèmes des Systèmes de Recommandation 58
18.1. Le problème du démarrage à froid (Cold Start Problem) 59
1.8.2. Le problème de la Stabilité Vs. Dynamicité (Stability Vs. Plasticity Problem) 62
1.8.3. Les systèmes statiques 62
1.8.4. Le problème de la parcimonie de données (Sparcity problem) 63
1.8.5. Le problème de l'évolutivité de données 64
1.8.6. Le problème de la sur-spécialisation 64
1.8.7. Bilan des approches et des problèmes inhérents aux Systèmes de Recommandation 64
1.9. La prédiction dans les Systèmes de Recommandation 65
1.9.1. La prédiction basée Modèle (basée factorisation matricielle) 66
1.9.2. La prédiction basée Mémoire 66
1.9.2.1. L a prédiction basée utilisateur 66
1.9.2.2. La prédiction basée item 67
1.10. Evaluation des Systèmes de Recommandation 68
1.10.1. Métriques d'évaluation de la prédiction 68
1.10.1.1. Erreur Moyenne Absolue [Mean Absolute Error MAE] 68
1.10.1.2. Erreur Quadratique Moyenne [Root Mean Squared Error RMSE] 68
1.10.1.3. Erreur moyenne Absolue Normalisée [Normalized MAE NMAE] 69
1.10.2. Métriques d'évaluation des recommandations (Top-N) 69
1.10.2.1. Précision 69
1.10.2.2. Rappel 70
1.10.2.3. La métrique F1 70
1.10.2.4. La mesure d'évaluation de la diversité 70
1.11. Critères d'évaluation pour les Systèmes de Recommandation 70
1.11.1.Les préférences de l'utilisateur 71
1.11.2.Précision de la prédiction 71
1.11.3.La Couverture 72
1.11.3.1.Couverture de l'espace d'items 72
1.11.3.2.Couverture de l'espace utilisateur 73
1.11.4. La confiance (Confidence) 73
1.11.3. La confiance (Trust) 74
1.11.4. La Sérendipité (Le hasard) 74
1.11.5. La Nouveauté 76
1.11.6. La Diversité 76
1.11.7. L'Utilité 77
1.11.8. La Robustesse 78
1.11.9. La Confidentialité 78
1.11.10. L'Aadaptabilité 79
1.11.12. L'évolutivité 79
1.12. Synthèse des approches de recommandation 80
1.13. Conclusion 81

CHAPITRE 2 : Travaux Connexes 83
2.1. Introduction 84
2.2. Les Systèmes de Recommandation 84
2.3. Le clustering dans le Filtrage Collaboratif 88
2.3.1. La méthode du Lissage basé cluster(Cluster-based smoothing) 88
2.3.2. L'algorithme FCM dans les Systèmes de Recommandation 89
2.3.3. La méthode de factorisation matricielle dans les systèmes de recommandation 91
2.3.4. Synthèse des travaux sur le custering dans le FC 96
2.3.5. Discussion 97
2.4. La diversité dans la Recommandation 100
2.4.1. Définition 100

2.4.2. Métriques de Diveristé 100
2.4.2.1. La Diversité Individuelle (Individual Diversity) 100
2.4.2.2. La Diversité Globale (Aggregate Diversity) 101
2.4.3. Algorithmes de diversification du contenu des recommandations 101
2.4.4. Synthèse des approches sur la diversité 120
2.4.5. Disscussion 124
2.5. Conclusion 125

CHAPITRE 3 : Un Système de Recommandation Hybride et Diversifiée 127

Partie I : Un Système de Recommandation Hybride Diversifiée basée Flou 128
3.1. Introduction 129
3.2. Un Système de Recommandation Hybride Diversifiée basée Flou 129
3.2.1. Objectifs et Motivations 129
3.2.2. Proposition 132
3.2.3. Processus de Recommendation Hybride Diversifiée basée flou 133
3.2.4. Notations 135
3.2.5. L'algorithme du Filtrage Collaboratif-basé Flou 136
3.2.5.1. L'algorithme du clustering-basé flou: Modified FCM to NMF (MFCM-NMF) 136
3.2.5.2. Identification du voisinage 140
3.2.5.3. La prédiction basée sur le filtrage collaboratif (CF-based prediction) 143
3.2.5.4. Algorithme du Filtrage Collaboratif basé flou 144
3.2.6. L'algorithme du Filtrage basé Contenu 145
3.2.6.1. Profil item (Item Profile) 145
3.2.6.2. Préférence des caractéristiques (Features' preferences) 146
3.2.6.3. Sélection des T-plus proches items (T-Nearest Items Selection) 147
3.2.6.4. La prédiction basée sur le contenu 148
3.2.6.5. Algorithme de Filtrage basé Contenu 148
3.2.7. La prédiction hybride 149
3.2.8. La recommandation Top-K basée sur la prédiction hybride 149
3.2.9. La recommandation Top-N Diversifiée 149
3.3.9.1. Les préférences des clusters 150
3.3.9.2. La recommandation Top-N 151
3.3.9.3. La recommandation Top-K 151
3.3.9.4. Algorithme de la Recommandation Top-N diversifiée basée flou 152
Partie II: Un Système de Recommandation Hybride Diversifiée basée sur le Niveau de Similarité 154
3.3. Un Système de Recommandation Hybride Diversifiée basée sur le Niveau de Similarité 155
3.3.1. Motivations et Objectifs 155
3.3.2. Proposition 156
3.3.3. Le processus du Niveau de Similarté 157
3.3.4. Recommandation Top-K Diversifiée basée sur le niveau de similarité 158
3.3.4.1. Génération de la recommandation similaire 159
3.3.4.2. Génération de la recommandation diversifiée basée sur le niveau de similarité 160
3.3.5. Algorithme de la recommandation Top-K diversifiée basée sur le Niveau de similarité 164
3.4. Une Nouvelle Approche Hybride pour la Recommandation Top-K (Une Nouvelle Approche de Recommandation Hybride Diversifiée basée Flou et Niveau de Similarité) 165
3.4.1. Algorithme de Recommandation Top-K diversifiée basée Flou et Niveau de Similarité 167
3.5. Conclusion 167

CHAPITRE 4 : Expérimentations 169

4.1. Introduction 170
4.2. Objectifs des expérimantions 170

4.3. Dataset 171
4.4. Métriques d'évaluation 171
4.4.1. Précision de l'algorithme de factorisation 171
4.4.2. Métrique d'évaluation de la prédiction basée modèle (RMSE) 172
4.4.3. Métrique d'évaluation de la prédiction (MAE) 172
4.4.4. Métriques d'évaluation des recommandations (Top-N) 172
4.4.4.1. les mesures de la précision 172
4.4.4.2. la mesure de la diversité 173
Partie I: DATASET 173
4.5. Expérimentation 1: Evaluation de l'Approche de Recommandation Hybride basée Flou 173
4.5.1. Evaluation de l'algorithme du Filtrage Collaboratif basé Flou 173
4.5.1.1. Performance de l'algorithme du clustering MFCM-NMF 174
4.5.1.2. Performance de l'algorithme de la sélection du voisinage 178
4.5.1.3. Performance de la prédiction basée sur le Filtrage Collaboratif Flou 181
4.5.1.4. Impact de l'évolutivité de données (Scalability Impact) 182
4.5.2. Evaluation de la prédiction hybride basée flou 183
4.5.2.1. Choix du paramètre Alpha 184
4.5.2.2. Performance de la prédiction hybride 185
4.5.2.3. Evaluation de la performance de la recommandation basée prédiction 186
4.5.3. Evaluation de la recommandation Top-K Diversifiée basée flou 187
4.5.3.1. Performance générale de la recommandation Top-K basée flou 188
4.5.3.2. Diversité Vs. Performance de la recommandation hybride basée flou 190
4.5.3.3. Performance de la recommandation avant Vs. après la prédiction et Impact de la clairsemée de données 190
4.5.4. Discussion des résultats 191
4.6. Expérimentation 2: Evalaluation de l'approche de diversité hybride basée sur le niveau de similarité 193
4.6.1. Test de l'ensemble des items divers à sélectionner 193
4.6.2. Comparaison avec l'approche hybride basée flou 194
4.6.3. Comparaison avec les méthodes de diversification de l'état de l'art 195
4.6.4. Discussion des résultats 197
4.7. Expérimentation 3: Evaluation de la nouvelle approche de recommandation hybride diversifiée basée flou et niveau de similarité 198
4.7.1. Test de la méthode de fusion 198
4.7.2. Performance de la nouvelle méthode hybride de diversité 200
4.7.3. Discussion des résultats 201
Partie II: ETUDE DE CAS (Scénario E-learning) 201
4.8. Intégration des approches proposées dans la plateforme *Moodle* 201
4.8.1. *Moodle* Dataset 202
4.8.2. Principales interfaces du Système 202
4.8.3. Evaluation et Résultats 208
4.8.4. Discussion 216
4.9. Conclusion 217

CONCLUSION GENERALE 216
REFERENCES BIBLIOGRAPHIQUES 220
ANNEXE A. Méthodes de classification non supervisées (FCM et NMF) 235
ANNEXE B. MovieLens Dataset 245

LISTE DES FIGURES

Figure 1.1 : Un exemple illustratif de la prédiction des évaluations manquantes 15
Figure 3.1 : Processus de la Recommandation diversifiée hybride basée flou.....134
Figure 3.2 : Exemple illustratif de la méthode d'initialisation rand Rrow 139
Figure 3.3 : Illustration de la notion du Niveau de Similarité.....158
Figure 3.4 : Processus de la recommandation diversifiée basée sur le niveau de similarité.....159
Figure 3.5 : Processus de la Nouvelle Approche de Recommandation Hybride Diversifiée basée Flou et Niveau de Similarité.....166
Figure 4.1. Valeur optimal des clusters K 174
Figure 4.2. Valeurs optimales des paramètres λz et λc.....175
Figure 4.3. Nombre maximal d'itérations *maxiter*.....176
Figure 4.4: RMSE pour différents algorithmes de clustering (petite valeur, meilleur performance) ..177
Figure 4.5: Valeur Optimale des plus proches clusters flous 179
Figure 4.6: Valeur Optimale des plus proches voisins flous.....179
Figure 4.7: Temps d'exécution pour le CCP Vs. la nouvelle mesure de similarité 180
Figure 4.8: MAE pour la prédiction basée FC Flou.....181
Figure 4.9: FC basé flou Vs. SF1 182
Figure 4.10: Impact de l'évolutivité 183
Figure 4.11: MAE des prédictions basée FC Flou et basée Contenu 184
Figure 4.12: Alpha vs. MAE avec différents ensembles de données 185
Figure 4.13: MAE des prédictions basée FC, BC et hybride 186
Figure 4.14: Performance de la recommandation basée prédiction 187
Figure 4.15: Performance de la Recommandation Top-K diversifiée basée flou Vs. Recommandation Top-N Ordinaire.....188
Figure 4.16: Comparaison des performances de la Recommandation Top-K.....189
Figure 4.17: Diversité Vs. Performance de la Recommandation Top-K basée Flou.....190
Figure 4.18: Comparaison de la performance de la Recommandation Top-K basée sur la prédiction 191
Figure 4.19: Comparaison de la performance des deux méthodes de sélection des items divers 194
Figure 4.20: Comparaison de la recommandation hybride floue avec la recommandation diversifiée basée sur le niveau de similarité.....195
Figure 4.21: Comparaison de la Diversité avec les méthodes de diversification de l'état de l'art 196
Figure 4.22: Comparaison de F1 avec les méthodes de diversification de l'état de l'art 196
Figure 4.23: Comparaison des méthodes d'hybridation 199
Figure 4.24: Performance de la nouvelle approche hybride.....200
Figure 4.25: Connexion de l'administrateur 205

Figure 4.26: Interface pour l'administrateur 205
Figure 4.27: Ajout d'une activité ou d'une ressource 206
Figure 4.28: Ajout d'une activité 206
Figure 4.29: Gestion des utilisateurs 207
Figure 4.30: Gestion des cours et catégories 207
Figure 4.31: Permission au bloc Administration 208
Figure 4.32: Connexion utilisateur 209
Figure 4.33: Liste des cours disponibles pour l'apprenant 209
Figure 4.34: Consultation des cours Ontologie 210
Figure 4.35: Evaluation du cours Ontologie 210
Figure 4.36: Déconnexion 211
Figure 4.37: Performance de la Recommandation basée flou sur le Dataset *Moodle* 212
Figure 4.38: Performance de la Recommandation Diversifiée basée sur le Niveau de Similarité sur le Dataset *Moodle* 213
Figure 4.39: Recommandation basée Flou Vs. Recommandation diversifiée basée sur le niveau de similarité sur le Dataset *Moodle* 214
Figure 4.40: Performance de la nouvelle approche hybride sur le Dataset *Moodle* 215

LISTE DES TABLEAUX

Tableau 1.1 : Bilan des approches et des problèmes inhérents aux SRs 65

Tableau 1.2 : Synthèse des approches de recommandation 80

Tableau 2.1 : Synthèse des travaux sur le clustering dans le FC..... 96

Tableau 2.2 : Synthèse de notre revue de la littérature sur la diversité dans les SRs 121

Tableau 4.1: Comparaison avec les méthodes d'initialisation 177

ACRONYMES

Pour des raisons de lisibilité, la signification d'un acronyme ou d'une abréviation n'est en général rappelée que lors de sa première utilisation dans le texte d'un chapitre. Par ailleurs, nous employons le terme français ou le terme anglais suivant l'usage le plus répandu.

SR	Systèmes de Recommandation
RI	Recherche d'Information
FC	Filtrage Collaboratif
FBC	Filtrage Basé Contenu
FM	Factorisation Matricielle
NMF	Factorisation en Matrice NonNégative
MFCM-NMF	Modified Fuzzy C-means to Non negative Matrix Factorization
SVD	Singular Value Decomposition
PMF	Probabilistic Matrix Factorization
ACLS	Alternating Constrained Least Squares
ALS	Alternating Least Squares
MFCM	Modified Fuzzy C-Means
FCM	Fuzzy C-Means
XFCM	eXponential FCM
ONMTF	Orthogonal Nonnegative Matrix Tri-Factorization
GRWNMF	Graph Regularized Weighted NonNegative Matrix Factorization
KPPV	K-Plus Proches Voisins
CPPVF	C-Plus Proches Clusters Flous
KPPVF	K-Plus Proches Voisins Flous
KPLV	K-Plus Loins Voisins
CCP	Coefficient de Corrélation de Pearson
SC	Similarité Cosinus

CPC	Corrélation de Pearson avec Contrainte
DQM	Différence Quadratique Moyenne
CCA	Corrélation Cosinus Ajustée
CRS	Coefficient du Rand du Spearman
SCF	Synthetically Collaborative Filtering
MAE	Mean Absolute Error
RMSE	Root Mean Squared Error
NMAE	Normalized Mean Absolute Error

INTRODUCTION GENERALE

Les Systèmes de Recommandations **[RES et VAR, 97]** identifient automatiquement les préférences des utilisateurs à travers leurs interactions avec le système en se basant sur le feedback implicite **[HU *et al.*, 08] [SAL 13] [SAL et *al.*, 13]** ou feedback explicite **[BOB *et al.*, 10]** ou les deux ensembles **[SHA et Sum, 13] [SAL et KAM, 13],** pour leur suggérer des recommandations en utilisant le filtrage d'information.

Pour générer des recommandations, un certain nombre d'approches ont été identifiées **[BUR 02],** où les plus utilisées sont le Filtrage Collaboratif **[BAL et SHO, 97]** et la Filtrage basé Contenu **[MOO et ROY, 99] [PAZ et BIL, 07**]. Les deux techniques ont leurs forces et faiblesses **[BUR 02],** où l'hybridation entre eux a été rapidement adoptée pour profiter de leurs avantages.

L'un des problèmes majeurs des systèmes de recommandation est le problème de la stabilité de ces systèmes par rapport au profil dynamique de l'utilisateur (Stability vs. Dynamicity Problem) **[BUR 02].** Ce problème revient au faite que si l'utilisateur s'intéresse à plusieurs items différents en même temps, comme il peut alterner ses préférences au fil du temps, et si son profil est créé dans le système, il devient difficile de changer ses préférences et prendre en considération ses différents choix et préférences. Cela limite la capacité des systèmes de recommandation à suivre l'évolution du profil de l'utilisateur et de s'adapter à ses différents choix et préférences, et par la suite lui recommander des items qui ne correspondent pas à ses différents choix et intérêts, ce qui conduit à un manque de diversité dans les listes de recommandation. En outre, les systèmes de recommandation, spécialement ceux utilisant le Filtrage Collaboratif souffrent du problème de l'évolutivité **[BUR 02],** lors de l'ajout d'un nouvel utilisateur ou d'un nouvel item, et souffre aussi du problème de la clairsemé de donnée [**BUR 02],** en raison des grandes matrices utilisateur-item contenant des évaluations dispersées.

L'objectif de la méthode du Filtrage Collaborative est de générer des suggestions d'items aux utilisateurs en utilisant les préférences de leurs voisins, en se basant sur des méthodes et métriques pour regrouper les utilisateurs et trouver l'ensemble des voisinages pertinents à travers l'utilisation des mesures de similarité, telles que la mesure de Corrélation de Pearson et la Corrélation Cosinus. Bien que, les mesures de similarité classiques calculent la similarité

entre l'utilisateur actif et l'ensemble des utilisateurs dans le système sans considérer leurs préférences ambigües et dispersées, ce qui nécessite des temps de calculs énormes et conduit à avoir un ensemble non pertinent de voisins. Une idée naturelle est de sélectionner les voisins les plus proches à partir des groupes similaires en utilisant des mesures de similarité efficaces pour réduire les calculs

En raison de la dispersion et l'imprécision dans les préférences des utilisateurs, les méthodes de la classification dure ne traitent pas efficacement l'ambiguïté dans les préférences de l'utilisateur, ce qui rend les méthodes floues plus appropriées à ces situations. L'algorithme des C-moyennes Floues (Fuzzy C-Means FCM) **[BUZ 73]** est l'une des méthodes qui ont montré des améliorations dans la performance du Filtrage Collaboratif **[FAN et LIU, 03][WAN *et al.*, 10][REN *et al.*, 11],** parce qu'il prend en considération l'incertitude et l'ambiguïté dans le comportement de l'utilisateur en lui affectant à différents clusters avec des probabilités appropriées.

Cependant, au cours des dernières années, de nouvelles qualités d'un bon système de recommandation ont été présentées dans la littérature, en plus de la performance des prédictions. Un système de recommandation efficace doit offrir des items nouveaux et divers aux utilisateurs, qui répondent à leurs différents intérêts et préférences, ce qui nécessite l'élaboration de nouvelles idées et techniques pour formuler des recommandations d'intérêt. Ainsi, les recommandations intéressantes devraient contenir des items divers et pertinents mais en considérant la performance du système, qui diminue inversement avec l'augmentation de la diversité.

Afin de trouver une solution au problème de la stabilité des systèmes de recommandation par rapport au profil dynamique de l'utilisateur et pour offrir des recommandations diversifiées aux utilisateurs, tout en considérant les effets de l'évolutivité et de la clairsemé de données, nous présentons dans cette thèse un système de recommandation hybride capable de générer des recommandations diversifiées qui répondent aux différents choix et intérêts de l'utilisateur, en intégrant deux approches distinctes, et la troisième sera une hybridation entre eux.

En raison de l'imprécision et des changements dans le profil de l'utilisateur, la première approche proposée [**MAA et SER, 12a**] **[MAA et SER, 15]** est une approche hybride qui combine entre le Filtrage Collaboratif et le Filtrage basé sur le contenu. La méthode du Filtrage Collaboratif proposée permet aux utilisateurs d'appartenir à différents groupes, ayant des intérêts similaires, avec des probabilités appropriées en élaborant un algorithme du Filtrage Collaboratif basé flou, qui combine entre l'algorithme FCM et la méthode de

factorisation en matrice non négative. Les utilisateurs actifs sont ensuite affectés aux groupes les plus similaires à l'aide des degrés d'appartenance. Ensuite, les voisins les plus proches sont sélectionnés en utilisant une nouvelle mesure de similarité basée sur la différence entre les degrés d'appartenance de l'utilisateur actif et les membres de groupes similaires. Pour augmenter la précision de la prédiction, l'algorithme du Filtrage Collaboratif basé flou est combiné avec un algorithme basé contenu pour prendre en considération le contenu des items dans le processus de la recommandation, tout en minimisant l'effet de la clairsemé de données.

Un algorithme de recommandation Top-N est ensuite présenté pour trouver les listes de recommandations dans les groupes similaires. Pour atteindre la diversité dans les listes de recommandation, le système génère des recommandations qui correspondent aux différents choix d'intérêt des utilisateurs actifs, en sélectionnant les Top-K items à partir de chaque liste générée pour construire la liste de recommandation finale. Le nombre d'items sélectionnés K est variable d'un cluster à l'autre, et il est calculé en fonction du degré d'appartenance de l'utilisateur à chaque plus proche cluster.

Afin de d'augmenter la diversité dans les listes de recommandation et sélectionner plus d'items nouveaux par rapport à ceux déjà vus par l'utilisateur, nous devons diversifier de plus en plus les contenus des recommandations et de suggérer des items qui ne sont jamais vus ni par l'utilisateur actif, ni son voisinage. Cependant, Recommander des items divers et très dissimilaires aux préférences de l'utilisateur, peut ne pas être utile pour l'utilisateur qui reçoit des items différents par rapport à ses choix et préférences. Afin de répondre à cette contrainte, nous avons proposé une nouvelle méthode de diversification basée sur le niveau de similarité entre l'utilisateur actif et les items, afin de juger l'importance des items divers recommandés aux utilisateurs.

La deuxième contribution vise à améliorer la diversité dans les systèmes de recommandation en proposant un algorithme de diversification des items recommandés, en regroupant les items non évalués par l'utilisateur en catégories. Ensuite, des poids jugeant la pertinence d'une catégorie par rapport à l'utilisateur actif sont calculés. La génération de divers items est effectuée en sélectionnant les items les plus pertinents à l'utilisateur à partir de la catégorie la plus pertinente, dont la pertinence est calculée sous forme d'un poids pour juger quel item sera sélectionné.

On propose dans cette deuxième partie, un schème hybride qui combine entre un processus de génération de recommandation similaire et un processus de recommandation d'items divers. Le premier processus se base sur un algorithme du Filtrage Collaboratif ordinaire

pour trouver les items similaires aux préférences de l'utilisateur actif pour assurer la précision des recommandations. Alors que le deuxième processus utilise une nouvelle méthode de diversification ayant comme but de sélectionner des items dissimilaires mais pertinents pour l'utilisateur en se basant sur une nouvelle formule pour calculer la pertinence des items.

La formule calcule le degré d'intérêt de l'item pour l'utilisateur en se basant sur une nouvelle notion, qui est la notion de niveau de similarité entre l'utilisateur et les items dissimilaires, en termes d'existence de voisin des voisins de l'utilisateur qui ont évalué les items. Ces degrés sont utilisés afin de juger la pertinence des items pour les utilisateurs en évitant par conséquence de recommander des items qui ne seront jamais intéressants pour l'utilisateur. Afin d'achever les objectifs attendus d'avoir une liste de recommandations avec des items divers et pertinents, tout en évitant la dégradation de la performance, les deux listes générées avec les deux processus sont fusionnées.

Pour équilibrer le compromis entre la précision et la diversité, et différemment des travaux de la littérature qui utilisent un paramètre pour contrôler le compromis entre la précision et la diversité, nous avons hybridé entre les deux approches proposées dans cette thèse, FC basé sur le flou et la méthode de diversification basée sur le niveau de similarité, comme troisième approche proposée.

Une partie importante de la thèse est dévolue au développement et à l'évaluation des différentes approches et notions proposées au sein d'un système de recommandation des films et un système d'apprentissage en ligne. Une série d'expérimentations est menée afin de prouver l'efficacité des approches proposées à rendre le système adaptatif aux changements des profils des utilisateurs, et leur capacité à augmenter la diversité dans les listes de recommandation, en générant des items divers et pertinents qui correspondent aux différents choix et intérêts des utilisateurs.

STRUCTURE DE LA THESE

Le mémoire est organisé en quatre chapitres comme suit :

- Dans le ***premier chapitre,*** nous présentons la théorie des systèmes de recommandation, ses origines, les concepts de base et notions liées, sa formalisation, les différentes approches et problèmes liés, en terminant par la présentation des métriques d'évaluation et la définition des nouveaux critères d'évaluation pour les systèmes de recommandation.

- Le ***deuxième chapitre*** est divisé en deux parties, *la première partie* contient un état de l'art sur les méthodes de *clustering* dans les systèmes de recommandation

basés sur le Filtrage Collaboratif en spécifiant les deux méthodes célèbres qui ont prouvé leur efficacité dans le domaine des recommandations : la méthode du Fuzzy C-Means pour le *clustering* flou et la méthode de Factorisation Matricielle.

La deuxième partie est consacrée à la définition du problème de la diversité dans les systèmes de recommandation, et la citation de différents algorithmes et métriques présentés dans la littérature visant à améliorer la diversité des contenus suggérés aux utilisateurs.

- ***Le troisième chapitre***, présente nos contributions visant à améliorer la qualité des recommandations générées en termes de diversification des contenus des recommandations, et à pallier aux problèmes des systèmes de recommandation en adressant en particulier les problèmes de la stabilité des systèmes, la clairsemé et l'évolutivité des données.

- ***Le quatrième chapitre***, présente l'élaboration des méthodes proposées au sein d'un système de recommandation sur la base de données de *MovieLens* et sur la plateforme d'apprentissage en ligne *Moodle*, en vue de prouver leurs efficacités. Les performances sont évaluées qualitativement et quantitativement avec différents tests et comparaisons.

A l'issue des études présentées, nous concluons avec un bilan de nos propositions, en ouvrant la porte vers certaines perspectives envisagées pour accomplir les lacunes des approches proposées et améliorer leurs performances.

CHAPITRE 1

LES SYSTEMES DE RECOMMANDATION

RESUME DU CONTENU

Dans ce chapitre, nous présentons de façon détaillée le domaine des systèmes de recommandation, en mettant l'accent sur les concepts de bases et les techniques du filtrage existantes, ainsi qu'une série des métriques d'évaluation de ces systèmes. Les nouveaux critères d'évaluation des systèmes de recommandation seront également présentés à la fin de ce chapitre.

1. 1. Introduction

Le développement du Web a créé un besoin de nouvelles techniques pour aider les utilisateurs à trouver ce qu'ils recherchent mais aussi pour faire savoir qu'une information existe, ces techniques sont appelées ***les Systèmes de Recommandation*** (SRs).

Une définition des systèmes de recommandation a été donnée dans ce chapitre ainsi que les fondements, concepts de base et notions liées au domaine. Ensuite, en se basant sur la manière dont les recommandations sont formulées, les principaux types de techniques de recommandation [**BUR 02**] sont identifiés, en précisant le *Filtrage Collaboratif* (*FC*) et le *Filtrage basé Contenu (FBC)*, ainsi que les problèmes dont souffrent ces systèmes. Une panoplie de mesures de similarité et d'évaluation a été présentée, ainsi que les nouveaux critères d'évaluation pour les systèmes de recommandation.

1. 2. Définition et fonctionnement des Systèmes de Recommandation

Les Systèmes de Recommandation (SR) **[RES et VAR, 97]** identifient automatiquement les préférences des utilisateurs à travers leurs interactions avec le système, en se basant sur le feedback implicite **[Hu *et al.*, 08] [SAL 13] [SAL *et al.*, 13]**, le feedback explicite **[BOB *et al.*, 10],** ou les deux feedbacks **[SHA et SUM, 13] [SAL et KAM, 13],** pour leur suggérer des recommandations en utilisant le filtrage d'information.

Le filtrage d'information est l'expression utilisée pour décrire une variété de processus dédiés à la fourniture de l'information adéquate aux personnes qui en ont besoin [**BEL et CRO, 92**]. Son but est de sélectionner et suggérer aux utilisateurs, à partir de larges volumes d'informations générés dynamiquement, les informations jugées pertinentes pour eux. Par conséquent, le filtrage d'information peut être vu aussi comme étant le processus d'élimination de données indésirables sur un flux entrant, plutôt que la recherche de données spécifiques sur ce flux.

Le processus du filtrage d'information commence lors de l'utilisation du système par des personnes ayant des désirs et des objectifs qui sont stables, à long terme ou périodiques. Par conséquent, cela conduit à des besoins réguliers en information, qui peuvent évoluer lentement au cours du temps au fur et à mesure que les conditions, les

objectifs et les connaissances changent. Le filtrage est basé sur des descriptions des individus et des groupes appelées *profils*, qui représentent généralement leurs intérêts à long terme, afin de répondre à leurs besoins. De tels intérêts ou profils engagent les utilisateurs dans un processus passif de recherche d'information, car ils reçoivent les informations à partir du système sans avoir à les rechercher.

D'un autre côté, les producteurs des items disponibles en ligne (incluant documents, services ou produits) prennent en charge la distribution de leurs produits dès qu'ils sont générés. Pour accomplir cette tâche, les items sont associés à une représentation de leur contenu appelée *profil item*, qui est ensuite comparée aux profils des utilisateurs ou des groupes.

Par la suite, les utilisateurs consultent les items suggérés et expriment leurs avis sur eux en les évaluant par rapport à leur réponse aux besoins des utilisateurs. Cette évaluation peut mener à la modification des profils des utilisateurs et des domaines de leurs intérêts.

Afin de générer des recommandations, un certain nombre d'approches de filtrage ont été présentées dans la littérature **[BUR 02],** où les plus utilisées sont le Filtrage Collaboratif (CF) **[BAL et SHO, 97],** et le Filtrage basé sur le Contenu (FBC) **[MOO et ROY, 99] [PAZ et BIL, 07].** Les deux techniques ont des forces et des faiblesses **[BUR 02],** où l'hybridation entre eux a été rapidement adoptée pour profiter de leurs avantages

1. 3. Historique des Systèmes de Recommandation

La notion du filtrage d'information a vu le jour dans le domaine de *la Recherche d'Information* (RI) [**BEL et CRO, 92**] quand la tendance s'est tournée vers la *personnalisation* des résultats présentés à l'utilisateur à la suite d'une requête. Puis, l'intérêt progressif porté à cette notion de filtrage et l'exploitation des méthodes de fouilles de données (*Data Mining)* dans ce contexte, a donné naissance aux systèmes de recommandation comme des systèmes voisins, mais indépendants des systèmes de RI du fait que les SRs sont spécialisés beaucoup plus dans le traitement des *profils à long terme* (les requêtes dans la RI représentent des *profils à court terme*).

Le premier système de recommandation, a été introduit par Goldberg [**GOL *et al*., 92**] et nommé *Tapestry*. Deux ans plus tard, les chercheurs du *Grouplens*[1] ont présenté leur premier Système de Recommandation [**RES et VAR, 97**] en parallèle avec le système

[1] http://www. grouplens. org

Ringo [**UPE 94**]. Jusqu'à l'année 1997, les auteurs ont utilisé le terme « Filtrage Collaboratif » au lieu de « Système de Recommandation ». Cette dernière appellation a été stabilisée en cette année par Resnick et Varian [**RES et VAR 97**].

L'année 1997 a reconnu également l'apparition du premier SR hybride '*Fab'* créé par Balavonic et Shoham **[BAL et SHO, 97],** et qui combine le Filtrage basé sur le Contenu avec le Filtrage Collaboratif.

Ensuite, en 2001, la notion du FC basé item a été introduite par Sarwar *et al*. [**SAR *et al.*, 01**] et elle a étendu le champ de popularité des SRs du secteur académique vers le secteur commercial. Cette approche a été exploitée plus tard par Linden [**LIN *et al.*, 03**] dans le portail Web d'Amazon.com[2].

1. 4. Filtrage d'information vs Recherche d'information

La différence entre le filtrage d'information et la recherche d'information n'est pas toujours claire; mais une première différence peut être tirée de leur définition. Généralement, un système de recherche d'information est un système ayant pour fonction d'apporter aux utilisateurs les documents qui leur permettent de satisfaire leurs besoins en information, en se basant sur les requêtes qu'ils fournissent [**BEL et CRO, 92**], tandis qu'un système de filtrage d'information fournit des documents (ou des items) à une personne ou ensemble de personnes, en se basant sur leurs profils à long terme [**CRO 93**].

Un système de recherche d'information a pour but de guider les utilisateurs à retrouver les items qui leur permettent de répondre à leurs besoins, alors qu'un système de filtrage leur recommande directement les items intéressants qui conviennent à leurs préférences et besoins.

Dans un système de recherche d'information, l'utilisateur saisie une requête et le système la compare avec les sources d'information disponibles, pour trouver un ensemble de documents répondant à sa requête. Les sources d'information utilisées par le système de recherche d'information sont un ensemble de données collectées dans des bases de données et souvent indexés par un ensemble de mots clés.

Dans un système de filtrage d'information, le système compare l'ensemble des informations disponibles aux profils des individus ou groupe d'individus, afin de leur suggérer les items\ et ou informations correspondants à leurs besoins. Dans ces

[2] www.amazon.com

systèmes, les items\ et ou informations proviennent directement de la part des producteurs ou bien collectés dans des bases de données.

Par conséquent, un certain nombre de points de distinction entre la recherche d'information et le filtrage d'information peuvent être cités:

- Dans un système de recherche d'information, les données sont sélectionnées à partir d'une base de données statique, alors que le filtrage d'information établit une sélection ou bien une élimination d'information à partir d'une source de données dynamique.
- Un système de recherche d'information ne peut traiter qu'une seule requête à la fois, ce qui le rend capable d'être utilisé que par une seule personne à un moment donné, tandis qu'un système de filtrage peut être utilisé par une ou plusieurs personnes en même temps.
- Un système de recherche d'information collecte et organise les documents selon la recherche des individus, tandis qu'un système de filtrage d'information distribue automatiquement des documents\ et ou des items aux personnes qui en ont besoin.
- Un système de recherche d'information utilise le principe de la découverte de l'information dans la base de données alors que le filtrage d'information utilise un processus d'élimination d'information du flux de données entrant.
- Un système de recherche d'information utilise les requêtes fournies par l'utilisateur qui représentent des intérêts à court terme. Par contre, le filtrage d'information utilise les profils des utilisateurs qui représentent généralement des intérêts à long terme, et qui peuvent être considérés comme des descriptions plus exactes des intérêts des utilisateurs que les requêtes.
- Un système de recherche d'information autorise l'utilisateur à interagir avec un document pendant une seule session de recherche, alors que le filtrage d'information permet l'interaction dans différentes sessions et prend en considération les changements de profils à travers les différentes sessions.

1. 5. Concepts de base, Notation et Notions liées

Dans cette section, nous définissons quelques concepts relatifs aux systèmes de recommandation, qui seront utilisés dans cette thèse.

1. 5.1. Les entités Utilisateur et Item

Dans tout système de recommandation, il existe deux entités importantes qui sont les utilisateurs et les items.

- ✓ *Utilisateur* : est une personne qui accède au système est fait l'enregistrement, en saisissant ses informations démographiques, ses centres d'intérêts et d'autres informations personnelles. L'ensemble des utilisateurs dans le système est représenté par U, où un utilisateur donné $u \in U$.
- ✓ *Item* : dans les systèmes de recommandation, un item est l'entité qui représente tout élément constituant une liste de recommandation et qui correspond aux besoins de l'utilisateur, incluant tout produit susceptible d'être vendu (livre, produits,…ect dans les sites du e-commerce tel que Amazon.com), vu (les films dans les sites de TV en ligne tel que Netflix), écouté (la musique) ou lu (tel que les informations dans les journaux en ligne, les revues dans les bibliothèques numériques), ainsi que les destinations de vacance, des restaurants, etc.

  Notons qu'un item peut être aussi un individu ou un ensemble d'individus suggérés à l'utilisateur dans les réseaux sociaux. L'ensemble des items disponibles dans le système est représenté par I, où $i \in I$.

1. 5.2. Evaluation (Note ou Vote)

Une évaluation est une valeur numérique dans une échelle quelconque (la plus utilisée est [1-5]) ou binaire (aimer\ Ne pas aimer, bon\ mauvais, etc.) qui représente la préférence ou non d'un item donné par un utilisateur. L'évaluation donné par un utilisateur *u* à un item *i* est représenté par $r_{u,i}$ ou par un triplet <*u, i, r*>. Où, une note de 5, par exemple, exprime une grande préférence et une note de 1 indique une faible préférence i.e. l'utilisateur n'a pas aimé l'item.

Une note peut être attribuée directement par un utilisateur à un item en donnant une valeur numérique ou binaire à travers l'interface du système appellée *évaluation explicite* **[BUR 02].** En outre, les préférences de l'utilisateur peuvent être déduites par le système en utilisant des algorithmes et techniques spécifiques **[REN *et al.*, 09] [LEE *et al.*, 08],** et dans ce cas appelée *évaluation implicite* **[OAR et KIM, 98] [BUR 02][KEL et TEE, 03].**

1. 5.3. Matrice d'évaluation utilisateur-item

L'ensemble de tous les triplets du système <*u, i, r*> sont enregistrés dans une base de données creuse appelée *Matrice d'Evaluation* (*Rating Matrix*) ou encore *Matrice utilisateur-item* (*user-item Matrix*) et elle est notée par *R*, où chaque ligne correspond aux évaluations fournies par un seul utilisateur et une colonne correspond aux évaluations qu'a eu un seul item par l'ensemble des utilisateurs.

La matrice d'évaluation utilisateur-item est l'entrée pour les systèmes de recommandation et la base des techniques du FC, qui utilisent les préférences (votes) pour la génération des recommandations. Afin d'améliorer la performance des systèmes de recommandation, plusieurs recherches ont apparues offrant de nouveaux scénarios pour introduire de nouvelles informations à la matrice d'évaluation **[SHI et HAN, 14].** Ces informations peuvent être divisées en deux catégories, selon leurs sources ou bien leurs associations à l'interaction avec le système.

Le premier type est celui de l'information littérale riche sur les utilisateurs (genre, âge, loisirs,… ect) ou sur les items (catégorie, contenu,…ect) **[SHI et HAN, 14],** qui est devenue importante et très utilisée surtout dans les réseaux sociaux et les technologies du Web 2.0 (tags, commentaire, contenu multimédia) **[SHI et HAN, 14].**
Le deuxième type d'information correspond à l'information associée à l'interaction de l'utilisateur avec le système, incluant le temps d'évaluation où d'achat des items **[KOR 09],** l'emplacement (local) de l'utilisateur ainsi que les commentaires et les avis des utilisateurs **[PHE *et al.*, 11][GUR et GAS, 13][LIU *et al.*, 10][LEE *et al.*, 14].** Une étude détaillée sur les scénarios impliquant des sources d'information dans les systèmes de recommandation et les techniques utilisées est présentée dans **[SHI et HAN, 14].**

1. 5.4. Notion de communauté

Une communauté ou groupe est un ensemble d'utilisateurs similaires qui partagent les mêmes préférences et goûts, et qui sont regroupés relativement à un critère donné. Plusieurs critères peuvent être utilisés par le processus de formation des communautés, citant par exemple le contenu des items évalués par les utilisateurs, les évaluations qu'ils ont données aux items, leurs données démographiques et leur domaine d'intérêt **[BOU 05].** Selon chacun de ces critères, les communautés créées par le système varient et les positions des utilisateurs dans ces communautés changent. Par conséquent, chaque

utilisateur peut appartenir à autant de communautés qu'il y a de critères utilisés pour leur formation **[NGU *et al.*, 06][NGU *et al.*, 06a].**

1. 5.5. Notion du profil

De façon générale, le profil d'un objet est un ensemble de caractéristiques permettant de l'identifier ou de le représenter. Deux types de profils peuvent être exploités dans les systèmes de recommandation, correspondant aux deux entités utilisées dans ces systèmes : l'utilisateur et l'item.

1) *Le profil utilisateur*: il s'agit d'une description des caractéristiques de l'utilisateur qui peuvent être ses centres d'intérêt, ses données démographiques, ou bien ses préférences exprimées sous forme d'évaluations, etc. Plusieurs approches d'acquisition des informations sur l'utilisateur afin de construire son profil existent, et peuvent être regroupées en approches manuelles et approches automatiques ou semi-automatiques **[BUR 02].** Les approches manuelles se fondent sur l'intervention de l'utilisateur, alors que les approches automatiques déduisent automatiquement le profil de l'utilisateur. Parmi les approches automatiques ou semi-automatiques, nous pouvons distinguer le '*profiling'* **[CHO *et al.*, 02]** et les approches par stéréotypes [**SHA *et al.*, 97**].

   Le '*profiling'* consiste à examiner, analyser et enregistrer les actions et successions d'actions d'un utilisateur lors des différentes sessions de recherche et d'interaction avec le système pour déterminer son profil. En revanche, l'approche par stéréotype est fondée sur l'identification des groupes d'utilisateurs et la détermination des critères de chaque groupe. Un stéréotype d'un utilisateur dans un système de recommandation consiste en un vecteur d'items et leurs appréciations qui augmentent d'une façon continue tant que l'utilisateur interagit avec le système tout au long du temps [**BUR 02**].
2) *Le profil item* : il correspond à la description de l'item avec un ensemble de caractéristiques, appelées aussi attributs ou propriétés, par exemple dans un système de recommandation de films, les items (films) sont représentés par leurs Ids, titre, genre, réalisateur, année de production, les acteurs principaux. Tandis que dans un système de recommandation de document, les caractéristiques sont des mots clés qui décrivent le contenu sémantique du document. Ces mots clés et leurs poids sont obtenus en général par une opération d'indexation [**RIJ 79**].

Les documents peuvent être représentés également par d'autres informations qui ne sont pas liées à leur contenu appelées métadonnées, telles que celles utilisées dans la *Dublin Core* pour décrire les documents (titre, auteur, sujet, etc.). De plus, les métadonnées peuvent se présenter sous forme de propriétés liées aux documents (type du document, profession de l'auteur, etc.), ou juste à des parties des documents (style, forme discursive, type d'unité documentaire, etc.) [**CRU 99**], afin de sélectionner les documents qui décrivent le mieux le sujet qu'ils traitent, et qui sont réellement utilisables et exploitables. Les métadonnées peuvent être utilisées aussi pour l'annotation qualitative des documents [**BER 02**].

1. 5.6. La Prédiction

La prédiction est le calcul de la note probable que l'utilisateur va attribuer à un item qu'il n'a pas encore vu ou évalué.

En général, les matrices d'évaluation ont seulement quelques cellules contenant des valeurs tandis que les autres ont des valeurs inconnues et dans la majorité des cas elles ont à l'intérieur un"0", ce qui donne des matrices creuses. Donc, la densité de ces matrices ne sera pas suffisante pour générer des recommandations précises. Par conséquent, les méthodes de prédiction des évaluations manquantes sont utilisées pour augmenter la densité de la matrice utilisateur-item en vue de faire des recommandations plus puissantes et plus pertinentes.

Le calcul de la prédiction se base sur l'utilisation des notes données par les voisins de l'utilisateur (*prédiction basée utilisateur*) ou attribuées aux items voisins de l'item test évalué par l'utilisateur actif (*prédiction basée item*), ou bien donné par un modèle (*prédiction basée modèle*). Ensuite, les items ayant les plus grandes valeurs de prédictions seront recommandés à l'utilisateur. Un petit fragment de la matrice d'évaluation est présenté dans la Figure 1.1 afin d'illustrer le rôle de la prédiction dans les systèmes de recommandation (sur une échelle de 1-5 et les ? indique les valeurs inconnues).

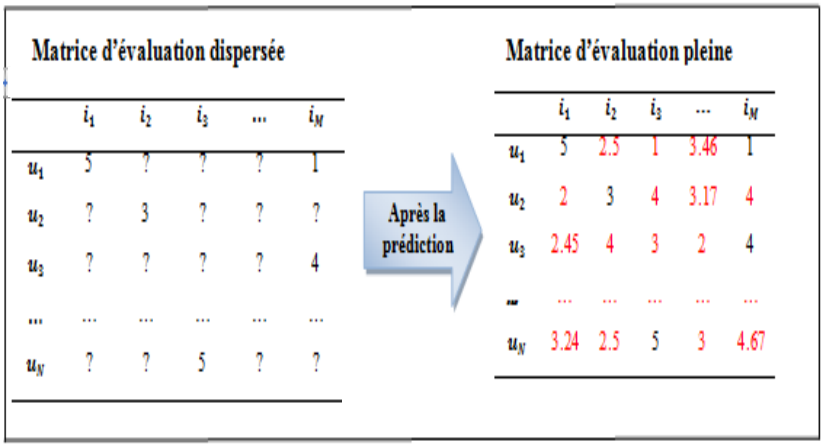


Figure 1.1. Un exemple illustratif de la prédiction des évaluations manquantes

La note prédite d'un item *i* pour l'utilisateur *u* peut être référencée par $\hat{r}_{u,i}$.

1. 5.7. La Recommandation

La recommandation est l'action de calculer une liste d'items (Top-N items) que l'utilisateur aimera le plus. Le calcul des listes de recommandation se fait en attribuant des scores pour les items selon leurs popularités ou leurs préférences **[WEN *et al.*, 08]**, par exemple. Contrairement à la prédiction, le calcul des recommandations ne se base pas strictement sur les évaluations.

1. 5.8. La Personnalisation

La personnalisation est une notion proche de la notion de recommandation mais elle est moins générale et elle consiste à adapter un item aux goûts, aux besoins et parfois même au comportement de l'utilisateur. Tandis qu'une recommandation génère une liste d'items plus ou moins adaptée aux besoins de l'utilisateur (c.à.d. elle ne garantie pas une personnalisation totale parce que les listes recommandées peuvent contenir des items nouveaux pour l'utilisateur ou différents de ces préférences, pour améliorer la satisfaction comme nous allons le voir plus tard dans ce chapitre).

1. 6. La formalisation des Systèmes de Recommandation

1.6.1. Formalisation du problème de la recommandation

La formalisation courante du problème de la recommandation se réduit à l'estimation de l'évaluation d'un item qui n'a pas encore été vu par l'utilisateur, en se basant sur ses

évaluations passées aux autres items ainsi que quelques informations additionnelles dépendantes du SR. Etant capable d'estimer ces évaluations, le SR pourra alors présenter à l'utilisateur les items qui se trouveront avec les meilleurs évaluations estimées.

D'une façon plus formelle : Soit U l'ensemble de tous les utilisateurs dans le système, I l'ensemble de tous les items, et soit *utility* une fonction d'utilité qui mesure l'utilité d'un item $i \in I$ pour un utilisateur $u \in U$, telle que: $utility : U \times S \rightarrow R$ où R est un ensemble totalement ordonné, qui peut prendre la forme la plus simple qui soit binaire (0, 1) = (item non préféré, item préféré); ou qui peut être un ensemble valeurs numériques (réels par exemple) plus sensible à l'utilité des différents items. Donc, pour chaque utilisateur $u \in U$, on veut choisir certain item $I_u^{'}$ qui maximise la fonction d'utilité *utility* :

$$\forall u \in U, I'_u = argmax_{i \in I} utility(u, I) \qquad (1.1)$$

Les utilisateurs et les items dans un SR sont définis par un ensemble de caractéristiques qui contribuent, selon l'approche employée, à la définition des paramètres de la fonction *utility*.

1.6.2. La formalisation des méthodes collaboratives

Les méthodes collaboratives recommandent à un utilisateur les items qui ont été bien évalués par les utilisateurs ayant des préférences similaires, et qui n'ont pas encore été évalués par cet utilisateur.

Formellement : l'utilité *utility(u,i)* d'un item *i* pour un utilisateur *u* est estimée en se basant sur les utilités $utility(u_j,i)$ assignées à l'item *i* par l'ensemble des utilisateurs u_j similaires à *u*.

Selon [**BRE 98**] les algorithmes collaboratifs peuvent être séparés en deux catégories : les algorithmes basés sur la mémoire et les algorithmes basés sur le modèle.

1.6.2.1. Les algorithmes basés Mémoire [Memory-Based Algorithms]

Ces algorithmes, appelés aussi basés voisinage, contiennent eux-mêmes deux types de relations du voisinage qui les divisent en deux catégories: les algorithmes basés sur l'utilisateur et les algorithmes basé sur les items.

- ***Les algorithmes basés utilisateur [User-Based Algorithms]***

Dans ces algorithmes, la prédiction des évaluations pour un utilisateur actif se fait à la base des évaluations passées qu'a eu l'item et le degré de similarité des utilisateurs qui l'ont évalué avec cet utilisateur.

$$\hat{r}_{u,i} = \bar{r}_u + f \sum_{v=1}^{n} sim(u,v)(r_{v,i} - \bar{r}_v) \quad (1.2)$$

Où $\bar{r}_u$ est la moyenne des évaluations faites par l'utilisateur actif *u*, *f* est un facteur de normalisation, et *sim(u,v)* c'est le degré de similarité entre *u* et *v*.

- ***Les algorithmes basés Item [Item-Based Algorithms]***

Ces algorithmes prédisent les évaluations pour un item *i* sur la base des évaluations qu'ont eu les items les plus proches de cet item et le degré de similarité entre eux. Cette approche calcule la similarité entre les items en fonction des évaluations des utilisateurs. Ces algorithmes utilisent toutes les évaluations qu'ont eues les items *i*, *j* pour calculer le degré de similarité entre eux. La similarité entre les items *i* et *j* peut être calculée comme suit :

$$sim(i,j) = \frac{\sum_{u=1}^{n}(r_{u,i} - \bar{r}_i)(r_{u,j} - \bar{r}_j)}{\sqrt{\sum_{u=1}^{n}(r_{u,i} - \bar{r}_i)^2 (r_{u,j} - \bar{r}_j)^2}} \quad (1.3)$$

Où *u* est l'ensemble des utilisateurs qui ont évalué *i* et *j* ensemble.

En plus de cette classification, Sarwar [**SAR *et al.*, 01**] a introduit une autre classification, qui est la recommandation collaborative hybride qui combine les recommandations collaboratives, basée item et basée utilisateur.

1.6.2.2. Les algorithmes basés Modèle [Model-Based Algorithms]

Contrairement aux algorithmes basés sur la mémoire, seules les évaluations faites par l'utilisateur sont prises en compte dans les algorithmes basés sur le modèle afin de construire un modèle de prédiction des évaluations.

Par exemple, pour l'approche probabiliste, une échelle d'entiers pour les évaluations est assumée et définie comme suit :

$$\hat{r}_{u,j} = E(r_{u,j}) = \sum_{k=0}^{m} \Pr(r_{u,i} = k \mid r_{u,a}, a \in I_u) \quad (1.4)$$

1.6.3. La formalisation des méthodes basées sur le contenu

Les méthodes basées sur le contenu recommandent les items similaires à ceux préférés par l'utilisateur dans le passé. Pour calculer la similarité entre les différents items des profils, des vecteurs de propriétés des items sont utilisés pour représenter leurs caractéristiques. Les profils sont parfois définis par un vecteur de poids (w_{c1}, ..., w_{ck}) tel que chaque poids dénote l'importance d'un mot clés *ki* pour l'utilisateur. Cette information (les mots clés) étant textuelle, les méthodes utilisées pour calculer ces poids sont dérivées du domaine de la RI **[BEL et CRO, 92].** L'une des mesures les plus connues dans ce contexte est TF.IDF [*Term Frequency / Inverse Document Frequency*].

En se basant sur un tel vecteur de caractéristiques, la fonction d'utilité pour les systèmes basés sur le contenu est généralement définie par :

$$utility_{u,i} = score(ContentBasedProfile(u), Content(i)) \quad (1.5)$$

Tels que les deux profils peuvent être représentés par des vecteurs TF.IDF, $\vec{w}_u$ (décrivant le profil d'un utilisateur contenant les évaluations qu'il a fait précédemment) et $\vec{w}_i$ (l'ensemble des poids dans le profil de l'item).

1. 7. Les techniques de recommandation

Nous présentons dans cette section les principales techniques de recommandation citées dans la littérature, en mettant l'accent sur les deux techniques les plus utilisées : FBC et FC.

1.7.1. Le Filtrage Basé sur le Contenu

Le filtrage basé sur le contenu peut être vu comme un système de recherche d'information, dont la fonction de correspondance entre une requête et un corpus de documents joue le rôle d'un filtre permanent entre un profil (sorte de requête à long terme et évolutif) et le flot de documents entrant (sorte de corpus évolutif) **[BUR 02].**

Un système de filtrage basé contenu suit deux fonctionnalités principales:

- La sélection des items et\ ou documents pertinents vis-à vis du profil de l'utilisateur;

- La mise à jour du profil de l'utilisateur en fonction du retour de pertinence fourni par l'utilisateur sur les items et\ ou documents qu'il a reçus. La mise à jour se fait par l'intégration des caractéristiques des items et\ ou des thèmes abordés dans les documents jugés pertinents.

Le filtrage basé sur le contenu compare premièrement les items non évalués par l'utilisateur avec son profil, représenté par l'ensemble des items qu'il a évalués, en calculant la similarité entre eux, i.e. c'est une corrélation item à item [**SCH 99**]. Ensuite, il recommande les items les plus proches des préférences de l'utilisateur [**MON *et al.*, 03**]. Donc, le processus de filtrage basé sur le contenu nécessite deux constituants essentiels qui sont : les profils des items et les profils des utilisateurs, car les recommandations se génèrent en se basant sur la corrélation entre les profils de ces deux entités. Nous présentons ci-dessous les méthodes les plus utilisées dans la littérature pour la construction des profils utilisateurs et items dans les systèmes à base de FBC.

1.7.1.1. Représentation des items (Profil Item)

Comme nous l'avons déjà défini précédemment dans ce chapitre, un profil d'un item consiste en un ensemble de caractéristiques qui représentent le contenu sémantique de l'item, de la même façon que pour l'indexation dans le domaine de la recherche d'information. Donc, cette approche aussi dépend fortement de la nature des données à représenter et elle trouve des difficultés pour la représentation des items, ce qui diminue sa précision d'un type de ressource à l'autre.

Les données textuelles sont les mieux représentées et avec lesquelles la méthode aboutit à la meilleure précision, alors que les autres types de donnée, i.e. données non textuelles telles que les données sonores, vidéos et les images, donnent de mauvaises précision car il est difficile d'extraire les caractéristiques à partir de ces types de données. Généralement, dans le cas des données non textuelles, l'approche utilise des métadonnées pour représenter leur contenu.

La plupart des systèmes de recommandation basés sur le contenu utilisent des données textuelles **[ADO et TUZ, 05],** dont les caractéristiques sont extraites à partir des pages Web, des courriels, des articles de presse ou des descriptions de produits. Cependant, ces données textuelles créent un certain nombre de complications lors de l'apprentissage d'un profil utilisateur, en raison de l'ambiguïté du langage naturel **[LOP et GEM, 11]**,

du fait que les profils sont représentés avec des mots-clés traditionnels, et il est difficile de capturer la sémantique des intérêts des utilisateurs. Différentes méthodes d'extraction des caractéristiques et attributs à partir des documents afin de les représenter ont été présentées dans la littérature **[PAZ et BIL, 07] [LOP et GEM, 11]**, dont la plus utilisée est la technique du modèle vectoriel.

1.7.1.1.1. Le Modèle d'Espace Vectoriel

Le Modèle d'Espace Vectoriel (ou Modèle Vectoriel MV) **[SAL *et al.*, 75]** est une représentation algébrique des documents textes, dont chaque document est représenté par un vecteur dans un espace à *n* dimensions. Chaque dimension correspond à un terme du vocabulaire d'une collection de documents donnée, et elle contient le poids associé au terme, représenté par cette dimension, dans le document donné.

Formellement, chaque document *d* est représenté comme un vecteur de poids *w* à long terme, où chaque poids indique le degré d'association entre le document et le terme *t* **[LOP et GEM, 11]**.

Soit D = $\{d_1, d_2, ..., d_N\}$ désignent un ensemble de documents ou de corpus, et T = $\{t_1, t_2, ..., t_n\}$ soit le dictionnaire qui contient l'ensemble des mots dans le document ou le corpus. Chaque document d_j est représenté comme $d_j = \{w_{1j}, w_{2j}, ..., w_{nj}\}$, où w_{kj} est le poids associé au terme t_k dans le document d_j, qui exprime l'importance de ce terme dans le document d_j. Généralement, l'ensemble des termes T est obtenu en appliquant certaines techniques de traitement du langage naturel standard, telles que la *tokenisation*, l'enlèvement des mots vides, et la *racinisation* ou *désuffixation* (le *stemming*) **[BAE et RIB, 99].** Ensuite, les vecteurs de poids des termes sont calculés.

Etant donné que les termes sont des informations textuelles, les méthodes utilisées pour calculer les poids de ces termes sont dérivées du domaine de la RI **[BEL et CRO, 92],** dont le système de pondération des termes le plus couramment utilisé dans ce contexte est le TF-IDF [*Terme Frequency-Inverse Document Frequency*].

Avec le système TF-IDF, les termes qui apparaissent fréquemment dans un seul document (fréquence du terme TF) mais rarement dans le reste du corpus (inverse de la fréquence du document IDF), sont plus susceptibles d'être pertinents pour le thème du

document **[LOP et GEM, 11].** Le poids du terme *t* dans le document *d* est calculé comme suit :

$$w_{t,d} = TF_{t,d} \times IDF_t = \underbrace{\frac{f_{t,d}}{max_z(f_{z,d})}}_{\mathbf{TF}} . \underbrace{log\,\frac{N}{n_t}}_{\mathbf{IDF}} \quad (1.6)$$

Où, $TF_{t,d}$ décrit l'importance du terme *t* dans le document *a* et IDF_t calcule la force discriminante du terme (dans tous les documents). $TF_{t,d}$ est défini par la fréquence du terme individuel dans un document ($f_{t,d}$) normalisé par la fréquence maximale de tous les termes dans *d*. IDF_t est défini par la fonction logarithmique du rapport entre *N* et n_t tel que *N* représente le nombre total de tous les documents et n_t est le nombre de documents contenant *t*.

A la fin de cette étape, le système obtient une liste des termes avec leurs poids indiquant leur importance pour chaque document. Afin de calculer la similarité entre les documents (ou vecteur de documents), la mesure de similarité la plus utilisée est la Similarité Cosinus (définie en plus bas dans ce chapitre).Un bon état de l'art sur les systèmes de recommandation basés contenu qui ont utilisé cette technique a été présenté dans **[LOP et GEM, 11]**.

L'inconvénient de cette technique est qu'elle ne considère pas la sémantique des termes. Cependant, de nouvelles techniques pour retirer les connaissances dans la phase d'indexation, en employant la sémantique par les ontologies ou à partir des sources de connaissances encyclopédiques sont présentées dans **[LOP et GEM, 11]**.

1.7.1.2. Méthodes pour l'apprentissage du Profil de l'utilisateur

Le profil de l'utilisateur peut être constitué d'un certain nombre de différents types d'informations qui représentent ses intérêts, dont deux types différents d'informations sont les plus considérés par les systèmes de recommandation **[PAZ et BIL, 07]** afin de créer le profil de l'utilisateur: Les préférences de l'utilisateur, et l'historique des interactions de l'utilisateur avec le système **[PAZ et BIL, 07]**. Ce dernier type d'information est important pour les systèmes de recommandation basés sur le contenu, vu qu'il peut être utilisé pour servir des données d'apprentissage pour les algorithmes d'apprentissage automatique, ayant comme but d'apprendre une fonction pour modéliser les intérêts de chaque utilisateur. La fonction utilisée par le modèle élaboré prédit si l'utilisateur sera intéressé par un item donné ou non, sous forme d'une estimation de la

probabilité que l'item est intéressant ou non pour lui, ou en prédisant directement une valeur numérique telle que le degré d'intérêt de l'item pour l'utilisateur.

La création d'un modèle de préférences de l'utilisateur à partir de son historique peut être vue comme une forme d'apprentissage d'une classification **[PAZ et BIL, 07]**, où l'ensemble de données d'apprentissage du modèle est divisé en catégories, par exemple en catégories binaires 'items que l'utilisateur a aimés' et 'items que l'utilisateur n'a pas aimés', en se basant soit sur les évaluations explicites ou implicites.

Nous distinguons différents types de méthodes d'apprentissage du profil de l'utilisateur utilisées par les systèmes de recommandation basés sur le contenu : les méthodes classiques inspirées des méthodes utilisées dans le domaine de la RI, et les méthodes adaptées pour apprendre un modèle des préférences de l'utilisateur utilisant les techniques de l'apprentissage automatique ou les modèles probabilistes. Les deux méthodes classiques les plus utilisées dans les SRs sont : le modèle vectoriel et la méthode du retour de pertinence.

1.7.1.2.1. Le Modèle Vectoriel

Ce modèle, présenté ci-dessus pour la représentation des items, est aussi appliqué par les systèmes de recommandation basés contenu pour apprendre le profil de l'utilisateur. Dans ce cas, les items d'intérêt pour l'utilisateur sont définis par leurs caractéristiques associées, où il apprend les profils des intérêts d'un utilisateur en se basant sur les caractéristiques des items qu'il a apprécié [**BUR 02**].

Donc, les profils des utilisateurs sont définis par un vecteur de poids ($w_{c1}, \ldots, w_{ck}$) tel que chaque poids dénote l'importance d'un terme (ou mot clé) pour l'utilisateur, et ils sont calculés par la méthode TF-IDF comme dans la section précédente, formule (1.6).
Les profils des utilisateurs dans ce cas sont considérés comme des modèles à long terme, et ils sont mis à jour à chaque fois que les préférences de l'utilisateur sont observées.

1.7.1.2.2. Le Retour de Pertinence

Le retour de pertinence est une technique adoptée aussi en Recherche d'Information, qui prend en considération les avis des utilisateurs afin d'affiner progressivement les requêtes en se basant sur les résultats de la recherche précédente.

Le principe général de cette technique est de permettre aux utilisateurs d'évaluer les documents retournés par le système de recherche par rapport à leur besoin d'information. Cette forme de retour d'information peut ensuite être utilisée pour affiner peu à peu la requête initiale. De manière similaire à l'évaluation des items, ils existent des moyens explicites et implicites pour collecter les données du retour de pertinence **[PAZ et BIL, 07].**

Dans le contexte des systèmes de recommandation, le retour de pertinence permet aux utilisateurs d'évaluer les documents ou items proposés par le système par rapport à leur besoin d'information. Cette forme de retour d'information peut ensuite être utilisée pour affiner progressivement le profil de l'utilisateur, ou entrainer l'algorithme d'apprentissage du classificateur qui déduit le profil de l'utilisateur **[LOP et GEM, 11]**.

L'algorithme de *Rocchio* **[ROC 71]** est l'algorithme du retour de pertinence le plus utilisé par les systèmes de recommandation basés sur le contenu. L'algorithme est fondé sur la modification d'une requête initiale par des prototypes pondérés différemment pour les documents pertinents et non pertinents. L'approche crée deux prototypes de documents en prenant la somme vectoriel sur tous les documents pertinents et non pertinents comme suit :

$$Q_{i+1} = \alpha Q_i + \beta \sum_{rel} \frac{D_i}{|D_i|} - \gamma \sum_{nonrel} \frac{D_i}{|D_i|} \quad (1.7)$$

Où, Q_i est la requête de l'utilisateur à l'itération *i.* α, β, et γ sont des paramètres qui contrôlent l'influence de la requête originale et les deux prototypes sur la requête modifiée résultante.

Le retour de pertinence a été utilisé dans plusieurs systèmes de recommandation basés sur le contenu, comme *YourNews* **[AHN et BRU, 07],** *Fab* **[BAL et SHO, 97]** et *NewT* **[SHE et MAE, 93].**

En plus des heuristiques traditionnelles basées sur les méthodes issues de la RI (Modèle Vectoriel et Retour de pertinence), beaucoup d'autres méthodes ont été utilisées comme les techniques de l'apprentissage automatique, les classificateurs Bayésiens et la classification automatique non supervisée (*Clustering)* etc.

Les techniques de l'apprentissage automatique, généralement utilisées dans la tâche d'induction des profils basés sur le contenu, ont été bien adaptées pour la catégorisation de texte **[SEB 02]**. Donc, le problème de l'apprentissage des profils des utilisateurs peut être considéré comme une tâche binaire de catégorisation de texte **[LOP et GEM, 11]**, où chaque document doit être classé comme intéressant ou pas par rapport aux préférences de l'utilisateur. Par conséquent, l'ensemble des catégories est C = {c+, c-}, où c+ est la classe positive (aimée par l'utilisateur) et c- est la classe négative (non aimée par l'utilisateur) **[LOP et GEM, 11]**.

De nombreux algorithmes d'apprentissage peuvent être utilisés pour apprendre le profil d'un utilisateur dans les systèmes de recommandation basés sur le contenu **[PAZ et BIL, 07]**, dont on présente ci-dessous les méthodes les plus utilisées par ces systèmes pour l'extraction de connaissances afin de construire le profil de l'utilisateur.

1.7.1.2.3. Classificateur Naïf de Bayes

La méthode Naïf de Bayes est une approche probabiliste pour l'apprentissage inductif, et elle appartient à la classe générale des classificateurs Bayésiens. Ces approches génèrent un modèle probabiliste basé sur les données observées précédemment.

L'objectif de ce modèle est d'estimer la probabilité d'appartenance a posteriori $P(c|d)$ du document d à la classe c. Cette estimation est basée sur la probabilité a priori $P(c)$ d'observer un document dans la classe c, la probabilité $P(d|c)$ d'observer le document d sachant la catégorie c, et la probabilité $P(d)$ d'observer le document d. En utilisant ces probabilités, le théorème de Bayes est appliqué pour le calcul de $P(c|d)$ comme suit :

$$P(c|d) = \frac{P(c)P(d|c)}{P(d)} \tag{1.8}$$

Pour classer le document d, la classe avec la plus forte probabilité est choisie:

$$P(c|d) = argmax_{c_j} \frac{P(c)P(d|c)}{P(d)} \tag{1.9}$$

$P(d)$ est généralement éliminé car il est le même pour toutes les classes c_j**[LOP et GEM, 11]**, alors que les valeurs de $P(d|c)$ et $P(c)$ sont estimées en observant les données d'apprentissage.

L'estimation $P(d|c)$ de cette manière est problématique, car il est très peu probable de voir le même document plus qu'une fois: les données observées ne sont pas généralement assez pour générer de bonnes probabilités. Le classificateur Naïf de Bayes surmonte ce problème en simplifiant le modèle à travers l'hypothèse d'indépendance : tous les mots dans le document observé *d* sont conditionnellement indépendants les uns des autres en considérant la classe *c*. Pour plus de détails sur l'algorithme Naïf de Bayes, veuillez consulter **[LOP et GEM, 11].**

Le classificateur Naïf de Bayes a été utilisé dans plusieurs systèmes de recommandation basés sur le contenu, comme *Syskill&Webert* **[PAZ *et al.*, 96] [PAZ et BIL, 97]**, *NewsDude* **[BIL et PAZ, 99]**, *Daily Learner* **[BIL et PAZ, 00],** *BALANCE* **[MOO et ROY , 00]**, et *LIBRA* **[MOO et ROY, 00].**

1.7.1.2.4. Les arbres de décision

Les arbres de décision construisent un arbre en partageant de manière récursive les données d'apprentissage, dans notre cas les items, en sous-groupes jusqu'à ce que ces sous-groupes ne contiennent que des instances d'une classe unique.

Une partition est formée par un test sur une caractéristique, dans le contexte de la classification de texte généralement la présence ou l'absence d'un seul mot ou d'une phrase. Le gain d'information et l'entropie sont les critères les plus utilisés pour sélectionner les caractéristiques les plus informatives pour les tests de séparation et pour choisir la durée sur laquelle faire fonctionner le partitionnement **[YAN et PED, 97]**.

La construction d'un arbre de décision nécessite la sélection d'un prédicat pour chaque nœud intérieur. Il ya plusieurs façons de choisir le meilleur prédicat, mais elles essaient toutes de prévoir que l'un des enfants reçoit la totalité ou la plupart des exemples positifs de l'ensemble d'apprentissage, et l'autre enfant obtient tous ou la plupart des exemples négatifs.

Une fois qu'un prédicat est sélectionné pour un nœud N, les items sont divisés en deux groupes: ceux qui satisfont le prédicat et ceux qui ne le font pas. Pour chaque groupe, on trouve le prédicat qui sépare les meilleurs exemples positifs et négatifs de ce groupe. Ces prédicats sont attribués aux enfants de N. Ce processus de division des exemples et de construction des enfants peut procéder à n'importe quel nombre de niveaux. On peut arrêter et créer une feuille, si le groupe des items pour un nœud est homogène, c.à.d. ils sont tous positifs ou tous négatifs.

Dans le contexte des systèmes de recommandation, un arbre de décision est considéré comme un arbre binaire formé d'un ensemble de nœuds organisés, où les feuilles contiennent des décisions, dans notre cas la décision serait «aimer» ou «ne pas aimer». Chaque nœud intérieur est une condition sur les objets étant classés, qui est un prédicat impliquant une ou plusieurs caractéristiques d'un item.

Pour classifier un item, le processus commence à la racine, et applique le prédicat à la racine de l'item. Si le prédicat est vrai, il faut aller à l'enfant de gauche, et s'il est faux aller à l'enfant de droite. Ensuite, le même processus se répète au niveau du nœud visité, jusqu'à ce qu'une feuille soit atteinte. Cette feuille classifie l'item comme souhaité ou non.

Plusieurs systèmes de recommandation ont utilisé les arbres de décision tel que le système proposé par **[KIM *et al.*, 01],** qui a appliqué les arbres de décision pour la personnalisation des publicités sur les pages Web, et le système *Syskill&Webert* **[PAZ *et al.*, 96] [PAZ et BIL, 97]** pour la recommandation des pages web. Ce système utilise un agent qui apprend le profil de l'utilisateur en analysant les informations sur une page, et évalue les pages web afin de sélectionner celles qui pourraient être intéressantes pour un utilisateur.

Récemment, **[KAR *et al.*, 13]** ont proposé deux extensions de cet algorithme, MPS et FDT pour l'augmentation de la vitesse de la construction de l'arbre et pour la construction des arbres de décision, respectivement. Le premier algorithme utilise la méthode de l'échantillonnage le plus populaire (*Most Popular Sampling MPS*), pour examiner seulement les items associés au nœud, qui sont les plus populaires chez les utilisateurs au lieu de vérifier tous les items candidats du nœud. La deuxième méthode

appelée arbre de décision factorisé (*Factorized Decision Tree FDT*), exploite une factorisation de la matrice pour prédire les évaluations au niveau des nœuds de l'arbre.

1.7.1.2.5. Les k-plus proches voisins

L'algorithme des plus proches voisins cherche à classer un item inconnu et non classé déjà, à la plus proche classe des items faisant partie de l'ensemble de données d'apprentissage stocké en mémoire, en utilisant une mesure de similarité. Dans ce cas, la mesure de similarité la plus utilisée est la similarité Cosinus.

Après avoir calculé la similarité entre l'item non classé et le reste des items, les *k*-plus proches items ayant la plus grande valeur de similarité sont sélectionnés et la classe de l'item non classé est définie en fonction des classes des plus proches voisins.

Cette méthode a été largement utilisée dans le domaine des systèmes de recommandation pour la classification des items, et la sélection des intérêts de l'utilisateur représentés par les plus proches items de son profil, citant à titre d'exemple les deux systèmes *Daily Learner* **[BIL et PAZ, 00],** et le système *Quickstep* **[MID *et al.*, 04]**.

Les systèmes de recommandation basés sur le contenu présentent un certain nombre d'avantages et de limitations **[BUR 02].**

1.7.1.3. Avantages

- La recommandation est générée selon les préférences personnelles d'un utilisateur particulier.
- La connaissance sur le domaine n'est pas nécessaire.
- La qualité des recommandations s'améliore avec le temps, en utilisant à chaque fois plus d'items évalués par l'utilisateur, ce qui conduit à apprendre de plus en plus les préférences de l'utilisateur et augmente la corrélation entre items.
- Un feedback implicite est suffisant.

1.7.1.4. Limites

- *La dépendance aux caractéristiques associées au contenu:* la méthode du FBC n'utilise que les caractéristiques pour représenter le contenu des items (leur profil), ce qui rend cette technique non efficace et faible dans certains contextes avec

certains types de données telles que les données multimédia, parce qu'il est très difficile d'extraire les caractéristiques à partir de ce type de données et par conséquence leur indexation est difficile. Un autre inconvénient de cette dépendance, est le cas où deux items distincts se représentent par les mêmes caractéristiques, ce qui rend la différentiation entre eux impossible.

- *Le problème de la sur-spécialisation*: Ce problème survient quand la recommandation se restreint aux items similaires à ceux déjà aimés par l'utilisateur. Dans les domaines de recherche en général de nouveaux axes apparaissent au fur et à mesure. Ces nouveaux axes utilisent des nouveaux termes pour décrire les nouveaux concepts. Donc, ces termes sont automatiquement inconnus par l'utilisateur et n'apparaissent pas dans son profil. Par conséquent, les documents portant ces termes ne seront jamais recommandés pour l'utilisateur car ils sont dissimilaires à son profil et le processus du filtrage les élimine directement. Donc, l'utilisateur ne pourra pas découvrir ce nouvel axe de recherche et exprimer sa préférence vers cet axe, sauf s'il aura une connaissance par ailleurs et il modifie manuellement son profil en ajoutant les nouveaux termes pertinents.

- L'incapacité du système à générer efficacement des recommandations qu'à partir d'un ensemble suffisant d'évaluations fournies par l'utilisateur (problème du nouvel utilisateur).

- L'incapacité de ces systèmes à intégrer des critères de pertinence des documents autres que les critères thématiques, malgré que plusieurs autres facteurs de pertinence peuvent être utilisés comme l'adéquation entre le public visé par l'auteur et l'utilisateur, la fiabilité de la source d'information ou encore la qualité scientifique des faits présentés dans les articles.

1.7.1.5. Exemple de systèmes de recommandation basés sur le contenu

Un système de recommandation basé sur le contenu a été proposé par Chandrasekaran *et al.,* [**CHA *et al.,* 08**], pour recommander les documents scientifiques susceptibles d'intéresser les auteurs connus de la base de données *CiteSeer*. Pour chaque auteur participant à l'étude, ils ont créé un profil utilisateur basé sur les documents publiés antérieurement. Sur la base de similitudes entre le profil utilisateur et les profils des documents de la collection, des documents seront recommandés à l'auteur.

Contrairement à la représentation traditionnelle, les profils des utilisateurs et des items ont été représentés par des arbres de concepts dans ce système. Ensuite, la similarité entre les profils utilisateurs et les profils documents est calculée à travers un algorithme de *matching* d'arbre en utilisant une mesure de distance arbre-édit. Les expérimentations ont démontré que l'algorithme de *matching* d'arbre de concepts performe beaucoup mieux que le modèle vectoriel, et que la méthode d'arbre de concepts offre de meilleures recommandations que la méthode de vecteur de concepts.

1.7.2. Le Filtrage Collaboratif

Le Filtrage Collaboratif compare les utilisateurs entre eux sur la base de leurs évaluations passées pour créer des communautés, et chaque utilisateur reçoit les items jugés pertinents par sa communauté [**BRE 98**].

Pour le filtrage collaboratif, les utilisateurs expriment leurs préférences envers les items en fournissant des évaluations (ou notes), qui constituent leurs profils. Ces évaluations sont comparées ensuite à celles données par les autres utilisateurs, en calculant la similarité entre eux pour sélectionner les plus proches utilisateurs. Ensuite, les prévisions des notes inconnues sont calculées, comme moyenne pondérée des évaluations faites par les plus proches utilisateurs. En d'autres termes, les recommandations pour un utilisateur donné sont basées sur le comportement et les préférences des utilisateurs similaires [**LIN *et al.*, 03**].

Cette approche résout les problèmes de l'approche basée sur le contenu; il devient possible de traiter n'importe quelle forme de contenu et de diffuser des ressources non nécessairement similaires à celles déjà reçues. Pour chaque utilisateur d'un système de filtrage collaboratif, un ensemble des plus proches voisins est identifié, et la décision de proposer ou non un item à un utilisateur dépendra des appréciations des membres de son voisinage.

Il existe trois catégories principales d'algorithmes du FC : *les algorithmes à base de mémoire, les algorithmes basés modèle et les algorithmes hybrides.*

1.7.2.1. La méthode basée modèle

L'approche basée modèle utilise les informations disponibles sur les utilisateurs et sur les items (les évaluations) pour élaborer un modèle qui génère les recommandations.

L'avantage pour ces méthodes par rapport à celles basées sur la mémoire, et qu'elles peuvent être conçues dans la phase hors ligne, en utilisant les données d'apprentissage, pour être utilisées rapidement et facilement en ligne.

Dans le cadre des approches basées modèles, la prédiction peut être faite de deux façons différentes:

- A partir de la prédiction fournie par le modèle lui-même, en construisant par exemple un modèle probabiliste pour l'estimation des valeurs de prédiction ou directement à partir du modèle.
- Ou bien, en regroupant les utilisateurs\ items par les méthodes de *clustering* et par la suite, les méthodes basées mémoires (basées utilisateurs ou basées items) seront utilisées pour prédire les évaluations pour les items.

Plusieurs méthodes ont été proposées dans la littérature utilisant des modèles, citant à titre d'exemple les réseaux Bayésiens **[BRE *et al.*, 98],** les systèmes flous **[YAG 03][MAA et SER, 10] [POR et al., 09],** les méthodes de la diagnostique de personnalité **[PEN *et al.*, 00]**, et les méthodes de Factorisation Matricielle (FM) **[PAA et TAP, 94] [LEE et SEU, 99]**, ainsi que les méthodes de l'apprentissage automatique comme les algorithmes génétiques **[KIM et AHN 08] [KIM et al., 10] [BOB *et al.*, 11a] [ATH et al., 14],** les Réseaux de Neurones Artificiels RNA (*Artificial Neural Netweorks ANN*) **[CHR et al., 07] [NIL et al., 14]**, les Machines à Vecteur de Support (*Support Vector Machines SVM*) **[MIN *et al.*, 05] [GHA et PRU, 10]**, les arbres de décision (*Decision Trees*) **[BEL et al., 08]**, le *clustering* **[SAR et al., 02]** et récemment la tendance s'est tournée vers les méthodes de l'apprentissage en profondeur (*Deep Learning*).

L'apprentissage en profondeur est appelé aussi apprentissage de caractéristiques en raison de sa puissante capacité d'apprendre les représentations de caractéristiques automatiquement **[OUY *et al.*, 14]**. En outre, les modèles en profondeur peuvent apprendre les caractéristiques d'ordre élevé des données d'entrée **[OUY *et al.*, 14],** ce qui les rend adéquats pour les systèmes de recommandation.

Actuellement, avec le développement du concept de l'apprentissage en profondeur, les chercheurs ont commencé à essayer d'employer l'inspiration de l'apprentissage en profondeur dans les systèmes de recommandation basés filtrage collaboratif **[OUY *et***

***al.,* 14]**. Plusieurs algorithmes d'apprentissage en profondeur ont été utilisés dans les systèmes de recommandation, citant par exemple l'approche basée sur les machines de Boltzmann restreintes (*Restricted Boltzmann Machines RBM*) **[SAL *et al.,* 07] [GUN et MEE, 08] [TES 15],** le modèle unifié IID qui est une extension de l'algorithme RBM **[GEO et NAK, 13],** les machines de Boltzmann non restreintes utilisées pour la modélisation conjointe des utilisateurs et des items **[PHU et VEN, 09],** et les Réseaux de Neurones Convolutionnels (*Convolutional Neural Network*) pour la recommandation aussi de musique **[VAN *et al.,* 13].** L'approche de l'apprentissage en profondeur a été utilisée aussi afin de rendre les systèmes adaptatifs au domaine pour les méthodes de classification des sentiments à partir des commentaires écris par les utilisateurs. L'approche de l'apprentissage en profondeur apprend pour extraire une représentation de haut niveau pour chaque commentaire. Ensuite, les classificateurs de sentiments utilisent cette représentation pour analyser les sentiments des utilisateurs afin d'améliorer la qualité des recommandations **[GLO *et al.,* 11].**

Nous présentons dans les sous sections suivantes les méthodes basées modèles connexes à notre travail, sachant la méthode de factorisation matricielle et la méthode du *clustering*.

1.7.2.1.1. Les modèles de Factorisation Matricielle

La Factorisation Matricielle (FM) est une méthode très utilisée en Analyse en Composantes Principales ACP qui consiste à décomposer une matrice en valeurs singulières (*Singular Value Decomposition SVD*) pour construire des facteurs orthogonaux deux à deux, ou bien en facteurs avec contraintes de non négativité sur leurs contenus **[PAA et TAP, 94] [LEE et SEU, 99].**

Les méthodes de FM ont reconnues une grande attention revenant à leur performance prouvée et leur efficacité à résoudre les problèmes des systèmes à grande échelle avec des données dispersées. La méthode de FM est adéquate à ces situations, en réduisant la dimensionnalité des matrices i.e. réduction de la clairsemé de données, et en rendant le système plus évolutif.

Cette technique a été utilisée largement dans plusieurs domaines : reconnaissance des formes, systèmes de recommandation, imagerie …ect. Où, elle a reconnu une grande importance dans les systèmes de recommandation et elle est devenue la méthode la plus

utilisées ces dernières années. Selon Agrawal et Chen **[AGR et CHE, 11]** la méthode de factorisation matricielle est classée comme étant l'approche moderne des systèmes de recommandation.

Nous allons présenter dans cette sous section, les deux principales méthodes de la factorisation matricielle.

- ***Décomposition en valeurs singulières (Singular Value Decomposition SVD)***

La décomposition en valeurs singulières (SVD) est une méthode très connue pour la factorisation matricielle qui fournit de meilleures approximations du rang inférieur de la matrice d'origine **[SAR *et al.*, 02a],** en la divisant en trois matrices principales.

Le calcul de la SVD pour la matrice d'évaluations *R* donne la factorisation suivante:

$$R = U.S.V' \qquad (1.10)$$

Où, $m = |U|$, $n = |I|$, U et V sont deux matrices orthogonales de taille $m \times k$ et $n \times k$, respectivement; S est une matrice diagonale de taille $k \times k$ ayant toutes les valeurs singulières de la matrice R comme ses entrées diagonales. Toutes les entrées de la matrice S sont positives et stockées dans l'ordre décroissant de leur importance.

Les lignes de la matrice U ($|U| \times k$) représentent l'intérêt des utilisateurs dans chacun des k topiques inférés, et les lignes de la matrice I ($|I| \times k$) représentent la pertinence de l'item pour chaque topique. Les valeurs singulières dans S sont des poids pour les préférences, représentant l'influence d'un topique particulier sur les préférences des utilisateurs envers les items dans le système.

En particulier, la clairsemé dans la matrice d'évaluation utilisateur-item soulève souvent des difficultés et la SVD conventionnelle est définie lorsque la connaissance sur la matrice est incomplète. En outre, la prédiction finale de notation est calculée en tenant compte aussi des prédicateurs de base (également connus sous le nom de préjugés (*biases*) ou interceptes). Une prédiction de base (i.e. approximation du premier ordre de bais) pour une note inconnue $r_{u,i}$ est dénotée par $b_{u,i}$ et tient compte des effets de l'utilisateur et de l'item :

$$b_{u,i} = \mu + b_u + b_i \qquad (1.11)$$

Où, μ est la note moyenne globale. Les paramètres $b_u$et b_i indiquent les écarts observés de l'utilisateur *u* et l'item *i*, respectivement, de la moyenne. Afin d'estimer $b_u$et b_i, on peut résoudre le problème des moindres carrés suivant:

$$min_{b_*} \sum_{(u,i)\in K}(r_{u,i} - \mu - b_u - b_i)^2 + \lambda(\sum_u b_u^2 + \sum_i b_i^2) \quad (1.12)$$

Ici, le premier terme $\sum_{(u,i)\in K}(r_{u,i} - \mu - b_u - b_i)^2$ vise à trouver des $b_u$et b_i qui correspondent aux évaluations attribuées. Le terme de régularisation $\lambda(\sum_u b_u^2 + \sum_i b_i^2)$ évite le sur-apprentissage en pénalisant les grandeurs des paramètres. Ce problème des moindres carrés peut être résolu assez efficacement par la méthode de la descente du gradient stochastique **[ADA 13].**

Puis, en utilisant un tel prédicateur de base, la note est prédite par la règle:

$$\hat{r}_{u,i} = \mu + b_u + b_i + q_i^T p_u \quad (1.13)$$

Où, $q_i^T p_u$ indique l'interaction entre l'utilisateur *u* et l'item *i* (interprété aussi comme l'intérêt général de l'utilisateur dans les caractéristiques de l'item) en utilisant les dimensions latentes calculées de la SVD.

Afin d'apprendre les paramètres du modèle *(b_u, b_i, p_u*et *q_i)*, l'erreur quadratique régularisée est minimisée:

$$min_{b_*,q_*,p_*} \sum_{(u,i)\in K}\left(r_{u,i} - \mu - b_u - b_i - q_i^T p_u\right)^2 + \lambda(b_u^2 + b_i^2 + \left|\left|q_i\right|\right|^2 + ||p_u||^2) \quad (1.14)$$

La constante λ, qui contrôle l'étendue de régularisation, est généralement déterminée par la validation croisée. La minimisation est généralement effectuée soit par la méthode de la descente du gradient stochastique (*Stochastic Gradient Descent SGD*) ou la méthode des moindres carrés alternés (*Alternating Least Squares ALS*).

La technique des moindres carrés alternés tourne entre la fixation de p_u pour résoudre q_i et de la fixation de q_i pour résoudre p_u. Notez que lorsque l'un d'eux est pris comme une constante, le problème d'optimisation est quadratique et peut être réglé de façon optimale **[BEL *et al.*, 07], [BEL et KOR, 07].** La méthode ALS est favorable parce qu'elle permet la parallélisation dans le système **[ADA 13].** Dans la méthode ALS, le système calcule chaque q_i indépendamment des autres facteurs de l'item, et calcule chaque p_u indépendamment des autres facteurs de l'utilisateur. De plus, la méthode est efficace pour les systèmes centrés sur les données implicites.

La méthode d'optimisation de la descente du gradient stochastique (SGD) [**FUN, 06] [KOR, 08]** utilise une approche d'apprentissage automatique standard, où elle parcourt toutes les évaluations dans l'ensemble de données d'apprentissage et pour chaque note $r_{u,i}$ donnée, une prédiction $\hat{r}_{u,i}$est faite, et l'erreur de prédiction associée $e_{u,i} = r_{u,i} - \hat{r}_{u,i}$est calculée. Pour un cas d'apprentissage $r_{u,i}$ donné, les paramètres sont calculés comme suit :

$$\begin{aligned} b_u &\leftarrow b_u + \gamma . e_{u.i} - \lambda . b_u \\ b_i &\leftarrow b_i + \gamma . e_{u.i} - \lambda . b_i \\ q_i &\leftarrow q_i + \gamma . e_{u.i} - \lambda . q_i \\ p_u &\leftarrow p_u + \gamma . e_{u.i} - \lambda . p_u \end{aligned} \tag{1.15}$$

- ***La factorisation en matrice non négative (Non Negative Matrix Factorization NMF)***

La factorisation en matrice non négative **[LEE *et al.*, 99]** est une méthode de réduction de la dimension adaptée aux situations où la matrice est positive et creuse (dispersée).

La factorisation d'une matrice X ($m \times n$) revient à la décomposer en deux matrices W et H de taille ($m \times k$) et ($k \times n$), respectivement, et qui ne contiennent que des valeurs positives ou nulle, tel que :

$$X \approx WH \tag{1.16}$$

Où, la recherche du rang de la factorisation $k << (m,n)$ joue un rôle cruciale pour l'obtention d'une bonne réduction. La factorisation se fait en minimisant la fonction objective suivante :

$$F(W,H) = ||X - WH||_F^2 \tag{1.17}$$

Plusieurs algorithmes ont été proposées pour résoudre cette fonction objective, citant comme exemple la méthode des Moindres Carrés Alternés (*Alternating Least Squares*) **[PAA and TAP, 94] [PIL *et al.*, 10],** les méthodes du Descente de Gradient (*Gradient Descente*) [**CHU *et al.*, 04],** la règle de Mise à Jour Multiplicative (*the Multiplicative Update Rule*) **[LEE and SEU, 01],** et les Moindres Carrés Alternés Non Négatifs (*Alternating Nonnegative Least Squares*) **[KIM and PAR, 07] [KIM and PAR, 08]**.

Parmi les critères de l'algorithme multiplicatif est qu'il converge vers un point stationnaire qui n'est pas forcément un minimum local, et si un élément des deux matrices prend la valeur 0, il ne sera pas changé, alors que l'algorithme de la descente du gradient pose un problème concernant le choix des pas de la descente. Alors que,

l'algorithme ALS exploite le fait que la convexité peut être en W ou H et non pas obligatoirement dans les deux.

Plusieurs méthodes d'initialisation de ces algorithmes ont été présentées dans la littérature **[LAN *et al.*, 06],** dont la plus connue est l'initialisation aléatoire. Plus de détails sur la méthode NMF et les algorithmes de minimisation seront présentés dans **[Annexe A].**

Une autre méthode très importante et très utilisée dans les systèmes de recommandation à base de modèle est la méthode du *clustering*, utilisée pour regrouper les items **[OCO et HER, 09] [GON 10]** ou les utilisateurs **[XUE et al., 05][GON 10]**, ou bien les deux (items et utilisateurs) en même temps (*bi-clustering*) [**ZHU et GON, 09][GEO et MER, 05].**

1.7.2.1.2. Le Clustering

Le *clustering* est une méthode de l'apprentissage non supervisé dont la connaissance préalable des paramètres et caractéristiques des clusters, n'est pas imposée. On peut dire aussi que le *clustering* désigne l'action de regrouper un ensemble de données (utilisateurs ou items dans notre cas) dans des groupes selon des critères de similarité, dont le nombre de groupe est prédéterminé. Ils existent plusieurs types d'algorithmes de *clustering* **[XU et WUN, 05],** dont les plus utilisés dans les systèmes de recommandation sont : le *clustering hiérarchique* et le *clustering par partitionnement*.

- ***Le Clustering Hiérarchique***

Le *clustring* hiérarchique est une méthode qui consiste à partitionner un ensemble de données hiérarchiquement en les disposants dans une structure d'arbre. L'algorithme suit les étapes suivantes :

1. Il commence avec un nombre de clusters égale au nombre d'entités *n* dans l'ensemble de données.
2. Il calcule la similarité (distance) entre une entité et une classe ou entre deux classes.
3. Il regroupe progressivement les clusters deux à deux, en choisissant les deux classes ayant les plus petites valeurs de distance en une seule.

4. Il répète ce processus en réunissant à chaque fois les classes similaires sous une classe plus grande jusqu'à l'étape (*n*-1), où on obtient une seule classe qui contient toutes les autres classes. Ce processus est reconnu aussi sous l'appellation : agglomération aux plus proches voisins.

Nous distinguons deux algorithmes différents de ce type du *clustering*, qui sont : la *Classification Hiérarchique Ascendante* (*CHA*) et la *Classification Hiérarchique Descendante (CHD)***[XU et WUN, 05]**. L'algorithme du *clustering* hiérarchique ascendant est le plus utilisé dans les systèmes de recommandation, citant comme exemple les travaux présentés par **[SCH et FAL, 07][PAR er SHE, 14][ZHE *et al.*, 13].**

- ***Le Clustering par partitionnement***

Ce type d'algorithmes consiste à regrouper les données en groupes en fonction de la similarité entre eux. L'algorithme le plus célèbre de cette catégorie est l'algorithme des *k-moyenne* (*k-means*) **[McQ 67],** qui consiste à regrouper les données en *k* groupes les plus dissimilaires que possibles, où les données au sein d'un groupe sont les plus similaires que possible.

Le principe de cet algorithme est de choisir *k* données aléatoirement, puis calculer la similarité entre le reste des données et ces *k* éléments pour les associer au *k* éléments choisis selon leur ressemblance. Ce processus se répète jusqu'à l'obtention de *k* groupes différents, où l'ajustement de leurs centres (centroids) se fait en fonction de leurs éléments.

Une variante très intéressante de cet algorithme et très utilisée est la méthode des *C-moyenne flou* (*Fuzzy C-means FCM*) [**BEZ 81**], [**BEZ 87**]. Cet algorithme regroupe les données en clusters tout en considérant la possibilité qu'une donnée puisse appartenir à différents clusters avec des degrés d'appartenance. Pour plus de détails sur l'algorithme, ses étapes et les méthodes de mise à jour, voir **[Annexe A].** L'algorithme FCM a été utilisé dans plusieurs travaux sur les systèmes de recommandation pour améliorer la phase du FC en tenant compte de l'ambiguïté dans les préférences des utilisateurs. Quelques travaux de la littérature seront présentés dans le chapitre suivant.

1.7.2.2. La méthode basée Mémoire (Voisinage)

Parmi les méthodes du filtrage collaboratif, les méthodes basées sur les plus proches voisins ont reconnues une grande popularité, en raison de leur simplicité, leur efficacité et leur capacité de produire des recommandations précises et personnalisées.

1.7.2.2.1. Les composants des méthodes du voisinage

La mise en œuvre d'un système de recommandation basé sur le voisinage nécessite trois facteurs très importants, en plus du bon choix de la méthode du voisinage utilisée, afin d'augmenter la précision et l'efficacité des recommandations ainsi que la qualité globale du système. Ces trois facteurs sont: *La normalisation des évaluations, le calcul des poids de similarité, et la sélection des voisins.*

Les principales métriques et méthodes utilisées dans ces trois étapes d'un algorithme basé sur le voisinage seront présentées dans cette section.

- ***La Normalisation des notes***

Lors de l'évaluation des items, chaque utilisateur du système de recommandation possède sa propre échelle de notation i.e. chaque utilisateur exprime ses préférences différemment des autres selon son point de vue. Par exemple, si un utilisateur *u* exprime une forte préférence d'un item *i* avec une note de 5, alors qu'un autre utilisateur *v* donne un max de note égale à 4 pour le même item pour indiquer qu'il a aimé. Selon le point de vue des deux utilisateurs, l'item *i* est supposé aimé avec satisfaction par les deux utilisateurs, mais du point de vue préférence, on juge que l'utilisateur *u* aime l'item *i* plus que l'utilisateur *v* l'a aimé. Afin d'avoir les bonnes valeurs de similarité entre les deux utilisateurs et des prédictions des évaluations précises, la normalisation des notes lors du calcul est recommandée.

Afin de convertir les évaluations individuelles à des évaluations universelles, différents schèmes de normalisation ont été proposés, où les deux schèmes les plus populaires et les plus utilisés sont: le centrage moyen et le Z-score **[DES et KAR, 11].**

- *Le centrage moyen*

L'idée du centrage moyen **[BRE *et al.*, 98] [RES *et al.*, 94]** est de déterminer si une note est positive ou négative en la comparant à la note moyenne.

Dans la recommandation basée sur l'utilisateur, une évaluation r_{ui}est transformée en une évaluation centrée moyen $h(r_{ui})$ en soustrayant à r_{ui} la moyenne des notes $\bar{r}_u$données par l'utilisateur *u* pour les items dans I_u :

$$h(r_{ui}) = r_{ui} - \bar{r}_u \tag{1.18}$$

En utilisant cette approche, la prédiction basée sur l'utilisateur d'une note r_{ui}est obtenue comme suit:

$$\hat{r}_{u,i} = \bar{r}_u + \frac{\sum_{m \in I_u} sim\,(u,v)(r_{v,i} - \bar{r}_v)}{\sum_{m \in I_u} sim\,(u,v)} \tag{1.19}$$

De la même manière, la normalisation centrée moyen basée item de r_{ui}est donnée par :

$$h(r_{ui}) = r_{ui} - \bar{r}_i \tag{1.20}$$

Où, $\bar{r}_i$ correspond à l'évaluation moyenne donnée à l'item *i* par les utilisateurs dans U_i. Cette technique de normalisation est le plus souvent utilisée dans la recommandation basée sur l'item, où une évaluation $r_{u,i}$ est prédite comme:

$$\hat{r}_{u,i} = \bar{r}_i + \frac{\sum_{j \in U_i} sim\,(i,j)(r_{u,j} - \bar{r}_j)}{\sum_{j \in U_i} sim\,(i,j)} \tag{1.21}$$

- *La normalisation Z-score*

La normalisation Z-score **[HER *et al.*, 99]** considère également les différences dans les échelles d'évaluation individuelles. Dans les méthodes basées sur l'utilisateur, la normalisation Z-score d'une évaluation $r_{u,i}$ consiste à diviser la note centrée-moyen basée utilisateur par l'écart type standard $\sigma_u$des notes attribuées par l'utilisateur *u* :

$$h(r_{ui}) = \frac{r_{ui} - \bar{r}_u}{\sigma_u} \tag{1.22}$$

La prédiction basée sur l'utilisateur de l'évaluation r_{ui} utilisant cette approche de normalisation serait donc obtenue comme suit :

$$\hat{r}_{u,i} = \bar{r}_u + \sigma_u \frac{\sum_{m \in I_u} sim\,(u,v)(r_{v,i} - \bar{r}_v)/\sigma_v}{\sum_{m \in I_u} sim\,(u,v)} \tag{1.23}$$

De même, la normalisation z-score de $r_{u,i}$ dans les méthodes basées item divise la note centrée-moyen basée item par l'écart type des notes données à l'item *i*:

$$h(r_{ui}) = \frac{r_{ui} - \bar{r}_i}{\sigma_i} \tag{1.24}$$

Par conséquent, la prédiction basée item de l'évaluation $r_{u,i}$ utilisant cette normalisation serait alors:

$$\hat{r}_{u,i} = \bar{r}_i + \sigma_i \frac{\sum_{j \in U_i} sim\,(i,j)(r_{u,j} - \bar{r}_j)/\sigma_j}{\sum_{j \in U_i} sim\,(i,j)} \tag{1.25}$$

Le choix du schème de normalisation à utiliser dans le calcul dépend des données disponibles (notes et contraintes d'évaluation) et des performances que la normalisation peut engendrer **[DES et KAR, 11]**. Les travaux de la littérature ont prouvé l'équivalence en performance entre ces deux normalisations **[HER *et al.*, 99],** alors qu'une étude récente a montré que Z-score possède des avantages plus importants que la normalisation centrée **[HOW et FOR, 08].** Si aucune normalisation ne peut être faite pour améliorer les résultats, la méthode du filtrage basé préférence est recommandée parce qu'elle se base sur l'utilisation des préférences relatives des utilisateurs au lieu des valeurs absolues de notation. Donc, prévoir les préférences relatives élimine la nécessité de normaliser les évaluations. Plus d'informations sur cette approche peuvent être trouvées dans **[COH, 98] [FRE *et al.*, 98] [JIN *et al.*, 03] [JIN et ZHA, 03].**

- ***Calcul de la similarité***

Une métrique ou mesure de similarité (MS) détermine la similitude entre les paires d'utilisateurs (FC basé utilisateur) ou la similitude entre les paires d'items (FC basé item), en utilisant les notes qu'a fait deux utilisateurs dans le système pour les items en commun (utilisateur-utilisateur) ou les notes données aux deux items par les utilisateurs du système (item-item).

Le calcul de similarité joue un rôle crucial dans les méthodes de recommandation basées sur le voisinage **[GEO et TSA, 10] [DES et KAR, 11]** parce qu'il permet de sélectionner les voisins de confiance, dont leurs évaluations sont utilisées dans la prédiction, et il fournit un moyen de donner plus ou moins d'importance à ces voisins dans la prédiction. De plus, la similarité possède un impact significatif sur la précision et la performance des systèmes de recommandation **[DES et KAR, 11].**

Une série de mesures de similarité statiques **[ADO et TUZ, 05]** ont été présentées dans la littérature, citant la Corrélation de Pearson, la Corrélation Cosinus, la Corrélation Pearson avec Contrainte et la différence quadratique moyenne **[DES et KAR, 11][BOB et al., 13].** Récemment, de nouvelles mesures de similarité ont été conçues pour répondre aux particularités et contraintes des systèmes de recommandation **[BOB *et al.*, 10] [BOB *et al.*, 12a],** et pour sélectionner l'ensemble de voisinage le plus pertinent **[BOB *et al.*, 12b] [WAN *et al.*, 08].**

Nous présentons dans cette section les métriques de similarité les plus utilisées dans la littérature: la Corrélation de Pearson (CP), la Corrélation Cosinus (COS), la Corrélation Cosinus Ajustée (CCA), la Différence Quadratique Moyenne (DQM), la Corrélation de Pearson avec Contrainte (CPC) et la Corrélation du Rang du Spearman (CRS).

- *La Corrélation Cosinus*

La Similarité Cosinus (SC) consiste à calculer la similarité du cosinus vectorielle (*Espace Vectoriel* ou *Cosine Vector CV*) **[BRE *et al.*, 98][BAL et SHO, 97] [BIL et PAZ, 00]** entre deux objets *a* et *b* représentés sous la forme de vecteurs $\vec{V}_a$ et $\vec{V}_b$, c.à.d. calculer la similarité du cosinus vectorielle entre leur vecteurs, comme suit:

$$COS(a,b) = \cos(\vec{V}_a, \vec{V}_b) = \frac{\vec{V}_a.\vec{V}_b}{||\vec{V}_a||_2.||\vec{V}_b||_2} \quad (1.26)$$

Dans le contexte de la recommandation d'items, cette mesure peut être utilisée pour calculer les similitudes entre utilisateurs ou entre items.

Pour calculer la similarité entre deux utilisateurs *u* et *v*, on considère les vecteurs de notes qu'ils ont données dans le système (qui représentent les vecteurs de leurs préférences ou de leurs profils). La similarité entre deux utilisateurs *u* et *v* serait alors calculée comme :

$$COS(\vec{V}_u, \vec{V}_v) = \frac{\sum_{i=1}^{Iu \cap Iv} r_{u,i}.r_{v,i}}{\sqrt{\sum_{i=1}^{Iu} r_{u,i}^2}\sqrt{\sum_{i=1}^{Iv} r_{v,i}^2}} \quad (1.27)$$

Où $I_{(u,v)}$ désigne l'ensemble des items évalués par les deux utilisateurs *u* et *v*. L'inconvénient de cette métrique de similarité et qu'elle ne prenne pas en compte la différence dans les échelles de notations (la moyenne et la variance des évaluations) faites par les utilisateurs *u* et *v*.

- *La Corrélation de Pearson*

La mesure de Corrélation de Pearson (CP) (*Pearson Correlation*) **[RES *et al.*, 94]** est la mesure la plus populaire dans les systèmes de recommandation du fait qu'elle considère la moyenne et la variance des évaluations. Dans le cas de la similarité entre utilisateurs, cette mesure soustrait la moyenne des notes attribuées par l'utilisateur dans le système à partir de chaque évaluation réelle inclue dans le calcul.

$$CP(u,v) = \frac{\sum_{i=1}^{Iu \cap Iv}(r_{u,i}-\bar{r}_u)(r_{v,i}-\bar{r}_v)}{\sqrt{\sum_{i=1}^{Iu \cap Iv}(r_{u,i}-\bar{r}_u)^2(r_{v,i}-\bar{r}_v)^2}} \quad (1.28)$$

La même idée peut être utilisée pour obtenir les similarités entre deux items *i* et *j* **[DES et KAR, 04] [SAR *et al.*, 01],** mais en comparant les évaluations faites par les utilisateurs qui ont noté ces deux items :

$$CP(i,j) = \frac{\sum_{u=1}^{U_i \cap U_j} (r_{u,i} - \bar{r}_i)(r_{u,j} - \bar{r}_j)}{\sqrt{\sum_{u=1}^{U_i \cap U_j} (r_{u,i} - \bar{r}_i)^2 (r_{u,j} - \bar{r}_j)^2}} \quad (1.29)$$

Bien que le signe d'un poids de similarité indique si la corrélation est directe ou inverse, sa magnitude (allant de 0 à 1) représente la force de la corrélation.

La Corrélation de Pearson souffre de problème du grand calcul de similitude entre les utilisateurs lorsque le nombre de notes en commun est peu. Cela peut être atténué par la fixation d'un seuil sur le nombre d'items évalués nécessaires pour une corrélation de 1, et l'augmentation de la similitude lorsque le nombre d'items évalués tombe en dessous de ce seuil.

- *Corrélation Cosinus Ajustée*

Les différences dans les échelles de notation des utilisateurs individuels sont souvent plus formulées que les différences dans les notes attribuées aux items individuels. Par conséquent, lors du calcul des similarités entre items, il peut être plus approprié de comparer les notes qui sont centrées sur la moyenne des utilisateurs, au lieu de la moyenne des items. La Corrélation Cosinus Ajustée (*Adjusted Cosine Correlation ACC*) **[SAR *et al.*, 01],** est une modification de la similarité de Corrélation de Pearson de l'item qui compare les notes moyennes centrées utilisateur:

$$CCA(i,j) = \frac{\sum_{i=1}^{U_i \cap U_j} (r_{u,i} - \bar{r}_u)(r_{u,j} - \bar{r}_u)}{\sqrt{\sum_{u=1}^{U_i \cap U_j} (r_{u,i} - \bar{r}_u)^2 (r_{u,j} - \bar{r}_u)^2}} \quad (1.30)$$

Dans certains cas, la Corrélation Cosinus Ajustée a été trouvée plus performante que la similarité du Corrélation du Pearson dans la prédiction des notes utilisant une méthode basée item **[SAR *et al.*, 01].**

Les trois mesures citées ci-dessus sont les plus utilisées dans la littérature, mais il existe plusieurs d'autres mesures qui ont été proposées pour calculer les similarités entre les utilisateurs ou les items, citant par exemple:

- *La Corrélation de Pearson avec Contrainte*

La Corrélation de Pearson avec Contrainte (*Constrained Pearson Correlation CPC*) **[SHA et MAE, 95]** a été proposée afin de corréler les valeurs absolues du type (aime\n'aime pas) plutôt que l'écart relatif utilisé dans la Corrélation de Pearson. Une valeur r_z est utilisée dans le calcul et elle est définie comme une valeur neutre de préférence (ni aimer ni détester), et qui se situe au milieu de l'intervalle de notation.

$$CPC(u,v) = \frac{\sum_{i \in I_u \cap I_v}(r_{u,i} - r_z)(r_{v,i} - r_z)}{\sqrt{\sum_{i \in I_u \cap I_v}(r_{u,i} - r_z)^2}\sqrt{\sum_{i \in I_u \cap I_v}(r_{v,i} - r_z)^2}} \quad (1.31)$$

- *La Différence Quadratique Moyenne*

La Différence Quadratique Moyenne (*Mean Squared Difference MSD*) **[SHA and MAE, 95],** calcule la similarité entre deux utilisateurs *u* et *v* comme l'inverse de la différence quadratique moyenne entre les notes données par *u* et *v* sur les mêmes items:

$$DQM(u,v) = \frac{|I_{u,v}|}{\sum_{i \in I_{u,v}}(r_{u,i} - r_{v,i})^2} \quad (1.32)$$

Cette mesure est apte à être modifiée pour calculer les différences sur les évaluations normalisées. La similarité DQM est limitée par rapport à la similarité du CP, car elle ne permet pas de capturer les corrélations négatives entre les préférences de l'utilisateur ou les évaluations des différents items **[DES et KAR, 11]** qui peuvent améliorer la précision des prédictions **[HER *et al.*, 02].**

- *La Corrélation du Rang de Spearman*

Une autre mesure de similarité bien connue est la Corrélation du Rang de Spearman (*Spearman Rank Correlation SRC*) **[KEN 48].** Alors que la similarité du CP utilise les valeurs de notes directement, la Corrélation du Rang de Spearman considère le rang de ces notes. Les items évalués par un utilisateur sont classés de telle sorte que l'item ayant la plus grande valeur d'évaluation prend le rang 1 et les items ayant les plus petites valeurs d'évaluation prennent des rangs plus grands. Le calcul est le même que celui de la Corrélation de Pearson, sauf que les rangs sont utilisés à la place des notes. La Corrélation du Rang de Spearman entre deux utilisateurs *u* et *v* est calculée comme suit:

$$CRS(u,v) = \frac{\sum_{i \in I_u \cap I_v}(k_{u,i} - \bar{k}_u)(k_{v,i} - \bar{k}_v)}{\sqrt{\sum_{i \in I_u \cap I_v}(k_{u,i} - \bar{k}_u)^2 (k_{v,i} - \bar{k}_v)^2}} \quad (1.33)$$

Où, $k_{u,i}$ est le rang de la note de l'item *i* dans la liste des items évalués par l'utilisateur u, et $\bar{k}_u$ est le rang moyen des items évalués par l'utilisateur u.

Le principal avantage de la CRS est qu'elle évite le problème de la normalisation des évaluations, en utilisant les classements des notes. D'autre part, cette mesure pourrait ne pas être la meilleure lorsque la plage de notes n'a que quelques valeurs possibles, ce qui crée un grand nombre de notes liées. En outre, cette mesure est généralement plus coûteuse que la CP parce que les évaluations doivent être triées afin de calculer leur rang.

De nombreuses nouvelles mesures de similarités ont été mises au point récemment pour améliorer la phase de la sélection du voisinage, afin de sélectionner que les voisins candidats, citant comme exemple les métriques *JMSD* **[BOB *et al.*, 10]**, *SM SING* (singularité) **[BOB *et al.*, 12b]** et la métrique *SM GEN* **[BOB *et al.*,11a]** basée sur les algorithmes génétiques utilisés pour trouver les poids de similarité, ainsi que les métriques adoptées pour assurer la confiance entre les utilisateurs telles que *Trust* **[JEO et CHO, 09].** Pour plus de détails sur ces mesures de similarité, veuillez consulter **[BOB 13].**

- ***Sélection du voisinage***

Le nombre des plus proches voisins à sélectionner et les critères utilisés pour cette sélection peuvent aussi avoir un impact sérieux sur la qualité du système de recommandation. La sélection des voisins utilisés dans la recommandation des items se fait normalement en deux étapes :

– Une étape du filtrage global, où seuls les voisins les plus proches sont conservés.
– Une étape de prédiction, où les meilleurs voisins sont choisis pour la prédiction.

➢ *Pré-filtrage des voisins*

Dans les grands systèmes de recommandation qui peuvent avoir des milliers d'utilisateurs et d'items, il est généralement impossible de stocker les similarités (non-zéro) entre chaque paire d'utilisateurs ou d'items, en raison des limites de mémoire **[DES et KAR, 11].** En outre, cela ne serait pas très rentable, car seules les plus

importantes de ces valeurs seront utilisées dans les prédictions. Le pré-filtrage des voisins est une étape essentielle qui rend les approches du voisinage plus faisables, en réduisant la quantité de poids de similarité à stocker, et de limiter le nombre de voisins candidats à prendre en compte dans la prédiction. Plusieurs façons sont possibles afin de pré-filtrer les voisins, dont les trois principales sont :

- ***Le Filtrage Top-N***

Pour chaque utilisateur ou item, seulement une liste des N plus proches voisins et de leur poids de similarité respectifs est maintenue. Pour éviter les problèmes avec l'efficacité ou la précision, N doit être choisi avec soin. Ainsi, si N est trop grand, une quantité excessive de mémoire sera nécessaire pour stocker les listes de voisinage et la prédiction des évaluations sera lente. D'autre part, la sélection d'une valeur trop petite pour N peut réduire la couverture de la méthode de recommandation, ce qui provoque que quelques items ne seront jamais recommandés.

- ***Le Filtrage par seuil***

Au lieu de garder un nombre fixe des plus proches voisins, cette approche garde tous les voisins dont l'ampleur des poids de similarité est supérieure à un seuil donné W_{min}. Bien que cette méthode soit plus souple que la technique du filtrage précédente, car seuls les voisins les plus importants sont conservés, la bonne valeur de W_{min}peut être difficile à déterminer. Une méthode de sélection des voisins efficace basée sur un seuil (appelée SNS pour Substitute Neighbors) a été présentée dans **[KIM and YAN, 07].**

- ***Le Filtrage Négatif***

En général, les corrélations des évaluations négatives sont moins fiables que celles positives. Intuitivement, cela est dû au fait qu'une forte corrélation positive entre deux utilisateurs est un bon indicateur de leur appartenance à un groupe commun (par exemple les fans des films d'horreur). Cependant, bien que la corrélation négative puisse indiquer l'appartenance à différents groupes, elle n'indique pas comment ces groupes sont différents, ou si ces groupes sont compatibles pour certaines d'autres catégories d'items. Les études expérimentales **[HER *et al.*, 99] [HER *et al.*, 04]** ont trouvé que les corrélations négatives ne fournissent aucune amélioration significative dans la précision de la prédiction.

- *Les voisins dans les prédictions*

Une fois la liste des voisins candidats a été calculée pour chaque utilisateur ou item, la prédiction de nouvelles notes se fait normalement avec les *k*-plus proches voisins, qui sont les *k* voisins ayant les plus grands poids de similarité. La question importante ici tourne autour de la bonne valeur à choisir pour *k*.

Lorsque le nombre de voisins est limité à l'aide d'un petit *k* (*k*<20), la précision de la prédiction est normalement faible. Quand *k* augmente, plus de voisins similaires contribuent à la prédiction, ce qui améliore la précision de la prédiction. Par contre, la précision diminue généralement lorsque trop de voisins sont utilisés dans la prédiction (par exemple *k*> 50), du fait que les quelques relations importantes sont atténuées par les nombreux relations faibles. Bien qu'un certain nombre de voisins entre 20 à 50 est le plus souvent décrit dans la littérature, voir par exemple **[HER *et al.*, 02] [HER *et al.*, 04],** la valeur optimale de *k* dépend des données et elle est en général déterminée par la validation croisée **[DES et KAR, 11].**

- *L'algorithme de recommandation basé k-Plus Proches Voisins*

L'algorithme de recommandation basé sur les *k*-plus proche voisins (KPPV) est l'algorithme référence pour les systèmes basés sur le filtrage collaboratif **[BOB *et al.*, 13]** et le plus utilisé, et ça revient à sa simplicité, la facilité de sa mise en œuvre et à l'exactitude des résultats qu'il génère et la bonne qualité des ses recommandations.

Cependant, cet algorithme est très sensible à la parcimonie de données, due à l'insuffisance de notes dans les matrices d'évaluation pour calculer la similarité entre les utilisateurs ou entre les items. De plus, l'évolutivité de données pose aussi un problème pour cet algorithme parce qu'il traite des matrices à grande échelle et qui augmentent en taille rapidement, ce qui conduit à un processus trop lent pour trouver le bon voisinage et pour recalculer la similarité avec chaque ajout d'une nouvelle entité.

L'algorithme des KPPV est basé sur les mesures de similarité afin de calculer les correspondances entre les utilisateurs ou entre les items, et sélectionner les entités ayant les plus grandes valeurs de similarité. Ensuite, les *k* plus proches voisins trouvés sont utilisés pour calculer les prédictions ou les recommandations pour les items inconnus en injectant la similarité dans la formule de calcul.

Les méthodes de similarité calculent généralement la similitude entre deux utilisateurs (utilisateur à utilisateur) en se basant sur les notes des deux utilisateurs, ou entre deux

items (item à item) en utilisant les notes qui les ont eues par l'utilisateur. Selon les deux similarités, on distingue deux versions de l'algorithme KPPV : l'algorithme des *k*-plus proches voisins basé utilisateur et l'algorithme des *k*-plus proches voisins basé item.

L'algorithme 1 résume les étapes de l'algorithme des *k*-PPV basé utilisateur.

Algorithme 1 : Algorithme de Recommandation des KPPV basé utilisateur

Entrée : Matrice d'évaluation R,
ua : l'utilisateur actif,
K: nombre d'utilisateurs dans le voisinage de l'utilisateur *N(ua)*,
N : nombre des items dans la liste de recommandation.
Sortie: La liste de recommandation L.
Pour (chaque utilisateur $x \in U$) & ($x \neq ua$) **faire**
Calculer la similarité entre *x* et *ua*,
Fin
Ordonner les similarités,
N(ua) ← les *k* utilisateurs les plus similaires,
Pour (chaque item $i \in I$) & ($r_{ua,i} = 0$) **faire**
Calculer la prédiction pour l'item *i* en utilisant les évaluations faites par les voisins dans *N(ua)*,
Fin
Ordonner les prédictions,
L ← les *top-N* items ayant les plus grandes valeurs de prédictions de la note $\hat{r}_{ua,i}$.

L'algorithme 2 présente la recommandation basée sur les *k*-PPV basé item.

Algorithme 2 : Algorithme de Recommandation des KPPV basé Item

Entrée : matrice d'évaluation R,
ua: utilisateur actif,
i: Item test,
T: nombre des plus proches items voisins de l'item test *i N(i)*,
N : nombre des items dans la liste de recommandation
Sortie : liste de recommandation L.
Pour (chaque item $j \in I$) & ($i \neq j$) **faire**
Calculer la similarité entre *i* et *j*.
Fin
Ordonner les similarités,
N(i) ← les *T* plus similaires items,
Pour (pour chaque item $i \in I$) & ($r_{ua,i} = 0$) **faire**
Calculer la prédiction pour l'item *i* en utilisant les évaluations données par l'utilisateur *ua* aux items dans *N(i)*.
Fin
Ordonner les prédictions
L ← les *top-N* items ayant les plus grandes valeurs de prédictions de la note $\hat{r}_{ua,i}$.

Les algorithmes des KPPV basés utilisateur ou basés item peuvent être combinés [**BUR 02] [WAN *et al.*, 06]** pour profiter de leurs avantages. Plus de détails sur les méthodes basées voisinage se trouvent dans **[DES et CAR, 11].**

1.7.2.2.2. Les avantages des méthodes basées sur les plus proches voisins

Les principaux avantages des méthodes du voisinage sont **[DES et KAR, 11] :**

- ✓ ***La simplicité :*** Les méthodes du voisinage sont intuitives et relativement simples à mettre en œuvre. Dans leur forme la plus simple, un seul paramètre (le nombre de voisins utilisés dans la prédiction) nécessit 46 églage.
- ✓ ***La justifiabilité :*** Ces méthodes fournissent également une justification concise et intuitive pour les prédictions calculées. Par exemple, dans la recommandation basée item, la liste des items voisins, ainsi que les évaluations fournies par l'utilisateur à ces items, peuvent être présentées à l'utilisateur comme une justification pour la recommandation. Cela peut aider l'utilisateur à mieux comprendre la recommandation et sa pertinence, et pourrait servir de base pour un système interactif où les utilisateurs peuvent sélectionner les voisins pour lesquels une plus grande importance devrait être accordée à la recommandation **[BEL *et al*., 07].**
- ✓ ***L'efficacité :*** L'un des points forts des systèmes à base de voisinage est leur efficacité. Contrairement à la plupart des systèmes à base de modèle, ils ne nécessitent pas de phase d'entrainement coûteuse, qui doit être effectuée à des intervalles fréquents dans les grandes applications commerciales. Alors que la phase de recommandation est généralement plus coûteuse que les méthodes basées sur des modèles, les plus proches voisins peuvent être pré-calculés dans une étape hors ligne, fournissant des recommandations instantanées proches. De plus, le stockage de ces plus proches voisins nécessite très peu de mémoire, ce qui rend de telles approches évolutives à des applications ayant des millions d'utilisateurs et d'items.
- ✓ ***La stabilité :*** Une autre propriété utile des systèmes de recommandation basés sur le voisinage est qu'ils sont peu affectés par l'ajout constant d'utilisateurs, d'items et de notes, qui sont généralement observées dans les grandes applications commerciales. Par exemple, une fois que la similarité entre items est calculée, un système basé item peut facilement faire des recommandations aux nouveaux utilisateurs, sans avoir re-entraîner le système. En outre, une fois quelques évaluations ont été saisies pour un nouvel item, seules les similitudes entre cet item et celles déjà dans le système doivent être calculées.

1.7.2.3. Avantages des méthodes collaboratives

- Possibilité d'indexation de tout genre d'items (e.g. son, multimédia, etc.), du fait que le contenu n'est pas pris en compte, et les items sont connus seulement par leur identifiant.
- L'élimination du problème de la sur-spécialisation, du fait que les items recommandés sont filtrés en fonction des évaluations qu'ils ont eues, qui représentent les préférences des utilisateurs, et non pas sur leur contenu. Malgré que Burke **[BUR 02]** a limité un peu cet avantage par l'imagination du cas où un item d'intérêt pour l'utilisateur qui n'est pas bien évalué par ses voisins, donc il ne sera jamais recommandé pour l'utilisateur actif.
- Une connaissance sur le domaine n'est pas demandée.
- Un feedback implicite est suffisant.
- La qualité des suggestions s'améliore avec le temps.

1.7.2.4. Limites des méthodes collaboratives

- *Le démarrage à froid (cold start) :* Lorsque le système est mis en utilisation pour la première fois, il nécessite une phase de collecte de données (utilisateurs et évaluations) afin de pouvoir calculer les similarités entre utilisateurs et entre items avant de générer les listes de recommandation.
- *Ajout de nouvel item* : Lors de l'ajout de nouvel item; le système n'est pas capable de recommander un item avant qu'il soit suffisamment évalué par les utilisateurs. Autrement, les nouveaux items ne peuvent être diffusés que si un minimum d'informations les concernant est collecté à partir de l'avis de l'un des utilisateurs.
- Les personnes ayant des goûts peu fréquents risquent de ne pas recevoir de recommandations.
- *Ajout de nouvel utilisateur*: Les nouveaux utilisateurs commencent avec un profil vide et doivent le constituer à partir de zéro. Même avec un profil de démarrage, une période d'apprentissage est toujours nécessaire avant que le profil ne reflète concrètement les préférences de l'utilisateur. Pendant cette période le système ne peut pas générer efficacement des items pertinents pour l'utilisateur.
- *Le problème de la stabilité par rapport au changement du profil des utilisateurs* : Une fois que le profil de l'utilisateur est créé dans le système, il est difficile de changer ses préférences.

- *La clairsemé de données* (*data sparcity*) : qui est une faiblesse principale pour ce type du filtrage du fait que la matrice d'évaluation est trop dispersée est contient peu de notes, ce qui rend la tâche de sélection des utilisateurs similaires très longue et donne un ensemble non pertinent de voisinage.
- *L'évolutivité de données* : un grand nombre d'utilisateurs et d'items sont ajoutés à chaque fois au système, ce qui rend difficile de les prendre en considération lors du calcul des similarités et des prédictions générées, parce que cela nécessite de recalculer les similarités à chaque fois dans les systèmes basés mémoire, et de re-entrainer le modèle conçu dans le cas du filtrage basé modèle.
- la vulnérabilité des systèmes de recommandation aux attaques.

1.7.2.5. Exemple de systèmes de recommandation basés sur le FC

➢ *Le système de recommandation BIPO* **[BAL et RIC, 08] :**

Le système BIPO (*Best Articles par Overlap*) vise à améliorer les systèmes du FC par l'adaptation de la fonction de similarité utilisateur-à-utilisateur utilisée dans l'étape de sélection des voisins, en tenant compte de l'utilisateur actif. Le système se base sur l'hypothèse qu'un voisin peut être amélioré si la similarité utilisateur-à-utilisateur est basée sur une sélection d'un sous ensemble des articles co-évalués en commun: les articles fortement corrélés avec l'article cible. Une méthode de sélection adaptative des items locaux est utilisée pour la sélection du meilleur voisin, où le calcul change dynamiquement en fonction des profils des utilisateurs. Ensuite, la méthode sélectionne le sous-ensemble d'articles ayant le plus grand poids de l'ensemble des articles co-évalués par les deux utilisateurs, dont la similarité est à déterminer.

➢ *Le système de recommandation Media Scout* **[SHA *et al.*, 07]**:

Ce système est mis en œuvre pour les laboratoires du télécom en Allemagne et il est utilisé dans les téléphones portables en Allemagne, pour recommander les items médias tels que les clips et les bandes annonces des films sur les téléphones cellulaires. Le système est conçu pour permettre aux mobiles, les portails médias en ligne ou les boîtes harnais des utilisateurs d'obtenir plus de contenu approprié.

Dans ce système, le profil de l'utilisateur correspond à une combinaison pondérée des stéréotypes de l'utilisateur, créés automatiquement par le biais d'un processus de regroupement. Où, chaque stéréotype est défini par une ontologie des attributs des items. Le système permet de classer les nouveaux utilisateurs à des groupes par le biais

d'un questionnaire interactif, généré automatiquement à partir des stéréotypes, après chaque mise à jour, alors que les utilisateurs actifs sont automatiquement classés à de nouveaux stéréotypes par le biais du processus de mise à jour sans subir à nouveau le questionnaire. Le système offre de bonnes recommandations pour les items évalués dans le passé et permet aussi de traiter les nouveaux items pour le système.

1.7.3. Le filtrage démographique

Ce type de filtrage utilise les données démographiques de l'utilisateur (âge, profession, ville d'origine, etc.) pour lui générer des recommandations. Dans ce cas, le filtrage se base sur une catégorisation des informations en fonction des données démographiques des individus. Ce qui conduit à limiter la performance de ce type du filtrage du fait que ces systèmes ne fonctionnent pas bien que pour un utilisateur avec beaucoup de proches voisins (similaires). De plus, les informations démographiques doivent être accumulées.

1.7.3.1. Exemple de système de recommandation basé sur le filtrage démographique

- *Le système de recommandation ALAMBIC* **[AIM *et al.*, 06]**

ALAMBIC est un système de recommandation basé sur le filtrage démographique, qui génère des recommandations en se basant sur les commentaires précédents des utilisateurs sur les mêmes caractéristiques démographiques (l'âge, le sexe, le niveau d'éducation, la richesse, la situation géographique, etc.). Ce système obtient de manière adéquate les objectifs de la protection de la vie privée dans les systèmes de recommandation du e-commerce, et il est basé sur une partie du tiers semi-confiant dans lequel les utilisateurs n'ont besoin que d'une confiance limitée. L'une des principales originalités de ce système est de séparer les données de l'utilisateur entre cette partie et le fournisseur de services, car il ne peut pas tirer les informations sensibles de la part de l'utilisateur seulement.

1.7.4. Le filtrage basé utilité

Ce système ne construit pas une généralisation à long terme pour ses utilisateurs, mais il base sa recommandation sur une évaluation de la correspondance entre les besoins des utilisateurs et l'ensemble des items disponibles. Dans ce type de filtrage, l'utilité de chaque item pour l'utilisateur est calculée afin de recommander les items les plus utiles.

Dans ce cas, le profil de l'utilisateur est la fonction d'utilité que le système dérive pour lui, et le système emploie des techniques de contraintes de satisfaction afin de localiser la meilleure correspondance.

Certaines limites sont associées à ce type du filtrage du fait que l'utilisateur doit introduire une fonction d'utilité au système. De plus, ces systèmes n'apprennent pas les préférences des utilisateurs ce qui conduit à une capacité de suggestion statique.

1.7.5. Le filtrage basé connaissance

Les SRs basés sur la connaissance font des suggestions en se basant sur des inférences des besoins et des préférences d'un utilisateur. Dans un certain sens, toutes les techniques de recommandation peuvent être décrites comme faisant un certain type d'inférence. Cette classe des systèmes s'inspire de la recherche en raisonnement à base de cas (*Case-based Reasoning CBR*) [**HAM 89] [KOL 93] [RIE et SCH, 89**].

Contrairement aux trois premières méthodes qui utilisent des algorithmes d'apprentissage, le filtrage basé connaissance exploite des connaissances du domaine et fait des inférences sur les besoins et préférences des utilisateurs, ce qui rend cette méthode plus appropriée dans certain cas que les autres surtout quand le système souffre d'un manque d'informations sur les utilisateurs ou sur les items (le problème de démarrage à froid) du fait qu'elle n'est pas sensible au manque d'informations.

Cependant, les approches dans ces systèmes se distinguent par le fait d'avoir une connaissance fonctionnelle, i.e. elles possèdent une connaissance décrivant comment un item particulier satisfait le besoin d'un utilisateur [**BUR 02**]. Par exemple, l'utilisateur peut recommander tout simplement un thé, et le système va connaître que le thé fait partie de la cuisine asiatique.

Le profil de l'utilisateur peut être n'importe quelle structure de connaissances qui supporte cette inférence. Dans le cas le plus simple, comme dans Google, elle peut être simplement la requête que l'utilisateur a formulée. Dans d'autres cas, elle peut être une représentation plus détaillée des besoins des utilisateurs [**TOW et QUI, 00**]. Ils existent trois types de connaissances qui sont impliqués dans un tel système :

- ***Catalogue des connaissances*** : Représente les connaissances sur les items à recommander et leurs caractéristiques. Par exemple, le système de

recommandation *Entree* devrait contenir la connaissance qui indique que la cuisine « *Thai* » est une sorte de cuisine « *asiatique* ».

- ***Connaissance fonctionnelle*** : Le système doit être capable de relier entre les besoins de l'utilisateur et l'item qui pourrait satisfaire ces besoins. Par exemple, *Entree* sait que le besoin d'une place pour un dîner romantique pourraient être satisfait par un restaurant qui a comme description « *Calme, avec une vue sur mer* ».
- ***Connaissance sur l'utilisateur*** : le système doit avoir des connaissances sur l'utilisateur qui pourraient prendre la forme d'informations démographiques générales ou des informations spécifiques sur les nécessités, pour lesquels une recommandation est demandée.

De ces types de connaissances, la dernière est la plus difficile, comme il est, dans le pire des cas, une instance du problème général de la modélisation de l'utilisateur [**TOW et QUI, 00**].

La connaissance utilisée dans un tel système peut prendre aussi différentes formes. Google par exemple utilise une information sur les liens entre les pages Web pour inférer la valeur de la popularité et d'authenticité [**BRI 98**].

1.7.5.1. Avantages

- Elle ne souffre pas du problème de démarrage à froid avec un nouvel utilisateur ou un nouvel item « *rump-up problem* ».
- Sensibilité aux changements des préférences.
- Inclusion des attributs divers (non pas seulement de l'item recommandé).
- Possibilité de relier les besoins des utilisateurs et les items.

1.7.5.2. Limites

- Capacité de suggestion statique.
- Une gestion des connaissances est requise.

1.7.5.3. Exemple de systèmes de recommandation basés connaissance

➢ *Le système de recommandation EntreeC* **[BUR 00]**

Le système *EntreeC* **[BUR 00]** est un système de recommandation des restaurants, et il a été basé sur le système de recommandation de restaurant *Entree* présenté initialement dans [**BUR 96 ; BUR 97**].

Le système génère ses recommandations par la recherche de restaurants dans une nouvelle ville similaires aux restaurants que l'utilisateur connaît et aime. Le système permet aux utilisateurs de naviguer, en indiquant leurs préférences par rapport à un restaurant donné, ainsi il affine leurs critères de recherche.

1.7.6. Les systèmes de recommandation hybrides

Les systèmes de recommandation hybrides combinent les techniques de recommandation dites pures pour atteindre une meilleure performance [**BUR 02**], en tirant profit des avantages des approches employées, et limitant les problèmes qui leurs sont liés. Les systèmes hybrides sont devenus les systèmes les plus communément utilisés ces dernières années.

En se basant sur les quatre techniques principales de recommandation citées ci-dessus (FC, BC, filtrage démographique et le filtrage basé connaissance), diverses techniques de recommandation hybrides ont été introduites et testées, et qui ont surpassé les systèmes à composant unique.

L'hybridation peut être établie en combinant des techniques différentes de types différents, où l'hybridation la plus fréquente est celle qui combine le filtrage collaboratif avec le filtrage basé contenu **[BUR 02] [WEN *et al.*, 08].** En outre, il est également possible de fusionner les différentes techniques du même type, par exemple, en hybridant les deux types du filtrage basé contenu utilisant l'algorithme Naïve de Bayes et l'algorithme des k-plus proches voisins, respectivement, ou les deux types du FC, basé mémoire et basé modèle **[CHE *et al.*, 09],** ou même le FC basé utilisateur avec FC basé item **[WAN *et al.*, 06].**

L'approche hybride a été conçue pour faire face aux problèmes des systèmes de recommandation classiques **[BUR 02]**, qui seront présentés dans la section suivante. Dont les principaux problèmes qui ont été largement abordés par les chercheurs dans ce domaine, sont le problème du démarrage à froid, les problèmes de la clairsemé et de l'évolutivité de données, et le problème de l'adaptabilité des systèmes au changement des profils des utilisateurs (la stabilité par rapport aux problèmes de plasticité).

En vue d'éviter les problèmes des méthodes de recommandation et surtout le problème du démarrage à froid **[BUR 02]** pour un nouvel utilisateur et pour un nouvel item **[WEN *et al.*, 08]**, l'hybridation entre le FC et le FBC est généralement adoptée en

utilisant le contenu des items, qui représente le profil des utilisateurs, pour détecter les similarités entre eux et construire les différentes communautés utilisées dans le filtrage collaboratif [**NGU *et al.,* 06**]. Ou bien, pour détecter la similarité entre les items les plus préférés par les voisins de l'utilisateur actif, générés par un processus du filtrage collaboratif, et les items évalués par l'utilisateur afin de lui générer ceux qui sont intéressant pour lui **[WEN *et al.,* 08]**.

Une autre hybridation puissante peut se réalisée aussi entre le FC et le filtrage basé connaissance surtout pour résoudre ce problème, du fait que ce type du filtrage est puissant et ne souffre pas des problèmes des autres approches.

Différentes possibilités de combinaison entre les méthodes de recommandation à donner naissance à sept différentes méthodes d'hybridation proposées par Burke **[BUR 02] :** la pondération, le mixage, la distribution, la combinaison de caractéristiques, cascade, l'augmentation de caractéristique, et le méta-niveau. Nous définissons dans les sous sections suivantes ces différentes possibilités d'hybridation.

1.7.6.1. L'Hybride Pondéré (weighted)

Un système de recommandation hybride pondéré est celui dans lequel le score d'un item recommandé est calculé à partir des résultats de toutes les techniques de recommandation disponibles dans le système. Donc, le plus simple hybride pondéré est celui qui combine les scores de recommandation de chaque algorithme en utilisant une formule linéaire.

L'avantage d'un système hybride pondéré est que toutes les capacités du système sont amenées à porter sur le processus de la recommandation d'une manière simple, et il est facile d'effectuer a posteriori l'affectation de poids et d'ajuster en conséquence l'hybride. Cependant, l'hypothèse implicite dans cette technique est que la valeur relative des différentes techniques est uniforme, plus ou moins uniforme dans l'espace des items possibles.

1.7.6.2. L'Hybride par Commutation (Switching)

Un hybride par commutation utilise un critère de bascule entre les techniques de recommandation. La question de cet hybride est la sélection de l'algorithme adéquat à chaque cas parmi les algorithmes de recommandation. Donc, un critère pour la sélection

de l'algorithme qui convient comme valeur de la confiance ou un critère externe devrait exister, avec lequel les composants de la méthode hybride pourraient avoir des performances différentes à des situations différentes.

Les commutations hybrides introduisent une complexité supplémentaire dans le processus de recommandation, puisque les critères de commutation doivent être déterminés, ce qui introduit un autre niveau de paramétrage. Cependant, l'avantage est que le système peut être sensible aux forces et aux faiblesses de ses recommandateurs constitutifs.

1.7.6.3. L'Hybride Mixte (Mixed)

Il s'agit d'un hybride qui est fondé sur l'intégration et la présentation de plusieurs listes de recommandation en une seule. Chacun des algorithmes de cet hybride devrait être en mesure de produire des listes de recommandation avec des rangs et l'algorithme de base de l'hybride mixte les fusionne en une seule liste classée. La question ici est de savoir comment les nouveaux scores classés (les scores de la méthode hybride mixte) devraient être produits. Un exemple simple est d'ajouter chaque score classé comme suit :

$$CF\_rank\ (3) + CB\_rank\ (2) \rightarrow CF\_Mixed\_rank\ (5)$$

L'avantage de l'hybride mixte est qu'il évite le problème du démarrage à froid d'un nouvel item: la composante basée sur le contenu peut être invoquée pour recommander de nouveaux items sur la base de leurs descriptions, même s'ils n'ont pas été notés par personne. Cependant, il souffre du problème de démarrage à froid pour un nouvel utilisateur puisque les deux méthodes, basée contenu et collaborative, ont besoin de quelques données sur les préférences de l'utilisateur pour pouvoir démarrer.

1.7.6.4. L'Hybride par Combinaison de caractéristiques (Feature Combination)

Un autre moyen pour la fusion contenu/collaboration est de traiter l'information collaborative comme simplement des données de caractéristiques supplémentaires associées à chaque exemple, et utiliser les techniques basées sur le contenu sur cet ensemble de données augmentées.

Ils existent deux composants de recommandation très différents pour cet hybride, la recommandation contribuée et la recommandation actuelle. Les recommandations

actuelles travaillent avec les données modifiées par la recommandation contribuée. La recommandation injecte les caractéristiques d'une source d'un algorithme à la source d'un autre algorithme.

L'hybride par combinaison de caractéristiques permet au système de tenir compte des données collaboratives sans compter exclusivement sur elles, alors il réduit la sensibilité du système au nombre d'utilisateurs qui ont noté un item. Inversement, il laisse le système avoir des informations sur la similarité inhérente des items qui sont par ailleurs opaques pour un système collaboratif.

1.7.6.5. L'Hybride en Cascade (Cascade)

Contrairement aux algorithmes hybrides précédents, cet algorithme implique un processus par étape. Dans cet algorithme, une technique de recommandation est employée en premier pour produire un classement en gros des items candidats, et une deuxième technique affine la recommandation parmi l'ensemble des items candidats.

L'hybride en cascade permet au système d'éviter d'employer la deuxième technique, de faible priorité, sur les items qui sont déjà bien différenciés par la première, ou qui sont mal-évalués et qui ne seront jamais recommandés. Parce que la deuxième étape de l'hybride en cascade se concentre uniquement sur les items pour lesquels une discrimination supplémentaire est nécessaire. En outre, cet hybride est de par nature tolérant au bruit dans le fonctionnement d'une technique de basse priorité (la deuxième), car seules les évaluations attribuées par le recommandateur de haute priorité (la première technique) peuvent être affinées, pas renversées.

1.7.6.6. L'Hybride par Augmentation de Caractéristiques (Feature augmentation)

L'hybride par augmentation de caractéristiques emploie une technique pour produire une note ou le classement d'un item et l'information, et ensuite cette information est incorporée dans le processus de la technique suivante.

Ce type d'hybride est similaire à l'hybride par combinaison de caractéristiques, mais différent en ce que le contributeur génère de nouvelles caractéristiques. Il utilise un algorithme pour calculer les caractéristiques qui seront utilisées comme entrée pour un autre algorithme. Il est plus flexible et ajoute de plus petite dimension que la méthode par combinaison de caractéristiques. La méthode par augmentation est intéressante, car elle offre un moyen d'amélioration de la performance d'un système de base. Une

fonctionnalité supplémentaire est ajoutée par des intermédiaires qui peuvent utiliser d'autres techniques pour augmenter les données elles-mêmes.

Bien que les deux techniques, en cascade et par augmentation de caractéristiques, enchaînent deux recommandeurs, avec le premier possède une influence sur le second, elles sont fondamentalement très différentes. Dans un hybride par augmentation, les caractéristiques utilisées par le second recommandeur incluent la sortie du premier, telles que les évaluations par exemple. Dans un hybride en cascade, la deuxième technique n'utilise pas la sortie de la première technique dans la production de ses classements, mais les résultats des deux recommandeurs sont combinés de manière prioritaire.

1.7.6.7. L'Hybride de Méta-niveau (Meta-level)

Une autre façon que deux techniques de recommandation peuvent être combinées est d'utiliser le modèle généré par l'un comme entrée pour un autre. Cela diffère de la méthode d'augmentation de caractéristiques, car dans un hybride d'augmentation, un premier modèle appris à générer des caractéristiques pour l'entrée d'un second algorithme, alors que dans un hybride de méta-niveau, l'ensemble du modèle devient l'entrée.

Pour cet hybride, une technique est employée pour produire une note ou un classement pour un item, ensuite l'information trouvée (note ou classement) sera incorporée dans le processus de l'autre technique de recommandation.

L'avantage de la méthode de méta niveau, en particulier pour l'hybridation contenu/collaboration est que le modèle appris est une représentation compressée de l'intérêt de l'utilisateur, et un mécanisme collaboratif qui suit peut fonctionner sur cette représentation de l'information dense plus facilement que sur les données de note brute.

1.7.6.8. Des exemples de systèmes de recommandation hybrides

Nous présentons dans cette section quelques systèmes présentés dans la littérature qui ont employé les différentes approches hybrides.

- *Le système de recommandation EntreeC* **[BUR 02] :**

Le système *EntreeC* [**BUR 02**] est un système de recommandation hybride en cascade qui recommande les restaurants pour les utilisateurs, en hybridant le filtrage basé

connaissance (basé raisonnement à base cas CBR) avec le filtrage social comme des techniques de prédiction.

- *Le système de recommandation de TV* [**ARD *et al.*, 03**] :

Un système de recommandation hybride pondérée de TV a été présenté par [**ARD *et al.*, 03**]. Dans ce système, trois techniques de prédiction ont été combinées: une technique fondée sur un stéréotype, une technique fondée sur des intérêts prévus explicitement par l'utilisateur, et une technique qui utilise un réseau de croyance bayésien, qui apprend de manière implicite les données du comportement de l'utilisateur recueilli. Les pondérations utilisées pour combiner les prédictions sont basées sur le score de confiance obtenu par les techniques individuelles.

- *Le système de recommandation P-Tango* [**CLA *et al.*, 99**] :

Le système *P-Tango* [**CLA *et al.*, 99**] utilise aussi un filtrage hybride pondéré. Il donne d'abord des poids égaux aux recommandations, collaborative et basée sur le contenu, mais ajuste progressivement le poids quand les prédictions sur les évaluations de l'utilisateur sont « confirmées » ou « infirmées ».

- *Le système de recommandation DailyLearner* **[BIL et PAZ , 00]**

Le système *Daily Learner* **[BIL et PAZ , 00]** est un service adaptatif de recommandation des informations (*News*), qui se base sur une méthode hybride par commutation. Il utilise un hybride basé sur le contenu/collaboration dans lequel une méthode de recommandation basée contenu est employée en premier. Si le système basé sur le contenu ne peut pas faire une recommandation avec suffisamment de confiance, alors une recommandation collaborative est tentée. Cette commutation hybride n'évite pas complètement le problème de démarrage à froid pour un nouvel item et un nouvel utilisateur, puisque les deux systèmes, collaboratif et basé contenu, souffre du problème de nouvel utilisateur. Cependant, la technique basée sur le contenu de *DailyLearner* est celle des plus proches voisins, qui ne nécessite pas un grand nombre d'exemples pour une classification précise.

1. 8. Les Problèmes des Systèmes de Recommandation

Comme tout domaine d'application, les systèmes de recommandation souffrent de plusieurs problèmes relatifs à la nature de ces systèmes et aux données nécessaires pour

leur fonctionnement. Dans cette section, nous allons identifier les problèmes inhérents aux systèmes de recommandation, et présenter quelques travaux réalisés en vue de résoudre ou d'alléger les effets de ces problèmes.

1.8.1. Le problème du démarrage à froid (*cold start problem*)

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. Selon le type de données qui manque, on peut distinguer trois types de problème du démarrage à froid **[BUR 02]** :

- ***Le démarrage à froid pour un nouveau système*** **[MID *et al.*, 02]:** Comme son nom l'indique ce type concerne les nouveaux systèmes qui souffrent d'une absence en informations (utilisateurs, items ainsi que les relations entre eux) , ce qui conduit certainement à de très mauvaises performances.

  Ce type de problème a été nommé dans **[BUR 02]** par le problème du démarrage (*The Start Up Problem*), et qui a été défini par la nécessité d'accumuler suffisamment de données pour être utilisées comme stéréotypes au démarrage. La solution traditionnelle de ce problème est de spécifier une phase, après la phase du développement du système, à la collecte de données qui seront utilisées comme données d'apprentissage, et d'utiliser les outils de classification automatique afin de trouver les corrélations entre les entités du système.
- ***Le démarrage à froid pour un nouvel item*** : Dans les systèmes à base du FC, 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 FBC avec le FC, afin de calculer les corrélations entre items **[SCH *et al.*, 01a]** en se basant sur leur contenu. De plus, les agents intelligents peuvent aussi être utilisés **[GOO 99]** pour évaluer automatiquement les items, malgré que cela peut fausser les recommandations parce qu'elles n'utilisent pas des évaluations réelles.

- ***Le démarrage à froid pour un nouvel utilisateur***: Le profil d'un utilisateur est constitué des évaluations qu'il a faites dans le système. Donc, si l'utilisateur est nouveau pour le système, cela veut dire qu'il n'a pas encore suffisamment d'évaluations pour construire son profil dans le système. Par conséquent, le système ne pourra pas fournir des recommandations pertinentes à cet utilisateur **[NGU *et al.*, 06],** parce qu'il a peu de connaissances sur ses préférences, et par la suite il n'arrivera pas à calculer sa similarité avec les autres utilisateurs afin de lui affecté à un groupe dans le cas du FC. Dans le cas du FBC, le système ne peut pas extraire les caractéristiques du contenu qui intéresse le nouvel utilisateur, ce qui conduit à des recommandations de mauvaise qualité.

Ces deux dernières variantes du démarrage à froid ont été nommées dans **[BUR 02],** par le problème Rump Up (*Rump Up Problem)*. Les solutions à ce problème diffèrent selon la méthode du filtrage utilisée (FC, FBC, ou encore filtrage hybride) et nécessitent la participation de l'utilisateur **[NGU *et al.*, 06a]** pour avoir un minimum d'informations afin de commencer le processus du filtrage**.** Où, dans le cas de la méthode du FC, une étape d'évaluation des items est demandée à l'utilisateur, dont ces items sont lui présentés par le système et ils n'ont aucune relation de préférence préalable avec les centres d'intérêt de l'utilisateur. Par conséquent, l'objectif des travaux, consacrés pour surpasser ce problème, est de chercher et présenter les meilleurs items aux nouveaux utilisateurs indépendamment d'eux **[RAS *et al.*, 02]** i.e. sans leur demander d'introduire un ensemble d'informations ainsi que d'évaluations.

Dans le cas du FBC et du filtrage hybride**,** la solution la plus courante de ce problème est de demander à l'utilisateur d'identifier ses centres d'intérêts à partir d'une liste de termes ou d'items qui correspondent le mieux à ses intérêts et préférences **[MEL *et al.*, 02], [CLA 99].** Cependant, cette étape est fatigante et ennuyante pour l'utilisateur du fait qu'il doit choisir à partir d'une grande liste de termes, ceux qui synthétisent leurs centres d'intérêts, ou bien il doit rechercher les items pertinents pour lui **[NGU *et al.*, 06a]**. Afin de réduire la tâche des utilisateurs, les données externes sur les utilisateurs peuvent être utilisées pour aider à trouver les items pertinents, automatiquement par le système **[NGU *et al.*, 06a]**, par exemple si le système dispose d'une ontologie du domaine, le profil de l'utilisateur peut être inféré automatiquement pour identifier ses

centres d'intérêts **[MID *et al.*, 02], [MID *et al.*, 04]**. En général, le profil résultant avec ces méthodes est incomplet et bruité **[NGU *et al.*, 06a]**.

Une autre solution pour ce problème, est d'associer les nouveaux utilisateurs aux profils types «stéréotypes» prédéfinis correspondants **[NGU *et al.*, 06a]**, en utilisant leurs informations démographiques ou une autre source d'informations externes telles que les pages Web personnelles **[PAZ 99]**, ou bien en exploitant les informations recueillies à partir des réponses fournies par l'utilisateur à une série de questions **[KRU 97].** Ces informations seront par la suite comparées par le processus du démarrage à froid avec les stéréotypes existants afin de trouver le plus approprié.

En plus de ces approches traditionnelles, beaucoup d'autres travaux ont été proposés dans la littérature pour alléger le problème du démarrage à froid. Nous citons à titre d'exemple, la technique de An-Te Nguyen *et al.* **[NGU *et al.*, 06a]***;* qui vise à éliminer le problème du démarrage à froid, plus précisément, celui du nouvel utilisateur.
La technique utilise le modèle des espaces de communautés **[NGU *et al.*, 06b]** qui emploie un processus de classification par règles, pour affecter automatiquement les nouveaux utilisateurs à leurs communautés initiales, qui leur correspondent le mieux, en se basant sur leurs données démographiques fournies au début de leur utilisation du système (disponibles à froid). La méthode n'utilise pas les autres données qui peuvent contribuer à la construction du profil de l'utilisateur **[BOU 05] (**évaluations des items ou description du contenu correspondant aux centres d'intérêt), car elles sont plus difficiles à obtenir à partir des nouveaux utilisateurs. Ensuite, une recommandation par niveau d'accord est générée, qui utilise l'unanimité entre les membres de la communauté pour juger la pertinence des items à recommander, et qui a prouvé de meilleures performances que celles fournies par les méthodes classiques pour le démarrage à froid.

Une autre technique était proposée par **[SCH *et al.*, 02]**, pour recommander les items qui combinent les contenus de données et la collaboration dans un cadre probabiliste. Ils ont fait la référence de leur algorithme autour du classificateur de Bayes, où ils ont tenu à recommander des items que personne dans la communauté n'a encore été évalué.

En vue d'alléger le problème du démarrage à froid ainsi d'améliorer la qualité de la recommandation, une technique puissante appelée CSHTR extension de la technique

HTR a été proposée par Li-Tung Weng **[WEN 08].** Cette technique étudie les relations implicites entre les préférences des items des utilisateurs et les préférences taxonomiques, avec laquelle ils ont vérifié que les utilisateurs qui partagent les mêmes préférences d'items partagent aussi les mêmes préférences taxonomiques. Les expérimentations menées en utilisant cette technique ont montré qu'elle a surpassé les autres techniques existantes, à la fois dans la qualité de la recommandation et l'efficacité du calcul.

1.8.2. Le problème de la Stabilité Vs. Dynamicité (*Stability Vs. Plasticity problem*)

Le problème de la stabilité des systèmes de recommandation revient à l'incapacité du système à suivre l'évolution du profil des utilisateurs, du fait que plus le système apprend les préférences des utilisateurs, plus le risque de stabiliser leurs profils augmente, ce qui rendra le changement de ces profils très difficile **[BUR 02].**

Une solution pour ce problème est de considérer que les nouvelles évaluations fournies par les utilisateurs sont plus fiables que les anciennes **[ALL 96] [LAM *et al.*, 96]**, en donnant un poids suffisant pour ces évaluations afin de virer le processus de recommandation vers les nouvelles préférences **[SCH *et al.*, 01].** Ainsi, les évaluations les plus récentes deviennent plus "importantes" pour les algorithmes d'apprentissage, en supposant qu'elles représentent mieux les intérêts des utilisateurs que les plus anciennes.

Une autre solution pour ce problème vise à apprendre les préférences des utilisateurs à propos de leurs intérêts courants, à partir d'une fenêtre de temps qui comprend les dernières observations pertinentes seulement **[KOY et SCH, 00].** Toutefois, si les intérêts de l'utilisateur changent souvent, un profil utilisateur précis ne peut pas être appris à partir d'une petite série d'observations récentes uniquement. Par conséquent, le système peut rechercher les dernières épisodes, où l'utilisateur a démontré un même ensemble d'intérêts, et essaye d'apprendre une description plus précise des intérêts de l'utilisateur en se souvenant des observations pertinentes et oubliant les observations non pertinentes **[KOY et SCH, 00].**

1.8.3. Les Systèmes Statiques

Les systèmes statiques sont des systèmes qui n'apprennent pas les préférences des utilisateurs au fur et à mesure de leurs connexions avec le système, mais ils considèrent

juste les informations qu'ils fournissent instantanément dans le processus de recommandation.

Ce type de systèmes peut être considéré comme idéal pour les systèmes à usage occasionnel tels que les systèmes de recommandation des restaurants et des musées. Cependant, si les clients utilisent le système d'une manière fréquente ça va être ennuyeux pour eux de re-saisir leurs besoins à chaque utilisation. Ce problème constitue l'inconvénient majeur des approches qui ne sont pas basées sur la mémoire, dont l'approche basée sur la connaissance est la plus concernée par ce problème malgré qu'il ne soit pas global pour cette approche car on peut imaginer des systèmes qui adaptent leurs connaissances en fonction du retour de pertinence des utilisateurs, et génèrent des recommandations qui sont de plus en plus automatisées.

Beaucoup de ces systèmes utilisent les méthodes d'apprentissage automatique pour l'acquisition des profils intéressants [**WEB 01**]. Toutefois, les intérêts des utilisateurs peuvent changer durant le temps, certains de ces systèmes sont fournis avec des mécanismes qui sont en mesure de suivre les dérivatives des intérêts des utilisateurs [**MIT *et al.*, 94**] [**GRA 98**] [**BIL et PAZ, 99**] [**SCH 00**].

1.8.4. Le problème de la parcimonie de données (*Sparcity problem*)

Ce problème est spécifique aux systèmes collaboratifs et il survient quand l'espace des évaluations est dispersé : peu d'utilisateurs ont évalué les mêmes items **[BUR02].** Dans les applications réelles, la matrice des évaluations comprend des milliers d'items et des millions d'utilisateurs. Toutefois, un utilisateur donné évalue seulement un petit nombre d'items. En conséquence, la matrice des évaluations présente de nombreuses cellules vides, ce qui conduit à avoir des difficultés pour calculer la similarité entre les utilisateurs, surtout pour la Corrélation de Pearson, et réduit énormément la précision des prédictions.

Plusieurs travaux ont été présentés dans la littérature afin de minimiser l'effet de la clairsemé de données, tels que l'algorithme de recommandation utilisant des pas aléatoires sur l'espace des items pour détecter les probabilités de transition entre les items, afin d'améliorer la matrice de similarité utilisée dans la phase de prédiction **[YIL et KRI, 08].** De plus, la clairsemé peut être atténuée grandement par les méthodes de réduction de la dimension de l'espace de données d'entrée, qui ont prouvé leur

efficacité à travers la littérature, telles que la méthode de factorisation matricielle **[CHE *et al.*, 09] [KOR 09],** et la méthode de décomposition en valeurs singulières **[KOR 08].**

1.8.5. Le problème de l'évolutivité de données

Le nombre des utilisateurs et des items dans les systèmes de recommandation peuvent être assez importants (souvent des millions d'utilisateurs et des milliers d'items). Cela risque de ralentir la procédure de recommandation de manière significative depuis les algorithmes basés sur la mémoire pour sélectionner les plus proches voisins, ce qui exigera trop de calculs **[SAR *et al.*, 01].**

1.8.6. Le problème de la sur-spécialisation

La sur-spécialisation appelée aussi (*The Portfolio effect*) dans **[BUR 02],** caractérise l'approche collaborative et surtout l'approche basée sur le contenu, du fait que des doublons de recommandations peuvent se produire lorsque des items distincts désignent un même contenu.

Pour l'approche collaborative, cet effet ne se surmontera que si les profils des utilisateurs au niveau d'un même groupe convergent vers des préférences proches et plus en plus spécialisées. Dans ce cas, la recommandation pour un utilisateur se restreint aux items similaires à ceux déjà appréciés par lui.

1.8.7. Bilan des approches et des problèmes inhérents aux Systèmes de Recommandation

Suite à l'étude que nous avons fait sur les approches et les problèmes de recommandation, nous avons établit le Tableau 1, présenté ci-dessous, qui représente une comparaison entre les approches de recommandation (non hybrides) en fonction de la présence ou non des différents problèmes (cités précédemment) au niveau de ces approches.

Problème / Approche	Nouvel utilisateur	Nouvel item	Nouveau système	Stabilité Vs. Dynamicité	Système statique	Dispersion	Evolutivité	Sur-spécialisation
Filtrage Basé Contenu	×	-	-	×	-	×	-	×
Filtrage Collaboratif	×	×	×	×	-	×	×	-
Filtrage Basé Connaissance	-	-	-	-	×	-	-	-

Tableau 1.1: Bilan des approches et des problèmes inhérents aux SRs

1. 9. La prédiction dans les Systèmes de Recommandation

La prédiction est le processus généré par les systèmes basés sur le filtrage collaboratif, visant à estimer l'intérêt d'un utilisateur donné pour un item donné sur la base d'une collection de profils d'utilisateurs. Généralement, ces profils résultent soit en demandant aux utilisateurs d'évaluer explicitement des items, ou ils sont déduits à partir du feedback implicite [**HOF *et al.*, 04**].

La recherche a été commencée avec les approches basées mémoire pour le filtrage collaboratif, qui peuvent être divisées en approches basées sur l'utilisateur [**BRE 98][HER 99][JIN 04][RES 94**] et approches basées items [**DES 04][SAR 01**].

Étant donné une évaluation test inconnue (d'un item test par l'utilisateur actif) pour être estimée, le FC basé sur la mémoire calcule tout d'abord les similitudes entre l'utilisateur actif et les autres utilisateurs (basé utilisateur), ou entre l'item test et les autres items (basé item). Ensuite, l'évaluation inconnue est prédite par la moyenne pondérée des évaluations connues de l'item test par les utilisateurs similaires (basé utilisateur), ou les évaluations connues des items similaires évalués par l'utilisateur actif (basé item) **[MA *et al.*, 07].**

En raison de la clairsemé de données de la matrice d'évaluation, de nombreuses évaluations liées (faites par des utilisateurs similaires ou attribuées aux items similaires) ne seront pas disponibles pour la prédiction, en utilisant soit la corrélation entre utilisateurs ou la corrélation entre items, ce qui réduit la précision des prédictions. Par conséquent, pour réduire l'effet de la dispersion de données et en vue d'améliorer les

prédictions, il est intuitivement souhaitable de fusionner les deux prédictions, basée utilisateurs et basée items.

L'utilisation pure des algorithmes basés mémoire pour le filtrage collaboratif donne souvent des prédictions pauvres, en particulier lorsque la matrice utilisateur-item est très dispersée. Par conséquent, la prédiction basée modèle semble plus efficace pour certaines situations critiques, pour réduire les effets de la clairsemé et de l'évolutivité de données. Toute fois, l'hybridation entre la prédiction basée mémoire et la prédiction basée modèle peut effectivement améliorer les performances du système [**WEN *et al*,. 06**] [**CHE *et al*,.09**].

Nous présentons ci-dessous les méthodes de prédiction des évaluations manquantes: la prédiction basée modèle et la prédiction basée mémoire.

1.9.1. La prédiction basée Modèle (basée factorisation matricielle)

La prédiction basée modèle dépend du modèle élaboré pour être utilisé dans la phase du FC, dont on est s'intéressé dans le présent document avec la prédiction que la méthode de factorisation matricielle fournit.

Après la convergence de l'algorithme de factorisation en matrice non négative de la matrice d'évaluation R en deux matrices W et H, une prédiction des évaluations manquantes peut être obtenue en utilisant les matrices finales. Où, l'estimation d'une nouvelle paire utilisateur-item $\hat{r}_{u,i}$ peut être obtenue par :

$$\hat{r}_{u,t} = \sum_{k=1}^{K} \widehat{w}_{u,k} \hat{h}_{k,t} \quad (1.34)$$

Où, $\widehat{w}_{u,k}$ et $\hat{h}_{k,t}$sont les facteurs optimaux obtenus après la factorisation de la matrice R.

1.9.2. La prédiction basée mémoire

On distingue deux types de prédictions relatives aux deux types du filtrage collaboratif basé mémoire: la prédiction basée utilisateur et la prédiction basée item.

1.9.2.1. La prédiction basée utilisateur

Le filtrage collaboratif basé utilisateur prédit l'intérêt de l'utilisateur actif à un item test en se basant sur l'information de notation des profils des utilisateurs similaires **[BRE and KAD 98][HER *et al.,* 99][RES *et al.,* 94].** Par conséquent**,** à chaque fois que le

nombre des utilisateurs similaires augmente, les évaluations qu'ils ont faites contribuent plus à la prédiction de la note de l'item test.

Afin d'obtenir des prédictions précises et pertinentes, l'ensemble des utilisateurs similaires à l'utilisateur actif doit être choisit soigneusement pour en sélectionner que les voisins candidats ayant des préférences les plus similaires à celles de l'utilisateur actif.

En effet, la sélection des voisins ou utilisateurs similaires se fait en utilisant les mesures de similarité, telles que celles citées au dessus, où la similarité cosinus et la corrélation de Pearson sont les mesures les plus populaires et les plus utilisées dans le FC **[BRE and KAD 98] [HER *et al.*, 99].** La similitude pourrait également être tirée à partir des données d'apprentissage **[JIN *et al.*, 04].**

La prédiction basée utilisateur vise à prédire la note prévue $\hat{r}_{u,i}$de l'item test i par l'utilisateur actif u, en utilisant les évaluations faites par les voisins pondérées par la similarité entre eux comme suit **[BRE and KAD 98][HER et al. 99] :**

$$\hat{r}_{u,i} = \bar{r}_u + \frac{\sum_{v \in KPPV} sim\,(u,v)(r_{v,i} - \bar{r}_v)}{\sum_{v \in KPPV} sim\,(u,v)} \quad (1.35)$$

Où, $\bar{r}_u$ et $\bar{r}_v$ représentent la moyenne des évaluations faites par les utilisateurs u et v, respectivement, et elles sont utilisées pour normaliser et ajuster les calculs. $v \in \{k$-plus proches utilisateurs de l'utilisateur actif $u\}$, et sim (u, v) est la similarité calculée par l'une des mesures identifiées précédemment.

Les méthodes existantes diffèrent dans leur traitement des notes inconnues des utilisateurs similaires ($r_{v,i} = \emptyset$). Dans la majorité des cas, les notes manquantes sont remplacées par un score de 0, mais cela peut engendrer une réduction de la prédiction, où la note moyenne de cet utilisateur similaire est utilisée [**BRE and KAD 98], [HER *et al.*, 99].** Dans **[XUE *et al.*, 05]** les notes inconnues sont remplacées par une interpolation de la note moyenne de l'utilisateur et la note moyenne de son groupe.

1.9.2.2. La prédiction basée item

Les approches basées item, telles que [**DES and KAR 04], [LIN *et al.*, 03] et** [**SAR *et al.*, 01**] appliquent la même idée, mais utilise la similarité entre les items au lieu des utilisateurs.

L'évaluation inconnue d'un item test par l'utilisateur actif peut être prédite en utilisant la moyenne des évaluations des items similaires de l'item test évalués par l'utilisateur actif [**SAR *et al.*, 01**], comme suit:

$$\hat{r}_{u,i} = \frac{\sum_{m \in TPPI} sim\,(i,m)(r_{u,m} - \bar{r}_u)}{\sum_{m \in TPPI} sim\,(i,m)} \quad (1.36)$$

Où, $m \in \{T$ plus proches items de $i\}$ et $sim\,(i,m)$ est la similarité entre les items i et m, et elle est calculée par la similarité cosinus vectorielle ou cosinus ajustée.

1. 10. Evaluation des Systèmes de Recommandation

Afin d'évaluer la performance des systèmes de recommandation, un certain nombre de mesures statistiques ont été proposées dans la littérature [**HER 04**]. On s'intéresse dans cette section par les mesures d'évaluation de la précision des prédictions et de la qualité des recommandations. Nous présentons dans ce qui suit deux types de métriques, pour l'évaluation de la prédiction et des recommandations Top-N, respectivement.

1.10.1. Métriques d'évaluation de la prédiction

Différentes mesures d'évaluation de la prédiction ont été présentées afin de juger la performance des systèmes à base du FC **[SAR 01],** en termes de précision des prédictions i.e. en testant la ressemblance entre les valeurs prédites et les évaluations réelles faites par l'utilisateur actif. Nous citons dans cette section les trois mesures célèbres d'évaluation des prédictions.

1.10.1.1. Erreur Moyenne Absolue [Mean Absolute Error MAE]

La métrique MAE est la métrique la plus utilisée dans les systèmes de recommandation, et elle est calculée en faisant la moyenne de l'écart absolu entre les valeurs prédites et les valeurs réelles. Formellement, elle est représentée comme suit:

$$MAE = \frac{\sum_{u,m} |r_{u,m} - \tilde{r}_{u,m}|}{|N|} \quad (1.37)$$

Où, N est le nombre des évaluations utilisées dans le test, $r_{u,m}$ est l'évaluation réelle faite par l'utilisateur u à l'item m, et $\tilde{r}_{u,m}$ est la note prédite correspondante. Plus MAE est petite, plus la performance est meilleure.

1.10.1.2. Erreur Quadratique Moyenne [Root Mean Squared Error RMSE]

L'erreur quadratique moyenne RMSE (appelée aussi écart quadratique moyen, RMSD) est une mesure fréquemment utilisée pour tester la performance d'un modèle, en

calculant la différence entre les valeurs prédites par le modèle et les valeurs réellement observées dans l'environnement qui est en cours de modélisation.

L'erreur RMSE d'une prédiction d'un modèle par rapport à une variable estimée $\tilde{r}_{u,m}$ est définie comme la racine carrée de la moyenne de l'erreur quadratique:

$$RMSE = \sqrt{\frac{\sum_{u,m}(r_{u,m}-\tilde{r}_{u,m})^2}{N}} \quad (1.38)$$

Où, $\tilde{r}_{u,m}$ est la valeur estimée par le modèle pour l'évaluation faite par l'utilisateur u à l'item m, et N le nombre total des évaluations utilisées dans le test. Cette mesure est plus sensible à l'erreur parce que la différence est mise au carré avant qu'elle soit sommée.

1.10.1.3. Erreur Moyenne Absolue Normalisée [Normalized MAE NMAE]

La métrique NMAE peut être calculée de deux façons **[PAR 06]**:

- ✓ *La Macro-MAE :* Calcule la *MAE* de chaque utilisateur séparément, puis elle calcule la moyenne de toutes les MAEs des utilisateurs.
- ✓ *La Micro-MAE :* Calcule directement l'erreur moyenne de toutes les évaluations de l'ensemble de tous les utilisateurs.

1.10.2. Métriques d'évaluation des recommandations (Top-N)

La performance des systèmes de recommandation est testée en termes de la précision des listes de recommandation générées. Les mesures d'évaluation de la performance des recommandations Top-N sont : la Précision, le Rappel et la métrique F1. Nous présentons aussi la mesure de diversité qui est l'une des mesures d'évaluation des recommandations proposées récemment pour en valider la performance des systèmes de recommandation et que nous allons utiliser dans nos tests.

1.10.2.1. Précision

Elle est définie par le rapport entre le nombre d'items pertinents sélectionnés et le nombre total des items sélectionnés. La précision représente la possibilité qu'un item sélectionné soit pertinent.

$$P = \frac{N_t}{N} \quad (1.39)$$

- N_t : Nombre des items pertinents trouvés.
- N : Nombre total des items.

1.10.2.2. Rappel

C'est le rapport entre le nombre d'items pertinents sélectionnés et le nombre total des items pertinents disponibles. Ceci, représente la probabilité qu'un item pertinent soit sélectionné.

$$R = \frac{N_t}{N_p} \tag{1.40}$$

- N_p : Nombre total des items pertinents.

Les deux mesures citées ci-dessus (*Précision* et *Rappel*) ne mesurent qu'une pertinence binaire (i.e. item pertinent/non pertinent), c'est-à-dire elles ne peuvent pas mesurer la qualité d'ordonnancement des items pertinents sélectionnés.

1.10.2.3. La métrique F1

C'est une combinaison des deux métriques précédentes, et elle est définie par :

$$F_1 = \frac{2PR}{P+R} \tag{1.41}$$

1.10.2.4. La mesure d'évaluation de la Diversité

La diversité dans les listes de recommandation est devenue un critère pour juger l'importance des items recommandés par rapport à l'utilisateur. Il existe deux types de diversité : la diversité individuelle et la diversité globale. Dans ce manuscrit, la diversité individuelle est utilisée comme mesure pour évaluer la diversité dans les listes de recommandation et elle est calculée comme suit :

$$ID_u = \frac{1}{N(N-1)} \sum_{i \in N} \sum_{j \in N, i \neq j} d(i,j) \tag{1.42}$$

Avec $$d(i, j) = 1\text{-}sim(i, j) \tag{1.43}$$

Où *sim* est la similarité entre deux items *i* et *j*.

1. 11. Critères d'Evaluation pour les Systèmes de Recommandation

Dans cette section, une série de propriétés des systèmes de recommandation est présentée, qui doivent êtres considérées lors de la sélection des approches de recommandation. Selon les besoins des applications, le concepteur du système décide quelles sont les propriétés importantes à considérer pour évaluer son application, où

certaines propriétés sont compromises (comme la précision et la diversité) et il faut bien comprendre et évaluer cette balance (ce compromis) et son effet sur la performance du système.

De plus, l'effet de ces propriétés sur l'utilisateur n'est pas clair et dépend de l'application, mais il faut montrer l'importance de chaque propriété considérée dans la pratique. Par conséquent, lorsque le concepteur suggère une telle approche pour améliorer une telle propriété, il faut évaluer comment les changements dans cette propriété affecte l'expérience de l'utilisateur, soit par le biais d'une étude de l'utilisateur ou par l'expérimentation en ligne **[SHA et GUN, 11].** Une fois les effets des propriétés spécifiques pour un système sur l'expérience de l'utilisateur ont été compris, on peut utiliser les différences de ces propriétés pour sélectionner un recommandeur.

1.11.1. Les préférences de l'utilisateur

Une propriété importante à tenir en compte dans les systèmes de recommandation est la préférence des utilisateurs vis-à-vis du système et des produits présentés. La préférence d'un même système ou un même item dépend d'un utilisateur à l'autre en termes de l'échelle des évaluations ou du retour d'information, comme elle se diffère chez le même utilisateur pour des systèmes différents. Par conséquent, une normalisation des préférences est nécessaire, ainsi que l'utilisation des poids pour donner de l'importance aux notes.

Donc, lorsqu'on souhaite améliorer un système, une étude de l'utilisateur est nécessaire afin de comprendre ses préférences et savoir pourquoi il préfère un système sur l'autre, en comparant les propriétés spécifiques. Pour simplifier de mesurer la satisfaction des utilisateurs, la segmentation de la satisfaction en éléments plus petits est utile pour comprendre le système et l'améliorer **[SHA et GUN, 11].**

1.11.2. Précision de la prédiction

La précision de la prédiction est la propriété la plus discutée dans la littérature des systèmes de recommandation. A la base de la grande majorité des systèmes de recommandation se trouve un moteur de prédiction. Ce moteur peut prédire les opinions des utilisateurs sur des items (par exemple les votes pour les films) ou la probabilité de l'utilisation (d'un produit par exemple dans la recommandation des achats).

Une hypothèse de base dans un système de recommandation est qu'un système qui fournit des prévisions plus précises sera préféré par l'utilisateur. Ainsi, de nombreux chercheurs ont tenté de trouver des algorithmes qui fournissent de meilleures prévisions.

La précision de la prédiction est en général indépendante de l'interface utilisateur, et peut être donc mesurée dans un essai en mode hors connexion. Une mesure de précision de la prédiction dans une étude de l'utilisateur mesure la précision donnée dans une recommandation.

De nouveaux critères et propriétés ont été apparus ces dernières années **[SHA et GUN, 11]** pour améliorer la qualité des recommandations et augmenter la satisfaction des utilisateurs. Nous présentons ci-dessous une liste de ces critères et propriétés.

1.11.3. La Couverture

La précision de la prédiction d'un système de recommandation, en particulier dans les systèmes du FC, augmente dans de nombreux cas avec la quantité de données. Cependant, certains algorithmes peuvent fournir des suggestions de haute qualité, mais seulement pour une petite partie des items alors que ces systèmes possèdent d'énormes quantités de données. Le terme couverture peut se référer à deux propriétés distinctes du système, couverture de l'espace d'items et couverture de l'espace d'utilisateurs, que nous discutons ci-dessous.

1.11.3.1. Couverture de l'espace d'items

Le plus souvent, le terme couverture se réfère à la proportion des items que le système de recommandation peut recommander. Ceci est souvent désigné comme couverture du catalogue **[SHA et GUN, 11].**

La mesure la plus simple de la couverture du catalogue est le pourcentage de tous les items qui ne peuvent jamais être recommandés. Cette mesure peut être calculée dans de nombreux cas donnés directement par l'algorithme sur l'ensemble de données d'entrée.

Une mesure plus utile est le pourcentage de tous les items qui sont recommandés aux utilisateurs lors d'une expérience, que ce soit hors ligne, en ligne, ou une étude de l'utilisateur. Dans certains cas, il peut être souhaitable de pondérer les items, par exemple, par leur popularité ou leur utilité. Ensuite, recommander les items avec le plus grand profil et éviter les items qui sont très rarement utilisés.

1.11.3.2. Couverture de l'espace utilisateur

La couverture peut aussi être la proportion des utilisateurs ou des interactions de l'utilisateur pour lequel le système peut recommander des items. Dans de nombreuses applications le système de recommandation ne peut pas fournir des recommandations pour certains utilisateurs en raison, par exemple, d'une faible confiance dans l'exactitude des prévisions pour cet utilisateur. Dans de tels cas, on peut préférer les systèmes de recommandations qui peuvent fournir des recommandations pour un large ensemble d'utilisateurs. Évidemment, ces systèmes doivent être évalués sur le compromis entre la couverture et la précision.

La couverture est mesurée par la richesse du profil de l'utilisateur requis pour faire une recommandation **[SHA et GUN, 11].** Par exemple, dans le cas du filtrage collaboratif cette propriété peut être mesurée comme le nombre d'items que l'utilisateur doit évaluer avant de recevoir des recommandations. Cette mesure peut être évaluée généralement dans des expériences hors ligne.

1.11.4. La Confiance (Confidence)

La confiance (*Confidence*) dans les systèmes de recommandation peut être définie comme la confiance du système dans ses recommandations ou ses prédictions **[SWE et SIN, 01] [HER *et al.*, 00]**. La confiance de la prédiction est basée sur un support de connaissance pour cette prédiction, soit en termes de ce qui est connu sur l'utilisateur ou sur l'item, et elle est généralement utilisée dans la phase d'évaluation hors ligne **[SHA et GUN, 11].**

Une mesure de confiance est importante car elle peut aider les utilisateurs à décider quels films à regarder, produits à acheter, et aussi aider un site e-commerce à prendre une décision sur quelles recommandations ne devraient pas être affichées, car une fausse recommandation peut diminuer la confiance des utilisateurs dans le système **[HER *et al.*, 04].**

De plus, l'utilisation d'une mesure de confiance est susceptible d'améliorer la satisfaction des utilisateurs et de modifier le comportement des utilisateurs dans un système de recommandation, où sa formule la plus simple est de calculer le nombre d'évaluations données à un item **[NEE *et al.*, 03].** Le système de recommandation affiche à l'utilisateur des explications sur les items recommandés ou bien il lui génère

un histogramme qui indique le nombre de personnes qui ont évalué l'item **[MLE *et al.*, 13].** Une large gamme d'algorithmes de recommandation afin de décider la confiance des recommandations et des prédictions, ainsi que des mesures de confiance ont été proposées dans la littérature **[MLE *et al.*, 13][ADO *et al.*, 07][HER *et al.*, 04][MAZ 13].**

1.11.5. La Confiance (Trust)

Bien que la confiance (*Confidence*) représente la confiance du système dans ses prédictions, la confiance (*Trust*) est la confiance de l'utilisateur dans le système de recommandation. Différentes définitions ont été apportées au *Trust* selon son type, où on peut distinguer deux principaux types **[ABD and HAI 98]** :

- La confiance interpersonnelle spécifique au contexte, dans laquelle un utilisateur fait confiance à un autre utilisateur par rapport à des situations particulières, et
- La confiance système-impersonnelle qui représente la confiance de l'utilisateur au système de recommandation. Par exemple, si le système recommande quelques items que l'utilisateur connaît déjà et aime, de cette façon, même si l'utilisateur ne gagne pas de valeur à partir de cette recommandation, il observe que le système fournit des recommandations raisonnables, ce qui peut accroître sa confiance dans les recommandations du système pour les items inconnus.

La méthode la plus évidente pour évaluer le *Trust* est de demander aux utilisateurs si les recommandations du système sont raisonnables ou non dans une étude de l'utilisateur **[HER *et al.*, 00][BON *et al.*, 06][CRA *et al.*, 08][PU and CHE, 06].**

La notion de la confiance (*Trust*) a reconnu une grande attention chez un certain nombre de communautés de recherche et différents points de vue ont été présentés sur la façon de mesurer et utiliser la confiance [**ADO and SMI, 05][MAS and BAH,04][MAS and AVE 07][PIT and MAR 08][PIT 09][YAN 14][WAN and KAO 13][BOB *et al.*, 11][ALE *et al.*, 15].**

1.11.6. La Sérendipité (Le hasard)

La sérendipité représente l'inhabituel ou la surprise dans les recommandations. Contrairement à la nouveauté, la sérendipité englobe le contenu sémantique des items,

et elle peut se considérer comme la distance entre les items recommandés et leurs contenus attendus.

Récemment, plusieurs chercheurs ont été intéressés par l'aspect de la sérendipité dans les systèmes de recommandation, et diverses définitions on été proposées pour ce concept. Par exemple, dans **[HER *et al.*, 04]** le hasard est défini comme une mesure dans laquelle les items recommandés sont attrayants et surprenants, en même temps, pour les utilisateurs. Cela signifie qu'une recommandation très fortuite permettrait à un utilisateur de trouver un item surprenant et intéressant. Donc, un item fortuit (*serendipitous item*) doit être non encore découvert ni prévu par l'utilisateur, et également il devrait être intéressant, pertinent et utile pour lui.

Des définitions similaires se retrouvent également dans d'autres œuvres. Par exemple, dans **[SHA et GUN, 11]** la sérendipité a été considérée comme une mesure de la façon dont les recommandations sont surprenantes et réussies.

La sérendipité est devenue un des critères utilisés pour juger et évaluer les recommandations **[OKU et HAT, 12] [GE *et al.*, 10]** et plusieurs méthodes pour l'assurer **[ZHA *et al.*, 02] [ZHA et HUR, 08],** ainsi que mesures ont été proposées dans la littérature afin de calculer le hasard dans les recommandations, à titre d'exemple la métrique *Unserendipity* **[ZHA 12],** la métrique *SRDP* **[GE *et al.*, 10]** et les deux métriques *Unexpectedness* et *Unexpectedeness_r* **[MUR *et al.*, 08].** La mise en œuvre de la sérendipité est en vue d'éviter de recommander les items évidents dans les systèmes basés sur le FC **[HER *et al.*, 04],** éviter aussi les problèmes de la sur-spécialisation dans les systèmes de recommandation basés sur le contenu **[IAQ *et al.*, 08]**, et pour aider les utilisateurs à révéler leurs intérêts inattendus **[KAM *et al.*, 05].**

La sérendipité est jugée à partir d'un sentiment subjectif **[GE *et al.*, 10].** Cependant, elle est associée d'un risque si les recommandations fortuites génèrent aux utilisateurs des items insatisfaisants ou inutiles, ce qui conduit les utilisateurs à cesser de suivre les recommandations ou même de ne pas utiliser le système de recommandation une autre fois **[SHA et GUN, 11].**

Finalement, on note que les recommandations très fortuites peuvent être obtenues à un coût d'une diminution de la précision, en basant ces recommandations sur quelques utilisateurs très similaires de l'utilisateur actif. Par exemple, le système pourrait trouver

l'utilisateur le plus similaire et recommande le nouvel item qui a reçu la plus grande note par cet utilisateur.

1.11.7. La Nouveauté

La nouveauté est l'une des métriques qui font l'objet d'intérêt ces dernières années pour évaluer la qualité des listes de recommandation. Différentes définitions ont été présentées dans la littérature pour définir le terme « Nouveauté » et qui dépend d'un domaine à l'autre **[ZHA 13**].

Une nouvelle recommandation est définie dans **[KON e*t al.,* 06]** comme la recommandation qui contient des items inconnus par l'utilisateur. Une autre définition de la nouveauté des recommandations est la considération des items non populaires (*Long Tail items*) pour la recommandation **[SHA *et al.,* 08][ZIE *et al.,* 05][CEL et HER 08]** en incluant la popularité dans la formule de prédiction. Dans **[ZHA 13],** La nouveauté des items dans les listes de recommandation était vue comme la caractéristique d'avoir des items inconnus, i.e. non vus déjà par l'utilisateur, qui sont satisfaisants et dissimilaires aux préférences de l'utilisateur**.** Plusieurs métriques ont été proposées afin d'injecter et de calculer la nouveauté dans les listes de recommandation **[ZIE *et al.,* 05][SHA *et al.,* 08][CAS *et al.,* 11][VAR 11][BOB *et al.,* 11].** Un bon état de l'art sur la nouveauté dans les SRs est présenté dans **[ZHA 13].**

1.11.8. La Diversité

La diversité est devenue l'une des propriétés les plus recherchées à nos jours dans les listes de recommandation afin de satisfaire les différents besoins des utilisateurs. Avoir une liste diversifiée revient à avoir une liste avec des items non similaires aux préférences de l'utilisateur. La diversité a été définie dans **[SHA et GUN 11]** comme l'inverse de la similarité. La notion de la diversité est fortement liée à la nouveauté dans **[VAR et CAS, 11][CAS *et al.,* 11][VAR 11],** parce que un nouvel item non vu déjà par l'utilisateur est un item dissimilaire à son profil comme on a déjà mentionné ci-dessus dans la sous section (1.11.7), donc il est sûrement différent des préférences de l'utilisateur et lui offre un plus grand choix. Par exemple, dans un système de recommandation de vacance **[SMY et MCC, 01],** le système recommande à l'utilisateur les tarifs et forfaits des vacances pour la destination qu'il a cherchée ou voulue. Mais si le système offre à l'utilisateur d'autres destinations de plus avec leurs forfaits, la

recommandation sera plus utile et plus satisfaisante pour l'utilisateur, parce qu'elle lui offre plus de choix qui seront peut être importants pour lui, et lui changent d'avis et de destination.

Il a été démontré dans les travaux connexes, que l'augmentation de la diversité dans la recommandation conduit à une diminution de la précision **[ZIE *et al.*, 05]][ZHA et HUR, 08].** Cependant, le défi majeur pour la diversité dans les systèmes de recommandation est la conservation de la précision des recommandations en présence de la diversité. Par conséquent, un paramètre de contrôle doit être considéré pour vérifier l'importance de la diversité dans la liste de recommandation, et pour équilibrer le compromis entre la diversité et la précision. En d'autre termes, on peut dire que le rôle de ce paramètre est de décider si les items divers recommandés sont pertinents pour l'utilisateur ou non.

Plusieurs méthodes ont été présentées dans la littérature pour augmenter la diversité dans les listes de recommandation **[ZIE *et al.*, 05][LAT *et al.*, 10][ADO et KWO, 09][ADO et KWO, 12][ZHA et HUR, 08]** ainsi que des métriques pour l'évaluer **[ZIE *et al.*, 05][CAS *et al.*, 11][VAR et CAS, 11][ADO et KWO 09,12]**, que nous allons les présenter dans le chapitre suivant. Une panoplie d'autres travaux sur la diversité dans les systèmes de recommandation est présentée dans **[GHA *et al.*, 13].**

1.11.9. L'Utilité

L'utilité dans un système de recommandation peut être considérée par rapport à l'utilisateur ou par rapport au site web **[SHA et GUN, 09].**

L'utilité d'une recommandation par rapport au site web représente le gain que le système obtient derrière une recommandation (en termes de revenues) **[SHA et GUN, 09].** Tandis que l'utilité d'une recommandation par rapport à un utilisateur est son intérêt et son efficacité par rapport à cet utilisateur, en calculant soit l'utilité d'un item à l'utilisateur **[BUR 02]** ou bien d'une liste d'items, comme la somme des utilités des items **[SHA et GUN, 09].**

Dans certains systèmes, l'utilité des recommandations est calculée pour juger la performance des systèmes au lieu du calcul de la précision. La diversité et la sérendipité peuvent être considérées comme étant des types d'utilité [**SHA et GUN, 09].** Pour plus

de détails sur l'utilité des recommandations et les fonctions utilisées, veuillez se référencer à **[CHE et PU, 04][HUA 11].**

1.11.10. La Robustesse

La robustesse d'un système de recommandation peut être vue comme la capacité du système à fournir les bonnes recommandations en présence de fausses informations insérées dans le système à travers des attaques **[MOB *et al.*, 07] [BUR *et al.*, 11],** qui ont pour but d'influencer les recommandations. Ces attaques sont engendrées par des utilisateurs malveillants qui veulent manipuler le système pour leurs besoins. Les utilisateurs malveillants falsifient les notes concernant leurs produits en les attribuant les valeurs maximales de vote, et dégradent les produits de leurs concurrents en les donnant les valeurs minimales de vote.

La robustesse des systèmes de recommandations a été élaborée dans plusieurs travaux [**LAM et RIE, 04] [BUR *et al.*, 11] [O'MA *et al.* 04] [O'MA *et al.*, 05],** ainsi que différentes stratégies et métriques ont été proposées pour l'évaluer [**BUR *et al.*, 11] [SEM *et al.*, 12] [CHI *et al.*, 05 [BUR *et al.*, 06][MOB *et al.*, 07][MEH et NEJ 08].**

1.11.11. La Confidentialité

Un utilisateur lors de son interaction avec un système de recommandation fournit des informations personnelles, ainsi que des évaluations pour exprimer ses préférences aux items et qui constituent son profil. Les utilisateurs préfèrent que personne ne peut accéder à leurs informations ni à leurs préférences, et qu'elles soient privées et sauvegardées dans le système d'une façon confidentielle. Cependant, le système doit sécuriser les informations privées de ses utilisateurs contre les attaquants, qui ont des accès au système et ses informations **[SHE et MIR, 09]** et qui représentent un risque sur la confidentialité du système **[RAM *et al.*, 01] [BOU *et al.*, 13].** Par conséquent, l'implémentation des algorithmes avancés est nécessaire pour assurer la confidentialité dans les systèmes de recommandation tels que l'algorithme proposé par **[SHE et MIR, 09],** qui assure la vie privée des utilisateurs dans le système Netflix, et l'algorithme de ré-identification naïf *Intersection d'ensemble* (*Set Intersection*) **[FRA *et al.*, 06]**. Pour plus de détails sur la confidentialité dans les systèmes de recommandation, veuillez lire **[PER *et al.*, 04] [FRA *et al.*, 06] [SHE et MIR, 09] [SHA et GUN 11].**

1.11.12. L'Adaptabilité

L'adaptabilité est la caractéristique qu'un système soit capable de s'adapter aux préférences de l'utilisateur **[MAH et RIC, 07]** ou aux changements de son profil **[KOY et SCH, 00],** en lui générant des recommandations tout en considérant les nouvelles notes qu'il donne aux items.

L'évaluation de l'adaptabilité peut être faite dans la phase en ligne en utilisant des interfaces interactives **[MAH et RIC, 09]**, ou hors ligne en ajoutant de nouvelles évaluations pour un tel utilisateur. Ensuite, une comparaison de l'ancienne liste de recommandation avec la nouvelle liste est faite, pour détecter les différences entre les listes avant et après l'ajout des nouveaux votes, en calculant le taux de variabilité entre les deux listes par l'indice de Gini ou l'entropie de Shanon **[SHA and GUN 09].**

Un autre type d'adaptabilité est la capacité du système à prendre en compte les nouveaux évènements et leur changement rapide, ainsi que collecter et recommander les items qui les correspondent pour une certaine période de temps **[SHA et GUN, 09],** telle que la recommandation d'articles de presse ou des histoires connexes dans les journaux en ligne **[KAW *et al.*, 07].**

1.11.13. L'Evolutivité

Les systèmes de recommandation sont conçus pour être utilisés dans des applications à grande échelle avec de grandes bases de données, qui augmentent grandement en taille avec le temps en ajoutant de nouveaux items et de nouveaux utilisateurs. Par conséquent, ces systèmes doivent élaborer des algorithmes performants qui les permettent d'augmenter leur performance en précision, en temps de réponse (i.e. temps pris pour fournir une recommandation) **[HER *et al.*, 02] [SAR *et al.*, 01],** et en espace mémoire **[KAR 01],** tout en considérant les nouvelles données ajoutées à la base.

L'évaluation de l'évolutivité d'un système de recommandation se fait en calculant l'évolution de la précision, ou bien la vitesse du système pour fournir les recommandations (temps nécessaire) sur différents ensembles de données de tailles différentes (du plus petit au plus grand ensemble de données). Pour plus d'information sur l'évolutivité dans les SRs et les méthodes utilisées pour l'assurer, veuillez se référencer à **[SAR *et al.*, 02][BEL et KOR, 07] [KOR 10] [KOR 12] [TAK *et al.*, 09].**

1. 12. Synthèse des approches de recommandation

A la fin de ce chapitre, nous présentons dans un tableau récapitulatif (*Tableau 1.2)* les principales approches de recommandation, sachant la méthode du FC, FBC et les approches hybrides, en citant les méthodes utilisées par chaque approche ainsi que les différentes métriques employées pour l'évaluation des SRs.

<table>
<tr><th colspan="3"></th><th>Méthodes et Métriques utilisées</th></tr>
<tr><td rowspan="3">Les types des SRs</td><td rowspan="2">Systèmes basés Filtrage Collaboratif</td><td>Méthodes basées Modèle</td><td>- Réseaux Bayésiens
- Systèmes flous
- Méthode de diagnostique de personnalité
- Méthodes de FM
- Algorithmes génétiques
- RNA
- SVM
- Les arbres de décision
- Apprentissage en profondeur
- Clustering
✓ Clustering hiérarchique (Ascendant, Descendant)
✓ Clustering par partitionnement (K-means, FCM)</td></tr>
<tr><td>Méthodes basées Mémoire</td><td>- La normalisation des notes
Le centrage moyen
La normalisation Z-score
- Calcul de la similarité
Corrélation Cosinus
Corrélation de Pearson
Corrélation Cosinus Ajustée
Corrélation de Pearson avec Contrainte
La Différence Quadratique Moyenne
Corrélation du rang de Spearman
- Sélection du voisinage
Pré-filtrage des voisins
✓ Filtrage Top-N
✓ Filtrage par seuil
✓ Filtrage négatif
KPPV</td></tr>
<tr><td rowspan="2">Systèmes par Filtrage Basé Contenu</td><td>Méthodes pour la représentation des items</td><td>- Modèle d'espace vectoriel</td></tr>
<tr><td>Méthodes pour l'apprentissage du profil de l'utilisateur</td><td>- Modèle vectoriel
- Retour de pertinence classificateur Naïf de Bayes
- Les arbres de décision
- KPPV</td></tr>
</table>

	Systèmes Hybrides	- Hybride Pondéré - Hybride Mixte - Hybride en Cascade - Hybride par commutation - Hybride par Méta-Niveau - Hybride par combinaison de caractéristiques - Hybride par Augmentation de caractéristiques
Evaluation des SRs	**Evaluation de la prédiction**	- Erreur Moyenne Absolue (MAE) - Erreur Quadratique Moyenne (MQE) - Erreur Moyenne Absolue Normalisée (NMAE)
	Evaluation de la recommandation	- Précision - Rappel - F1
	Nouveaux critères d'évaluation	- Couverture - Confidence - Trust - Sérendipité (le hasard) - Nouveauté - Diversité - Utilité - Robustesse - Confidentialité - Adaptabilité - Evolutivité

Tableau 1.2: Synthèse des approches de recommandation

1. 13. Conclusion

Nous avons présenté dans ce chapitre, le domaine des systèmes de recommandation, les principales notions liées, ses approches et ses problèmes ainsi que les mesures d'évaluation utilisées pour valider la performance des prédictions et des recommandations générées. Pour plus de détails sur les systèmes de recommandation veuillez se référencer à **[ADO *et al.*, 05][ADA 13][BOB *et al.*, 13][SU et KHO, 09]**et **[LEE *et al.*, 12].**

Cependant, ces dernières années de nouveaux critères et propriétés ont été présentés afin d'augmenter la performance et la faisabilité des SRs, incluant la nouveauté et la diversité dans les listes de recommandation, le hasard, la couverture, la confiance, le

trust, l'utilité, la robustesse, la confidentialité, l'adaptabilité et l'évolutivité **[SHA et GUN 11].**

Dans ce manuscrit, nous nous sommes intéressés aux deux critères d'évaluation : l'adaptabilité des systèmes de recommandation et la diversité dans les listes de recommandation. Ainsi, nous présenterons dans le chapitre suivant un état de l'art des travaux connexes sur les méthodes et métriques de la diversité, ainsi que ceux liés aux approches proposées.

CHAPITRE 2

TRAVAUX CONNEXES

RESUME DU CONTENU

Ce chapitre est divisé en trois grandes parties : la première partie présente un tour d'horizon sur les systèmes de recommandation, puis un état de l'art sur les algorithmes du filtrage collaboratif utilisant les méthodes de *clustering* des utilisateurs et des items, sera présenté dans la section deux. Nous nous sommes intéressés aux méthodes de *clustering* flou FCM et de factorisation matricielle et leur utilisation dans le filtrage collaboratif. La troisième partie contient un état de l'art sur l'utilisation de la notion de diversité dans les systèmes de recommandation afin d'améliorer la qualité des recommandations et d'évaluer leur pertinence pour les utilisateurs.

2.1. Introduction

Nous allons présenter dans ce chapitre un état de l'art sur les systèmes de recommandation, le *clustering* des utilisateurs et des items dans ces systèmes, et l'incorporation de la notion de la diversité dans la liste de recommandation avec une panoplie de travaux réalisés dans la littérature.

2.2. Les Systèmes de Recommandation

Les Systèmes de Recommandation (SRs) **[RES et VAR, 97]** identifient automatiquement les préférences des utilisateurs par le biais de leurs interactions avec le système en se basant sur le feedback implicite [**Hu *et al.*, 08] [SAL, 13] [SAL *et al.*, 13]** ou feedback explicite **[BOB *et al.*, 10]** ou les deux **[SHA et SUM, 13] [SAL et KAM, 13],** pour leur suggérer des recommandations en utilisant le filtrage d'information. Pour faire des recommandations, un certain nombre d'approches ont été identifiées **[BUR, 02],** où les plus utilisées sont basées sur le filtrage collaboratif (FC) **[BAL et SHO, 97],** et sur le filtrage basé Contenu (FBC) **[MOO et ROY, 99] [PAZ et BIL, 07].** Les deux techniques ont leurs forces et faiblesses **[BUR, 02],** où l'hybridation entre eux a été rapidement adoptée pour profiter de leurs avantages.

Les systèmes de recommandation sont construits généralement en se basant sur deux types de méthodes différentes qui sont le filtrage basé sur le contenu (FBC) et filtrage collaboratif (FC) **[BUR 02].** L'approche basée contenu génère des recommandations de contenu en fonction des caractéristiques des utilisateurs ou des items, comme dans le système de recommandation présenté par Sarwar et al. **[SAR *et al.*, 01],** où pour un item donné, les items similaires évalués par l'utilisateur actif dans le passé sont identifiés et utilisés pour la recommandation. Cette corrélation est appelée la corrélation «item-à-item» **[BUR 02].**

Les algorithmes du FC fournissent des recommandations ou des prédictions pour les items sur la base des opinions des voisins de l'utilisateur actif. Ces avis peuvent être obtenus explicitement ou implicitement par l'utilisation de certaines mesures. Cette corrélation est appelée une corrélation «peuple-à-peuple» ou corrélation «utilisateur-à-utilisateur» **[BUR 02].**

La méthode du FC utilise seulement les évaluations effectuées par les utilisateurs similaires sur les articles pour prédire les évaluations inconnues des nouvelles paires utilisateur-item. Le FC utilise le principe que deux utilisateurs probablement continuent à choisir des items similaires s'ils ont déjà choisi des items similaires, et il est devenu plus populaire que l'approche basée sur le contenu ces dernières années, en raison de son rendement général supérieur.

Les premiers travaux dans le domaine du FC ont été publiés à la fin des années 1990, où Goldberg *et al.,* ont présenté dans **[GOL *et al.,* 01],** le système *Tapestry* qui utilise le filtrage collaboratif pour filtrer les courriers à partir de plusieurs listes de diffusion (*mailing lists*), simultanément, en utilisant les opinions des autres utilisateurs sur les lectures. Ces deux techniques ont leurs forces et leurs faiblesses **[BUR 02],** où les approches hybrides sont rapidement adoptées pour résoudre les problèmes individuels de chaque approche par les avantages de l'autre **[BUR 02].** Néanmoins, les résultats de l'hybridation ne sont pas suffisamment efficaces pour résoudre tous les problèmes des systèmes de recommandation.

L'hybridation entre les approches du FC et du FBC a fait l'objet d'intérêt dans plusieurs travaux sur les systèmes de recommandation, afin de profiter de leurs avantages **[BUR 02][WEN *et al.,* 08].** Une bonne étude sur le domaine des systèmes de recommandation, y compris les méthodes de recommandation utilisées, est présentée dans **[ADO et TUZ, 05]** et **[Su et KHO, 09][BOB *et al.,* 13].**

Le filtrage collaboratif vise à prédire les évaluations d'un utilisateur test pour de nouveaux items basés sur une collecte d'informations d'autres utilisateurs ayant les mêmes idées. Il suppose que les utilisateurs partageant les mêmes évaluations sur les items passés ont tendance à s'entendre sur de nouveaux items.

Jusqu'à présent, la recherche sur le filtrage collaboratif peut être principalement classée en deux catégories: le filtrage à base de mémoire **[RES *et al.,* 94], [SAR *et al.,* 01]** et le filtrage à base de modèle**[BIL et PAZ, 00].** Les Méthodes basées sur la mémoire considèrent l'information de voisinage utilisant des méthodes fondées sur la similarité entre les items **[SAR *et al.,* 01] [LIN *et al.,* 03] [DES et KAR 04],** ou entre les utilisateurs **[BRE *et al.,* 98] [HER *et al.,* 99] [JIN *et al.,* 04] [RES *et al.,* 94] [SCH *et al.,* 99] [BOB *et al.,* 10], [BOB *et al.,* 11], [SHA et LU, 11],** ou une fusion entre eux **[DHI 01] [HOF and PUZ 99] [WAN *et al.,* 06].**

Ces méthodes, d'abord calculent les similitudes entre l'utilisateur actif et d'autres utilisateurs (basées utilisateur), ou entre l'item test et autres items (basées items), et de les appliquer pour identifier les *K* plus proches voisins de l'élément actif. Ensuite, l'évaluation inconnue est prévue en combinant les évaluations connues des voisins. Deux défis majeurs pour les méthodes à base de mémoire sont l'évolutivité et la clairsemé de données.

Pour le premier problème, ces algorithmes reposent sur des corrélations exactes des deux vecteurs utilisateurs/item, ce qui conduit ces algorithmes de sacrifier la couverture et la précision des systèmes de recommandation. Plus concrètement, depuis le coefficient de corrélation n'est défini qu'entre les utilisateurs qui ont évalué au moins deux items en commun, ou les items qui ont été achetés, puis de nombreuses paires utilisateur/items n'auront pas de corrélation à tous. En conséquence, les systèmes de recommandation basés sur la mémoire ne peuvent pas calculer exactement le voisinage et identifier les items à recommander, qui va certainement mener à des recommandations pauvres.

Concernant le deuxième problème, dans la pratique, le nombre d'utilisateurs et des items à la fois peuvent être assez importants (souvent des millions d'utilisateurs et des milliers d'items), cela risque de ralentir la procédure de recommandation de manière significative car les algorithmes basées sur les *K* plus proche voisins exigera trop de calculs dans ce cas **[SAR *et al.*, 01].**

Pour pallier les limitations des méthodes à base de mémoire, les approches basées sur des modèles ont été proposées, qui établissent un modèle, en utilisant les évaluations observées, capable d'interpréter les données fournies et de prédire les évaluations inconnues. Des exemples de cette catégorie comprennent les arbres de décision **[BRE *et al.*, 98]**, les réseaux bayésiens [**BRE *et al.*, 98],** les modèles de *clustering* **[UNG and FOS 98], l**es modèles à facteur latent **[CAN 02],** les modèles aspect **[HOF and PUZ 99],** les méthodes de réduction de la dimension **[GOL *et al.*, 01],** le modèle à base de factorisation matricielle [**DIN *et al.*, 06] [SRE *et al.*, 05] [KOR 09] [SAL and MNI 08]** etc.

Généralement, les algorithmes basés modèles sont considérés comme plus efficaces que ceux à base du mémoire (voisinage). Cependant, tout comme l'a souligné **[XUE *et al.*, 05],** la production et la mise à jour d'un modèle nécessitent beaucoup de temps car il y a généralement beaucoup de paramètres libres à régler. Donc, l'hybridation entre ces deux

types du FC est recommandée car ils sont souvent complémentaires et la meilleure performance est obtenue en les combinant afin d'éliminer les inconvénients d'une méthode par les avantages de l'autre **[BUR 02] [BEL *et al.,* 07] [KOR 08].**

Les algorithmes du FC basés mémoire font des prédictions ou des recommandations aux utilisateurs sur la base des avis des voisins, qui nécessitent la recherche de voisins candidats à partir de l'ensemble total des utilisateurs en utilisant des mesures de similarité comme la Corrélation Cosinus et la Corrélation de Pearson **[BRE *et al.,* 98] [HER *et al.,* 99].**

Pour améliorer les méthodes de sélection du voisinage dans le FC basé mémoire, les chercheurs ont été menés vers la proposition de nouvelles mesures de similarité. **[CHO et SUH, 13]** ont proposé une nouvelle mesure de similarité qui calcule différents voisinages pour différents items, en ajoutant la similarité entre les items comme poids à la similarité basée utilisateur. **[BOB *et al.,* 11]** ont proposé une mesure de similarité pondérée qui utilise les algorithmes génétiques pour calculer les poids afin d'améliorer les performances de recommandation. Citant également les mesures proposées dans **[BOB *et al.,* 10] [LIU *et al.,* 14]** qui combinent des informations contextuelles avec les valeurs d'évaluation et la préférence globale, respectivement. **[SUN *et al.,* 12]** ont proposé une mesure de similarité appelé Distance de l'Opérateur Uniforme de Jaccard (*Jaccard Uniform Operator Distance JacUOD)* dédiée à être utilisée dans les espaces vectoriels multidimensionnels.

D'autres chercheurs ont utilisé l'entropie de l'information, pour trouver les voisins comme mesure de similarité dans les situations de démarrage à froid **[ZHA *et al.,* 11],** ou prendre en considération sa valeur lors du calcul des similitudes basées-utilisateur et basées-item **[PIA *et al.,* 09].** Récemment, Kaleli **[KAL 14]** a proposé une nouvelle mesure de similarité pour sélectionner les voisins, qui combine la similarité avec l'entropie entre les entités.

Les méthodes basées sur les K-plus proches voisins regroupent les utilisateurs en calculant les similarités entre eux, ce qui nécessite un temps de calcul énorme. Cependant, les méthodes à base de *clustering* (méthode basée modèle) réduit les temps de calcul en introduisant les modèles de regroupement.

Les méthodes basées sur des modèles suggèrent un modèle pour représenter les items ou les utilisateurs et les relations entre eux. Plusieurs méthodes basées sur des modèles ont

été proposées, où certaines des méthodes efficaces et réussites sont celles qui utilisent la factorisation matricielle (FM) pour représenter les utilisateurs et les articles tels que des facteurs ou des matrices résultantes après la factorisation de la matrice d'évaluations utilisateur-article. Les techniques de factorisation matricielle ont été largement appliquées au filtrage collaboratif [**CAN 02], [SAL et MNI, 08], [SRE *et al.*, 05], [WIT *et al.*, 09], [GU *et al.*, 10], [HOF 04], [BEL et KOR, 07], [KOR *et al.*, 09], [NGU et SHU, 13] [PIL *et al.*, 10]** et **[LUO *et al.*, 13],** ce qui prouve leur efficacité dans les recommandateurs basés FC **[YIN et PEN, 12].**

Comme nous l'avons déjà mentionné, les algorithmes basés sur les modèles augmentent la performance du système, dont les méthodes les plus performantes et les plus utilisées à travers la littérature sont le *clustering* et la factorisation matricielle. Nous nous sommes intéressés dans le présent manuscrit avec ces deux méthodes. Nous présentons dans les sections suivantes quelques travaux de la littérature qui ont utilisé la factorisation matricielle et\ou le *clustering* (plus précisément le *clustering* flou et la factorisation en matrice non négative), dont quelques uns constituent les fondements de notre première approche proposée.

2.3. Le clustering dans le Filtrage Collaboratif

Le *clustering* a été utilisé dans les systèmes de recommandation pour regrouper les items **[OCO et HER, 09] [GON 10]**, les utilisateurs **[XUE *et al.*, 05][GON 10]**, ou bien les deux en même temps (*bi-clustering*) [**ZHU et GON, 09][GEO et MER, 05].** Les types de *clustering* les plus utilisés dans les systèmes de recommandation sont le *clustering* hiérarchique ascendant **[XU et WUN, 05],** qui a été élaboré dans **[SCH et FAL, 07][PAR et SHE, 14][ZHE *et al.*, 13],** et les algorithmes de *clustering* par partitionnement : *k-means* **[KIM et YAN 05][BRI et KEL, 02][XUE *et al.*, 05]** et FCM **[LIN et al., 03][FAN et LIU, 03][WU et LI, 08][REN et al., 11]**. D'autres méthodes et travaux connexes sur le *clustering* flou seront présentés ci-dessous**.**

2.3.1. La méthode du lissage basé cluster (Cluster-based Smoothing)

Dans [**XUE *et al.*, 05**], la notion d'affectation d'un utilisateur à plusieurs groupes est présentée mais sans l'utilisation de l'algorithme FCM. Xue *et al*. ont proposé une méthode de lissage basé cluster qui regroupe les utilisateurs à l'aide de l'algorithme *K*-

means d'abord, et prédit ensuite toutes les données manquantes sur la base des évaluations des top-*N* utilisateurs les plus similaires dans les groupes similaires.

La simulation de cette méthode a montré que cette méthode a généré de meilleurs résultats que d'autres algorithmes de filtrage collaboratif. Cependant, la méthode basée cluster limite la diversité des utilisateurs de chaque cluster, et les résultats de regroupement des *K-means* s'appuie sur les K utilisateurs présélectionnés. En outre, si un utilisateur ne possède pas suffisamment d'utilisateurs similaires, l'algorithme Top-N génère par la suite un grand nombre d'utilisateurs différents, qui feront certainement diminuer la précision de la prédiction pour des utilisateurs actifs.

Pour les problèmes du FC, il semble généralement plus raisonnable de permettre que les utilisateurs appartiennent à des classes différentes. FCM prend cette idée et classifie chaque utilisateur dans des classes différentes avec des probabilités appropriées. Dans la sous section suivante, nous présentons quelques travaux de la littérature qui utilisent le *clustering* flou dans la phase du filtrage collaboratif.

2.3.2. L'algorithme FCM dans les systèmes de recommandation

Dans les systèmes de recommandation, les utilisateurs peuvent évaluer des articles avec des goûts différents, ce qui rend difficile la décision sur leurs profils. Une idée naturelle pour faire face à cette situation est d'affecter les utilisateurs à différents groupes correspondant à leurs différents goûts et intérêts. L'algorithme des C-moyenne floues (Fuzzy C-Means FCM) **[BEZ, 73]** est adéquat à cette idée, car il regroupe les utilisateurs dans différents groupes avec des probabilités appropriées. L'algorithme FCM est largement utilisé dans les systèmes de recommandation pour regrouper les utilisateurs et les items [**FAN et LIU 03]**, **[WAN *et al.*, 10]**, **[REN *et al.*, 11],** mais en utilisant des versions modifiées de l'algorithme FCM pour faire face aux problèmes de l'évolutivité et la clairsemée des données**.**

2.3.2.1. La méthode de Lingras, Yan et West

[LIN *et al.*, 03] ont utilisé l'algorithme FCM pour segmenter les utilisateurs du web. Ils ont appliqué cet algorithme sur trois sites éducatifs analysés antérieurement par Lingras et al. **[LIN *et al.*, 02].** Les groupements flous qui en résultent également fournissent une représentation raisonnable des comportements des utilisateurs pour les trois sites web.

2.3.2.2. Un Système de Recommandation flou pour le web

Un exemple de système de recommandation qui utilise l'algorithme FCM est le système de recommandation des pages web proposé par Fang et Liu **[FAN et LIU, 03],** qui utilise l'algorithme FCM pour regrouper les utilisateurs et les items (les pages web) dans les approches basées utilisateurs et basées items.

2.3.2.3. Un Système de Recommandation Flou pour les e- Élections (FRS)

Terán et Meier **[TER et MEI 10]** ont proposé un système de recommandation flou (*Fuzzy Recommender System FRS*) pour la recommandation dans le domaine des élections électroniques (*e-Elections*) en utilisant un algorithme de classification basé flou. Le système fournit des informations sur les profils aléatoires des candidats les plus proches de l'électeur (l'utilisateur), et une distribution des parties politiques qui sont organisées en clusters flous. Pour éviter le problème de l'initialisation aléatoire de la matrice des centres de clusters, le système FRS génère des clusters flous en utilisant une version modifiée de l'algorithme FCM (FCM modifié), qui initialise la matrice des centres avec un membre aléatoire de chaque partie politique. La sortie est une matrice de partition floue qui contient le degré d'appartenance de l'électeur et des candidats par rapport à chaque groupe, et fournit une représentation graphique des parties politiques distribuées dans les clusters, pour aider les citoyens à analyser le comportement des politiciens sur la base de similitudes entre eux.

2.3.2.4. Un Système de Recommandation Automatique basé sur FCM

Gao Ren et al. **[REN *et al.*, 11],** ont proposé un nouveau système de recommandation automatique basé sur l'algorithme FCM (ARS), les ensembles flous et la théorie des ensembles approximatifs, y compris trois étapes principales: la discrétisation des données, établissement des règles et le raisonnement flou. Le système utilise l'algorithme FCM pour regrouper l'espace d'entrée. Les degrés d'appartenance sont utilisés avec des attributs pour atteindre la discrétisation des données, en vue de surmonter l'inconvénient de la discrétisation des données d'entrée par le regroupement à chaque fois. Les règles de raisonnement flou avec l'explosion combinatoire ont été résolues en utilisant les ensembles approximatifs afin de réduire les attributs des produits, pour accélérer la vitesse du raisonnement flou.

2.3.2.5. Un Système de Recommandation personnel hybride avec FCM Modifié

Shinde et Kulkarni **[SHI et KUL 11],** ont proposé un système de recommandation personnalisé hybride flou en utilisant une version modifiée de l'algorithme FCM

(*Modified FCM Hybrid Personal Recommender System MFCMHPRS*) pour regrouper les éléments de la matrice utilisateur-item dans une phase de prétraitement. Une méthode d'initialisation de la matrice des degrés d'appartenance a été proposée, où les centres des groupes initiaux sont calculés au lieu de les sélectionner de façon aléatoire. Une version modifiée de la similarité cosinus a été proposée pour calculer la similarité entre l'utilisateur actif et les différents groupes en introduisant un coefficient pour pondérer la similarité de l'utilisateur. Le coefficient calcule le compromis entre le nombre d'items communs notés par deux utilisateurs et le nombre total des items en commun. Ensuite, la prédiction des évaluations est calculée en utilisant l'approche basée sur l'utilisateur, et par la suite la recommandation des Top-N items est faite.

2.3.2.6. FCM Exponentiel

Un algorithme de classification floue exponentielle appelé (*XFCM pour eXponential FCM*) a été proposé dans [**TRE et JAR, 12**], en reformulant la fonction objective de l'algorithme FCM avec une équation exponentielle afin de regrouper les items. Où, la fonction d'objective a été formulé sur la base qu'elle soit égale à 1 si les données appartiennent au groupe et égale à 0 sinon, et elle est considérée comme une fonction de distance pour attribuer les items aux clusters. La prédiction des évaluations manquantes produites par l'algorithme (XFCM) se fait en fonction de la similarité entre les données et les centres des clusters, en tenant compte seulement des groupes pertinents.

Dans la section suivante, nous présentons quelques travaux de la littérature sur les méthodes de factorisation matricielle dans les systèmes de recommandation.

2.3.3. La méthode de factorisation matricielle dans les systèmes de recommandation

Les méthodes de factorisation matricielle ont été largement utilisées dans les systèmes de recommandation, et ont prouvé de très bonne performance vis-à-vis de la réduction des effets des deux problèmes du filtrage collaboratif, qui sont la parcimonie et l'évolutivité de données, en plus de l'amélioration de la performance des prédictions des évaluations. Nous citons dans cette section quelques travaux qui ont utilisé les méthodes de factorisation matricielle dans le FC et ont montré l'efficacité de ces méthodes dans le FC.

En raison de son efficacité dans le traitement d'énormes ensembles de données, le modèle à base de factorisation matricielle est devenu l'un des modèles les plus populaires parmi les méthodes fondées sur un modèle, citant par exemple la décomposition en valeurs singulières (*singular value decomposition SVD*) **[KOR, 08],** la factorisation en matrice à rang faible pondéré (*Weighted Lower rank Matrix Factorization WLRMF*) [**SRE and JAA 03**], la factorisation en matrice non négative pondérée (*Weighted Nonnegative Matrix Factorization WNMF*) [**ZHA *et al.*, 06**], la factorisation en matrice à marge maximale (*Maximum Margin Matrix Factorization MMMF*) [**SRE *et al.*, 04**], la factorisation en matrice probabiliste (*Probabilistic Mtarix Factorization PMF*) [**SAL and MNI 07**] et la factorisation par tri des matrices probabilistes (*Probabilistic Matrix Tri Factorization PMTF*)[**YOO and CHO 09**].

L'une des réalisations les plus réussies des modèles à facteurs latents utilisant la factorisation matricielle est celle présentée par Koren **[KOR 09],** qui a prouvé que la technique de factorisation matricielle est devenue une méthodologie dominante dans les systèmes de recommandations à base du filtrage collaboratif. Mais, en réalité pour faire de bonnes prédictions et pour améliorer la performance des recommandeurs basés sur le filtrage collaboratif, les méthodes de factorisation matricielle sont fusionnées avec les méthodes basées sur la mémoire **[BEL *et al.*, 07],** en citant comme exemple :

2.3.3.1. Le modèle SVD++

Koren dans **[KOR 08]** a proposé un modèle combiné nommé SVD++ qui améliore la précision de la prédiction en tenant compte des informations implicites, représentées par l'ensemble des items qui ont été évalués indépendamment de la valeur de leur note.

2.3.3.2. Hybridation entre FM et le FC basé voisins [HMF]

Afin d'augmenter la performance des schèmes hybrides, différentes variantes des méthodes de factorisation matricielle et du filtrage collaboratif basé voisins ont été proposées par Takacs et al. **[TAK *et al.*, 08],** en examinant différents scénarios de régularisation pour la méthode de factorisation matricielle. Ensuite, deux méthodes du filtrage collaboratif basé voisins sont introduites : l'une basée sur les coefficients de corrélation et l'autre sur les moindres carrés linéaires.

2.3.3.3. Filtrage Collaboratif basé sur la Factorisation par Tri des Matrices Non-négatives Orthogonales (FCBFTMNNO)

Bien que les informations internes du voisinage aient été largement utilisées dans des procédés basés sur la mémoire, elles sont rarement utilisées dans les méthodes fondées

sur un modèle. Pour en profiter des capacités prouvées des méthode de factorisation matricielle et des informations internes, nécessaires pour la bonne prédiction des évaluations manquantes des items pour un utilisateur actif, [**CHE *et al.,* 09**] ont proposé un nouveau cadre pour le filtrage collaboratif, afin de réduire les problèmes des approches à base de mémoire et à base de modèle, en appliquant la factorisation par tri en matrice non négative orthogonale (*Orthogonal non negative Matrix Tri-factorization ONMTF*) [**DIN *et al.,* 06**].

L'algorithme applique pour la première fois la factorisation par tri en matrice non négative orthogonale pour regrouper les lignes et les colonnes de la matrice utilisateur-item simultanément. Puis, le calcul de similarité entre l'utilisateur actif et les centres des classes est effectué en utilisant l'information incluse dans les matrices obtenues. Après avoir les différentes similarités, une sélection des C plus proches groupes est faite afin de sélectionner les K-plus proches voisins de cet utilisateur de la même façon que dans [**XUE *et al.,* 05**]. Pour obtenir une bonne prédiction des évaluations inconnues, une combinaison linéaire des trois prédictions (basée modèle, basée utilisateur et basée item) a été proposée.

Cet algorithme possède les caractéristiques suivantes:

1. Le problème de clairsemé a été contourné en raison de l'application de la factorisation matricielle.

2. Le problème d'évolutivité a été atténué par l'application de la technique de factorisation ONMTF, puisque les utilisateurs et les items sont regroupés en même temps.

3. Cette méthode fusionne les résultats de prédiction des différents types d'algorithmes (approche basée modèle et basée mémoire), qui a donné de meilleures performances en fonction des résultats des expériences.

Cependant, un inconvénient pour cet algorithme est l'utilisation des coefficients de fusion qui contrôlent les valeurs des poids, et qui nécessitent beaucoup de temps de calcul pour trouver les bonnes valeurs de coefficients qui donnent la meilleure prédiction. En plus, d'après le modèle proposé, ils ont donné beaucoup d'importance à la prédiction des ONMTF alors que les informations sur les utilisateurs et sur les items ont aussi une grande importance pour la prédiction des évaluations manquantes dans le processus du FC.

2.3.3.4. Un Graphe Régularisé Pondéré par Factorisation en Matrices Non négatives

Ainsi que les informations internes possèdent un important intérêt pour la prédiction des évaluations manquantes, les informations externes telles que les informations démographiques de l'utilisateur, et les genres des items jouent un rôle très important dans les systèmes de recommandation basés sur le contenu. Par exemple, les utilisateurs des professions similaires peuvent avoir des intérêts similaires des produits, les films du même genre peuvent être appréciés de façon similaire par les gens. Néanmoins, cette information externe est généralement négligée dans les systèmes basés sur le filtrage collaboratif.

Sur la base des observations précédentes, [**GU *et al.*, 10**] ont proposé un modèle unifié du filtrage collaboratif basé sur un graphe régularisé pondéré par factorisation en matrice non négative (*a graph regularized weighted nonnegative matrix factorization GRWNMF*). Ils ont construis deux graphes sur les utilisateurs et les items, respectivement, pour exploiter les informations internes et externes. Ensuite, ils ont présenté un modèle basé sur un graphe régularisé pondéré par factorisation avec tri en matrice non négative (*a graph regularized weighted nonnegative matrix tri-factorization model*) comme une extension, qui est plus approprié pour la régularisation graphique sur les items et les utilisateurs simultanément. Les expériences sur des ensembles de données de référence ont démontré que les méthodes proposées ont une meilleure performance que les méthodes du filtrage collaboratif existantes.

Ainsi, cette méthode hérite des avantages des méthodes fondées sur un modèle, et possède également les avantages des méthodes à base de mémoire qui tiennent compte de l'information sur le voisinage. En outre, cette méthode a la capacité d'utiliser les informations démographiques de l'utilisateur et les informations sur le genre d'items qui sont utilisés par les systèmes de recommandation basés sur le contenu et toute information supplémentaire concernant les réseaux utilisateur-utilisateur. En raison de l'utilisation de ces informations, cette méthode est capable de trouver une représentation de basse dimensionnalité plus facilement interprétable pour les utilisateurs et les items, qui, améliorent les performances de recommandation. Par contre, cette méthode nécessite beaucoup de paramètres pour le calcul et la régularisation des graphes.

Il a été montré dans [**DIN *et al.*, 05]**, **[DIN *et al.*, 06]** que l'algorithme de factorisation en matrice non négatives est équivalent à une forme détendue de l'algorithme *K-means* lors

de l'utilisation des moindres carrés comme fonction objective de l'algorithme NMF, en considérant le premier facteur de la matrice comme centres de gravité du cluster et le second comme des indicateurs d'appartenance au groupe . Ce qui donne un fondement théorique pour l'utilisation de la méthode de factorisation en matrices non négatives NMF pour le *clustering* des données.

Certaines approches puissantes ont été développées pour résoudre les problèmes du filtrage collaboratif en combinant entre les idées des méthodes à bases de factorisation matricielle FM et d'autres algorithmes de *clustering* tels que *k-means* et les *Fuzzy C-means* (FCM) etc. Un exemple typique est le modèle du FCM modifié (*Modified FCM MFCM*) proposé par **[WU et LI 08]** qui combine entre les deux méthodes FM et FCM.

2.3.3.5. Algorithme FCM Modifié

L'idée dans cet algorithme est la transformation de la fonction objective de l'algorithme FCM en une fonction objective de la FM en ajoutant des termes de pénalités pour gérer les contraintes de la fonction objective de FCM, ensuite la minimisation de la fonction objective du FCM revient à résoudre la fonction objective du FM en utilisant la descente du gradient au moment 0.

WU et LI **[WU et LI 08]** ont développé deux extensions de cet algorithme MFCM 1 et MFCM2 pour réaliser MFCM. Dans MFCM1, ils ont utilisé un terme de pénalité λ pour pénaliser les paramètres $z_{u,k}$quand ils ne remplissent pas les contraintes c.à.d. proches de 0. Parce que pour un élément donné, un indice proche de 1 indique une forte association au cluster et un indice proche de 0 indique une petite association au cluster correspondant. Ainsi l'objectif des termes de pénalisation λ est de diminuer les paramètres pour atténuer le sur-apprentissage et de contraindre les paramètres à satisfaire les contraintes. La fonction objective de l'algorithme MFCM1 est la suivante :

$$H_1(Z,C) = \frac{1}{2}\sum_{(u,m)\in P}[(r_{u,m} - z_u c_m)^2 + \lambda \left|\left|c_m\right|\right|_2^2 + \lambda(||z_u||_2^2 + (z_{u1}-1)^2 + ||\mathrm{z_{u-}}||_2^2] \quad (2.1)$$

Cette fonction est une version modifiée de la fonction objective de factorisation matricielle, c'est pourquoi les méthodes utilisées pour la minimisation de la fonction de FM peuvent être utilisées pour minimiser la fonction objective (2.1).

Dans l'algorithme MFCM2, ils ont remplacé $z_{u,k}$ par la probabilité d'appartenance de l'utilisateur *u* au groupe *k* comme suit

$$\widetilde{H}(Z,C) = \sum_{(u,m)\in P}(r_{u,m} - \sum_{k=1}^{K} p_{u,k}c_{k,m})^2 \quad (2.2)$$

Où $p_{u,k} = e^{z_{u,k}} / \sum_{l=1}^{K} e^{z_{u,l}}$. $P=(p_{u,k})$ satisfait toutes les contraintes $Z_1 = 1$; $Z \geq 0$ automatiquement. $z_{u,k}$est régularisé vers 0 puisque $z_{u,k}$= 0 (k = 1,..., K) signifie que l'utilisateur *u* appartient à chaque cluster avec la même probabilité. Lorsque cela est pris en considération, la fonction objective finale est la suivante :

$$H_2(Z,C) = \frac{1}{2}\sum_{(u,m)\in P}(r_{u,m} - \frac{1}{\sum_{k=1}^{K} e^{z_{u,k}}}\sum_{k=1}^{K} e^{z_{u,k}} c_{k,m})^2 + \lambda \left||c_m|\right|_2^2 + ||z_u||_2^2 \quad (2.3)$$

Les deux algorithmes ont été minimisé en utilisant la méthode de descente du gradient au moment 0, et ils ont montré de bonnes prédictions avec convergence rapide que l'algorithme FCM et une précision comparable à celle des FM mais avec une meilleur interprétation, ce qui prouve l'efficacité des méthodes de factorisation matricielle pour les problèmes du FC. Cependant, il a été démontré dans **[LAN *et al.*, 06]** qu'il existe d'autres méthodes de minimisation de la fonction objectives de la FM beaucoup plus efficaces que la méthode de la descente du gradient utilisée dans **[WU and LI, 08]**.

2.3.4. Synthèse des travaux sur le clustering dans le FC

Dans le *Tableau 2.1*, nous présentons un résumé des méthodes présentées ci-dessus.

Critère / Référence	Clustering		FCB Factorisation	FCB mémoire
	Flou	Dur		
Cluster-based Smoothing **[XUE *et al.*, 05]**	-	×	-	×
[FAN et LIU, 03]	×	-	-	-
[WAN 10]	×	-	-	-
ARS **[REN *et al.*, 11]**	×	-	-	-
méthode de Lingras, Yan et West **[LIN *et al.*, 03]**	×	-	-	-
FRS **[TER et MEI, 10]**	×	-	-	-
MFHPRS **[SHI et KUL, 11]**	×	-	-	-
ARS **[TER et JAR, 12]**	×	-	-	-
CFONMTF **[CHE et al., 09]**	-	-	×	×
[TAK *et al.*, 08]	-	-	×	×

***GRWNMF* [GU *et al.*, 10]**	-	-	×	-
***Temporal Dynamics* [KOR 09]**	-	-	×	-
***SDV*++[KOR 08]**	-	-	×	-
[BEL et KOR, 07]	-	-	×	×
***MFCM* [WU et LI, 08]**	×	-	×	-

Tableau 2.1. Résumé des méthodes du FC basé sur le Clustering et sur la FM dans les SRs

2.3.5. Discussion

Une synthèse des travaux de la littérature sur les méthodes du filtrage collaboratif a été présentée dans les sections ci-dessus, incluant les nouvelles mesures de similarité conçues pour améliorer le FC basé sur le voisinage.

Nous avons vu que la majorité des travaux ont utilisé un FC basé modèle pur, ce qui dégrade la précision et la pertinence des items recommandés. Néanmoins, quelques uns ont hybridé entre le FC basé modèle et basé voisinage **[XUE *et al.*, 05][KOR 08][CHEN *et al.*, 09][TAK *et al.*, 08]** et **[GU *et al.*, 08],** dont ils se basent sur un FM et *k-means* pour le regroupement des utilisateurs et des items. Ces méthodes ont apporté des améliorations à la précision de la prédiction car elles incluent l'information sur le voisinage, en sélectionnant l'ensemble candidat des voisins pour être employé dans le processus de prédiction. L'inconvénient de ces méthodes est qu'elles réalisent un regroupement des utilisateurs sans la considération de l'ambiguïté dans les préférences des utilisateurs, comme il est le cas avec l'algorithme FCM.

Une méthode performante est celle proposée par **[WU et LI, 08]** parce qu'elle profite des avantages de l'algorithme FCM, qui permet l'affectation de l'utilisateur à différents groupes avec des degrés d'appartenance, et la méthode de FM qui réduit énormément la taille de la matrice et la clairsemé de données. Cependant, les algorithmes proposés dans **[WU et LI, 08]** sont basés sur le modèle seulement, ce qui rend l'hybridation de cette méthode avec l'algorithme du FC basé sur les plus proches voisins un point très intéressant pour augmenter de plus en plus la performance et la précision des prédictions.

Tout au long de notre étude sur les SRs (chapitre I et chapitre II), nous avons constaté que l'algorithme basé sur le voisinage sélectionne toujours les plus proches voisins de

l'utilisateur et lui génère des items sur la base de leurs préférences. Donc, l'utilisation de la version floue de cet algorithme pour sélectionner les plus proches voisins flous, qui n'a pas été déjà élaborée dans le FC, peut engendrer plus d'items pertinents aux recommandations.

Les deux travaux de **[CHE *et al.*, 09] et [XUE *et al.*, 05],** ont proposé la sélection des plus proches voisins à partir des plus proches clusters, ce qui réduit énormément le temps de calcul pour ne pas calculer la similarité avec la totalité de l'ensemble d'utilisateurs du système.

Les méthodes sont très coûteuses en temps de calcul pour trouver le voisinage candidat, en utilisant les mesures de similarité proposées dans la littérature. Une nouvelle métrique de distance qui profite des résultats du *clustering* pour la sélection de l'ensemble candidat en se basant sur les degrés d'appartenance, réduit le temps de calcul même par rapport aux travaux de **[CHE *et al.*, 09] et [XUE *et al.*, 05].**

Les travaux présentés ont tous comme but d'augmenter la précision de la prédiction en utilisant le FC, alors que le filtrage basé contenu donne plus de pertinence au contenu généré parce qu'il considère la similarité entre les items et les préférences de l'utilisateur. Donc, l'hybridation entre les deux estimations, basée FC et basée FBC, est envisagée dans notre travail.

Le processus de génération de recommandation Top-N attribut des scores aux items non évalués par l'utilisateur et lui recommande ceux ayant les grands scores. La clairsemé peut affecter la précision des recommandations, ce qui conduit à des recommandations non pertinentes. Notre travail vise à réduire l'effet de la clairsemé, en estimant les valeurs pour les entrées creuses de la matrice d'évaluation avec le processus de prédiction hybride avant de lancer le processus de recommandation.

Le but derrière cette étude est de rechercher les algorithmes qui permettent l'affectation multiple des utilisateurs à différents clusters tout en considérant l'ambigüité dans leurs préférences, d'un côté, et de performer la phase du filtrage collaboratif autant que possible, pour réduire les effets de ces problèmes (clairsemé, et évolutivité), de l'autre. Ainsi que la recherche de la manière avec laquelle le système est rendu adaptatif et génère des recommandations avec un contenu diversifié correspondant aux différents goûts des utilisateurs, tout en garantissant leur pertinence pour l'utilisateur.

Cependant, malgré l'affectation multiple faite dans les travaux cités, le processus de recommandation ou de prédiction ne génère qu'un sous ensemble d'items sur la base des plus proches voisins d'un seul groupe ou même de différents groupes, sans un critère pour juger la pertinence ou non des items recommandés à l'utilisateur.

Afin d'aller plus loin sur la notion de la diversité dans les systèmes de recommandation, nous allons présenter et discuter dans la section suivante les principales méthodes élaborées et métriques proposées dans la littérature pour assurer la diversification des contenus.

2.4. La diversité dans la recommandation

2.4.1. Définition

La diversification peut être définie comme le processus de génération d'une variété d'items dans la liste de recommandation afin d'améliorer la qualité des recommandations et atteindre la satisfaction des utilisateurs en leurs livrant différents items. Ce qui nous mène à la deuxième définition de la diversité, d'une façon plus au moins formelle, diversifier est quantifier ou mesurer combien les items d'une liste de recommandation, respectivement les listes de recommandation, sont différents (es) les uns (es) des autres. D'où, des mesures de diversité basées sur l'utilisation des distances (dis-similarité) entre items ou listes d'items ont été définies et présentées afin de pouvoir évaluer la diversité dans les recommandations. Une explication des recommandations générées est parfois souhaitable pour aider à améliorer l'acceptation et la compréhension des recommandations par l'utilisateur **[GE *et al.*, 11] [JON et PU, 08].**

Il a été démontré dans la littérature **[SMY et MCC, 01][MCS, 02][ZIE *et al.*, 05] [ZHA *et al.*, 09] [GHA *et al.*, 13]** que la diversité dans un ensemble d'items peut être augmentée à un coût de la réduction de la précision du système. Bien que, la fonction de la diversité est contrastée à l'exactitude, de nombreux chercheurs ont tenté d'apporter la congruence entre les deux **[ZIE *et al.*, 05] [ZHA *et al.*, 09]**. Nous présentons dans cette section quelques travaux de la littérature ayant comme but d'augmenter la diversité dans les listes de recommandation.

2.4.2. Métriques de Diversité

2.4.2.1. La diversité individuelle (Individual Diversity) [SMY et MCC, 01]

La diversité individuelle pour l'utilisateur *u* est donnée par la dis-similarité moyenne de toutes les paires d'articles recommandés appartenant à la liste de recommandation N. Formellement, la diversité de la liste de recommandation *Top-N* générée à l'utilisateur *u* est représentée comme suit :

$$ID_u = \frac{1}{N(N-1)} \sum_{i \in N} \sum_{j \in N, i \neq j} d(i,j) \qquad (2.4)$$

Avec : $d(i, j) = 1\text{-}sim(i, j)$. (2.5)

Où, *sim(i,j)* est la similarité entre les item *i* et *j*, qui peut être obtenue en utilisant des mesures de similarité ou de distance sur les évaluations d'entrée directement ou bien à partir des méta-données et des caractéristiques des items. *d(i,j)* est la dis-similarité entre les deux items *i* et *j*.

2.4.2.2. La diversité Globale (Aggregate Diversity) [ADO et KWO, 12]

La diversité globale est la diversité calculée pour un groupe, et on peut la mesurer par la distance de *Hamming* ou la Couverture. La distance de *Hamming* calcule la différence entre les emplacements ou les positions des top-N items dans les listes de recommandations des utilisateurs.

La couverture calcule le pourcentage des items différents que le système de recommandation est capable de générer pour tous les utilisateurs.

$$AD = \frac{1}{|I|} \mid \cup N_u \mid \quad (2.6)$$

Où I est l'ensemble des items et |I| est le nombre d'items de cet ensemble. N_u est la liste de recommandation générée pour l'utilisateur u. $|\cup Nu|$ est le nombre total des items recommandés à tous les utilisateurs. Plus la valeur est grande, plus les items recommandés aux utilisateurs sont divers.

2.4.3. Algorithmes de diversification du contenu des recommandations

Plusieurs travaux ont été présenté ces dernières années afin d'améliorer la qualité de recommandations générées en terme de diversification du contenu délivré aux utilisateurs correspondant à leurs différents choix et intérêts. Quelques travaux ont présenté des algorithmes de reclassement de la liste de recommandation et d'autres ont présenté des mesures de diversité utilisées pour trouver les articles ou items divers.

2.4.3.1.Diversité Globale et Reclassement basé popularité des items

[ADO et KWO, 09] ont utilisé une mesure de diversité, en plus de la précision, pour mesurer la précision des recommandations. La mesure de diversité est calculée comme

le nombre total d'items distincts recommandés pour tous les utilisateurs, et elle est appelée mesure de diversité globale (*Aggregate diversity measure*).

$$Diversity - in - top - N = | \bigcup_{u \in U} L_N(u) | \quad (2.7)$$

La popularité des articles a été utilisée pour augmenter la diversité de la recommandation en tant que critère de classement au lieu de la prédiction, afin de recommander les articles les moins populaires aux utilisateurs.

Quatre méthodes de classement ont été présentées dans **[ADO et KWO, 09]** pour augmenter la diversité dans les listes d'items à recommander:

- **Valeur de note prédite inversée (*Reverse Predicted rating value***): en classant les articles du plus faible au plus élevé.
- **Evaluation moyenne** (***Average rating***): en classant les articles sur la base de la moyenne d'évaluation de chaque article comme un rang.
- **likeability absolue (*Absolute likeability***): en calculant le nombre d'utilisateurs qui ont aimé l'article (qui ont donné des notes supérieures à un seuil à cet article).
- **likeability relative** (***Relative likeability***): en calculant le pourcentage entre le nombre d'utilisateurs qui ont aimé l'article et le nombre total d'utilisateurs qui l'ont évalué.

Pour classer les articles, ceux qui ont des valeurs prédites au dessus d'un seuil de classement sont calculés avec les méthodes proposées et ceux au dessous du seuil sont calculés avec les méthodes de classement standard. Enfin, tous les articles au dessus du seuil sont recommandés.

[ADO et KWO, 12] ont proposé des techniques de reclassement pour améliorer la diversité dans les recommandations Top-N, en considérant le compromis entre la précision et la diversité. Dans leur travail, la diversité globale a été envisagée, i.e. la diversité parmi tous les utilisateurs (nombre d'articles divers parmi tous les utilisateurs). De plus, ils ont présenté deux autres métriques pour atteindre la diversité: la variance des notes de l'article (*item rating variance*) et la variance des notes du voisin (*neighbor's rating variance*). Dans la première métrique, ils calculent un rang (*rank*) comme la variance des évaluations effectuées par les utilisateurs sur l'article, et dans la deuxième, ils calculent le rang (*rank*) en utilisant la variance des évaluations effectuées par les voisins de l'utilisateur pour un article particulier i.e. les plus proches voisins qui ont évalué l'article. Ils ont testé l'efficacité de ces méthodes de reclassement sur

plusieurs ensembles de données: MovieLens, Netflix et Yahoo en calculant la perte de précision et le gain de diversité. Ils ont également présenté une technique aléatoire qui améliore la diversité en choisissant des items au hasard. Pour améliorer la précision et la diversité, ils ont proposé de recommander moins d'articles en fixant un nouveau seuil supérieur ou égal au seuil de notation (*rating threshold*) utilisé dans le classement pour filtrer les articles.

2.4.3.2.Similarité intra-liste et Diversification des Topics

[ZIE *et al.* 05] [ZIE et LAU, 09] ont proposé une nouvelle mesure pour calculer la diversité de la liste des articles appelée la similarité intra-liste (*Intra-list similarity*). Ils ont proposé une méthode de diversification des thèmes (topiques). Tout d'abord, ils ont proposé une nouvelle similarité appelée la similarité basée taxonomie (*Taxonomy-based similarity*) pour calculer la similarité entre les articles en fonction de leur classification.

$$ILS(L) = \frac{\sum_{i \in L} \sum_{j \in L, i \neq j} sim(i,j)}{2} \quad (2.8)$$

Où, *sim(i,j)* est la similarité basée taxonomie proposée dans **[ZIE *et al.*, 04].** Une valeur de la similarité intra-liste élevé indique une petite diversité.

Ils ont utilisé une liste ordonnée d'articles en entrée (classement standard pour la recommandation) et ce qui donne le rang standard $rang_s$. Ensuite, un reclassement des items est effectué en utilisant la similarité basée taxonomie pour trier les items dans l'ordre inverse de leur similarité pour obtenir le rang de dissimilarité $rang_s$. Ce rang est fusionné ensuite avec le rang standard en utilisant un facteur de diversification φ pour avoir le rang final avec lequel les items seront arrangés.

$$Rang(i) = (1 - \varphi).rang_s(i) + \varphi.rang_{div}(i) \quad (2.9)$$

Le facteur de diversification φ a pour rôle d'ajuster le compromis entre la diversité et la précision.

2.4.3.3.Partitionnement du profil de l'utilisateur

[VAR et CAS, 13] ont proposé une nouvelle méthode basée sur le partitionnement du profil de l'utilisateur à des sous-profils différents et de générer des recommandations pour chaque sous-profil. Ensuite, les différentes listes de recommandation sont regroupées pour former la recommandation. Les sous-profils sont générés sur la base de

la catégorisation des articles. Les recommandations sont calculées par estimation de la pertinence de chaque catégorie d'un article pour le sous-profil de l'utilisateur.

Ils ont utilisé le principe de la reformulation des requêtes et le classement des résultats obtenus par cette méthode proposée par **[SAN *et al.*, 10]**, et l'employer pour atteindre la diversification dans les systèmes de recommandation en utilisant des recommandations pour les sous-profils.

Les sous-profils sont calculés en utilisant une valeur de préférence des articles en se basant sur la préférence originale (l'évaluation) multipliée par un poids, qui représente l'appartenance ou non d'un article à la catégorie.

$$r(u_c, i) = W(c, i) * r(u, i) \quad (2.10)$$

Où, W est une valeur binaire.

2.4.3.4. Diversité Temporelle

[LAT et al., 10] ont considéré le changement temporel dans les ensembles des utilisateurs et des articles et dans les préférences des utilisateurs pour générer des recommandations diversifiées. Une fonction de score a été proposée pour classer les articles en utilisant l'évaluation prédite et la confiance dans la prédiction. Comme suite

$$S_{\hat{r},i} = \left(\hat{r}_{u,i,t} - 3\right) * confidence(\hat{r}_{u,i,t}) \quad (2.11)$$

La confiance est le nombre des articles utilisés dans la prédiction. 3 est le mi-point du l'échelle, car ils ont utilisé une échelle d'évaluation de 5 étoiles. Lorsque deux articles ont le même score, ils ont proposé d'utiliser la moyenne d'évaluation datée au dernier article évalué qui prend le rang le plus élevé.

Ils ont proposé de créer une liste de recommandation pour les utilisateurs à l'instant *t* en utilisant trois algorithmes pour montrer l'efficacité. Ensuite, ils mettent à jour cette liste à l'instant (t+μ) (μ est la taille de la fenêtre qui incrémente le temps avec la période de temps que l'utilisateur n'a pas interagit avec le système à partir de la dernière utilisation.). Puis, ils ont proposé de calculer la diversité atteinte au cours du temps (i.e. entre deux listes classées consécutives) comme un rapport entre le nombre de différents articles de ces deux listes et le nombre N d'articles recommandés.

Etant donné deux listes de recommandation L_1 et L_2, théoriquement la différence entre ces deux listes est le nombre d'items qui apparaissent dans la 2ème liste et qui n'existent pas dans la première.

$$L_2 \backslash L_1 = \{x \in L_2 | x \notin L_1\} \tag{2.12}$$

Une mesure de diversité a été proposée à propos de ce fondement théorique tout en considérant la taille des listes N comme suit :

$$diversity(L_2, L_1, N) = \frac{|L_2 \backslash L_1|}{N} \tag{2.13}$$

Dans le cas où les deux listes sont les mêmes, la diversité sera égale à 0, et si elles sont totalement différentes on obtient une diversité égale à 1.

L'inconvénient de cette mesure est qu'elle ne considère que la différence entre la totalité des items appartenant à une liste alors que l'on peut avoir la diversité avec quelques items seulement. Cependant, une nouvelle mesure de nouveauté des listes de recommandation est définie comme suit :

$$Novelty(L_1, N) = \frac{|L_1 \backslash A_t|}{N} \tag{2.14}$$

Cette mesure calcule le rapport entre le nombre de nouveaux items par rapport à l'ensemble des articles recommandés au jour A_t et le nombre N d'articles recommandés. Ensuite, pour calculer la diversité offerte par les systèmes du FC, ils ont proposé de calculer les valeurs moyennes obtenues à partir des valeurs de la diversité (diversité moyenne) et de la nouveauté (nouveauté moyenne) entre toutes les Top-N listes générées à l'instant *t* et la dernière liste générée pour chaque utilisateur.

Pour promouvoir la diversité temporelle, ils ont proposé deux nouvelles méthodes:

- ✓ *Commutation hybride temporelle* (*temporal hybrid switching*): cet algorithme consiste de basculer entre deux algorithmes du FC: KNN et SVD. Où, un algorithme d'entre eux est appliqué à chaque mise à jour pour formuler des recommandations, ce qui a augmenté la diversité.
- ✓ *Reclassement des recommandations individuelles de l'utilisateur* (*re-ranking individual user's recommendations*): cette méthode consiste à augmenter la diversité en sélectionnant N articles à partir d'une liste de M articles (N <M).

Puis, en remplaçant *d**N articles de cette liste (*d*: la diversité) en les sélectionnant au hasard à partir de [N + 1 ... M].

2.4.3.5.Diversités basées Distance (Distance-based diversity) :

[ABB *et al.*, 13] ont proposé une version modifiée de l'algorithme du Hachage sensible à la localité (*Locality Sensitive Hashing*) appelé LSH* pour trouver les plus proches voisins d'un article.

Ils ont proposé une nouvelle extension de l'algorithme dLSH appelé dLSH* qui intègrent la diversité dans le calcul des seaux (*buckets*) à trouver pour calculer seulement les articles potentiellement intéressants. Ils ont utilisé deux types de distances: Distance de Pertinence (*Relevance Distance*) et la Distance de Diversité (*Diversity Distance*).

La Distance de Pertinence calcule la dis-similarité.

La Distance de Diversité calcule combien deux articles sont différents. Cinq mesures de diversité ont été proposées, où, quatre parmi elles utilisent les commentaires sur les articles et la cinquième utilise le contenu des articles.

- **La Distance basée Pertinence:**

$$d_{rel}(a_i, a) = 1 - Jaccard(x, y) \leq r \qquad (2.15)$$

a_i et a sont des articles et x, y leurs caractéristiques. La mesure de Jaccard calcule la similarité entre deux articles comme le nombre de caractéristiques communs divisés par le nombre total de caractéristiques. r est un seuil de rayon. Les caractéristiques sont extraites en utilisant Open Calais (OC).

- **La Distance basée Diversité**

1. La diversité basée commentaire:

Ils ont utilisé les commentaires pour définir quatre diversités:

a) Utilisation des entités:

$$d(a_i, a_j) = 1 - Jaccard(OC(discuss(a_i)), OC(discuss(a_j)) \qquad (2.16)$$

Où, $discuss(a_i)$ est un document contenant tous les commentaires sur l'article a_i, ce dernier contient les sujets ou thèmes (topiques) des articles, et les entités (personnes, pays et villes) qui représentent les caractéristiques du document de discussion *discuss*.

b) Utilisation des sentiments:

Ils ont proposé d'extraire les sentiments à partir du document des commentaires et le considérer comme sentiment positif S_i^+ ou sentiment négatif S_i^- et lui attribuer un intervalle [-1, +1] ou un sentiment neutre 0.

$$div(a_i, a_j) = \sqrt{(S_i^+ - S_j^+)^2 + (S_i^- - S_j^-)^2} \quad (2.17)$$

Où, S_{ai}^+ et S_{ai}^- sont les pourcentages des commentaires positifs (resp. negatifs) postés sur l'article a_i .

c) Utilisation de l'identificateur de l'utilisateur (User Ids):

Compter les utilisateurs qui lisent l'article *a* et calculer les utilisateurs identifiés par *User-IDs(a)* (ensemble d'utilisateurs qui commentent *a*) parce que les utilisateurs similaires lisent des articles similaires.

$$div(a_i, a_j) = 1 - JACCARD(user - IDs(a_i) - User - IDs(a_j)) \quad (2.18)$$

d) Utilisation du Location de l'utilisateur:

Un ensemble de pays est extrait de l'ensemble des utilisateurs qui commentent l'article *a* référencé par *Users-Countries(a)*, parce que les utilisateurs de la même région ont des intérêts similaires que différentes régions.

$$div(a_i, a_j) = 1 - Jaccard\ (users - countries(a_i) - users - countries(a_j)) \quad (2.19)$$

2. La Diversité basée Contenu:

$$div(a_i, a_j) = 1 - Jaccard\ (OC(a_i) - OC(a_j)) \quad (2.20)$$

Calcule la similarité en fonction des caractéristiques similaires extraites à partir des commentaires.

- **La pertinence et la diversité basées ensemble (Set-based Relevance and Diversity):**

$$Rel(R) = avg_{ai \in R}(1 - d_{rel}(a_i, a)) \quad (2.21)$$

R: L'ensemble des articles pertinents.

Ensuite, ils sélectionnent K articles, où K⊆R, qui correspondent à la formulation de la diversité comme suit :

$$max_{S \subset R, |S|=K} min_{ai,aj \in S}\, d_{div}(a_i, a_j) \quad (2.22)$$

Ce qui signifie que l'ensemble *K* des articles différents sélectionnés à partir de la liste *R* en prenant les articles qui maximisent la différence (distance minimale au centroid a_i).

2.4.3.6.Sélection des K-plus loin voisins (KPLV)

[SAI *et al.*, 12] ont proposé un système de recommandation orthogonale qui sélectionne les k-plus loin voisins et recommande les articles les moins aimés (détestés) par eux. Ils calculent la similarité entre les utilisateurs ayant au moins cinq articles co-évalués (sans prendre les évaluations moyennes), en utilisant les mesures de similarité ordinaires. Ensuite, ils ont pris les plus dissemblables (moins similaires) et ont recommandé les articles non aimés (*disliked*) par eux. Afin d'augmenter la diversité et atteindre une petite perte en précision, ils ont proposé d'alterner entre l'algorithme K-plus loin voisins (*K-Farthest Neighbours KFN*) et l'algorithme des K-plus proches voisins (*K-Nearest Neighbours KNN*) avec une probabilité proportionnelle à la précision.

2.4.3.7.Modèle du FC Synthétique (Synthetically CF model)

[WAN et YIN, 13] ont proposé un cadre du FC qui combine entre les deux techniques du FC : basé utilisateur et basé item, appelé modèle du FC synthétique (*Synthetically CF model SCF*).

Il a été démontré dans **[WAN et YIN, 13]** que le FC génère des recommandations fondées sur la popularité et le FBC génère les articles les moins populaires (*long-tail items*) i.e. des articles divers. En conséquence, pour formuler des recommandations avec divers articles sans perte en précision, ils ont combiné entre les deux algorithmes. Où, ils génèrent deux sous-ensembles d'articles N1 et N2 générés par les deux algorithmes

du FC basé utilisateur et FC basé item, respectivement. Ensuite, les deux listes sont fusionnées pour composer la liste de recommandation N. Où, la liste N1 contient les articles les plus populaires (ceux préférés) et la liste N2 contient les articles différents aux besoins de l'utilisateur.

$$N = \{i/i \in N_1 \cup N_2, N_1 \cap N_2 = \emptyset\} \quad (2.23)$$

Le modèle SCF donne aux utilisateurs la possibilité d'ajuster la diversité dans les listes recommandées en fonction de leurs besoins en utilisant deux paramètres: taux d'intérêt et taux de nouveauté. Le taux d'intérêt contrôle le nombre d'articles populaires dans le filtrage collaboratif basé utilisateur et le taux de nouveauté contrôle le nombre d'articles les moins populaires (*long tail*) dans le filtrage collaboratif basé item, et ils ont utilisé un seuil de popularité trouvé dynamiquement en fonction du nombre d'utilisateurs dans le système pour sélectionner les articles dans les sous-listes N1 et N2.

SCF Recommendation Algorithm

Entrée: matrice d'évaluation, vecteur de popularité des items, N : nombre des items dans la liste de recommandation, α: taux de prévalence, β : taux de nouveauté, δ : seuil de popularité des items.

Sorite: L : la liste de recommandation

Début

1. N1= (N*α) items populaires par le FCBU (popularité de l'item > δ).
2. N2= (N*(1-α)) items par le FCBI, où [N * β (β≤(1-α))] appartient aux items non populaires (popularité de l'item < δ) , et le reste des items sont produits directement par le FCBI.
3. Renvoyer N= N1+ N2.

Fin

2.4.3.8.Algorithme ClusDiv et mesure z-diversité

[AYT et KAR, 14] ont proposé un algorithme basé sur le regroupement (*clustering-based algorithm*) pour améliorer la diversité, en regroupant l'ensemble de tous les items, et non seulement ceux notés par l'utilisateur, pour suggérer des listes de recommandation.

Une nouvelle métrique de diversification a été proposée pour mesurer la diversité appelée *z-diversité* pour éviter les problèmes de la parcimonie des évaluations et la

difficulté de la diversification avec lesquels souffrent les anciennes mesures de diversité.

$$ZD\big(L(u)\big) = \frac{D\big(L(u)\big) - D(I)}{SD(I)} \tag{2.24}$$

Où, I est l'ensemble des items dans la base de données, *D(L(u))* et *D(I)* sont les diversités des items dans *L(u)* et *D(I)*, respectivement.

SD(I) est l'écart type des dis-similarités de toutes les paires dans *I* tel que défini ci-dessous:

$$SD(I) = \sqrt{\frac{1}{N(N-1)} \sum_{i \in I} \sum_{j \in I, i \neq j} (d(i,j) - D(I))^2} \tag{2.25}$$

Où, $N=|I|$, *D(I)* est la diversité des items dans *I*, et *d(i,j)* est la dis-similarité entre les items *i* et $j \in I$, et elle est définie comme (1-*sim*(*i*,*j*)), avec *sim* calculée avec la similarité cosinus.

Avec la parcimonie des évaluations, Les notes manquantes sont mises à 0 et les similarités entre les éléments tendent à 0. Ainsi, la dissimilarité (1-similarité) sera très proche de 1 et les listes de recommandation apparaissent superficiellement très diversifiées.

Le deuxième problème, comme mentionné dans **[AYT et KAR, 14]** est qu'il existe des ensembles de données difficiles à se diversifier parce que l'écart entre les similitudes de leurs paires de données est faible, ce qui est le cas pour les ensembles de données des livres.

- **Clus Div algorithm:**

L'algorithme Clus Div comporte deux phases: en ligne et hors ligne. Dans la phase hors ligne (offline), ils regroupent les articles de l'ensemble de données des items à N clusters, où N est le nombre d'articles dans la liste de recommandation, par l'application de l'algorithme k-means en utilisant seulement les évaluations sur les articles sans aucune information sur le contenu. Ensuite, un algorithme pour calculer des poids pour les clusters est proposé (*Cluster Weights CW*) comme le nombre d'articles à sélectionner de chaque groupe d'articles. Exemple : si le poids pour le groupe C est égal à 5, cela signifie qu'il doit choisir les Top-5 articles dans la liste d'items du groupe C, qui est formée après le processus de prédiction.

Les poids des clusters sont initialisés par le nombre d'items appartenant au cluster correspondant et à la liste de recommandation générée par l'algorithme de prédiction, en même temps.

Puis, des étapes sont générées pour distribuer les poids entre les clusters, où les clusters ayant des poids supérieurs à un seuil donné donnent des poids aux clusters avec de petits poids pour augmenter la diversité autant que possible. Le seuil est utilisé pour ajuster le compromis entre la diversité et la performance. Où, un petit seuil augmente la diversité et diminue la performance (rappel) tant dis qu'un seuil élevé diminue la diversité et augmente la performance. Le seuil est réglé par chaque utilisateur à son propre besoin de diversité.

Après le calcul de tous les poids des clusters, la liste de recommandation est construite en ajoutant les articles appartenant à des groupes ayant des poids supérieurs à 0, où le nombre d'articles sélectionnés dans chaque groupe est égale au poids du cluster et à chaque ajout est diminué de 1 jusqu'à ce que tous les poids deviennent 0.

2.4.3.9. Classement par Effet sur la Diversité Totale (Total Diversity Effect Ranking TDE rank)

[PRE *et al.*, 13] ont proposé deux nouvelles méthodes de classement hybride pour améliorer la diversité dans les listes de recommandation et un nouveau mécanisme a été présenté pour contrôler le poids entre la diversité et la précision.

La méthode de classement est appelé Classement à Effet de diversité totale (*Total Diversity Effect Ranking*) pour montrer l'effet de chaque article recommandé sur la diversité totale des listes de recommandation Top-N.

Pour contrôler le degré de diversité et de précision, ils ont proposé un nouveau mécanisme. Il a été démontré que chaque item a un impact sur la diversité et la précision des listes de recommandation. Ils ont utilisé la distance de Jaccard (indice de Jaccard) pour calculer la similarité entre les items en utilisant leurs caractéristiques comme suit :

$$dist(x, y) = \frac{|A_x \cup A_y| - |A_x \cap A_y|}{|A_x \cap A_y|} \quad (2.26)$$

Où A_x et A_y sont les ensembles d'attributs des items x et y.

Puis, l'algorithme du FC basée sur l'utilisateur a été utilisé pour prédire les évaluations en utilisant les deux algorithmes : La moyenne de la somme pondérée ajustée (*Adjusted weighted sum average*) et la moyenne de la somme pondérée (*weighted sum average*). Ensuite, quatre méthodes de classement ont été proposées pour être utilisées. Où, deux sont les méthodes de bases et les deux autres sont des méthodes hybrides qui combinent les deux premières:

- *La méthode de classement standard* (*standard ranking SRank*): où les articles sont classés en fonction des résultats de la prédiction dans un ordre décroissant et seulement les meilleurs articles sont sélectionnés pour la recommandation.
- *La méthode de classement à Effet sur la diversité totale* (*Total Diversity Effect*) : cette méthode calcule les distances entre chaque paire d'items dans la liste de recommandation, puis, pour chaque item elle calcule le total (somme) de dis-similarités (différences) avec les autres articles.

L'effet sur la diversité totale TDE de l'article c_i dans une liste de recommandation L_u pour un utilisateur actif *u* est présenté comme suit:

$$TDE(c_i) = \sum_{j=1\ldots|L_u|} dist(c_i, c_j);\ c_i \neq c_j; c_i, c_j \in L_u \qquad (2.27)$$

Où c_j est un article dans la liste de recommandation et *dist* est calculée avec la distance de Jaccard.

Le classement avec la méthode de l'effet sur la diversité totale (TDE Rank) est appliqué en sélectionnant les articles ayant le plus grand effet sur la diversité totale.

Les deux autres méthodes de classement sont des méthodes hybrides qui combinent le classement standard (SRank) avec le classement basé sur l'effet de diversité totale (TDE) de deux façons, pour améliorer la diversité et réduire la perte de précision causée par TDE et la perte de la diversité causée par le SRank, simultanément. Le principe de l'hybridation est d'utiliser l'une des méthodes en premier, puis utiliser ses résultats comme entrées pour la deuxième. Les deux méthodes hybrides sont :

- Classement à effet sur la diversité totale-standard (*Standard-total diversity effect Rank S-TDE Rank*): dans ce classement la méthode SRank est appliquée pour générer Top-N + S articles (S est un entier positif ≥1). Ensuite, la liste de

recommandations résultante est utilisée par la méthode TDE Rank comme entrée.

- Classement Standard à effet sur la diversité totale *(Total diversity effect Rank-Standard TDE-S Rank*): cette méthode utilise le classement TDE d'abord pour générer une recommandation de Top-N+S items. Ensuite, le classement standard est appliqué sur cette liste pour générer la liste de recommandations top-N.

Pour évaluer la performance des propositions, trois mesures ont été proposées.

- *La précision moyenne*:

$$avgprec(TL) = \frac{\sum_{Lu \in TL} prec(Lu)}{|TL|} \quad (2.28)$$

Avec : $$prec = recall = \frac{|Lu \cap TL|}{|Lu|} \quad (2.29)$$

Où, *Lu* est la liste de recommandation pour l'utilisateur *u*; et *TL* est la liste de tous les utilisateurs qui ont reçu la liste de recommandation.

La précision calcule le rapport entre le nombre d'items pertinents réels | Lu∩TL | et le nombre d'items pertinents prédits dans la liste de recommandation de l'utilisateur *u*, |*Lu*|.

- *La diversité moyenne:* La diversité d'une liste de recommandation est calculée comme suit:

$$div(Lu) = \frac{\sum_{j \in 1..|Lu|} dist(c_i, c_j)}{|Lu|.(Lu-1)/2}; i \neq j;\ c_i, c_j \in Lu \quad (2.30)$$

Ensuite, la diversité moyenne est calculée en faisant la moyenne de la diversité de toutes les listes de recommandation de chaque utilisateur comme suit :

$$avgdiv(TL) = \frac{\sum_{Lu \in TL} div(Lu)}{|TL|} \quad (2.31)$$

- *Moyenne Harmonique de la Diversité et de la Précision (Harmonic means of precision and diversity HMPD*) : cette méthode est proposée afin d'équilibrer le compromis entre la précision et la diversité comme suit :

$$HMDP\big(avgdiv(TL), avgprec(TL)\big) = 2 \times \frac{AvgDiv \times Avg\ Prec}{AvgDiv + AvgPrec} \quad (2.32)$$

Où, S-TDE Rank avec la méthode de prédiction à base de similarité pondérée ajustée Adj-WS avec 1 surplus article supplémentaire fournit la plus grande valeur de HMDP.

2.4.3.10. Diversification des Voisins (Neighbors' diversification)

Zhang a proposé un algorithme pour balancer entre la diversité et la précision en sélectionnant divers voisins pour accroître la diversité aux systèmes de recommandation basés sur la confiance (*trust-based RSs*). L'algorithme proposé comporte trois phases:

- *Calcul des scores des items :*

Tout d'abord, des scores sont calculés pour les articles comme score d'intérêt en fonction de leur emplacement dans la taxonomie d'items et à leurs frères (items dans le même niveau de la taxonomie) comme suit:

$$Sco(pm) = K * \frac{sco(P_{m+1})}{sib(P_{m+1})+1} \quad (2.33)$$

$sib(P_{m+1})$: Nombre de frères.

- *Sélection des divers voisins*

La deuxième étape est de sélectionner divers voisins pour faire des recommandations avec divers articles. A ce propos, un algorithme de diversité basé sur la confiance (*trust-based diversity algorithm*) a été présenté utilisant une nouvelle mesure de similarité. Cette dernière calcule la similarité entre les profils des utilisateurs représentés par des taxonomies, en utilisant les scores des thèmes (topiques) calculés et non pas les valeurs d'évaluation comme suit:

$$sim(v_u, v_s) = \frac{\sum_{k=0}^{|D|}(V_{uk}-\overline{V}_u).(V_{sk}-\overline{V}_s)}{\sqrt{\sum(V_{uk}-V_u)^2.(V_{sk}-V_s)^2}} \quad (2.34)$$

V_u : Le score du thème (topique) de l'utilisateur *u*. $\overrightarrow{V_u} = (V_{u1}, V_{u2}, \dots V_{u|D|})$ est le vecteur du profil de l'utilisateur *u* et V_{uk} est le score d'intérêt pour le topique $d_K \in$ D.

Ensuite, un schème a été proposé pour optimiser le compromis entre *sim* et *div* avec un paramètre d'optimisation λ comme suit:

$$W(u_k) = (1-\lambda) \times sim(T, u_k) + \lambda \times div(T, u_k) \quad (2.35)$$

Où, *div* est la diversité calculée comme *1-sim*.

Ensuite, les voisins confiants différents (*diverse trust neighbors*) sont choisis en sélectionnant les utilisateurs ayant les plus grandes valeurs $W(u_k)$comme ensemble d'utilisateurs candidats.

- *Prédiction des évaluations*

Enfin, Zhang a utilisé la prédiction basée sur l'utilisateur ajustée pour prédire les notes des items en utilisant l'ensemble candidat choisi.

2.4.3.11. Diversification des thèmes (Topic diversification)

[ZHA, 08] a proposé d'utiliser une mesure de similarité basée sur la similarité entre thèmes (topiques). Une nouvelle métrique de diversité a été proposée à partir de cette similarité dont sa performance a été dans les recommandeurs basés FC.

La représentation taxonomique a été utilisée pour classer les items, où la racine est le terme général (domaine) et les nœuds sont les différents thèmes (catégories) et les nœuds feuilles sont les items (les films dans leur cas).

- **Métrique de similarité d'une liste (*List similarity metric*)**

La similarité entre deux articles est calculée en fonction de l'emplacement des articles dans la taxonomie, où les articles de la même sous-catégorie ont une similarité égale à 1 et les articles dans le même niveau de la taxonomie ont une similarité supérieure à celle avec différents niveaux. La mesure de similarité est définie comme suit:

$$similarity(I_i, I_j) = \frac{2*Depth(LCA\,(I_i,I_j))}{Depth(I_i)+Depth(I_j)} \qquad \textbf{[GAN et GAR, 03]} \ (2.36)$$

I_i et I_j sont le $i^{ème}$ et le $j^{éme}$ items. $Depth(I_i)$ le nombre d'arêtes de la racine à l'items I_i. $LCA\,(I_i, I_j)$ est le nœud qui est un ancêtre des deux items I_i et I_j.

- **Métrique de diversité d'une liste (*List diversity metric*):**

La dis-similarité moyenne a été utilisée pour calculer la différence entre toutes les paires des items dans la liste de recommandation, comme suit :

$$diversty(I_1, I_2, \dots I_n) = \frac{\sum_{i=1,n}\sum_{j=i,n}(1-similarity(I_i,I_j))}{\frac{n}{2}*(n-1)} \qquad (2.37)$$

La mesure de diversité est calculée en exploitant la connaissance du domaine hiérarchique (*hierarchical domain knowledge*) pour compléter la similarité entre les items.

2.4.3.12. Diversité Relative

[SMY et McC, 01] ont présenté une bonne étude sur l'utilité et l'intégration de la diversité dans les systèmes de recommandation et ils ont présenté 5 algorithmes (FC, BC et popularité avec deux extensions hybrides) et ils ont utilisé différents degrés de diversité pour montrer la satisfaction des utilisateurs. Ils ont proposé deux extensions : FC avec diversité relative (CFDR) et FC avec diversité Fixe (CFDF). Dans ce dernier, ils conservent les N premiers articles à partir des K articles avec N = Nombre de recommandation * X%).

Dans CFDR: ils sélectionnent les Top-K articles, ensuite, ils placent les autres articles un par un en sélectionnant à chaque fois l'article qui maximise la fonction de diversité calculée comme suit :

$$\mathrm{sim}(\mathrm{i}_1, \mathrm{i}_2) = \frac{\sum_{j=1,n} Wj * sim_{attribute=j}(i_1, i_2)}{\sum_{j=1.n} Wj} \quad (2.38)$$

Où:

$$RelDiversity(i, C) = \begin{cases} 0 \; si \; C = \{\}; \\ \sum_{j=1.n}(1 - sim(I, Cj)) \, /n \end{cases} \quad (2.39)$$

2.4.3.13. Modèle Statistique

[ZHA 09] a abordé le problème de la diversité à partir de deux points différents: la perspective de l'utilisateur et la perspective du système. Ensuite, il a proposé un modèle statistique qui résume le problème de la recommandation qui se base sur la similitude entre la paire utilisateur-item. Une distribution gaussienne a été utilisée pour représenter la similarité et les ensembles de données, où il a généré trois modèles pour différents recommandeurs.

Il a proposé de résoudre le problème par trois algorithmes:

- ✓ *Algorithme 1:* vise à optimiser la fonction objective qui représente le compromis entre la diversité et la précision en utilisant un paramètre d'ajustement $\theta \in [0,1]$ comme suit :

$$y^* = argmax(1 - \theta)\alpha y^T D_y + \theta\beta m_u^T y \quad (2.40)$$

s.t. $1^T y = p$, y(i)∈{0,1} ∀i∈1…M.

Avec, m_u est le vecteur de M-Dimension, m(i) représente la similarité entre c_i et le profil de l'utilisateur. α et β sont des paramètres de normalisation. Différents algorithmes ont été utilisés pour achever cette fonction objective : *Greedy*, *Relaxation* et *quantification*.

- ✓ *Algorithme 2:* cet algorithme se concentre sur le partitionnement du profil de l'utilisateur à des groupes, puis générer des recommandations à chaque sous-profil. Pour atteindre la nouveauté. L'algorithme basé top-N item (*Item-based Top-N algorithm*) a été utilisé pour générer des recommandations à chaque sous-profil. Puis, à la fin, toutes les listes sont regroupées pour former la recommandation finale.
  Il a proposé de se concentrer sur les nouveaux groupes qui correspondent aux nouveaux goûts.
- ✓ *Algorithme 3:*La troisième méthode se base sur la méthode de réduction de dimension pour améliorer la similarité entre les items et par conséquent améliorer la classification. L'algorithme SVD a été utilisé pour factoriser la matrice d'évaluation. Ensuite, les premiers *l* items de la matrice résultante sont pris pour les utiliser dans le calcul de la similarité/dis-similarité dans les algorithmes de *clustering*.

2.4.3.14. Diversité basée sur le rang et la pertinence des items

[VAR et CAS, 11] ont présenté une étude de la littérature sur les métriques de la nouveauté et de la diversité et ont proposé une nouvelle métrique pour mesurer la diversité basée sur l'ordre et la pertinence des items dans la liste de recommandation *R*. La nouvelle métrique de diversité est basée sur une nouvelle métrique de nouveauté qui calcule la distance relative entre les items.

$$div(R\backslash u) = \sum_{i_k, i_l \in R, k \neq l} C_k disc(k) disc(l|k) p(rel|i_k, u) p(rel|i_l, u) d(i_k, i_l) \quad (2.41)$$

Où, $disc(l|k) = disc(\max(1, l-k))$ qui reflète la réduction relative au rang (*relative rank discount*) d'un item à la position l sachant que la position k est atteinte. Avec la constante de normalisation, $C_k = C / \sum_{i_l \in R - \{i_k\}} disc(l|k) p(rel|i_l, u)$.

Cette formule de la diversité est la formule de calcul de distance intra-liste moyenne avec introduction de la sensitivité au rang et à la pertinence.

2.4.3.15. Diversité basée sur le Choix, la Découverte et la Pertinence des items

[VAR et CAS, 11] ont proposé un cadre probabiliste pour représenter la diversité et la nouveauté dans les systèmes de recommandation en considérant la pertinence des choix et la découverte. Le même travail a été présenté dans **[CAS et al., 11**], Où ils ont présenté de nouvelles métriques de diversité et de nouveauté spécifiées aux SRs pour atteindre leurs besoins en tenant compte de deux concepts: la similarité entre items et l'interaction utilisateur-item, et trois conditions: Choix, Découverte et Pertinence pour représenter l'interaction utilisateur-item et ils ont introduit le rang et la pertinence en tant que propriétés de ces deux métriques. Où

- **Choix**: indique si l'item est choisi ou sélectionné par l'utilisateur.
- **Découverte**: indique si l'item est vu ou non par l'utilisateur.
- **Pertinence**: combien l'utilisateur aime l'article, le degré de préférence (valeur de l'évaluation).

Ils ont présenté des métriques en considérant la nouveauté et d'autres en considérant la diversité. Pour atteindre la nouveauté des items, deux modèles ont été présentés: l'un basé sur la popularité et le second sur la distance entre les items, et ils ont considéré deux variantes pour chaque formulation: variante Générique et variante Relative pour un ensemble d'items comme suit:

$$Novelty(i) = -log_2 p(i) \text{ et } Novelty(i/u) = -log_2 p(i/u) \qquad (2.42)$$

Où *p(i)* est la probabilité que l'article *i* est observé i.e. est choisi par l'utilisateur. La Nouveauté correspond au logarithme de l'inverse de la popularité.

- **Le modèle de la popularité basée sur la découverte (Discovery-*based popularity model*)**: examine les items familiés par l'utilisateur et non pas ceux choisis par lui comme suit :

$$Novelty(i) = 1 - p(K/i) \text{ et } Novelty(i/u) = 1 - p(K/i,u) \qquad (2.43)$$

- **La Nouveauté d'article basée sur la distance (*Distance-based item Novelty*)**: cette métrique est défini comme la distance moyenne ou minimale entre l'article cible et d'autres articles.

$$Novelty(i/S) = \sum_{j \in S} p(i/S) d(i,j) \text{ ou } Novelty(i/S) = min_{j \in S} d(i,j) \qquad (2.44)$$

d: est la mesure de distance.

Ensuite, plusieurs estimations ont été proposées pour répondre à chaque cas (nouveauté, découverte et pertinence). La liste de recommandation est calculée comme suit:

$$m(R) = \sum_{i \in R} p(i/R)\, novelty(i) \tag{2.45}$$

R: est la liste des items recommandés.

Pour incorporer la pertinence, la formule est transformée à:

$$m(R/u) = \sum_n disc(n/p(rel\backslash i_n, u) novelty(i_n \backslash u) \tag{2.46}$$

Disc: fonction d'actualisation i.e. en termes de pertinence.

- **Métrique de Nouveauté pour la recommandation:**

$$Novelty(R\backslash u) = \sum disc(n) p(rel\backslash i_n, u) p(j\backslash u) d(i,j) \tag{2.47}$$

- **La métrique de Diversité basée Nouveauté:**

$$Diversity(R\backslash u) = \sum_{n,k} disc(n) p(rel\backslash i_n, u) p(i_k \backslash R) d(i_n, i_k) \tag{2.48}$$
$$= 2 \sum_{K<n} disc(n) disc(K) p(rel\backslash i_n, n)$$

Cette formule inclue le rang et la pertinence dans la liste de recommandation

$$P(choose) \sim p(seen) p(rel) \tag{2.49}$$

2.4.3.16. Diversité basée sur les Médoïdes Prioritaires

Dans **[BOI *et al.*, 11],** une nouvelle méthode pour améliorer la diversité dans la recommandation Top-N a été présentée pour équilibrer entre la similarité et la diversité des items recommandés dans les listes Top-N, en utilisant la méthode des médoïdes prioritaires (*Priority-medoids*) (un médoïde est le représentant d'un groupe). Lors du calcul de la similarité, la priorité est ajoutée comme une référence pour inclure l'item qui a le score le plus élevé. Cette méthode est appliquée après le processus du FC et le calcul des scores des Top items.

Ils ont utilisé l'arbre de couverture de priorité (*Priority Cover Tree*) pour représenter les items d'une manière descendante en fonction de leurs notes. Ensuite, les K items de recommandation sont choisis en utilisant quatre différentes méthodes (Nidification, Séparation, Couverture, et Priorité).

- *Nidification*: Si un nœud est associé à un item *i*, alors l'un de ses enfants doit également être associé avec *i*.
- *Séparation*: Tous les nœuds au niveau *l* sont au moins $\frac{1}{2^l}$ loin d'un autre item.
- *Couverture*: Chaque nœud au niveau *l* est à distance $\frac{1}{2^l}$ à ses enfants au niveau *l*+1.
- *Priorité*: Chaque nœud a une note supérieure ou égale à celles de ses enfants.

Le processus de sélection commence à partir de la racine de l'arbre, en cherchant pour le premier niveau *l* dans l'arbre qui contient au moins k nœuds, où $|C_{l-1}| < k$ et $|C_l| \geq k$. En suite, les *k* items représentants sont choisis à partir de C_l, où si le nombre de nœuds dans C_l est égale à *k* alors ils sont tous choisis. Sinon, si $|C_l| > k$ alors un sous ensemble de C_l égale à *k* est choisis, en sélectionnant les items en C_{l-1}, puis en ajoutant le reste du $(k - C_{l-1})$ à partir de $(C_l - C_{l-1})$.

2.4.3.17. [Ge et al. 11]

Dans **[Ge *et al.*, 11],** une méthode de classement des items a été proposée, qui place les items ayant les plus grandes valeurs de diversité (dis-similarité) en haut de la liste de recommandation pour aider les utilisateurs à percevoir la diversité.

Une bonne étude avec plus de détails sur la diversité dans les systèmes recommandation, ainsi que les méthodes, métriques d'évaluation et approches proposées, est présentée dans **[HUR et ZHA] [CAS *et al.*, 13].**

2.4.4. Synthèse des approches sur la diversité

Nous présentons dans le (*Tableau 2.1*) un résumé des méthodes utilisées pour diversifier les listes de recommandation. Les critères utilisés dans ce tableau permettent de :

- ✓ Classifier les méthodes selon le type de diversification (Gl : Globaleest, Ind : individuelle) ou la méthode utilisée pour diversifier les recommandations (classement, sélection),
- ✓ Indiquer l'utilisation ou non d'un paramètre d'ajustement,
- ✓ Nom de la méthode et ses principales étapes,
- ✓ La métrique de diversification proposée ou utilisée par l'algorithme.

	Paramètre d'ajustement	Diversité		classement	Sélection	Méthode de diversification	Métrique et Schème
		Gl	Ind				
[ADO et KWO, 09 12]	Seuil du classement	×	-	×	-	Reclassement des items selon : - *Note prédite inversée* - *Evaluation moyenne* - *Likeability absolue* - *Likeability relative* - *Variance des notes des items* - *Variance des notes des voisins*	$div = \lvert \bigcup_{u \in U} L_N(u) \rvert$
[Zie *et al.*, 05 09]	Facteur de diversification φ	-	×	×	-	Diversification des topiques *-Reclassement selon similarité basée taxonomie*	$Rang(i) = (1-\varphi).rang_s(i) + \varphi.rang_{div}(i)$ $ILD_u = \frac{1}{N(N-1)} \sum_{i \in N} \sum_{j \in N, i \neq j} d(i,j)$
[Var et CAS 13]		-	×	-	×	Partitionnement du profil de l'utilisateur	
[LAT *et al.*, 10]	-	×	-	×	-	Diversité temporelle : - *Commutation hybride temporelle*	$div(L_2, L_1, N) = \frac{\lvert L_2 \backslash L_1 \rvert}{N}$ $Novelty(L_1, N) = \frac{\lvert L_1 \backslash A_t \rvert}{N}$

[ABB *et al.*, 13]	-	-	×	-	×	Diversité basée distance - *Distance de pertinence* - *Distance de diversité*	$d_{rel}(a_i, a) = 1 - Jaccard(x, y) \leq r$ $d(a_i, a_j) = 1 - Jaccard(OC(discuss(a_i)), OC(discuss(a_j))$ $div(a_i, a_j) = \sqrt{(S_i^+ - S_j^+)^2 + (S_i^- - S_j^-)^2}$ $div(a_i, a_j) = 1 - JACCARD(user - IDs(a_i) - User - IDs(a_j))$ $div(a_i, a_j) = 1 - Jaccard\ (users - countries(a_i) - users - countries(a_j))$
[SAI *et al.*, 12]	-	-	-	-	×	K Plus Loin Voisins - *Sélection des KPLV* - *Recommandation des items non aimés par les KPLV.*	-
[WAN et YIN, 13]	-Taux d'intérêt -Taux de nouveauté	-	-	-	×	Modèle du FC Synthétique *-N1 : RE basé popularité* *-N2 : RE basé Long Tail* *N=N1+N2*	-
[AYT et KAR, 14]	Seuil de contrôle des poids	×	-	-	×	Algorithme ClusDiv *-clustering des items* *-calcul des poids pour les clusters* *-sélection des items selon le poids du cluster*	$ZD(L(u)) = \frac{D(L(u)) - D(I)}{SD(I)}$
[PRE *et al.*, 13]	La formule HMDP	-	×	×	-	Classement à effet sur la diversité totale *-TDE* *-TDE-S* *-S-TDE*	$TDE(c_i) = \sum_{j=1..\lvert L_u \rvert} dist(c_i, c_j);\ c_i \neq c_j$ $avgdiv(TL) = \frac{\sum_{Lu \in TL} div(Lu)}{\lvert TL \rvert}$ $HMDP(TL) = 2 \times \frac{AvgDiv \times Avg\ Prec}{AvgDiv + AvgPrec}$

[ZHA 08]	-	-	×	-	×	Diversification des topiques	$diversty(I_1, I_2, \dots I_n) = \frac{\sum_{i=1,n}\sum_{j=i,n}(1 - similarity(I_i, I_j))}{\frac{n}{2} * (n-1)}$
[ZHA F. 09]	λ	-	-	-	×	Diversification des voisins *-Calcul des scores pour les items* *-Sélection de divers voisins confiant DVC* *-Prédiction avec DVC*	$W(u_k) = (1-\lambda) \times sim(T, u_k) + \lambda \times div(T, u_k)$
[SMY *et al.*, 01]		-	×	×	-	Diversité relative	$RelDiversity(i, C) = \begin{cases} 0 \; si \; C = \{\}; \\ \sum_{j=1.n} (1 - sim(I, Cj))/n \end{cases}$
[ZHA 09]	θ	-	-	-	×	*1) Algorithme probabiliste* *2) Partitionnement du profil* *3) Réduction de dimensionnalité*	1) $y^* = argmax(1-\theta)\alpha y^T D_y + \theta\beta m_u^T y$
[VAR et CAS, 11]	-	-	×	-	×	Diversité relative au rang et à la pertinence des items	$Diversity(R\backslash u) = \sum_{\substack{i_k, i_{l \in R} \\ k \neq l}} c_k disc(k) disc(l\|k) p(rel \backslash i_k, u) p(rel\|i_l, u)\; d(i_k, i_l)$
[CAS et VAR, 11]	-	-	×	-	×	Diversité relative au choix, à la découverte et à la pertinence des items	-$Novelty(R\backslash u) = \sum disc(n) p(rel\backslash i_n, u) p(j\backslash u) d(i,j)$ - $Diversity(R\backslash u) = \sum_{n,k} disc(n) p(rel\backslash i_n, u) p(i_k \backslash R) d(i_n, i_k)$
[BOI *et al.*, 11]	-	-	×	×	×	Diversité basée sur les médoïdes prioritaires	-
[GE et al., 11]	-	-	×	×	-	Classement des items divers en tête de la liste de recommandation	-

Tableau 2.2. Résumé de notre revue de la littérature sur la diversité dans les SRs

2.4.5. Discussion

Selon les critères présentés dans le tableau, nous pouvons considérer les algorithmes de diversité selon le type de recommandation générée en deux classes : les algorithmes de diversité basée sur le classement ou le reclassement des listes de recommandation, et les algorithmes de diversité basée sur la sélection des items divers.

Avec le premier type, le classement\ reclassement des items se fait sur la base de certains critères et méthodes. L'inconvénient majeur de cette méthode est le seuil de classement qui joue un rôle crucial pour cette méthode, où le choix d'une valeur pour ce paramètre affecte grandement la diversité voulue et la performance du système surtout quand il est laissé à l'utilisateur pour le régler **[ADO et KWO, 12].** Par conséquent le système devient ennuyeux pour l'utilisateur, qui peut perdre sa confiance au système qui est incapable de s'adapter à ses besoins, ce qui rend la diversité basée sur la sélection plus favorable que celle basée sur le classement.

D'un point de vue du contenu diversifié, nous pouvons dire que la méthode basée sur la sélection peut offrir plus d'items divers parce qu'elle sélectionne les items à partir de l'ensemble total des items dans le système. Cependant, la performance du système s'affecte et elle diminue énormément, ce qui nécessite l'élaboration d'un schème d'ajustement ou de formule qui inclue la diversité et la précision ensemble pour équilibrer le compromis entre eux.

L'utilisation d'un schème d'ajustement nécessite l'utilisation d'un paramètre ayant pour rôle d'ajuster le compromis entre précision et diversité, tel que utilisé dans **[ZIE *et al.*, 05][ZHA 09][PRE *et al.*, 09][WAN et YIN, 13][AYT et KAR, 14][ZHA F., 09]**. Ce qui est l'inconvénient majeur de ces méthodes, du fait qu'il nécessite l'intervention de l'utilisateur ou de l'administrateur du système pour choisir la bonne valeur de ce paramètre afin d'avoir des recommandations satisfaisantes sans une grande perte de performance. De plus, l'ensemble d'items divers sélectionnés peut ne pas être pertinent pour l'utilisateur ce qui conduit l'utilisateur à quitter le système avec le temps.

A partir de l'état de l'art présenté, nous pouvons voir que les études menées consistent à augmenter la diversité dans les listes de recommandation, et à mesurer son impact sur la performance du système **[CAS *et al.*, 13]** ou sur la satisfaction des utilisateurs **[JON**

10]. En outre, nous pouvons aussi constater que la majorité des méthodes ont élaboré la diversité en agissant sur le contenu des items c.à.d. elles utilisent un filtrage basé sur le contenu pour assurer la diversité [**ZIE *et al.*, 05][ZHA *et al.*, 08][LAT *et al.*, 08][BOI *et al.*,11],** avec seulement deux travaux **[SAI *et al.*, 12] [ZHA F., 09]** qui ont utilisé le filtrage collaboratif.

Le premier est basé sur les plus loin voisins et recommande les items qu'ils n'ont pas aimé et le deuxième utilise un ensemble de divers voisins et recommande les items qu'ils ont aimés. La première méthode basée sur les KPLV garantie la génération des items divers, mais cela conduit à une perte considérable de la précision des recommandations. De plus, ces items peuvent ne pas être pertinents pour l'utilisateur du faite qu'ils sont très dissimilaires à ces préférences, et s'il continu à recevoir des recommandations non pertinentes et qui ne correspondent pas à ces besoins, il s'ennuie et finira par quitter le système. Le deuxième algorithme est plus performant que le premier parce qu'il sélectionne des voisins divers confiants et utilise leurs préférences pour le calcul des recommandations en utilisant un paramètre pour ajuster le compromis entre diversité et similarité. Cependant, elle nécessite aussi un réglage du paramètre d'ajustement et elle ne garantie pas la pertinence des items recommandés pour l'utilisateur en raison de la dis-similarité entre l'utilisateur et les voisins choisis.

A notre connaissance et à partir de l'étude de l'état de l'art que nous avons fait, aucun travail n'a utilisé un filtrage collaboratif qui assure la pertinence et la diversité, simultanément, dans les listes d'items générées à l'utilisateur et sans utiliser un paramètre ou fonction pour l'ajustement, ce qui rend le système plus adaptatif aux besoins de l'utilisateur.

2.4.6. Conclusion

Dans la première partie du chapitre, nous avons présenté quelques méthodes existantes dans la littérature sur le filtrage collaboratif, en montrant leurs efficacités à prouver la performance de prédiction des évaluations, tout en appuyant sur les deux méthodes puissantes : FCM et NMF.

La deuxième partie du chapitre présente la notion de diversité dans les SRs, en citant différents algorithmes et mesures de diversités présentés dans la littérature.

En se basant sur les différentes notions et travaux présentés, nous présentons dans le chapitre suivant deux nouvelles approches pour les systèmes de recommandation et une troisième approche qui les combine, afin de rendre ces systèmes plus adaptatifs aux besoins des utilisateurs, tout en réduisant les effets de la clairsemé et l'évolutivité de données, et pour assurer la diversité dans les listes de recommandation.

CHAPITRE 3

UN SYSTEME DE RECOMMANDATION HYBRIDE ET DIVERSIFIEE

RESUME DU CONTENU

Dans ce chapitre, deux contributions ont été présentées en vue d'améliorer la qualité des recommandations générées pour l'utilisateur en terme de contenu diversifié en plus des items pertinents, afin d'éviter de perdre la performance de la prédiction. La première contribution présente un système de recommandation hybride diversifiée basée flou, qui combine entre un Filtrage Collaboratif 'FC' flou et un Filtrage Basé Contenu 'FBC' pour améliorer la précision des prédictions des évaluations inconnues, et qui servent comme données d'entrée pour un processus de recommandation diversifiée. La deuxième contribution présente un système de recommandation hybride basée sur une nouvelle notion qui est la notion du niveau de similarité. Cette notion se base sur la similarité entre les items non vus et les préférences de l'utilisateur en termes de voisins des voisins qui les ont évalués, pour justifier la pertinence des items divers à recommander. Une troisième approche hybride est également proposée dans ce chapitre, qui combine entre l'algorithme du FC basé flou et le processus de diversification basé sur le niveau de similarité.

PARTIE 1 :

UN SYSTEME DE RECOMMANDATION HYBRIDE DIVERSIFIEE BASEE FLOU

3.1. Introduction

Les choix et les goûts des utilisateurs sont vagues et incertains parce que des milliers de différents items sont disponibles en ligne, et les utilisateurs font leurs évaluations pour peu de produits seulement, qui peuvent être de la même catégorie comme de différentes catégories.

La première partie de ce chapitre présente un Système de Recommandation hybride diversifiée basée flou qui est capable de recommander des items aux utilisateurs qui correspondent à leurs différents choix et intérêts en considérant l'incertitude et l'ambiguïté dans leurs préférences. Le système intègre deux approches distinctes, dont la première vise à performer la phase du FC en utilisant un algorithme basé sur le flou et la deuxième approche vise à performer le FBC.

La deuxième partie du chapitre présente une nouvelle approche hybride pour augmenter la diversification des recommandations qui se base sur le principe de la nouveauté, en introduisant une nouvelle notion qui est la notion du niveau de similarité entre les items et l'utilisateur actif.

Une troisième proposition sera également présentée dans la deuxième partie, qui est une approche hybride qui combine entre le FC basé sur le flou avec la méthode basée sur le niveau de similarité.

3.2. Un Système de Recommandation Hybride Diversifiée basée Flou

En vue d'améliorer la qualité des recommandations et d'atteindre la satisfaction des utilisateurs, cette première partie présente une nouvelle approche de recommandation hybride visant à rendre les systèmes de recommandation plus adaptatifs et capables de considérer les différents choix et goûts des utilisateurs, par la diversification du contenu des items recommandés, en améliorant la phase du filtrage collaboratif.

La section suivante présente les objectifs visés par cette approche ainsi que les motivations derrière chaque étape de l'approche.

3.2.1. Objectifs et Motivations

Nos objectifs visés dans cette première approche sont:

- La génération des recommandations qui tiennent compte des différents choix et goûts des utilisateurs, afin d'éliminer le problème de la stabilité des systèmes de recommandation par rapport au changement du profil de l'utilisateur.
- Amélioration de la précision des prédictions par la combinaison entre les prédictions, basée FC et basée filtrage par contenu.
- La réduction des problèmes du FC, évolutivité et clairsemé des données.
- Amélioration de la qualité des recommandations Top-N, en réduisant le problème du manque de données par l'utilisation des résultats de la prédiction dans la phase de la recommandation afin d'avoir des matrices pleines et des recommandations pertinentes.

Notre objectif principal visé dans cette première approche est la prise en compte des différents goûts et choix des utilisateurs, en assurant plus de diversification dans le contenu des recommandations générées, afin de satisfaire tous leurs besoins. Pour se faire, la diversification du voisinage de l'utilisateur est visée afin de lui générer des items différents et pertinents en même temps.

Notre deuxième objectif principal est de réduire le problème de la clairsemé des données de la matrice d'évaluation, en profitant des résultats de la prédiction des évaluations manquantes dans la phase de la recommandation Top-N. Après la phase de la prédiction, les valeurs des items que l'utilisateur actif n'a pas encore évalués sont prédites, et par conséquent le processus de la recommandation sera appliqué sur des données complètes pour l'utilisateur (presque tous les items ont une note connue), ce qui augmente la précision des calculs et des recommandations générées.

Notre troisième objectif est la réduction des deux problèmes de la méthode du FC, qui sont la clairsemé et l'évolutivité des données.

Comme la recherche du voisinage est une étape importante du processus du FC, notre quatrième objectif, qui est un objectif secondaire, est la conception d'un algorithme du FC qui permet de trouver le bon voisinage de l'utilisateur.

Par conséquent, nous avons cherché à concevoir un algorithme du FC pour améliorer la première phase de la recommandation, qui inclut la création des groupes dans le système et l'affectation des nouveaux utilisateurs aux groupes pertinents, la recherche de l'ensemble du voisinage candidat, ainsi que la prédiction des évaluations inconnues.

Après avoir accompli l'étude des travaux utilisés dans le domaine du filtrage collaboratif sur le *clustering* des utilisateurs et la prédiction des évaluations manquantes

dans la matrice utilisateur-item. Nous avons constaté qu'un algorithme efficace du FC doit porter les points suivants:

- Construction automatique des clusters en utilisant un algorithme efficace pour la segmentation des données : l'ensemble des utilisateurs et l'ensemble des items.
- Affectation de l'utilisateur actif aux différents groupes avec différents degrés d'appartenance.
- La recherche des voisins pertinents de l'utilisateur actif qui est une étape importante de l'algorithme du filtrage collaboratif.
- Une bonne prédiction des évaluations manquantes dans la matrice d'évaluation pour l'utilisateur actif.

Afin de répondre à ces critères, et en se basant sur les travaux présentés dans l'état de l'art, nous proposons d'utiliser un algorithme du FC qui satisfait les points suivants:

- Construction automatique des groupes dans le système à partir de la matrice d'évaluation en utilisant la méthode de factorisation en matrices non négatives. La raison derrière ce choix est que :
  - La méthode de factorisation matricielle permet de segmenter les ensembles à large volume de données, contenant plus qu'un type de données ce qui est le cas dans la matrice d'évaluation qui contient les utilisateurs ainsi que les items.
  - L'utilisation de cet algorithme dans le FC a prouvé une réduction énorme du problème de l'évolutivité des données lors de l'ajout d'un nouvel utilisateur ou d'un nouvel item, ainsi que le problème de la clairsemé de données car après la factorisation nous obtenons des matrices à rang plus inférieur que la matrice originale contenant que les informations significatives.
  - Cette extension des méthodes de factorisation matricielle est conçue pour être appliquée sur des matrices positives qui contiennent des valeurs ≥ 0, ce qui correspond à notre cas (la matrice d'évaluation ne contient que des valeurs positives ou 0 pour les items non évalués).
- Prise en compte de l'ambiguïté dans les préférences des utilisateurs en utilisant l'algorithme Fuzzy C-Means 'FCM', qui permet leur affectation à différents groupes selon des degrés d'appartenance.
- Combinaison entre les idées de l'algorithme FCM et la méthode de factorisation en matrices non négatives qui a prouvé son efficacité dans le domaine du FC, afin de profiter de leurs avantages.

- La sélection d'un ensemble candidat des voisins pour améliorer la précision des prédictions.
- Génération d'une bonne prédiction pour améliorer la qualité des recommandations, en combinant entre les différentes prédictions basées FC.

Un cinquième objectif est l'amélioration de la précision des prédictions en incluant le contenu des items non évalués. La majorité des travaux génèrent des prédictions en utilisant le FC alors que la prédiction basée contenu joue un rôle important pour la précision des prédictions, qui tient compte de la pertinence du contenu des items par rapport aux préférences de l'utilisateur.

3.2.2. Proposition

A l'issue des problèmes discutés et des objectifs visés, nous proposons une nouvelle approche de recommandation hybride basée sur la notion du flou, afin de considérer tous les choix et préférences incertaines des utilisateurs et d'atteindre tous leurs besoins.

L'approche prend en considération les notions de base du FC, qui sont le *clustering* des utilisateurs, le choix des voisins candidats et la prédiction des évaluations manquantes de la matrice utilisateur-item, et elle vise à réduire les deux problèmes majeurs du FC, la parcimonie et l'évolutivité des données. Les étapes de l'approche basée FC proposée sont les suivantes:

- ✓ Construction automatique des groupes dans le système à partir de la table d'évaluations en utilisant la méthode de factorisation en matrices non négatives. La raison derrière ce choix d'utilisation et que cet algorithme réduit d'une façon énorme le problème d'évolutivité des données lors de l'ajout d'un nouvel utilisateur ou d'un nouveau item.
- ✓ Affectation multiple des utilisateurs aux différents groupes selon des degrés d'appartenance, en utilisant le principe de l'algorithme FCM dans la méthode FMNN, afin de considérer l'ambiguïté dans les préférences des utilisateurs lors de la construction des clusters.
- ✓ Ensuite, pour la sélection du voisinage, Nous proposons de prendre en considération juste les *k*-plus proches voisins appartenant aux *C*-plus proches clusters suivant le principe de **[XUE et al., 05], [CHE et al., 09]** mais en utilisant l'extension floue de l'algorithme. Nous proposons d'utiliser les informations incluses dans la matrice

d'appartenance résultante après la factorisation, pour la sélection du voisinage de l'utilisateur actif. Notre avantage est le parcours d'une petite matrice ce qui réduit énormément le temps de calcul alors que les méthodes traditionnelles basées sur le voisinage ont généralement besoin de chercher toute la base de données, qui souffre définitivement du problème d'évolutivité.

- ✓ La prédiction des évaluations manquantes dans la matrice des évaluations.

Pour améliorer la qualité des recommandations, en tenant compte du contenu des items, un algorithme de filtrage basé sur le contenu est proposé et fusionné avec l'algorithme du FC basé flou afin de prédire les évaluations manquantes de la matrice utilisateur-item.

Un processus de diversification des recommandations Top-N sera proposé, par la suite, en performant la précision des recommandations par l'utilisation des résultats de la phase de la prédiction hybride.

La diversification visée sera assurée en diversifiant le voisinage de l'utilisateur afin de lui générer des items différents et pertinents en même temps. Premièrement, dans la phase du FC, un ensemble divers des plus proches voisins est choisi comme ensemble candidat afin d'aboutir une prédiction plus précise des évaluations manquantes. Ensuite, la diversification est achevée dans la phase de la recommandation Top-N, en recommandant des items divers à partir des plus proches clusters de l'utilisateur.

Les détails de cette première proposition sont présentés dans les sections suivantes.

3.2.3. Processus de Recommandation Hybride Diversifiée basée Flou

Le processus de la recommandation diversifiée basée flou est conçu comme une combinaison de plusieurs composants spécialisés en différentes phases de la recommandation. Les étapes du processus de la recommandation sont schématisées dans *la Figure 3.1*.

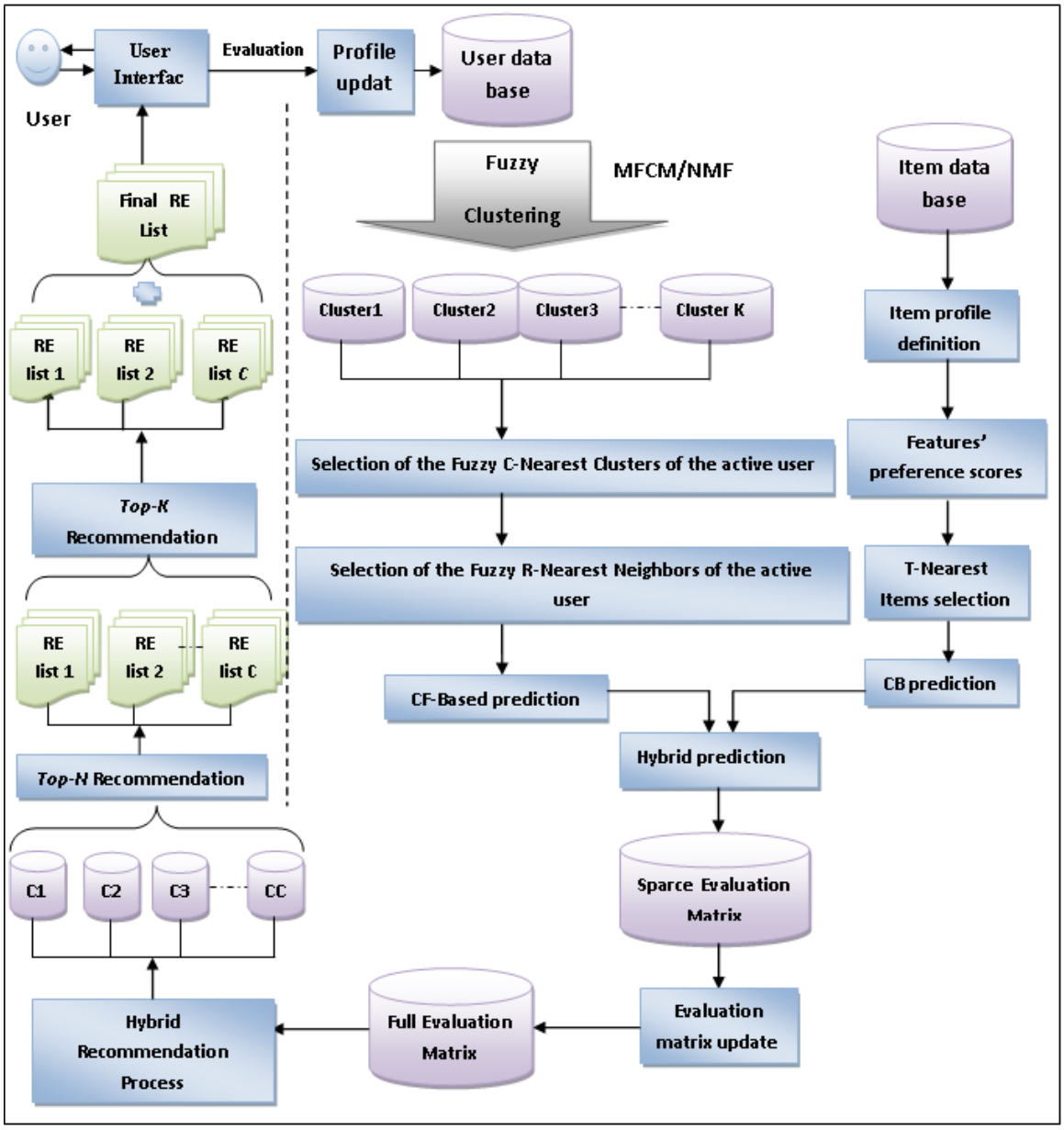


Figure 3.1 : Processus de la Recommandation diversifiée hybride basée flou

Le processus de la recommandation diversifiée basée flou peut être considéré comme deux grandes parties. La première partie, à droite, est la partie consacrée à la construction des clusters flous et la multi-affectation des utilisateurs actifs en clusters jugés pertinents, en considérant leurs intérêts vagues et imprécis. Puis, les plus proches voisins sont sélectionnés parmi les plus proches clusters comme ensemble de voisinage candidat. Après cela, une prédiction hybride des évaluations manquantes est faite en combinant entre les prédictions basées filtrage collaboratif flou et filtrage basé contenu.

La deuxième grande partie, à gauche, représente le processus de la recommandation diversifiée. Tout d'abord, un processus de recommandation est généré afin de calculer

la prédiction. Après avoir calculé les scores pour les items, les Top-N items sont sélectionnés en tant que liste de recommandations pertinentes générées dans chaque groupe similaire. Ensuite, la liste de recommandation finale est construite en choisissant les Top-K items à partir de chaque Top-N liste selon le degré d'appartenance de l'utilisateur aux groupes correspondants. Les détails de ces processus sont présentés dans les sections suivantes, mais d'abord nous allons présenter quelques notations utilisées tout au long de la suite du chapitre.

3.2.4. Notations

Nous définissons ici quelques notations utilisées au cours de notre approche. Comme tous les systèmes de recommandation, notre système utilise deux entités différentes, les utilisateurs représentés par *u* et les items par *t*.

- L'évaluation faite par l'utilisateur *u* à l'item *t* est $r_{u,t}$, où toutes les évaluations faites par l'utilisateur construisent un vecteur $\vec{V}_{u,t}$ qui représente le *profil de l'utilisateur*.
- La matrice utilisateur-item contenant toutes les évaluations est *R*.
- Soit $z_{u,k}$ la probabilité que l'utilisateur *u* appartient au cluster *k*, et Z = ($z_{u,k}$) est la matrice de probabilité $U \times K$, où *U* et *K* représentent le nombre des utilisateurs et des clusters, respectivement.
- Soit $c_{k,t}$ le centre de gravité de l'item *t* dans la classe *k (la moyenne des évaluations faites par les membres du groupe k pour l'item t)*, et C = ($c_{k,t}$) est la matrice des centres de gravité $K \times T$, où *T* est le nombre des items.

Notez que dans notre système, les items sont des films et les utilisateurs sont modélisés par leurs évaluations de ces films appartenant à un intervalle de [1,5], ce qui est le cas dans l'ensemble de données *Movilens*.

Nous entamons maintenant la présentation des différentes parties de l'approche proposée. En commençant par l'algorithme du filtrage collaboratif, puis l'algorithme du filtrage basé sur le contenu. Ensuite, la prédiction hybride sera présentée. Après avoir prédit les évaluations manquantes dans la matrice utilisateur-item, le processus de la recommandation diversifiée est généré.

3.2.5. L'algorithme du filtrage collaboratif-basé flou

Cette section contient les étapes détaillées de notre algorithme du filtrage collaboratif composé de trois étapes importantes : le regroupement des utilisateurs et l'affectation de l'utilisateur actif à différents groupes; la sélection de l'ensemble candidat du voisinage flou utilisé dans la phase de prédiction; et la troisième étape est l'étape de la prédiction.

3.2.5.1. L'algorithme du clustering basé flou: Modified FCM to NMF (MFCM-NMF)

Pour regrouper les utilisateurs, nous avons utilisé une version ajustée de l'algorithme MFCM2 proposé par **[WU and LI, 08],** qui factorise la matrice d'évaluation *R* en deux matrices *Z* et *C*, où *Z* est la matrice de probabilité et *C* est la matrice des centres de clusters, mais en utilisant la méthode NMF au lieu de FM standard, pour imposer la contrainte de non négativité sur les éléments de la matrice *C*. Tant que *C* est la matrice des centres de clusters, où chaque cellule représente la moyenne des évaluations faites par les membres d'un groupe à un item, ses éléments doivent être ≥0.

Nous nous sommes orientés vers cet algorithme car il hérite les avantages de l'algorithme FCM, qui permet d'affecter les utilisateurs à plusieurs clusters avec des probabilités appropriées, ainsi qu'à partir de l'algorithme de NMF, qui permet de regrouper les grands ensembles de données contenant différents types de données, simultanément, et produit des matrices positives, ce qui correspond à notre cas. En outre, nous voulons profiter de la matrice de probabilité résultante pour calculer les similarités entre les utilisateurs avec une nouvelle mesure de distance proposée ci-dessous, ce qui réduit efficacement les calculs et les problèmes à grande échelle du FC de manière plus efficace.

En ajoutant la contrainte de positivité au problème formulé dans **[WU and LI, 08],** l'algorithme est nommé MFCM-NMF (*Modified FCM to NMF*) et est présenté comme suit :

$$H(Z,C) = \frac{1}{2}\sum_{(u,t)\in P}\left(r_{u,t} - \frac{1}{\sum_{k=1}^{K} e^{z_{u,k}}}\sum_{k=1}^{K} e^{z_{u,k}} c_{k,t}\right)^2 + \lambda_c \left||c_t|\right|_2^2 + \lambda_z ||z_u||_2^2 \quad \text{[WU and LI, 08] (3.1)}$$

$$\text{St. } Z_1 = 1 \text{ ; } Z \geq 0 \text{ ; } C \geq 0.$$

Où, $r_{u,t}$ est l'évaluation faite par l'utilisateur *u* à l'item *t*, et $e^{z_{u,k}}$ est la probabilité que cet utilisateur appartienne au cluster *k*, ayant comme centre $c_{k,t}$ qui est la moyenne des

évaluations faites par les membres de ce groupe à cet item, avec *k* un nombre positive. $0<=\lambda_c,\lambda_z<=1$ sont des petits paramètres de régularisation définis à travers les expérimentations. Sachant que nous avons utilisé une formule de normalisation de la matrice de probabilité Z pour mettre la somme des éléments de chaque ligne égale à 1.

3.2.5.2.1. L'algorithme de résolution

Comme nous l'avons déjà mentionné dans le chapitre 1 (sous section 1.7.2.1.1), plusieurs algorithmes ont été proposés pour résoudre la fonction objective de la méthode de factorisation en matrices non négatives.

Dans notre travail, nous avons utilisé l'algorithme des Moindres Carrés Contraints Alternés (*Alternating Constrained Least Squares)* (ACLS) proposé par **[LAN *et al.*, 06],** qui est une extension de l'algorithme des Moindres Carrés Alternés ALS (*Alternating Least Squares*). Cet algorithme exécute une étape des moindres carrés standards (sans contraintes) **[PAA and TAP, 94],** et après tous les éléments négatifs dans le vecteur de la solution sont mis à 0. La raison derrière ce choix est que, chaque étape de l'algorithme ACLS résout un petit système de matrice $k \times k$, ACLS est l'un des algorithmes de NMF disponibles les plus rapides (plus rapide que les algorithmes SVD). (Voir **[LAN *et al.*, 06]** section 3.4 pour des temps d'exécution comparatives, où ACLS converge rapidement et donne des facteurs de NMF très précis). De plus, cet algorithme impose la non négativité sur le contenu des deux facteurs en mettant simplement tous les éléments négatifs égaux à zéro à chaque étape, ce qui correspond à notre besoin, i.e. tous les éléments des matrices Z et C doivent être >=0.

Enfin, cet algorithme initialise seulement un facteur (une des matrices résultante et non pas les deux), ce qui réduit considérablement le temps d'exécution. L'algorithme 3.1 décrit les étapes de l'algorithme ACLS [LAN *et al.*, 06]. Avec une modification dans l'ordonnancement des étapes de l'algorithme, où on a commencé par l'initialisation du deuxième facteur qui est la matrice C, puis après, nous l'avons considéré comme étant le facteur fixe et nous avons calculé Z, parce que la méthode d'initialisation que nous allons proposer initialise C et non pas Z. I est une matrice identité.

Algorithme 3.1. Algorithme ACLS pratique pour NMF

Entrée: λ_c, λ_z, p
Sortie: *Z et C*
Début
1. C = *rand*(*u*,*p*);
2. Pour *i* = 1 : *maxiter* faire,
3. (cls) Résoudre Z dans l'équation de la matrice ($CC^T + \lambda_Z I$) Z^T = CR^T. % pour C fixé, trouver Z
4. (nonnég) Mettre tous les éléments négatifs dans Z à 0.
5. (cls) Résoudre C dans l'équation de la matrice ($Z^T Z + \lambda_C I$) C =Z^TR. % pour Z fixé, trouver C
6. (nonnég) Mettre tous les éléments négatifs dans C à 0.
7. Fin Pour
Fin

3.2.5.2.2. Méthode d'initialisation

Tous les algorithmes de la méthode de factorisation en matrices non négatives sont itératifs et il est bien connu qu'ils sont sensibles à l'initialisation des deux facteurs **[WIL 03].** Certains algorithmes exigent que les deux facteurs doivent être initialisés **[HOY 02], [HOY 04], [LEE and SEU 99],[LEE and SEU 01], [PAU et al. 05],** tandis que d'autres nécessitent une initialisation d'un seul facteur **[PAA 94], [PAA 97], [SHA 05], [SHA 06].** Dans tous les cas, une bonne initialisation peut améliorer la vitesse et la précision des algorithmes, comme elle peut produire une convergence plus rapide à un minimum local **[LAN *et al.*, 06].** Une bonne initialisation peut éviter certains problèmes de convergence. Pour plus d'information sur les différentes méthodes d'initialisation veuillez consulter **[LAN *et al.*, 06].**

Un avantage des algorithmes ALS, comme ACLS, est qu'ils ne nécessitent qu'une initialisation d'un seul facteur (la matrice *C* dans notre cas). Dans les algorithmes ALS, une fois le premier facteur est connu, le deuxième facteur est calculé rapidement par un calcul des moindres carrés. Par conséquent, nous nous sommes intéressés aux techniques de calcul d'un bon C^0.

Comme nous l'avons déjà dit, l'utilisation de l'algorithme ACLS nécessite seulement l'initialisation de la matrice des centres de clusters C, où chaque ligne contient la moyenne des évaluations faites par les membres d'un cluster à tous les items. Pour initialiser C, nous proposons une version ajustée de la méthode *random Acol initialization* proposée dans **[LAN *et al.*, 06],** qui initialise les colonnes d'une matrice

aléatoirement. La méthode ajustée initialise chaque ligne de la matrice C en faisant la moyenne de *p* lignes choisies aléatoirement à partir de la matrice d'évaluation *R* (i.e. faire la moyenne des vecteurs d'évaluations de *p* utilisateurs choisis aléatoirement). Nous appelons la nouvelle méthode ajustée : *méthode d'initialisation Rlignes aléatoires* (*rand Rrows initialization method*). Notons que *p* est un entier positif prédéterminé et fixé à la valeur 10 (*p*=10), c.à.d. 10 lignes de la matrice *R* sont choisies aléatoirement pour initialiser chaque ligne de la matrice C.

Nous avons choisi cette méthode d'initialisation, car il est plus logique de construire les vecteurs de base d'un facteur à partir des données existantes dans la matrice d'évaluation (à factoriser), que de former des vecteurs de façon aléatoire, comme l'initialisation aléatoire (*Random initialisation*). De plus, il a été prouvé que cette méthode est très peu coûteuse, et se situe entre l'initialisation aléatoire et l'initialisation centroïde en termes de performances **[LAN 05.a] [LAN 05.b].**

Exemple : Pour bien comprendre le principe de *l'initialisation rand Rrow*, l'exemple suivant explique comment cette méthode initialise la première ligne de la matrice C. Supposons que p=4, alors nous sélectionnons quatre utilisateurs aléatoirement à partir de la matrice R comme membres initiaux du cluster C1, où leurs indices sont choisis aléatoirement et stockés dans un tableau vecteur *n* de taille 4, avec les évaluations suivantes [1,0,2,0,…,3,5], [0,5,4,0,…,0,2], [3,1,0,0,…,0,4], [0,0,2,0,…,5,3]. Ensuite, la 1ère ligne de la matrice *C* sera initialisée par [1, 1.5, 2,0,…, 2, 3.5] qui est le vecteur moyen des quatre vecteurs comme montré dans la Figure 3.2.

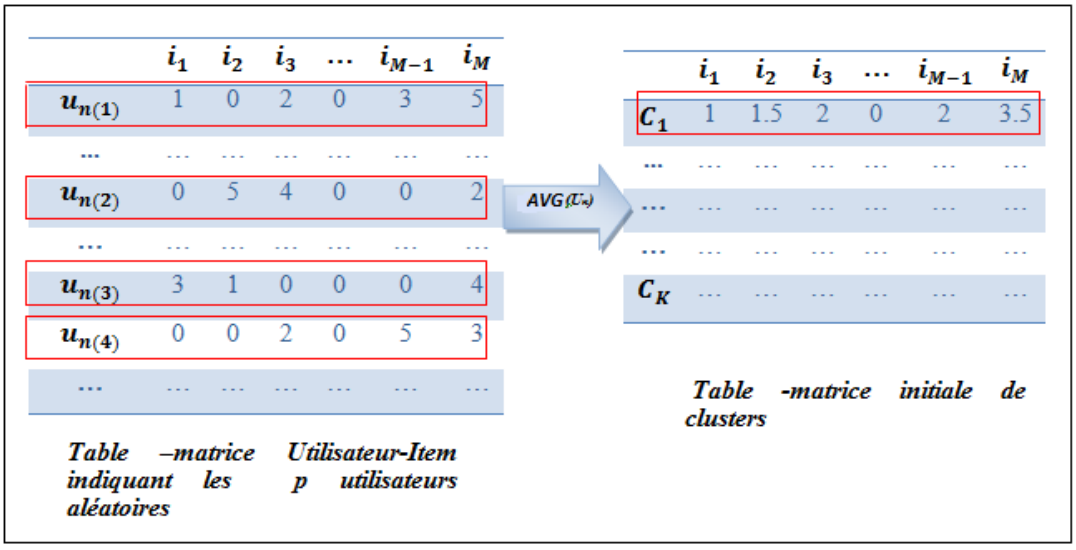


Figure 3.2. Exemple illustratif de la méthode d'initialisation rand Rrow

3.2.5.2.3. Critère de convergence

La majorité des algorithmes de NMF utilisent un critère de convergence le plus simple possible, qui est un nombre fixe d'itérations noté, *maxiter*. Ce critère est utilisé souvent parce que le critère naturel, arrêter quand $\|R - ZC\| \leq \varepsilon$ nécessite la recherche de la bonne valeur de ε, comme seuil de l'erreur de factorisation, ce qui conduit à plus de calculs.

Notez que *maxiter* était le critère de convergence de l'algorithme ACLS utilisé dans **[LAN *et al.*, 06].** Cependant, un nombre d'itérations fixé au préalable par l'utilisateur n'est pas la bonne façon pour vérifier la convergence d'un algorithme parce que la valeur la plus appropriée pour *maxiter* est un problème dépendant, i.e. il dépend des données d'entrées et des valeurs de paramètres choisies pour l'algorithme. Ainsi, nous proposons d'utiliser ce critère pour contrôler la convergence de notre algorithme mais en trouvant la valeur optimale de *maxiter* expérimentalement et non pas en la supposant par l'utilisateur.

3.2.5.2. Identification du voisinage

Les algorithmes basés sur le filtrage collaboratif génèrent des prédictions ou des recommandations pour les utilisateurs actifs en se basant sur les opinions de leurs voisins, ce qui nécessite la recherche des voisins candidats à partir de l'ensemble total d'utilisateurs en utilisant des métriques de similarité populaires comme la Corrélation Cosinus et la Corrélation de Pearson **[BRE *et al.*, 98] [HER *et al.*, 99].**

Une idée logique pour trouver l'ensemble candidat de voisins est qu'ils se trouvent dans les clusters les plus proches de l'utilisateur, ce qui nécessite de chercher seulement dans les plus proches groupes plutôt que parcourir toute la base de données. Cette idée a été achevée dans **[XUE *et al.*, 05]** et **[CHE *et al.*, 09],** où les *k*-plus proches voisins ont été sélectionnés à partir des *C*-plus proches clusters, en utilisant les méthodes ordinaires de similarité basées utilisateur.

En raison de la capacité des algorithmes du *clustering* flous, à considérer la nature ambigüe des voisins pendant la détermination du voisinage de l'utilisateur actif; nous proposons des versions adoptées et modifiées des algorithmes des *C*-plus proches clusters flous CPPCF (*Fuzzy C-Nearest Clusters FCNC*) et des *K*-plus proches voisins flous KPPVF (*Fuzzy K-Nearest Neighbors FKNN*) **[KEL *et al.*, 85].** Ces algorithmes

seront employés pour sélectionner l'ensemble candidat des voisins qui sera utilisé par la suite dans les calculs des prédictions et des recommandations. Le principe de la sélection du voisinage est le suivant:

- Tout d'abord, nous sélectionnons les *C*-plus proches clusters flous, en utilisant les degrés d'appartenance de l'utilisateur aux clusters comme indice de similarité entre l'utilisateur et les groupes. Où, les degrés d'appartenance sont sélectionnés à partir de la matrice Z qui résultent de la phase de factorisation. La raison derrière l'utilisation du degré d'appartenance comme indice de similarité est notre supposition qu'une forte appartenance indique automatiquement une grande similarité dans les préférences de l'utilisateur et des membres du groupe en question. De plus, nous avons choisi de prendre en considération juste les *C*-plus proches groupes flous parce qu'il est inintéressant de considérer les préférences des groupes très dissimilaires de ceux de l'utilisateur (ex: il n'est pas important de considérer les préférences des membres d'un groupe *c* lors de la génération des recommandations pour l'utilisateur, où le degré d'appartenance de cet utilisateur à ce groupe est égale à 0.003). Donc, nous choisissons juste les clusters jugés pertinents pour l'utilisateur ayant les plus grandes valeurs d'appartenance.
- Ensuite, nous calculons la similarité entre l'utilisateur actif et les membres des *CPPCF* trouvés, pour choisir les *K*-plus proches voisins flous. Une nouvelle mesure de similarité est utilisée pour calculer la correspondance entre les utilisateurs en se basant sur les degrés d'appartenance aux groupes, afin de minimiser le temps de calculs et de bénéficier des résultats de la factorisation.

Nous allons présenter ci-dessous les étapes détaillées de notre méthode pour identifier les *K*-plus proches voisins flous de l'utilisateur actif.

3.2.5.2.1. Sélection des C-plus proches Clusters flous

L'algorithme des C-plus proches clusters flous vise à chercher les plus proches clusters à partir de l'ensemble total des clusters, sur la base du critère d'ambiguïté et du flou dans les préférences générales des groupes.

Nous proposons de chercher la ligne du vecteur correspondant à l'utilisateur actif dans la matrice de probabilité Z, et de prendre seulement les C clusters (avec $C<k$) pour lesquels l'utilisateur a eu les plus grandes probabilités (les plus grands degrés

d'appartenance), ce qui élimine les calculs de similarité et minimise considérablement le temps de calcul.

Les étapes de l'algorithme sont présentées dans la *l'algorithme 3.2.*

Algorithme 3.2. C- Plus Proches Clusters Flous (*CPPCF*)

Entrée: u_a : l'utilisateur actif, *C :* une valeur ∈[1,…, *k*] (*k* est le nombre de clusters)
Sortie : *CPCF*
Début
1. NP = 0
2. Pour (*i = 0 à k*) faire
3. Si (*NP* ≤ *C*) alors
4. Inclure c_i dans l'ensemble *CPCF*
5. NP := NP+1
Sinon
7. c_m ← le cluster dans *CPCF* avec la valeur la plus petite valeur de probabilité ($z_{a,m}$ est la plus petite)
8. Si ($z_{a,i} > z_{a,m}$) Sinon
9. Supprimer c_m à partir de *CPCF*;
10. Inclure c_i dans l'ensemble *CPCF*;
11. Fin Si;
12. Fin Si;
13. Fin Pour;
14. Renvoyer *(CPCF)*
Fin.

L'algorithme de sélection CPPCF, sélectionne tout d'abord l'ensemble des C clusters qui se situent en première position de la matrice Z. Ensuite, à chaque fois qu'il trouve un cluster avec un degré d'appartenance plus grand que ceux dans l'ensemble sélectionné, il l'ajoute à l'ensemble et retire le cluster ayant la plus petite valeur d'appartenance.

3.2.5.2.2. Sélection des K-plus proches voisins flous (KPPVF)

Pour sélectionner l'ensemble candidat des voisins flous KPPVF, nous proposons une nouvelle mesure de similarité qui utilise la différence entre les degrés d'appartenance à un seul cluster, entre l'utilisateur actif et les membres des *CPCF*, pour minimiser les calculs comme suit :

$$sim(ua, v) = |Z_{ua,c} - Z_{v,c}| \quad , \{v \in CPCF\} \tag{3.2}$$

Où $Z_{ua,c}$et $Z_{v,c}$ sont les probabilités d'appartenance de l'utilisateur actif *ua* et l'utilisateur *v* au cluster *c*, avec $c \in$ *CPCF*. La similarité entre deux utilisateurs

augmente lorsque la différence entre leurs degrés d'appartenance à un même cluster tant vers 0.

L'algorithme 3.3 résume les étapes de sélection des *K*-plus proches voisins flous.

Algorithme 3.3. les Plus Proches Voisins Flous (KPPVF)

Entrée: u_a: l'utilisateur actif,
K : une valeur ∈[1,…,$|U_c|$] ($|U_c|$ Nombre d'utilisateurs dans les *CPCF*)
Sortie *: KPVF*
Début
1. neighbors =0;
2. Pour (i=1 à $|U_c|$) faire ;
3. Si ($u_i \in$ *CPCF*) alors,
4. Si (neighbors ≤ *K*) alors,
5. Inclure $u_i$dans l'ensemble de *KPVF*
6. neighbors← neighbors+1
7. Sinon
8. u_{ds} ← l'utilisateur le plus éloigné (ou le moins similaire) des *KPVF*
9. Si ($| z_{i,c} - z_{a,c}| < |z_{ds,c} - z_{a,c}|$) alors
10. Supprimer u_{ds} à partir de l'ensemble des *KPVF* ;
11. Inclure u_i dans l'ensemble des *KPVF* ;
12. Fin Si
13. Fin Si
14. Fin Si
15. Fin Pour
16. Renvoyer (*KPVF*)

Fin

Après le calcul de similarité entre l'utilisateur actif et les membres des plus proches clusters flous, nous choisissons les *K* voisins les plus similaires (avec les plus petites valeurs de dissimilarité). *K* et C sont des nombres positifs définis à travers les expérimentations.

3.2.5.3. La prédiction basée sur le filtrage collaboratif (CF-based prediction)

Afin d'avoir une bonne performance et en se basant sur les résultats présentés dans [**CHE *et al.*, 09],** une combinaison linéaire des prédictions similaires à celui de Chen *et al.*, [**CHE *et al.*, 09]** est utilisée pour fusionner les trois prédictions du FC: basée MFCM-NMF, basée-utilisateur et basée-item, respectivement.

$$CFB\_pred(ua,t) = \lambda\, \widehat{nr}_{ua,t} + \delta(1-\lambda)\widehat{ur}_{ua,t} + (1-\delta)(1-\lambda)\widehat{ir}_{ua,t} \quad \text{[CHE et al., 09]} \quad (3.3)$$

Où, $\widehat{nr}_{ua,t}$, $\widehat{ur}_{ua,t}$et $\widehat{ir}_{ua,t}$ représentent les résultats de prédiction des algorithmes basés sur MFCM-NMF, FC basé utilisateur et FC basé items, respectivement, comme présentés dans le chapitre 1, section 1.9 (Formules 1.29, 1.30 et 1.31, respectivement). $0 \leq \lambda, \delta \leq 1$ sont des coefficients de fusion pour contrôler le poids entre les différentes prédictions, que nous allons définir expérimentalement.

3.2.5.4. Algorithme du Filtrage Collaboratif basé Flou

L'algorithme 3.4 résume les étapes du processus du filtrage collaboratif flou proposé ci-dessus.

Algorithme 3.4. Filtrage Collaboratif basé Flou

Entrée: R : matrice d'évaluation, ua : utilisateur actif, *k*: nombre de facteurs,
C, K : nombres des plus proches clusters et des plus proches voisins, respectivement,
Sortie: CFB_pred
Début
1. (Z,C) = *MFCM-NMF* (*R*,*k*) ; factorisation de la matrice R en Z et C ;
2. CPCF = *CPPCF* (*Z*,*C*) ; sélection des plus proches clusters flous ;
3. KPVF= *KPPVF* (*CPCF, Z, K*), sélection des plus proches voisins flous ;
4. Pour (*t* = 1:T) faire;
5. Si $r_{ua,t}$ = 0 alors,
6. $\widehat{nr}_{ua,t}$ = prédiction basée sur MFCM-NMF;
7. $\widehat{ur}_{ua,t}$ = prédiction basée utilisateur;
8. $\widehat{ir}_{ua,t}$ = prédiction basée item;
9. $CFB\_pred(ua,t) = \lambda\, \widehat{nr}_{ua,t} + \delta(1-\lambda)\widehat{ur}_{ua,t} + (1-\delta)(1-\lambda)\widehat{ir}_{ua,t}$.
10. Fin Si
11. Fin Pour
Fin

Après avoir appliqué l'algorithme du filtrage collaboratif (*Algorithme 3.4*), les valeurs des votes inconnus sont prédites, ce qui réduit considérablement la clairsemé de la matrice d'évaluation. Cependant, comme nous l'avons déjà dit, ce type de filtrage se base en grande partie sur le calcul du voisinage, et les préférences des voisins. Donc, si l'ensemble choisi des voisins les plus proches n'est pas candidat ou le modèle élaboré n'est pas performant, les prédictions obtenues ne seraient pas fiables. De plus, les items recommandés ne seront pas pertinents du fait que la prédiction donne des estimations des valeurs de préférence en utilisant juste les évaluations faites par les voisins de l'utilisateur, sans compter la similarité et la faisabilité du contenu des recommandations à l'utilisateur. De plus, parfois nous obtenons un ensemble insuffisant des voisins proches de l'utilisateur actif ce qui provoque la non prédiction de quelques évaluations,

et par conséquent certains items pertinents pour l'utilisateur ne seront pas recommandés.

Ainsi, l'utilisation d'un algorithme du filtrage basé sur le contenu est recommandé afin d'améliorer la performance du FC, pour prédire les évaluations manquantes tout en considérant le contenu des items évalués par l'utilisateur actif. Nous proposons dans la section suivante un algorithme de prédiction basé sur le contenu des items.

3.2.6. L'algorithme du Filtrage-basé Contenu

Pour faire des prédictions basées sur le contenu, un ensemble de caractéristiques est nécessaire pour décrire le contenu des items et de les corréler. Où, quelques caractéristiques sont plus représentatives et contribuent plus à la distinction entre les items que d'autres caractéristiques.

Dans notre système, les items sont des films et seules les caractéristiques les plus représentatives sont considérées pour représenter les items. Ainsi, nous proposons de prendre un sous-ensemble à partir des caractéristiques des films qui est composé de l'ensemble des genres des films, que nous jugeons plus pertinents pour différencier entre les préférences des utilisateurs envers ces films. Les sections suivantes contiennent les étapes détaillées du processus de la prédiction basée sur le contenu.

3.2.6.1. Profil Item (Item Profile)

Dans le présent manuscrit, nous avons utilisé la base de données des films *Movilens*[3]. Dans cette base de données, les informations sur les films (leurs caractéristiques) sont représentées par une liste séparée par des tabulations. Ces caractéristiques sont: *film id, titre de film, date de sortie, date de sortie de la vidéo, IMDb URL, inconnu, Action, Aventure, Animation, Enfants, Comédie, Policier, Documentaire, Drame, Fantastique, Film-Noir, Horreur, Musical, Mystère, Romance, Science-Fiction, Thriller, guerre, Western.*

Où, les 19 dernières caractéristiques correspondantes aux genres des films semblent plus importantes pour la production des recommandations et ce sont les caractéristiques que nous avons jugé pertinentes et que nous avons considéré dans notre système.

[3] www.movilens.com

Par conséquent, nous proposons de sélectionner une sous-matrice contenant les genres seulement et la considérer comme la matrice représentative des films. Donc, le profil d'un film est vu comme un vecteur de taille 19 comme suit:

$$i_m = < genre_1, genre_2, genre_3, \dots, genre_{19} >$$

Où, un film peut appartenir à plusieurs genres (catégories) à la fois et l'appartenance ou non d'un film à un genre (une catégorie) est indiquée de façon binaire, 1 indique qu'un film est de ce genre et 0 indique qu'il n'est pas.

Le processus de la prédiction basée sur le contenu est décrit ci-dessous.

3.2.6.2. Préférence des caractéristiques (Features' preferences)

Pour calculer combien l'utilisateur actif préfère un item donné, un score est attribué à chaque caractéristique pour promouvoir les items en fonction de la fréquence d'apparition de chaque caractéristique dans l'ensemble des items évalués par l'utilisateur actif et leurs évaluations. La raison est que deux caractéristiques (genres) g_1 et g_2 appartenant à deux items $t_1$et t_2, respectivement, peuvent avoir la même fréquence d'apparition dans l'ensemble des items évalués par l'utilisateur u_a, mais l'item t_1 a une meilleure évaluation par rapport à l'item t_2. Par conséquent, la préférence de l'utilisateur doit promouvoir le genre $g_1 \in t_1$sur $g_2 \in t_2$ à travers leurs scores dans le vecteur des préférences. Le score d'une caractéristique (genre) g_n dans le vecteur des préférences de l'utilisateur u_a est calculé comme suit :

$$score\ (g_n, \vec{V}_{ua}) = \frac{\sum_{g_n \in t(ua)} (r_{ua,t} . Occur(g_n))}{|t(ua)|} \quad (3.4)$$

Où, *|t(ua)|* est le nombre d'items évalués par l'utilisateur actif u_a, $r_{ua,t}$ est l'évaluation faite par l'utilisateur u_a pour l'item *t* contenant le genre g_n, et $Occur(g_n)$ représente la fréquence d'apparition de ce genre dans l'ensemble d'items évalués par u_a.

Comme exemple, nous considérons que le genre « Crime » possède une fréquence d'apparition égale à 4, i.e. il y a 4 items évalués par l'utilisateur u_a ayant comme genre « Crime ». Supposons que les évaluations faites par cet utilisateur pour ces items sont (4, 2, 5, 3), et le nombre total des items qu'il a évalué est égal à 20. Donc, le score du genre « Crime » sera calculé comme suit :

$$score\ (g_{crime}, \vec{v}_{ua}) = \frac{\sum_{g_{crime} \in t(u)} (rating(ua,t)\ .\ Occur(g_{crime}))}{|t(ua)|}$$

$$score\ (g_{crime}, \vec{v}_{ua}) = \frac{(4*4)+(2*4)+(5*4)+(3*4)}{20} = 2.8$$

Après avoir calculé les scores de toutes les caractéristiques (genres) dans l'ensemble des items évalués par l'utilisateur actif, nous obtenons une matrice de scores qui contient tous les genres avec leur scores obtenus.

3.2.6.3. Sélection des T-Plus Proches Items TPPI(T-Nearest Items Selection)

Après avoir donné un score pour chaque caractéristique (genre), nous calculons la similarité entre l'item test et l'ensemble des items évalués par l'utilisateur actif pour sélectionner les *T*-plus proches items voisins de cet item. Nous proposons d'utiliser la mesure de similarité Cosinus (présentée dans le chapitre 1) pour calculer la similarité entre deux vecteurs d'items contenant des genres. Comme nous l'avons déjà vue, la similarité cosinus entre deux items est calculée comme suit:

$$t\_sim\ (\vec{v}_i, \vec{v}_j) = \frac{\vec{v}_i . \vec{v}_j}{|\vec{v}_i| . |\vec{v}_j|} \quad (3.5)$$

Cette similarité entre deux items *m* et *t* peut être formulée comme suit :

$$sim(t,m) = \frac{\sum_{n=1}^{nb_{genre}} t(g_n).m(g_n)}{|t(g_n)|^2.|m(g_n)|^2} \quad (3.6)$$

Où $t(g_n)$ et $m(g_n)$ sont des valeurs indiquant l'existence ou non d'un genre g_n dans le vecteurs correspondant aux items *t* et *m*, respectivement.

Cependant, cette formule de similarité n'est pas vraiment représentative parce qu'elle calcule la similarité entre deux items en utilisant juste des valeurs binaires qui représentent l'existence ou non d'un genre dans un item.

Pour améliorer le calcul de cette similarité, nous proposons d'intégrer le score de chaque genre dans le calcul en pondérant chaque vecteur d'item, contenant des valeurs binaires, par les scores des genres.

$$sim(t,m) = \frac{\sum_{n=1}^{nb\_genre} [score(g_n)\ .\ t(g_n)].[score(g_n).m(g_n)]}{|\ t(g_n)|^2.|m(g_n)|^2} \quad (3.7)$$

Où, $score(g_n)$ est le score calculé pour le genre g_n.

Après le calcul de la similarité entre l'item test et les items évalués par l'utilisateur actif, les T-plus proches items (*TPPI*) de l'item test sont sélectionnés comme ensemble d'items voisins candidat.

3.2.6.4. La prédiction basée sur le contenu

La prédiction de l'évaluation inconnue pour un item *t* est calculée en utilisant la similarité de cet item avec ses *T*-plus proches items multipliée par les évaluations faites par l'utilisateur actif pour ces items comme suit:

$$CB\_pred(u_a, t) = \frac{\sum_{m \in TPPI(t)} \boldsymbol{r_{u,m}} \,.\, sim(t,m)}{\sum_{m \in TPPI(t)} sim(t,m)} \qquad (3.8)$$

Afin d'achever de bonnes prédictions, les méthodes basées sur le contenu doivent utiliser un ensemble de caractéristiques les plus représentatives que possible et qui influencent sur la pertinence et la faisabilité des recommandations pour l'utilisateur; chose qui limite l'efficacité de ces méthodes en cas de caractéristiques représentatives insuffisantes. Pour remédier aux limitations des prédictions basées sur le FC et sur le FBC, nous avons effectué une hybridation entre eux.

3.2.6.5. Algorithme du Filtrage basé Contenu

L'algorithme 3.5 résume les étapes de la prédiction basée sur le contenu.

Algorithme 3.5. Filtrage basé Contenu

Entrée : R : matrice d'évaluation, ua : utilisateur actif, M : ensemble des items
T : nombres des plus proches items,
Sortie : CB_pred
Début
1. Pour (chaque genre $g \in 1{:}|G|$) faire;
2. Occur(g)=0;
3. Si (t$\in r_{ua}$) & (g$\in$t) alors
4. Occur(g)= Occur(g)+1;
5. $score$ (g, $\vec{V}_{ua}$) $= \sum_{g_n \in t(ua)} (r_{ua,t} \,.\, Occur(g)) \backslash |t(ua)|$;
6. Fin Si
7. Fin Pour
8. Pour (chaque item $m \in 1$: M) faire;
9. $sim(t,m) = \sum_{n=1}^{nb\_genre} [score(g_n) \,.\, t(g_n)].[score(g_n).m(g_n)] \,/\, |t(g_n)|^2.|m(g_n)|^2$
10. Sélectionner les TNI;
11. Fin Pour
12. $CB\_pred(ua,t) = \left(\sum_{m \in TNI(t)} r_{u,m} \,.\, sim(t,m)\right) / \sum_{m \in TNI(t)} sim(t,m)$.
Fin

3.2.7. La prédiction hybride

Afin d'obtenir plus de précision dans les prédictions, nous proposons un schème de prédiction hybride qui combine linéairement la prédiction basée sur le FC (*CFB_pred*) avec la prédiction basée sur le contenu (*CB_pred*), et cette prédiction hybride est référencée comme (*Hybrid-pred)*. Formellement:

$$Hybrid\_pred(u_a, t) = \alpha \times CFB\_pred(u_a, t) + (1 - \alpha) \times CB\_pred(u_a, t) \quad (3.9)$$

Où, $0 \leq \alpha \leq 1$ est une variable utilisée pour contrôler le poids entre les deux prédictions.

3.2.8. La Recommandation Top-K basée sur la prédiction hybride

Après la phase de la prédiction hybride des évaluations manquantes de la matrice d'évaluation, les items seront classés dans un ordre croissant en fonction de leurs valeurs prédites. Ensuite, une recommandation basée sur la prédiction est générée en sélectionnant les *Top-K* items ayant les plus grandes valeurs de prédiction.

La recommandation basée sur la prédiction génère les *Top-K* items à l'utilisateur ayant les plus grandes valeurs de note prédites, en se basant sur un processus flou pour la classification et la sélection du voisinage de l'utilisateur, ce qui engendre forcément une sorte de diversification dans les items générés. Cependant, la question qui se pose tourne autour de la pertinence ou non des items divers recommandés.

Néanmoins, notre but dès le début était la génération de la recommandation *Top-N* diversifiée en se basant sur les résultats de la prédiction, afin d'améliorer la performance de la recommandation en utilisant des bases pleines d'un côté, et d'augmenter le taux de diversification dans les listes de recommandation, de l'autre, en justifiant d'une manière indirecte la pertinence des items diversifiés à recommander. Ainsi, un processus de diversification des recommandations *Top-N* sera élaboré dans la section suivante.

3.2.9. La Recommandation Top-N Diversifiée

Après avoir appliqué l'algorithme de prédiction hybride cité et expliqué ci-dessus, les évaluations des items non vus déjà par l'utilisateur actif sont obtenues. Ensuite, le processus de recommandation *Top-N* est appliqué. Premièrement, les scores des items

sont calculés au niveau de chaque groupe afin de sélectionner les *Top-N* items. Ensuite, la liste des *Top-N* items dans chaque groupe (cluster) est générée en sélectionnant les *N* items ayant les plus grand scores. Enfin, une liste contenant les *Top-k* items sera sélectionnée à partir de chaque *Top-N* liste pour générer les recommandations aux utilisateurs actifs. Les détails du processus de recommandation sont présentés dans les sous-sections suivantes.

3.2.9.1.Les préférences des clusters

Pour identifier les préférences des groupes, tous les items évalués par les membres de chaque groupe sont explorés et leurs scores de préférence sont calculés au niveau de chaque groupe. Les items préférés sont déterminés par le nombre des plus proches utilisateurs qui ont évalué l'item (popularité de l'item) et la moyenne des évaluations explicites des items données par ces utilisateurs.

Plus précisément, la préférence générale de l'item $t \in T$ par le cluster $c \in CPPCF$ peut être calculée par la formule suivante:

$$C\_pref(c,t) = \beta * moy(c,t) + (1-\beta) * pop(c,t) \quad \textbf{[WEN } \textbf{\textit{et al.}}\textbf{, 08]} \ (3.10)$$

Où, *moy(c,t)* est la moyenne des évaluations explicites faites par les membres du cluster *c* pour l'item *t*, et elle est obtenue à partir de la matrice des centres de clusters *C, et pop(c,t)* est la popularité de l'item *t* dans le cluster *c*, calculée comme le nombre d'utilisateurs appartenant au cluster *c* qui ont évalués l'item *t.* β est une variable utilisée pour ajuster le poids entre la moyenne des évaluations des items et la popularité de l'item.

Formellement, la moyenne des évaluations et la popularité de l'item peuvent être représentées comme suit :

$$moy(c,t) = C(c,t) \quad (3.11)$$

$$pop(c,t) = |c_t| \quad (3.12)$$

Avec, $C(c,t) \in$ C (matrice des centres des clusters) et $|c_t|$ le nombre d'utilisateur du cluster *c* qui ont évalués l'item *t*.

Pour des raisons de flexibilité, nous proposons de donner à β la valeur trouvée par la moyenne des évaluations pour donner la chance aux nouveaux items, parce que si un

item donné a été bien évalué, cela veut dire qu'il mérite d'être favorisé à travers sa moyenne d'évaluation même s'il n'est pas populaire (i.e. la première partie de la formule prendra plus de poids).

3.2.9.2.La recommandation Top-N

Pour faire la recommandation *Top-N*, un *rang (rank*) est attribué à chaque item afin de classer les items selon leurs scores de préférence et leurs similarités avec le vecteur de l'utilisateur, en adaptant la formule proposée par **[WEN *et al.*, 08].**

$$Rank_{ua,t} = \theta . C\_pref(c,t) + (1-\theta) Sim(\vec{V}_{ua}, t) \quad (3.13)$$

Avec, $0 \leq \theta \leq 1$ est un facteur pour ajuster le poids. *Rank* est calculé comme la somme pondérée de la préférence de l'item par les membres du cluster *c* et la similarité entre cet item et le vecteur des items évalués par l'utilisateur actif, qui est calculé avec la formule (3.7) (formule de similarité entre items identifiée dans la partie du filtrage basé contenu).

Nous proposons de fixer θ à la valeur des préférences du cluster, parce que si l'utilisateur actif est un nouvel utilisateur, cela signifie qu'il a peu d'évaluations. Ainsi, des items lui sont recommandés en donnant plus de poids aux préférences du groupe plutôt que la valeur de similarité. Dans l'autre cas, où l'item a un score de préférence faible, i.e. il n'est pas populaire, et possède une très faible moyenne d'évaluations, des items similaires à ses préférences sont recommandés à l'utilisateur.

Après le calcul des scores pour les items, ces derniers sont classés dans un ordre décroissant, et la liste de recommandation à générer dans le cluster *c* est choisie en sélectionnant les *Top-N* items ayant les *N* plus grands scores trouvés.

3.2.9.3.La Recommandation Top-K

Tant que l'utilisateur actif appartient à plusieurs groupes avec des degrés d'appartenance différents. Nous proposons de lui générer une recommandation contenant les *Top-K* items à partir de chaque Top-*N* liste sélectionnée au niveau des clusters les plus proches de cet utilisateur. Les *Top-K* items sont définis en fonction du degré d'appartenance de l'utilisateur au groupe comme suit :

$$K = z_{ua,c} * N \quad (3.14)$$

Où, *N* est le nombre d'items recommandés dans le cluster *c* et $z_{ua,c}$ est le degré d'appartenance de l'utilisateur actif *ua* à ce groupe obtenu à partir de la matrice des degrés d'appartenance *Z.*

Ainsi, la recommandation diversifiée est représentée formellement comme suit :

$$\sum_{uc \in FCNP(u)} TOP - K\ (uc, i) \qquad (3.15)$$

3.2.9.4.Algorithme de la recommandation Top-N diversifiée basée Flou

L'algorithme 3.6 résume les étapes du processus général de la recommandation hybride diversifiée basée flou.

Algorithme 3.6. Recommandation Top-N Diversifiée basée Flou

Entrée: R : matrice d'évaluation, M : ensemble des items (films), ua : utilisateur actif, *K,n,N,k,C,T*: des nombres entiers positifs
Sortie: L_{Top-N}
Début
1. Pour t=1 :T faire,
2. Si $r_{ua,t}$= 0 alors,
3. CFB_pred = *Filtrage Collaboratif basé Flou* (*ua, R, k, K, C*) ;
4. CB_pred = *Filtrage basé Contenu* (*ua, R, T, M*);
5. Hybrid_pred = α**CFB_pred + (1-α) * CB_pred;*
6. Fin Si
7. Fin Pour
8. Pour c = 1:CPPCF faire,
9. Pour t = 1:T faire,
10. Si $r_{ua,t}$= 0 alors,
11. C_pref(c,t) = β*moy(c,t) + (1-β)*pop(c,t) ;
12. Rank(ua,t) = θ*C_pref(c,t) + (1-θ)*Sim($\vec{V}_{ua}$, t);
13. Fin Si
14. Fin pour
15. Liste_ordon(c) = classer↓ (*Rank*) ;
16. Top_N (c) = Liste_ordon(1:N);
17. Top-n(c) = {Top-N(1:n) | n= degré d'appartenance (ua,c)*N};
18. Fin Pour
19. $L_{Top-N} = \sum_{c=1:FCNP} Top - n(c)$;
20. Renvoyer (L_{Top-N});
Fin

L'approche hybride basée flou a prouvé son efficacité pour générer des items divers par rapport à la recommandation traditionnelle, par la diversification des clusters et du voisinage de l'utilisateur, et la génération des recommandations agrégées similaires à leurs goûts et préférences **[MAA et SER, 12a] [MAA et SER, 15].** Cependant, cette approche ne donne pas le vrai sens de la diversité, car elle génère des items aimés par les voisins de l'utilisateur qui sont en réalité semblables à ses choix, ce qui nous a menés

vers une nouvelle méthode de diversité qui combine entre diversité et nouveauté des items.

Afin d'augmenter la diversité dans les listes de recommandation et sélectionner plus d'items nouveaux par rapport à ceux déjà vus par l'utilisateur, soit par évaluation directe ou recommandés par le système, nous devons diversifier de plus en plus de contenus et de recommander des items qui ne sont jamais vus ni par l'utilisateur actif, ni son voisinage. Une nouvelle approche de diversité sera présentée dans la deuxième partie de ce chapitre.

PARTIE 2 :

UN SYSTEME DE RECOMMANDATION HYBRIDE DIVERSIFIEE BASEE SUR LE NIVEAU DE SIMILARITE

3.3. Un Système de Recommandation Hybride Diversifiée basée sur le Niveau de Similarité

Les systèmes de recommandation visent à améliorer la performance à travers la prédiction des préférences des utilisateurs en leurs générant des recommandations avec les Top-N items prédits. Récemment, il a été prouvé que la performance des SRs du point de vue prédiction des items n'est pas suffisante pour achever la satisfaction de l'utilisateur. Pour avoir un bon système de recommandation, il faut qu'il soit capable d'offrir des listes de recommandation pertinentes contenant des items divers pour accomplir tous les besoins et goûts de l'utilisateur.

La diversité des items dans les listes de recommandation est devenue un sujet d'intérêt pour les recherches récentes sur les systèmes de recommandation (comme nous l'avons montré dans le chapitre II). Il a été démontré que l'augmentation de la performance des recommandations, diminue la diversité des résultats, ce qui a créé un grand défit pour les SRs pour répondre à ces deux critères et pour balancer le compromis entre eux.

Dans cette partie, nous présentons une nouvelle approche de recommandation hybride basée sur le niveau de similarité.

3.3.1. Motivations et Objectifs

L'importance de la diversité pour atteindre la satisfaction de l'utilisateur et lui offrir plus de possibilité pour découvrir de nouveaux items ont été les principales raisons que nous ont motivées pour proposer cette nouvelle approche. De plus, la prise en compte de la précision avec la diversité sans avoir recours à un paramètre ou fonction pour l'ajustement du compromis entre les deux constitue une originalité pour notre approche.

Donc, notre objectif principal est l'augmentation de la diversité des items dans les recommandations Top-N, en sélectionnant des items divers pertinents pour garantir l'achèvement de la diversité avec de petite perte en précision. De plus, nous voulons mettre ou proposer une stratégie pour balancer le compromis entre la précision et la diversité pour éviter la perte en eux et sans l'utilisation de paramètre d'ajustement. En d'autres termes, nous voulons proposer une nouvelle méthode de diversification qui vise à avoir des items qui ne ressemblent pas aux choix de l'utilisateur et même aux items

recommandés dans la phase du FC, et qui ne sont pas trop dissimilaires par rapport aux besoins de l'utilisateur, en recommandant les items populaires chez les plus loin voisins de l'utilisateur.

3.3.2. Proposition

En vue d'améliorer la diversité dans les listes de recommandation, on propose de recommander des items qui ne sont pas encore vus ni par l'utilisateur ni par ses plus proches voisins, parce que l'utilisateur sera peut être intéressé avec ces nouveaux items que nous jugeons très dissemblables.

Afin d'achever notre objectif, tout en évitant les inconvénients cités dans le (chapitre II), nous proposons une nouvelle approche de recommandation hybride basée sur une nouvelle notion qui est la notion de niveau de similarité entre les items et le profil de l'utilisateur. Nous nous sommes amenés à utiliser le niveau de similitude entre le profil de l'utilisateur actif et les items dissimilaires en termes d'existence de voisins des voisins, afin de juger la pertinence des items pour les utilisateurs en évitant par conséquent de recommander des items qui ne seront jamais intéressants pour l'utilisateur.

Afin d'atteindre la pertinence des items divers sélectionnés, et sans l'utilisation d'un paramètre d'ajustement du compromis entre précision et diversité, on propose de fusionner une liste d'items dissimilaires avec une liste d'items similaires aux préférences de l'utilisateur trouvée par un algorithme du FC standard.

Donc, notre méthode se situe dans la catégorie des méthodes basées sur la sélection des items, et elle offre de la diversité au contenu des listes similaires, en fusionnant deux listes (similaire et dissimilaire) ce qui rend l'utilisation d'un paramètre d'ajustement n'est pas recommandée. De plus, la pertinence des items dissimilaires recommandés est justifiée par l'utilisation du nouvel algorithme basé sur le niveau de similarité entre les items et le profil de l'utilisateur.

Afin d'augmenter la diversité en assurant la pertinence des items nous voyons que la sélection de la catégorie dissimilaire la plus pertinente pour l'utilisateur et que choisir les items dissimilaires pertinents de cette catégorie sera plus intéressant pour lui alors que lui générer des items différents appartenant à différentes catégories, parce que

l'utilisateur ne sera pas sûrement intéressé par tous ces items très dissimilaires appartenant à différentes catégories dissimilaires.

C'est pourquoi nous proposons de générer à l'utilisateur les items dissimilaires pertinents appartenant à la catégorie dissimilaire la plus pertinente. Ensuite, si l'utilisateur aime les items recommandés c.à.d. il donne de bonnes évaluations pour les items dissimilaires recommandés, le processus se répète et il recevra par la suite les recommandations à partir d'autres catégories dissimilaires qui contiennent le reste des items dissimilaires non vus par l'utilisateur.

Nous introduisons dans la sous section suivante la notion du niveau de similarité proposée.

3.3.3. Le processus du Niveau de Similarité

La notion du niveau de similarité proposée dans ce manuscrit signifie la distance entre l'utilisateur actif et un item dissimilaire donné en termes d'existence ou non de voisins des voisins qui ont évalué l'item. Etant donné un item *m* non évalué par l'utilisateur actif, ni par ses voisins les plus proches, le niveau de similarité entre cet utilisateur et l'item *m* est envisagé comme suit:

- S'il existe au moins un utilisateur *v* qui partage un seul item avec l'utilisateur actif, que l'on appelle plus loin voisin, et si le même utilisateur a évalué l'item *m*, alors le niveau de similarité de cet item est égal à 1, parce qu'il existe une relation directe entre l'utilisateur actif et l'item qui est le voisin le plus éloigné. La motivation est que ce voisin est sûrement l'un des voisins des plus proches voisins de l'utilisateur actif, ce qui revient à l'hypothèse que si l'utilisateur reçoit un item déjà évalué par un voisin de ces plus proches voisins sera peut être content et satisfait de l'item malgré qu'il est différent de ses préférences et plus probant que de lui recommander des items qui n'ont aucune relation avec les préférences de l'utilisateur.
- Si aucun voisin n'a évalué cet item, donc nous passons aux voisins des voisins. S'il y a au moins un utilisateur parmi les voisins des voisins de l'utilisateur qui a évalué cet item, alors nous lui donnons un niveau de similitude égal à 2, et nous continuons avec cette méthode pour rechercher le niveau de similarité de l'item par rapport à l'utilisateur actif.

Notons qu'un item avec un niveau de similarité égal à 1 est très intéressant à l'utilisateur actif qu'un item avec un niveau de similitude égale à 0 (dans le cas où l'item n'était pas évalué par aucune personne) et il est prioritaire qu'un item avec les niveaux de similarité égaux à 2 ou 3. Pour plus de compréhension, la *Figure 3.3* illustre la notion de niveau de similarité.

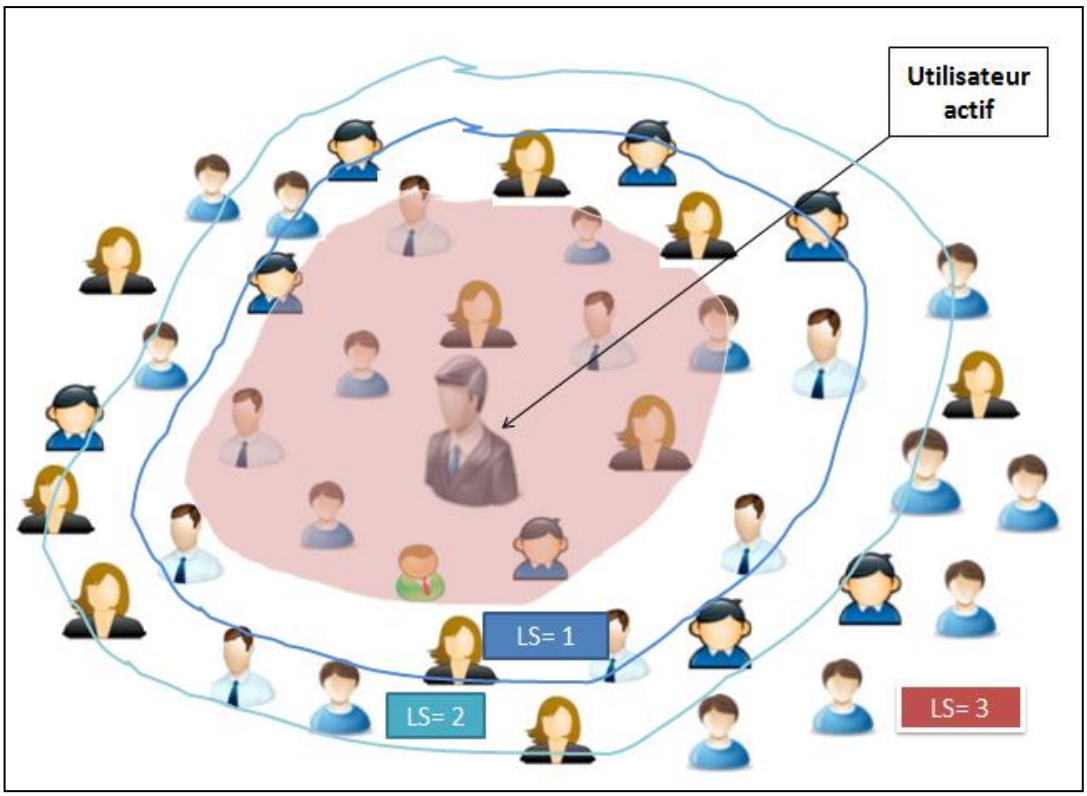


Figure 3.3. Illustration de la notion du Niveau de Similarité

Les détails de la nouvelle approche de recommandation sont présentés dans la section suivante.

3.3.4. Recommandation Top-K diversifiée basée sur le niveau de similarité

Nous proposons dans cette deuxième partie, une nouvelle approche de recommandation hybride qui combine entre un processus de génération de recommandation similaire et un processus de recommandation d'items divers. Le premier processus se base sur un algorithme de filtrage collaboratif ordinaire pour trouver les items similaires aux préférences de l'utilisateur actif pour assurer la précision des recommandations. Alors que le deuxième processus utilise une nouvelle méthode de diversification ayant comme but de sélectionner des items dissimilaires mais pertinents pour l'utilisateur en se basant

sur une nouvelle formule pour calculer la pertinence des items. Afin d'achever les objectifs attendus, avoir une liste de recommandation avec des items divers et pertinents en même temps, nous proposons de fusionner les deux listes générées par les deux méthodes.

Les étapes du processus de la recommandation hybride diversifiée basée sur le niveau de similarité sont schématisées dans *la Figure 3.4*.

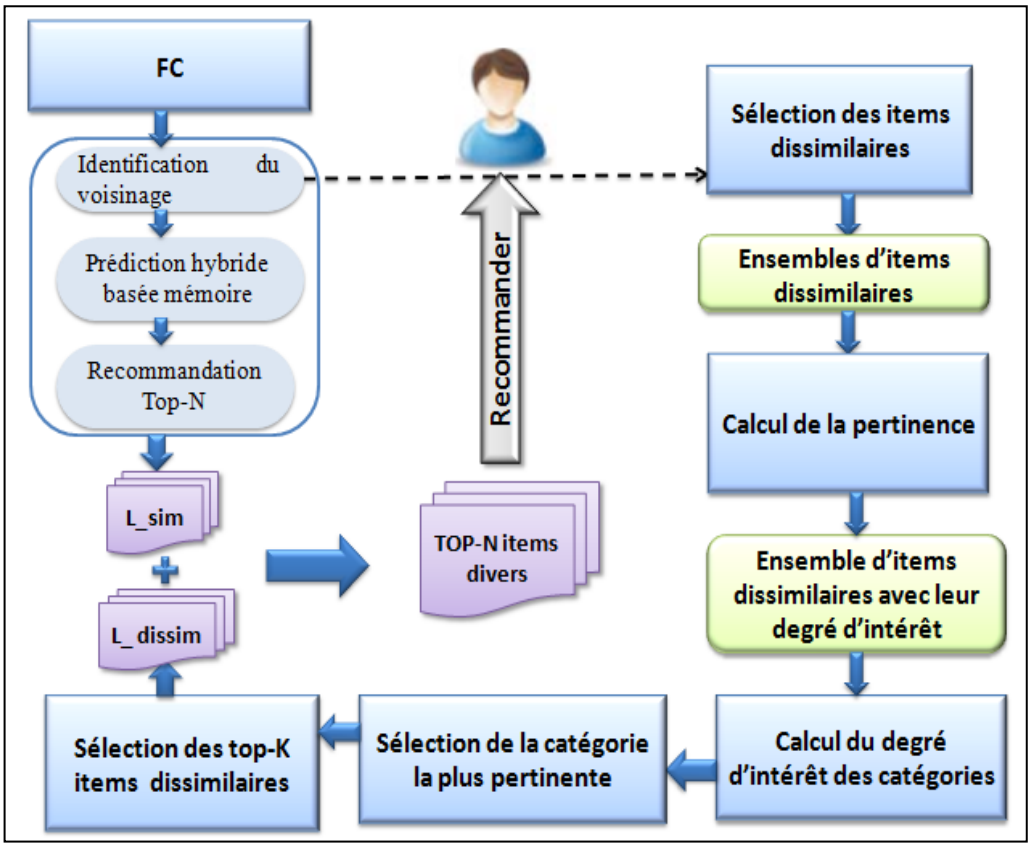


Figure 3.4. Processus de la Recommandation Diversifiée basée sur le Niveau de Similarité

Les différentes étapes de la nouvelle approche de recommandation basée sur le niveau de similarité seront détaillées dans les sous-sections suivantes.

3.3.3.1.Génération de la recommandation similaire

Comme nous l'avons déjà indiqué ci-dessus, pour la génération de recommandation avec des items similaires nous avons proposé d'utiliser un algorithme du FC ordinaire, qui permet de recommander à l'utilisateur les items préférés par ses plus proches voisins.

A cette fin, la première étape à faire est l'identification du voisinage de l'utilisateur, en calculant la similarité entre l'utilisateur actif et les autres utilisateurs, et la sélection des plus proches voisins comme ensemble candidat.

Pour le calcul de la similarité entre utilisateurs, nous avons utilisé la corrélation de Pearson:

$$sim(ua, v) = \frac{\sum_{i=1}^{I_u \cap I_v}(r_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i=1}^{I_u \cap I_v}(r_{u,i} - \bar{r}_u)^2 (r_{v,i} - \bar{r}_v)^2}} \quad (3.16)$$

Ensuite, nous calculons les prédictions pour les items non évalués en utilisant une combinaison entre les prédictions basées sur le voisinage (basée utilisateur et basée item) en utilisant les formules de prédiction utilisées dans la première partie et citées dans le chapitre 1 (Formule (1.30) pour la prédiction basée utilisateur et la formule (1.31) pour la prédiction basée item). Les deux prédictions sont fusionnées par un paramètre λ comme suit :

$$CFB_{pred(ua,t)} = \lambda . \widehat{ur}_{ua,t} + (1 - \lambda)\widehat{ir}_{ua,t} \quad (3.17)$$

Après la génération de la recommandation basée sur le filtrage collaboratif, en sélectionnant les Top-N items ayant les plus grandes valeurs de prédiction, nous obtenons donc un ensemble des items les plus similaires à ceux évalués par l'utilisateur actif, et préférés par ses voisins.

Maintenant, la diversité sera le cœur de la nouvelle méthode proposée visant la diversification du contenu, que nous allons la présenter en détails ci-dessous.

3.3.3.2.Génération de la Recommandation Diversifiée basée sur le Niveau de Similarité

Afin d'augmenter la diversité dans les listes de recommandation, nous proposons de générer des items différents de ceux préférés par l'utilisateur et qu'il n'a pas encore vus, mais en offrant une notion de garantie vis-à-vis de la pertinence des items divers pour l'utilisateur. Donc, la deuxième phase de l'approche proposée dans cette deuxième partie est dédiée à la nouvelle approche de la diversité basée sur le niveau de similarité.

3.3.3.2.1. Catégorisation des items

Comme nous l'avons déjà vu dans la partie concernant la présentation du contenu des items (sous section 3.2.6.1), les items dans l'ensemble de données possèdent plusieurs caractéristiques et selon ces caractéristiques, un item peut appartenir à une ou plusieurs catégories en même temps.

Comme le *Dataset* que nous avons utilisé ne contient pas une classification des différents items dans des catégories, ce qui est le cas pour la majorité des ensembles de données disponibles, nous proposons tout d'abord de construire une table qui contient toutes les catégories (qui représentent les différents genres des films dans notre cas) et d'affecter les items aux catégories appropriées.

Cette étape sert à connaître les items appartenant à chaque catégorie, afin de savoir par la suite le nombre des items évalués par l'utilisateur dans chaque catégorie. Ce nombre sera utilisé dans la formule de calcul du degré d'intérêt des items présentée en plus bas.

3.3.3.2.2. Sélection des items non vus

L'idée de notre approche de diversification repose en grande partie sur la notion de nouveauté dans les items. Fournir des items divers n'implique pas qu'ils sont nouveaux pour l'utilisateur, mais le contraire est vrai. Recommander un item jamais vu par l'utilisateur, qu'il est dissimilaire à ses préférences implique forcément que l'item est différent par rapport à ce que l'utilisateur a l'habitude de recevoir. De ce fait, nous proposons de sélectionner l'ensemble des items non évalués ni par l'utilisateur ni par son voisinage pour garantir au maximum d'avoir un ensemble d'items dissimilaires.

Premièrement, l'ensemble des items évalués par les plus proches voisins de l'utilisateur actif est trouvé, ce qu'on a appelé I_{NN}, où :

$$I_{NN} = \{\forall i \in T, \forall\, v \in KPPV, r_{v,i} \neq 0\} \tag{3.18}$$

Deuxièmement, les items évalués par l'utilisateur actif sont regroupés dans un ensemble nommé I_{ua}, avec :

$$I_{ua} = \{\forall i \in T, r_{ua,i} \neq 0\} \tag{3.19}$$

Ensuite, l'ensemble des items non encore évalués est extrait de l'ensemble total des items, en éliminant les items évalués par l'utilisateur actif et par ses voisins les plus

proches, afin d'avoir de nouveaux items, qui accomplissent la caractéristique de dissimilarité autant que possible avec les choix de l'utilisateur. Cet ensemble est représenté comme suit:

$$I_{\psi} = I|(I_{ua} \cup I_{NN}) \quad (3.20)$$

Où, I_{ψ} est l'ensemble des items divers.

Recommander des items divers à l'utilisateur peut ne pas être satisfaisant pour lui, quand les items suggérés ne sont pas pertinents. Donc, nous proposons de calculer le degré d'intérêt ou de pertinence des items divers à recommander pour l'utilisateur pour ne sélectionner que ceux jugés pertinents.

3.3.3.2.3. Calcul de la pertinence des items divers

Recommander des items divers et très dissimilaires aux préférences de l'utilisateur, peut ne pas être utile pour l'utilisateur qui reçoit des items qui ne lui intéresse pas.

Ainsi, à partir de l'ensemble des items divers, les items jugés intéressants appartenant à la catégorie intéressante pour l'utilisateur actif seront choisis pour la recommandation, afin d'augmenter la diversité des contenus délivrés. Donc, cette étape est consacrée à la recherche des items pertinents à partir de la catégorie la plus pertinente. C'est pour quoi, nous proposons de calculer un degré d'intérêts pour les items divers afin de juger leurs pertinences pour l'utilisateur.

La formule de calcul du degré d'intérêt proposée se base sur le niveau de similarité entre l'utilisateur et les items en termes d'existence de voisins qui ont évalué l'item ou non. La raison est que l'utilisateur sera peut être intéressé par des items qui ont été évalués par les voisins de ses voisins même s'ils sont différents de ses préférences, que des items non aimés ni par ses voisins ni leurs voisins.

Nous avons mis quelques prémisses (principes) à partir desquelles nous avons proposé la formule de calcul du degré d'intérêt. Ces prémisses sont les suivantes:

- A chaque fois que le nombre des plus loin voisins qui ont évalué l'item augmente (l'item devient plus populaire) → le degré d'intérêt augmente.

Justification : Recommander un item populaire chez les voisins des voisins est intéressant pour l'utilisateur.

- A chaque fois le niveau de similarité augmente (prend une plus grande valeur) → le degré d'intérêt diminue.

Justification : à chaque éloignement des voisins des voisins de l'utilisateur, l'utilisateur devient non intéressé par les items qu'ils aiment parce qu'ils deviennent trop dissimilaires.

- A chaque fois que le nombre des items évalués par l'utilisateur dans la catégorie de l'item divers *a* augmente → le degré d'intérêt diminue.

Justification : pour augmenter la diversité, on ne veut pas recommander les items similaires aux préférences de l'utilisateur.

Par conséquent, pour sélectionner les items dissimilaires intéressants pour l'utilisateur actif, le degré d'intérêt affecté à chaque item *a* de l'ensemble divers I_{ψ}, est calculé comme suit:

$$interest\_deg(a) = \frac{|U_a^{X_i^{ua}}|}{\omega * |R_{ua}^{c(a)}|} \quad (3.21)$$

Où, $|U_a^{X_i^{ua}}|$ est le nombre d'utilisateurs qui ont évalué l'item *a* et partagent au moins un item avec l'utilisateur actif *ua*, avec X_i^{ua} est l'ensemble des utilisateurs ayant au moins un item en commun avec l'utilisateur actif, représenté comme suit:

$$X_i^{ua} = \{y \in U \mid r_{y,a} \neq 0, \quad \text{Où } \exists\, j \neq a \in I \ \text{ s.t } r_{y,j} \neq 0 \text{ et } r_{ua,j} \neq 0\} \quad (3.22)$$

ω est un poids qui exprime le niveau de similarité entre les items et l'utilisateur actif, et $|R_{ua}^{c(a)}|$ est le nombre d'items évalués par l'utilisateur actif *ua* dans la catégorie *c* de l'item *a*. Pour éviter les cas où la valeur de $|R_{ua}^{c(a)}|$ est égale à 0 (qui donne une division par 0), nous avons initialisé le nombre des items évalués dans chaque catégorie par 1.

La formule du degré d'intérêt (*interest_deg*) vise à calculer l'intérêt ou la pertinence d'un item dissemblable en fonction du nombre d'utilisateurs (voisins des voisins), avec au moins un item en commun avec l'utilisateur actif, qui ont évalué cet item, le nombre des items évalués par l'utilisateur actif qui appartiennent à la catégorie de l'item dissimilaire, et le niveau de similarité entre cet item et le profil de l'utilisateur en termes de niveau de similarité entre les voisins des voisins de l'utilisateur ayant évaluer l'item.

Finalement, nous proposons de sélectionner la catégorie la plus intéressante pour l'utilisateur et recommander les items pertinents qu'elle contient. La pertinence d'une catégorie d'items est jugée sur la base d'un poids que nous allons calculer comme le

degré d'intérêt moyen des degrés des items qu'elle contient. Où, ce poids représente combien une catégorie est importante (pertinente) à l'utilisateur sachant que l'utilisateur n'a pas évalué aucun item ou un nombre très restreint d'items de cette catégorie, et les items de cette catégorie sont dissimilaires en contenu par rapport à ceux évalués par l'utilisateur. Donc, le degré d'intérêt pour chaque catégorie *c* est calculé comme suit :

$$interest\_cat(c) = \frac{\sum_{i \in c} E_v(i)}{|c|} \quad (3.23)$$

Ensuite, la catégorie possédant le plus grand degré d'intérêt ($interest\_cat$) est choisie, et les items intéressants ayant les plus grand degrés d'intérêt et appartenant à cette catégorie seront sélectionnés pour être recommandés.

3.3.5. Algorithme de la Recommandation Top-K diversifiée basée sur le Niveau de Similarité

Les étapes de l'approche proposée sont résumées dans l'algorithme 3.7.

Algorithme 3.7. Recommandation Top-N Diversifiée basée sur le Niveau de Similarité

Entrée: *ua:* l'utilisateur actif, *K* et *N*: deux entiers positifs, U : l'ensemble des utilisateurs, I : ensemble des items.
Sortie: L_{TOP-N}: liste de recommandation
Début
1: L_{sim} = *Filtrage Collaboratif (I, U, K, N, ua)*;
2: I_{NN}= {∀u∈ KPPV et ∀i∈I| $r_{u,i}$≠0 };
3: I_{ua} = {∀ i∈I| $r_{ua,i}$≠0 };
4: I_{ψ} = I \ ($I_{NN} \cup I_{ua}$);
5: L = ϕ;
6: Pour (chaque item a∈I_{ψ}) faire
7: X_i^{ua}= {y∈U | $r_{y,a} \neq 0$, Où ∃ j ≠ a ∈ I s.t $r_{y,j} \neq 0$ et $r_{ua,j} \neq 0$};
8: E_v = *interest_deg*(a) = $|X_i^{ua}|/(\omega * R_{ua}^{c(a}$ 161
9: L = L ∪ (a, E_v);
10: Fin pour
11: Pour (chaque catégorie c∈C) faire
11: interest= interest_cat(c)= $\sum_{i \in c} E_v(i)/|c|$;
12: Fin Pour;
13 : pertin_cat = {c | interest_cat(c) = max (interest_cat)} ;
13: L_{diss}= sort(L)↓ selon E_v;
14 : L_{div} = {i∈ L_{diss} et i∈pertin_cat}
14: L_{TOP-N} = {L_{sim}[1…|L_{sim}|-| L_{div}|] + L_{div} s.t |L_{sim} + L_{div}|=N};
15: renvoyer (L_{TOP-N});
Fin

La méthode basée sur le niveau de similarité produit des recommandations diversifiées, mais la diversification est proportionnelle au fait qu'elle combine entre deux listes, l'une très similaire et l'autre dissimilaire, ce qui réduit sûrement la précision par rapport à la recommandation similaire.

Afin d'augmenter encore plus la notion de la diversité dans les listes de recommandation sans perdre considérablement la précision dans les listes de recommandation, et pour éviter d'utiliser un paramètre d'ajustement préfixé par l'utilisateur ou toute autre méthode pour l'initialiser, une nouvelle approche hybride est proposée. L'approche combine entre le filtrage collaboratif basé sur le flou et la méthode basée sur le niveau de similarité, et elle est présentée ci-dessous.

3.4. Une Nouvelle Approche Hybride pour la Recommandation Top-K (Une Nouvelle Approche de Recommandation Hybride Diversifiée basée Flou et Niveau de Similarité)

Afin d'équilibrer le compromis entre la précision et la diversité, et différemment des travaux de la littérature qui utilisent un paramètre pour contrôler le changement dans la précision et la diversité, nous proposons d'hybrider les deux approches proposées dans cette thèse, FC basé sur le flou et la méthode de diversification basée sur le niveau de similarité, comme troisième approche proposée.

La nouvelle approche utilise l'algorithme du filtrage collaboratif basé flou, présenté dans la première partie de ce chapitre, pour générer des recommandations dans les plus proches clusters flous tout en considérant l'ensemble des plus proches voisins flous comme une première phase de recommandation. Puis, elle utilise la méthode de diversification basée sur le niveau de similarité, présentée dans la deuxième partie du chapitre, pour générer des items divers et nouveaux pour l'utilisateur.

Ensuite, les deux listes générées par les deux processus seront fusionnées pour constituer la liste de recommandation *Top-K* finale, afin d'améliorer la diversité des choix dans les listes de recommandation. Le processus de la nouvelle approche hybride est synthétisé dans la *Figure 3.5*.

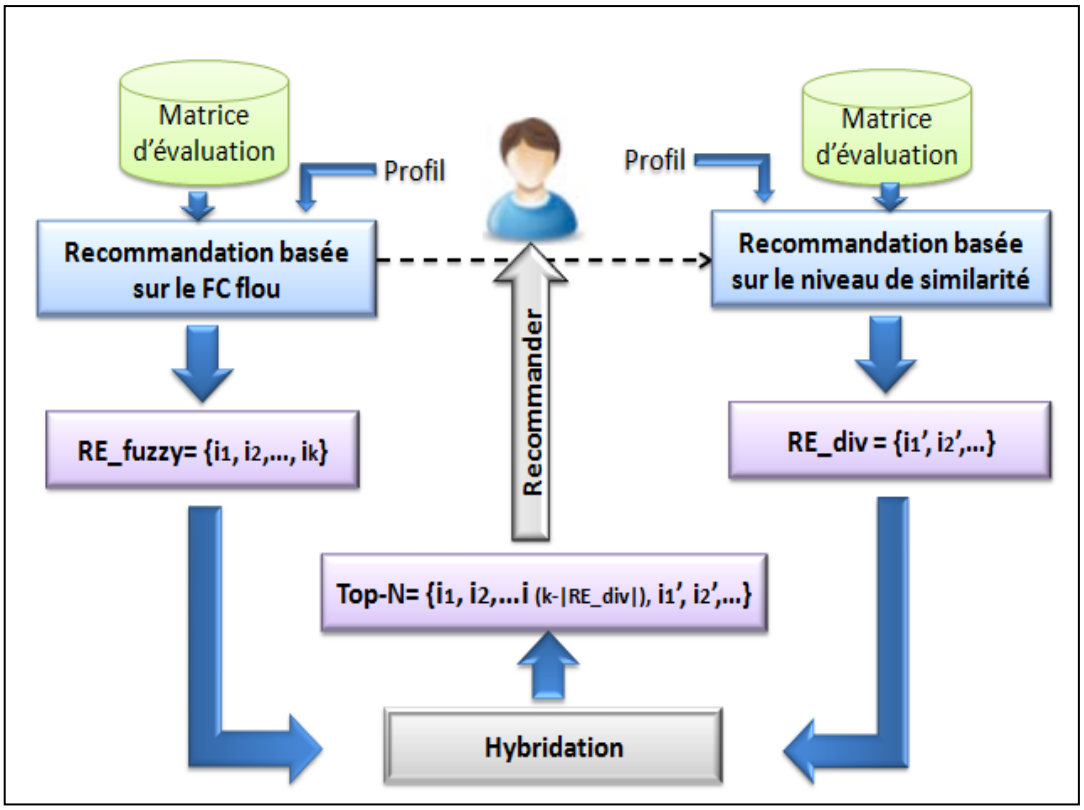


Figure 3.5. Nouvelle Approche de Recommandation Hybride Diversifiée basée Flou et Niveau de Similarité

Le processus de recommandation présenté dans la *Figure 3.5* explique comment les deux approches utilisées sont combinées, où l'approche basée sur le FC flou s'effectue en premier lieu, afin de regrouper les utilisateurs et sélectionner l'ensemble du voisinage candidat de l'utilisateur actif. Ensuite, l'approche basée sur le niveau de similarité est effectuée, et une liste de recommandation est générée avec chaque approche.

L'algorithme du FC basé sur le flou génère une liste de recommandation *Fuzzy-RE* qui contient les items préférés par les plus proches voisins flous appartenant aux plus proches clusters flous. Tandis que la méthode de diversification basée sur le niveau de similarité, génère une liste de recommandation *RE_div* contenant les items divers non vus déjà par l'utilisateur ni son voisinage.

La liste de recommandation finale générée est une concaténation des deux listes avec la contrainte qu'on sélectionne les premiers items de la première liste, où le nombre d'items à sélectionner est égal à (*K*-|*RE_div*|). La liste de recommandation finale se représente comme suit :

L_{TOP-K} = {*Fuzzy_RE* (1…| *Fuzzy_RE* |-| *RE_div* |) + *RE_div* | s.t | *Fuzzy_RE* + *RE_div* |=N} (2.24)

3.4.1. Algorithme de Recommandation Top-K diversifiée basée Flou et Niveau de Similarité

L'algorithme 3.8 résume les étapes de la nouvelle approche hybride de diversification.

Algorithme 3.8. Recommandation Top-N Diversifiée basée Flou et Niveau de Similarité

Entrée: *ua:* l'utilisateur actif, *K, k, C* et *N*: des entiers positifs, U : ensemble des utilisateurs, I : ensemble des items, R : matrice d'évaluation.
Sortie: L_{TOP-N}: liste des items divers
Début
1: L_{divF}= *Filtrage Collaboratif basé Flou* (*ua, k, C, K, N*) ;
2: I_{NN}= {∀u∈ KPPV et ∀i∈I| $r_{u,i}$≠0 };
3: I_{ua} = {∀ i∈I| $r_{ua,i}$≠0 };
4: I_{ψ} = I \ (I_{NN}∪I_{ua});
5: L = ϕ;
6: Pour (chaque item a∈I_{ψ}) faire
7: X_i^{ua}= {y∈U | $r_{y,a}$ ≠ 0, Où ∃ j ≠ a ∈ I s.t $r_{y,j}$ ≠ 0 et $r_{ua,j}$ ≠ 0};
8: E_v = *interest-deg*(a) = $|X_i^{ua}|/(\omega * R_{ua}^{c(a)})$;
9: L = L ∪ (a, E_v);
10: Fin pour
11: Pour (chaque catégorie c∈C) faire
11: *interest-cat*(c)= $\sum_{i\in c} E_v(i)/|c|$;
12: Fin Pour;
13 *pertin_cat* = {c | *interest_cat*(c) = max (*interest_cat*)} ;
14: L_{diss}= sort(L)↓ selon E_v;
15 : L_{div} = {i∈ L_{diss} et i∈*pertin_cat*}
16: L_{TOP-N} = {L_{divF}(1…|L_{divF}|-| L_{div}|) + L_{div} | s.t |L_{divF} + L_{div}|=N};
17: renvoyer (L_{TOP-N});
Fin

3.5. Conclusion

Deux contributions principales ont été présentées dans ce chapitre visant à éliminer le problème de la stabilité des systèmes de recommandation et du manque de diversité dans le contenu des recommandations générées aux utilisateurs.

Les deux méthodes sont des méthodes hybrides, dont la première vise à rendre le système plus adaptatif aux préférences des utilisateurs en utilisant un algorithme du FC basé sur le flou combiné avec un algorithme du FBC, et la deuxième méthode vise à augmenter la diversité dans les listes de recommandation en générant une liste des items

dissimilaires pertinents, en se basant sur le niveau de similarité entre les items et les utilisateurs, combinée avec une liste d'items similaires trouvée par un algorithme du FC simple.

Une troisième proposition a été également présentée en combinant entre l'algorithme du FC basé sur le flou et l'algorithme de diversification basé sur le niveau de similarité, pour augmenter la diversité dans les listes de recommandation tout en conservant leurs précisions.

La validation des différentes propositions seront présentées dans le chapitre suivant.

CHAPITRE 4

EXPERIMENTATIONS

RESUME DU CONTENU

Dans ce chapitre, une série d'expériences est présentée afin de valider les différentes approches et notions proposées au niveau du chapitre précédent pour prouver leurs efficacités dans un système de recommandation des films, puis dans un système de recommandation dans le domaine du *e-learning*.

4.1. Introduction

Les différentes approches proposées au niveau du chapitre III sont établies et testées sur des ensembles de données des films afin de prouver leur efficacité en recommandant des listes de recommandation pertinentes et diversifiées. Ensuite, les trois approches seront intégrées dans un système d'apprentissage en ligne afin de tester comment elles agissent dans un scénario concret.

4.2. Objectifs des expérimentations

Nous menons une série d'expériences pour examiner l'efficacité et la faisabilité des différentes propositions présentées dans le chapitre précédent, en commençant par la validation de notre approche de recommandation hybride basée sur le flou. Tout d'abord, le processus du filtrage collaboratif basé flou sera testé en termes d'évolutivité et d'efficacité, puis, la performance des prédictions et la qualité des recommandations seront évaluées. Ensuite, nous testons l'efficacité de la nouvelle méthode de diversification basée sur le niveau de similarité, et nous terminerons par la validation de l'approche hybride. En particulier, nous abordons les questions suivantes:

1. Est-ce que la mesure de similarité proposée est efficace ou non? les expériences sont menées pour examiner l'efficacité de la mesure de similarité en termes de temps de calcul en comparant nos résultats avec la mesure de similarité du CCP.
2. Est-ce-que la prédiction hybride augmente la performance du système ou non ? en comparant sa performance avec celles de la prédiction basée sur le FC flou et le FBC.
3. Est-ce-que l'utilisation d'une phase de prédiction avant la recommandation augmente la précision des recommandations ou non ? en testant les résultats avec le processus de recommandation ordinaire.
4. Est-ce-que le processus de recommandation diversifiée basé flou est efficace et augmente la précision et la diversité des listes de recommandation ou non ? en testant sa performance avec la recommandation ordinaire.
5. Est-ce-que la nouvelle méthode basée sur le niveau de similarité offre plus de diversité ou non ? en testant sa performance avec des méthodes de diversification de la littérature et la méthode basée flou.
6. Est-ce-que la nouvelle approche hybride améliore les performances ou non ? en comparant les résultats avec les deux premières approches proposées.

4.3. Dataset

Nous avons utilisé dans nos expérimentations la base de données MovieLens[4] 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.

Nous avons utilisé trois ensembles de données qui contiennent 100, 200 et 300 utilisateurs avec plus de 20 évaluations et 1000 items comme ensembles d'entrainement, qui sont dénotés comme *ML_100*, *ML_200* et *ML_300*, respectivement, et un autre ensemble qui contient 200 utilisateurs comme ensemble de test.

En outre, nous avons choisit au hasard 5, 10 et 20 items évalués par l'utilisateur actif que nous avons les arrangés dans trois différents ensembles de données qu'on a appelés *Given-5*, *Given-10*, et *Given-20*, respectivement, et que nous utiliserons pour le test de performance de la prédiction.

Les expériences sont effectuées sur un PC avec un 2,40 GHz Intel Core 2 Duo, CPU E4600 et 1 Go de mémoire.

4.4. Métriques d'évaluation

Nous présentons dans cette partie les différentes mesures d'évaluation utilisées dans nos expérimentations, pour tester la performance de chaque partie des propositions.

4.4.1. Précision de l'algorithme de factorisation

Nous avons choisi d'utiliser la mesure de précision de Frobenius utilisée dans **[HOY 04]** et **[LAN et al.06]** comme métrique d'évaluation, comme suit :

$$||\mathrm{V} - \mathrm{ZC}||_2 = \frac{1}{2}(\mathrm{V} - \mathrm{Z} * \mathrm{C})^2 \qquad (4.1)$$

Plus la valeur de précision est inférieure, plus la performance est meilleure.

[4] http://www. grouplens. org

4.4.2. Métrique d'évaluation de la prédiction basée modèle (RMSE)

Pour évaluer la performance de la prédiction fournie par le modèle de factorisation floue, nous avons utilisé l'Erreur Quadratique Moyenne RMSE, calculée comme suit :

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{j,m}(r_{j,m} - \tilde{r}_{j,m})^2}{N}} \tag{4.2}$$

Où, $\hat{r}_{j,m}$ est la valeur estimée par le modèle et N le nombre total des évaluations utilisées dans le test.

4.4.3. Métrique d'évaluation de la prédiction (MAE)

Nous allons utiliser la métrique MAE pour évaluer la performance des prédictions

$$\mathrm{MAE} = \frac{\sum_{j,m} |r_{j,m} - \tilde{r}_{j,m}|}{|N|} \tag{4.3}$$

Où N est le nombre des évaluations utilisées dans le test. Plus MAE est inférieure, plus la performance est meilleure.

4.4.4. Métriques d'évaluation des recommandations (Top-N)

4.4.4.1. Les mesures de la précision

Afin d'évaluer la performance de notre système de recommandation, nous avons utilisé les métriques du Rappel, Précision et la métrique F_1. Où, F1 est calculée comme suit :

$$F_1 = \frac{2PR}{P+R} \tag{4.4}$$

Où *P* et *R* sont la *Précision* et le *Rappel* respectivement, et ils sont calculés comme suit :

$$P = \frac{N_t}{N} \tag{4.5}$$

$$R = \frac{N_t}{N_p} \tag{4.6}$$

- N_p : Nombre total des items pertinents.

- N_t : Nombre des items pertinents trouvés.
- N : Nombre total des items.

4.4.4.2. La mesure de la diversité

Pour évaluer la performance du système du côté de la diversité des listes de recommandation, la métrique de diversité individuelle proposée par Ziegler **[ZIE *et al.*, 05]** sera utilisée**.**

$$ID_u = \frac{1}{N(N-1)} \sum_{i \in N} \sum_{j \in N, i \neq j} d(i,j) \quad (4.7)$$

Avec $$d(i, j) = 1\text{-}sim(i, j) \quad (4.8)$$

Où *sim* est la similarité entre deux items i et j.

La partie expérimentation sera divisée en deux grandes parties, où la première partie, appelée Evaluation avec Dataset, sera consacrée à l'évaluation des différentes approches sur l'ensemble de données des films *MovieLens,* et la deuxième partie sera dévouée à l'application des différentes approches proposées dans un scénario *e-learning*.

PARTIE I : DATASET

4.5. Expérimentation 1 : Evaluation de la Performance de l'Approche de Recommandation Hybride basée Flou

Nous évaluons dans cette section la performance de la première approche proposée : la recommandation hybride basée sur le flou, en testant la performance de chaque partie de l'approche (FC basé flou, Filtrage Basé Contenu, la Prédiction hybride et la recommandation Top-N diversifiée basée flou).

4.5.1. Evaluation de l'algorithme du Filtrage Collaboratif basé Flou

Dans cette partie, les trois grandes étapes principales pour effectuer le FC basé flou seront testées chacune à part. Ensuite, la performance générale de l'algorithme du FC sera évaluée en utilisant l'erreur absolue moyenne.

4.5.1.1.Performance de l'algorithme du clustering MFCM-NMF

L'algorithme de factorisation MFCM-NMF nécessite un choix des valeurs et paramètres utilisés (K, λ_c *et* λ_z) afin d'avoir une bonne convergence de l'algorithme.

Dans cette partie nous présentons les différentes expérimentations effectuées afin de trouver les valeurs optimales des paramètres. En plus, nous menons des expériences pour comparer la performance de notre algorithme de factorisation avec quelques algorithmes cités dans la littérature en utilisant l'erreur RMSE, et pour tester la sensibilité de l'algorithme au problème de l'évolutivité de données. Notons que l'effet de la clairsemé de données sera testé en plus bas dans la partie d'évaluation de la recommandation.

4.5.1.1.1. Choix du nombre *k* des clusters

Afin de sélectionner un nombre k approprié de clusters, nous avons utilisé l'ensemble de données d'entrainement *ML_200*. Nous avons appliqué l'algorithme proposé MFCM-NMF en utilisant six valeurs différentes de k (10, 30, 50, 100, 150 et 200), et nous avons choisit la valeur pour laquelle la fonction objective prend la petite valeur de converge.

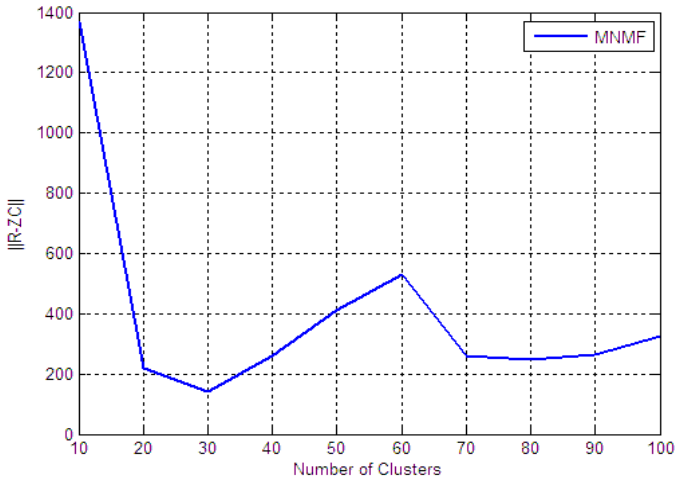


Figure 4.1. Valeur optimale des clusters k

Comme le montre la Figure 4.1, la meilleure valeur pour laquelle la courbe de la fonction atteint la plus petite valeur est k=30 clusters.

Donc, nous allons fixer la valeur de k à 30 clusters tout au long du reste des expérimentations.

4.5.1.1.2.Choix des valeurs optimales des paramètres de régularisation λ_c et λ_z

Comme le montre l'équation 3.1 (chapitre 3), λ_c *et* λ_z sont des petits paramètres de régularisation qui équilibrent le compromis entre l'erreur d'approximation et les contraintes. Avec $0 < \lambda_c, \lambda_z < 1$.

Nous avons effectué deux expériences sur l'ensemble d'entrainement *ML_300* avec *k*=30 pour identifier les valeurs optimales des paramètres de régularisation :

- Tout d'abord, nous avons fixé $\lambda_c = 0$ et nous avons testé les propriétés de λ_z.
- Ensuite, le paramètre λ_z est fixé à la valeur optimale trouvée et nous avons continué à tester les propriétés de λ_c.

La Figure 4.2 présente les deux graphes de la fonction objective avec différentes valeurs de λ_z (en haut) et λ_c (en bas) appartenant à l'intervalle [0 - 0,1].

A partir des résultats obtenus, nous avons choisi les deux valeurs, λ_z=0.02 et λ_c=0.08, parce que ces valeurs sont les valeurs pour lesquelles la fonction a aboutit aux petites valeurs de convergence malgré que sur les sous figures elles ne sont pas vraiment claires.

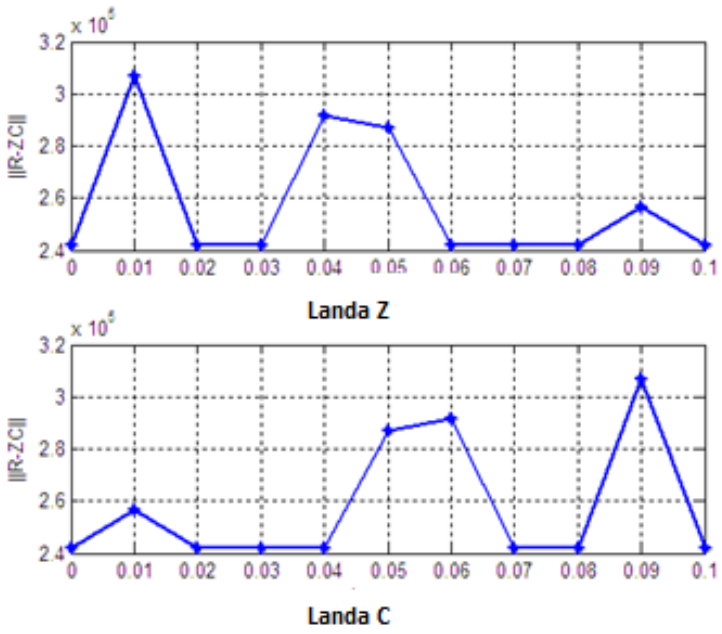


Figure 4.2. Valeurs optimales des paramètres λ_z et λ_c

4.5.1.1.3.Critère d'arrêt (nombre maximal d'itérations maxiter)

Nous avons testé l'évolution de l'erreur RMSE pour trouver la valeur optimale du nombre maximal d'itérations pour laquelle l'algorithme de factorisation converge et nous avons obtenu la plus petite valeur de l'erreur. Les expériences ont été menées sur l'ensemble de données ML_300, avec les valeurs fixées pour les paramètres comme trouvées ci-dessus (k=30, λ_z=0.02 et λ_c=0.08), en variant le nombre des itérations de [10 à 200].

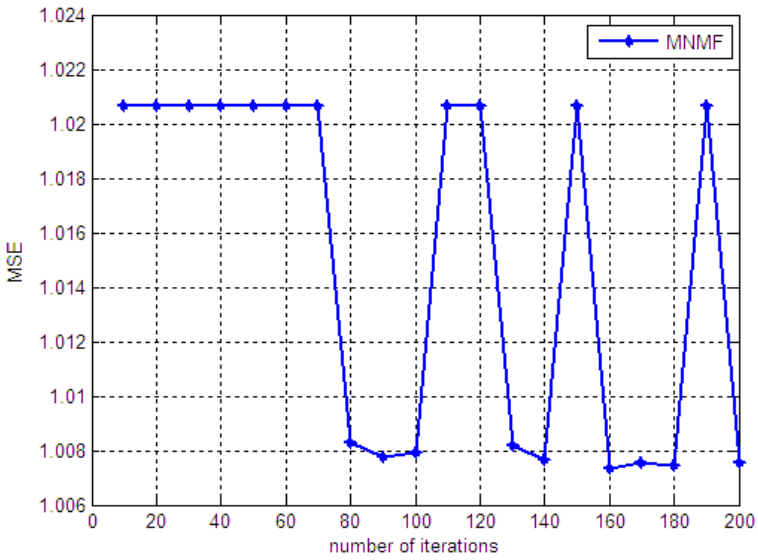


Figure 4.3. Nombre maximal d'itérations maxiter

À partir de la *Figure 4.3*, le nombre d'itérations pour lesquelles l'erreur RMSE a abouti la plus petite valeur est 160. Alors, le nombre d'itération maxiter sera fixé à 160 itérations dans le reste des évaluations.

4.5.1.1.4.Performance de la Méthode d'initialisation *rand Rrow*

Nous allons comparer dans cette sous section notre méthode d'initialisation *rand Rrow* de la matrice initiale C^0 avec la méthode la plus utilisée dans les algorithmes de factorisation matricielle, qui est la méthode d'initialisation aléatoire (*random initialization method*) en terme de temps d'exécution.

En utilisant la méthode *rand Rrow*, chaque ligne de la matrice C^0 est formée par une moyenne de 10 lignes choisies aléatoirement à partir de la matrice d'évaluation *R*.

Les temps d'exécution sont présentés dans le *Tableau 4.1*.

Initialization Method	Execution time in seconds
Rand Rrow	2.03s
Random	2.47s

Tableau 4.1. Comparaison entre les méthodes d'initialisation

À partir du *Tableau 4.1,* les expérimentations ont montré que l'utilisation de la méthode rand Rrow conduit à une convergence plus rapide que l'utilisation de la méthode d'initialisation aléatoire, car elle initialise les vecteurs de base de la matrice C^0 à partir des données incluses dans la matrice des évaluations, qui sont plus proches des meilleurs vecteurs trouvés, alors que la méthode *aléatoire* initialise, au hasard, les vecteurs de C^0 avec des valeurs différentes à celles du contenu de la matrice des notes, ce qui nécessite plus de temps pour atteindre la convergence.

4.5.1.1.5.Performance du modèle de la factorisation MFCM-NMF

Afin de prouver la performance de l'algorithme de factorisation par rapport à l'algorithme FCM pour le *clustering*, et l'efficacité de la méthode de la résolution utilisée, la précision de l'algorithme MFCM-NMF utilisant l'algorithme ACLS, est comparée aux algorithmes les plus populaires cités dans la littérature, NMFDIV **[LEE et SEU, 01]** et NMFALS **[PAA et TAP, 94]** et l'algorithme FCM **[BEZ 73],** en utilisant l'erreur RMSE. Dans ces expériences, nous avons utilisé l'ensemble de test avec *k*=30, λ_c=0.08, λ_z=0.02 et *maxiter* =160. Les résultats sont présentés dans la *Figure 4.4.*

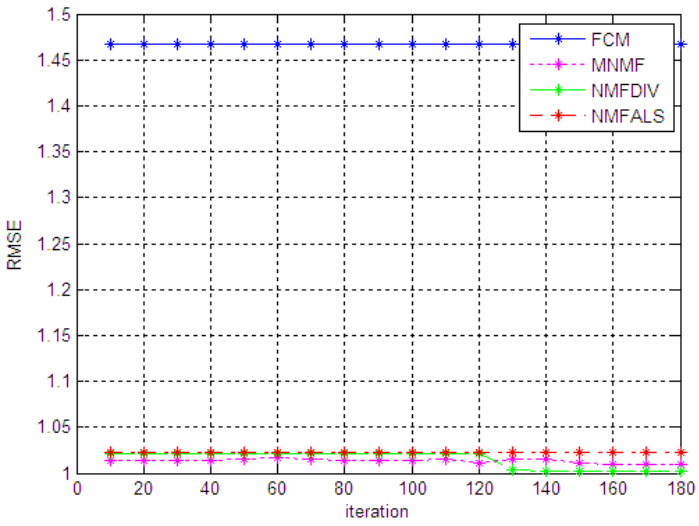


Figure 4.4: RMSE pour différents algorithmes de clustering (petite valeur, meilleur performance)

Comme on le voit dans la *Figure 4.4*, l'erreur RMSE diminue avec de petites variations ce qui nous apparait comme stable, mais en réalité il n'y a que l'erreur produite par l'algorithme FCM qui est stable depuis la convergence de l'algorithme.

Il est clairement démontré que les algorithmes de factorisation surpassent l'algorithme FCM en termes de précision (erreur RMSE), ce qui signifie qu'ils sont plus efficaces pour le *clustering* des utilisateurs. De plus, les algorithmes basés sur ALS offrent une précision semblable à celle de l'algorithme NMFDIV, mais l'erreur RMSE produite par l'algorithme ACLS est plus petite que les autres, ce qui confirme son efficacité à faire de bonnes factorisations non négatives.

4.5.1.2.Performance de l'algorithme de sélection du voisinage

Pour tester la performance de l'algorithme de la sélection du voisinage, deux expériences ont été réalisées:

- La première pour trouver les valeurs optimales du nombre des clusters flous C et le nombre des voisins flous K.
- La deuxième pour tester l'efficacité de la nouvelle mesure de similarité basée sur les degrés d'appartenance des utilisateurs, en comparant les résultats du même algorithme mais en utilisant la mesure de similarité basée sur le Coefficient de Corrélation de Pearson.

4.5.1.2.1. Choix des valeurs optimales du C et K

Pour trouver les valeurs optimales du C et K, deux expérimentations ont été faites:

- Dans la première expérience, nous avons cherché la valeur optimale des plus proches clusters flous C en utilisant l'ensemble de données *ML_200* avec k = 30, λ_z=0.02, λ_c=0.08, et *maxiter*=160, dont on a sélectionné à chaque fois un nombre différent des plus proches clusters, allant de [5 à 30] pour sélectionner la valeur pour laquelle notre algorithme aura la meilleure performance.
- La *Figure 4.5* montre l'évolution de l'erreur MAE par rapport au nombre des plus proches clusters sélectionnés en utilisant les trois ensembles de données *Given 5*, *Given 10* et *Given 20*.

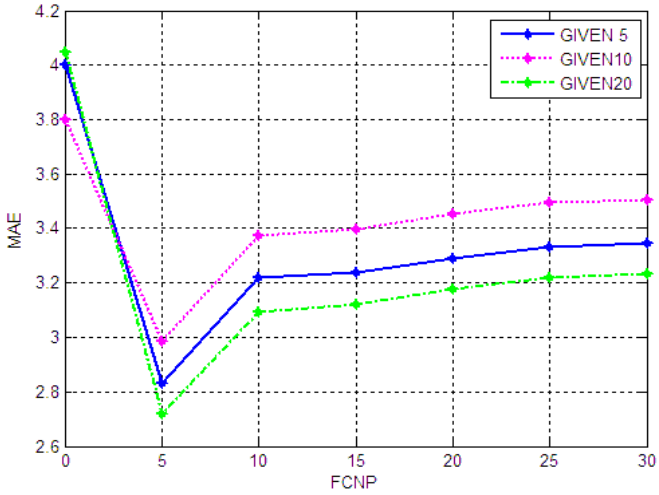


Figure 4.5: Valeur optimale des plus proches clusters flous

Comme nous pouvons le voir dans la *Figure 4.5*, la valeur la plus appropriée des plus proches clusters flous qui produit la plus petite MAE est 5 clusters.

- La deuxième expérience concerne la recherche de la valeur optimale de *K* en utilisant l'ensemble de données *ML_200* avec *k* = 30 et C = 5, tout en sélectionnant différentes valeurs des plus proches voisins K allant de 5 à 50. La *Figure 4.6* montre l'évolution de l'erreur MAE en fonction du nombre des plus proches voisins avec *Given _10* et *Given_20.*

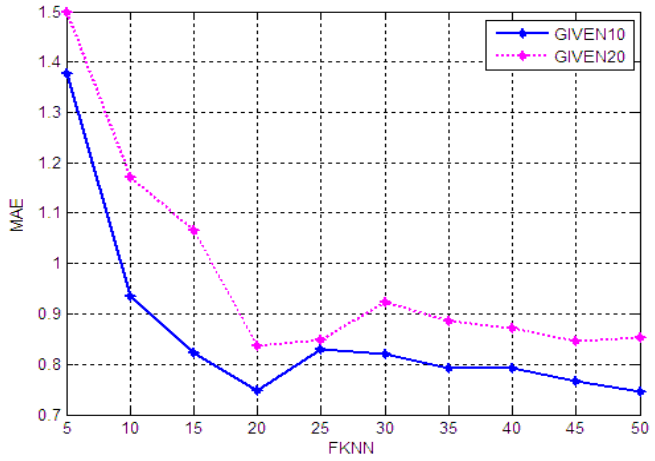


Figure 4.6: Valeur optimale des plus proches voisins flous

A partir de la *Figure 4.6*, la valeur optimale de K pour laquelle notre algorithme a eu la meilleure performance MAE pour les deux ensembles de données est R = 20.

Dans les sections suivantes de l'évaluation, le nombre C des plus proches clusters flous sélectionnés est considéré comme 5 clusters et le nombre K des plus proches voisins flous sélectionnés est considéré comme 20 voisins.

4.5.1.2.2.Performance de la nouvelle mesure de similarité

Des expériences sont menées afin de tester la performance de la nouvelle mesure de similarité en comparant son efficacité avec la mesure de Corrélation de Pearson en termes de temps d'exécution. La *Figure 4.7* présente le temps d'exécution produit par l'algorithme proposé sur les trois ensembles de données *Given 5*, *Given 10* et *Given 20* en utilisant les deux mesures de similarité.

Figure 4.7: Temps d'exécution pour le CCP et la nouvelle mesure de similarité

Les résultats présentés dans la *Figure 4.7* montrent que le temps d'exécution produit par la prédiction à base du FC utilisant la nouvelle mesure de similarité est très court par rapport à la similarité du CCP, et il augmente dans les deux cas avec le nombre d'items évalués.

Ce résultat revient à l'utilisation de la matrice de probabilité dans la sélection des plus proches clusters directement, et dans le calcul de similarité pour sélectionner les voisins similaires plutôt que parcourir toute la base de données, ce qui réduit énormément les temps de calcul. Ainsi, la similarité proposée est plus efficace pour trouver l'ensemble de voisinage candidat et pour réduire les calculs.

4.5.1.3.Performance de la prédiction basée sur le Filtrage Collaboratif Flou

Pour évaluer l'efficacité du processus de FC basé flou, sa performance est comparée à celle des prédictions, basée items, basée utilisateur et basée modèle MMFCM-NMF (MNMFB-Pred dans la *Figure 4.8*). Les expériences sont effectuées sur *ML_200* avec les trois ensembles de données *Given 5*, *Given 10* et *Given 20*, en utilisant les valeurs optimales des paramètres trouvés dans les expériences précédentes. Les paramètres de fusion dans le schème de la prédiction basée sur le FC flou sont fixé à $\lambda=0.6$ et $\delta=0.5$ expérimentalement.

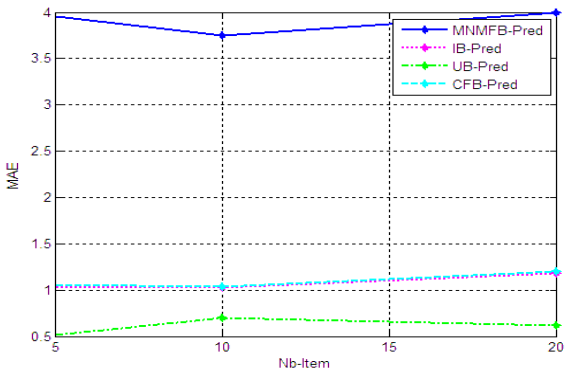


Figure 4.8: MAE pour la prédiction basée sur le FC Flou

Nous pouvons voir sur la *Figure 4.8*, que l'algorithme basé sur l'utilisateur surpasse les autres algorithmes pour toutes les valeurs des items sélectionnés et l'algorithme à base du modèle MFCM-NMF donne la mauvaise prédiction.

Cependant, l'algorithme à base d'item donne des performances similaires à celles de la prédiction basée sur le FC flou, meilleures que la prédiction MFCM-NMF et mauvaises que celles de la prédiction basée sur l'utilisateur.

Cela nous donne une conclusion que la combinaison de prédictions est plus avantageuse pour avoir de bonnes prédictions, mais en donnant plus d'importance à la prédiction basée sur l'utilisateur que les prédictions, basée sur le modèle MFCM-NMF et basée item, en variant les paramètres de fusion.

4.5.1.3.1.Comparaison de la prédiction basée sur FC flou avec le modèle SF1

La performance de la prédiction à base du FC flou est comparée à celle de l'algorithme *SF1* **[WAN *et al.*, 06],** pour montrer l'efficacité du schème de fusion. Notez que *SF1* est

un schème linéaire qui combine la prédiction basée item et la prédiction basée utilisateur comme suit :

$$SF1 = \lambda\, \widehat{ur}_{u,t} + (1-\lambda)\widehat{ir}_{u,t} \qquad \textbf{[WAN et al., 06]} \ (4.9)$$

Où, $\widehat{ur}_{u,t}$ et $\widehat{ir}_{u,t}$ sont les résultats des prédictions, basée utilisateur et basée item, respectivement. λ est un paramètre de régularisation tel que $0 \geq \lambda \geq 1$.

Les résultats de l'expérience sur *ML_200* avec λ=0.6 et δ=0.5 dans notre schème de fusion pour combiner les trois prédictions, et λ=0.5 pour le schème *SF1* sont présentés dans la *Figure 4.9*. Sachant que les valeurs des paramètres ont été fixées par les expériences.

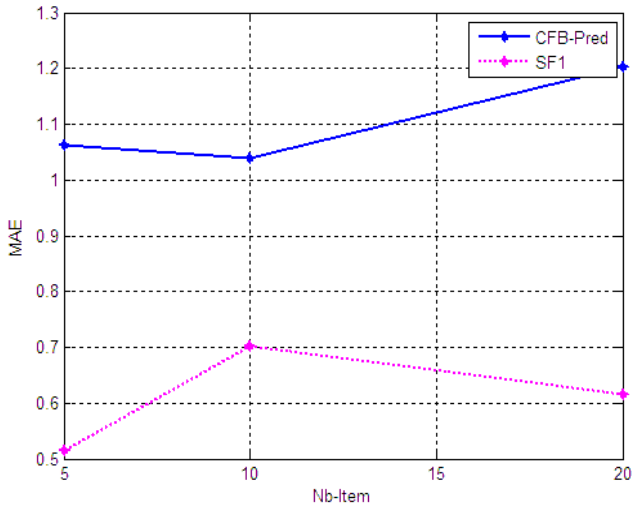


Figure 4.9: FC basé flou Vs. SF1

La *Figure 4.9* montre que l'algorithme *SF1* dépasse l'algorithme du FC flou en terme de prédiction, en raison de la fusion de la prédiction basée factorisation avec les prédictions, basée item et basée utilisateur. Néanmoins, nous ne pouvons pas ignorer la prédiction fournie par le modèle MFCM-NMF car il donne des prédictions dans les ensembles de données à grande échelle avec données dispersées.

4.5.1.4.Impact de l'évolutivité de données (Scalability Impact)

Dans cette sous section, nous menons une expérience sur l'ensemble de données *ML_300* avec le nombre des plus proches voisins flous est fixé à 20 (comme trouvé ci-dessus), mais en variant à chaque fois le sous-ensemble des voisins flous à sélectionner en les augmentant de 10% à 100%. L'évolution de la métrique MAE sera

vérifiée en utilisant les différents ensembles de prédiction *Given_5*, *Given_10* et *Given_20*. Les résultats sont présentés dans la *Figure 4.10*.

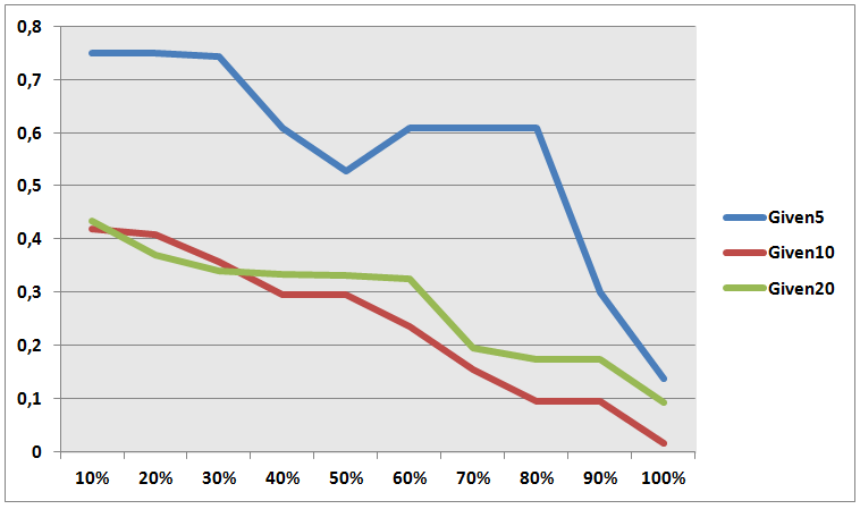


Figure 4.10: Impact de l'évolutivité

A partir de la *Figure 4.10*, on voit clairement que la performance de la prédiction augmente avec le nombre des plus proches voisins flous sélectionnés dans les trois cas d'ensembles de données. Où, la valeur de MAE diminue avec le nombre des voisins sélectionnés et avec chaque ensemble de données, et cela revient à la bonne prédiction des évaluations manquantes en utilisant plus de voisins candidats dans la prédiction basée utilisateur.

4.5.2. Evaluation de la prédiction hybride basée flou

Avant d'évaluer la performance de la prédiction hybride floue, les performances des deux prédictions, basée sur le contenu et basée sur le FC flou, ont été testées.

Dans la *Figure 4.11*, la performance de la prédiction basée FC flou est comparée à la prédiction basée sur le contenu en utilisant les ensembles de données *Given_5*, *Given_10* et *Given_20*, avec *T* (nombre des plus proches items à sélectionner dans la phase du FBC) est fixé à 10 items.

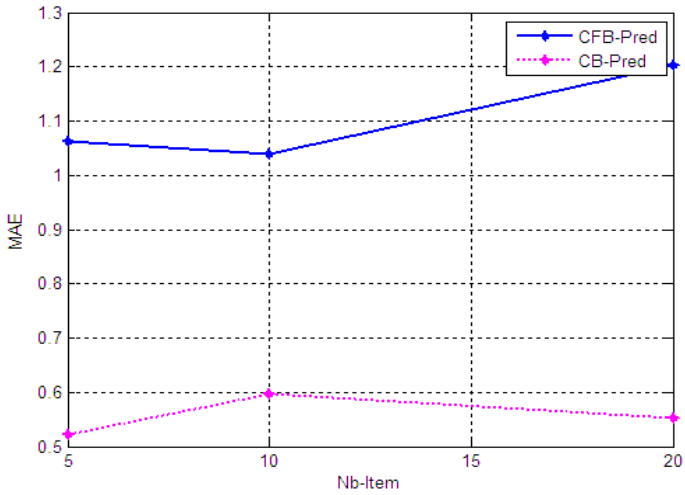


Figure 4.11: MAE des prédictions, basée FC Flou et basée Contenu

Les résultats présentés dans la *Figure 4.11* montrent que la prédiction basée sur le contenu dépasse la prédiction à base du CF flou lors de la prédiction des évaluations manquantes avec les trois ensembles de données, en générant des prédictions plus précises grâce à l'utilisation des contenus des items similaires aux préférences de l'utilisateur. Ce qui confirme l'importance de prendre en considération ce type de prédiction dans les systèmes de recommandation, ce qui est négligée par la majorité des travaux visant à améliorer les prédictions dans les systèmes de recommandation.

Cela nous a menés à l'hybridation entre ces deux types de prédictions afin d'améliorer la précision des prédictions.

4.5.2.1.Choix du paramètre Alpha

Rappelons de l'équation (3.9) Chapitre 3, la prédiction à base de FC est combinée linéairement avec la prédiction basée sur le contenu par un facteur de pondération α, pour faire une prédiction générale des évaluations.

Afin de déterminer la meilleure valeur de α, des expériences sont menées en utilisant la métrique MAE. La *Figure 4.12* montre les valeurs MAE résultantes avec différentes valeurs de α allant de 0.1 à 1, qui sont obtenues en utilisant *ML_200* sur *Given_5*, *Given_10 et Given_20.*

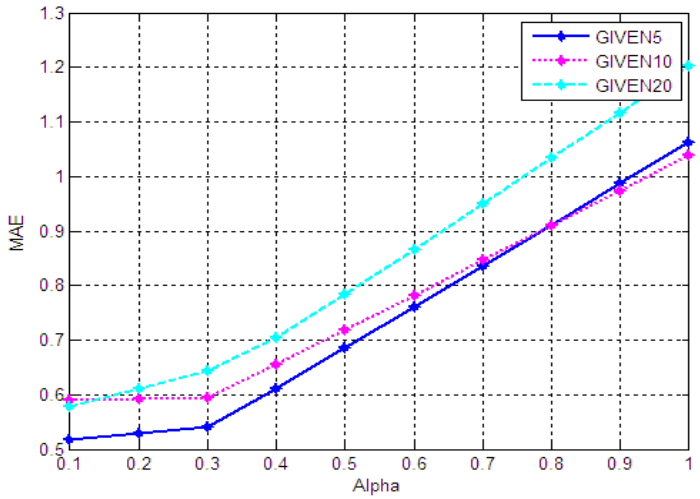


Figure 4.12: Alpha vs. MAE avec différents ensembles de données

Comme nous pouvons le voir dans la *Figure 4.12*, l'erreur MAE est presque constante entre 0,1 et 0,3 et elle diverge rapidement à partir de la valeur 0,3, et prend des valeurs très grandes. Ainsi, nous avons conclut que la valeur la plus appropriée de α est de 0,3. Par conséquent, la prédiction basée contenu prend le poids supérieur lors de la prédiction des évaluations manquantes par rapport à la prédiction basée sur le FC afin de réaliser la prédiction hybride, i.e. en tenant α = 0,3, ce poids est associé à la prédiction basée FC tandis que le poids de la prédiction basée sur le contenu est considéré comme 0,7 (1-α).

4.5.2.2.Performance de la prédiction hybride

Les résultats de la performance de la prédiction hybride sont comparés à ceux des méthodes de prédiction basée FC flou et prédiction basée FBC dans la *Figure 4.13.*

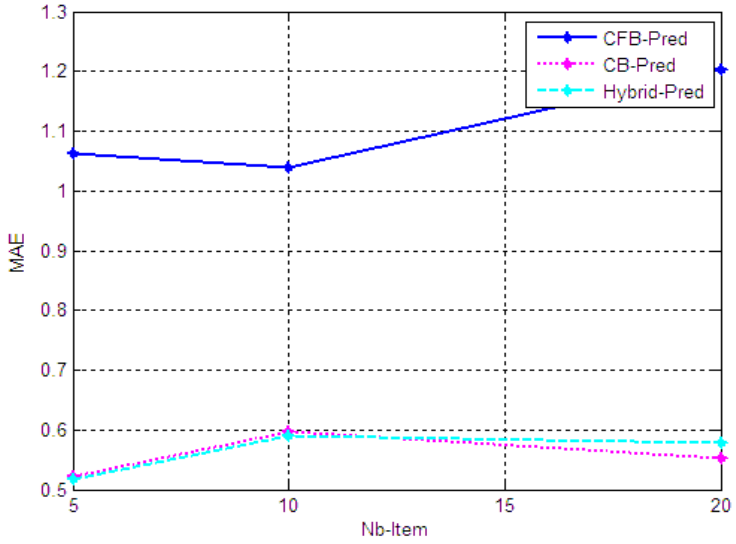


Figure 4.13: MAE des méthodes de la prédiction basée FC flou, basée Contenu et hybride

Nous pouvons voir sur la *Figure 4.13* que la prédiction hybride fournit de meilleures performances que la prédiction basée sur le FC flou lors de la prédiction des évaluations manquantes avec les trois ensembles de données, et elle fournit une performance similaire à celle de la prédiction basée contenu.

Ainsi, les résultats confirment que l'hybridation entre les méthodes de prédiction basée FC et basée contenu est plus avantageuse que d'utiliser le FC seul lors de la prédiction des évaluations.

4.5.2.3.Evaluation de la performance de la recommandation basée prédiction

Après la prédiction des évaluations manquantes pour l'utilisateur actif, les items sont classés dans l'ordre décroissant de leurs prédictions, et les Top-N items sont générés à l'utilisateur. Nous voulons évaluer cette recommandation pour tester sa performance malgré qu'on ait un autre schème de recommandation. Les résultats de performance (Rappel, Précision, et F1) sont présentés dans la *Figure 4.14*.

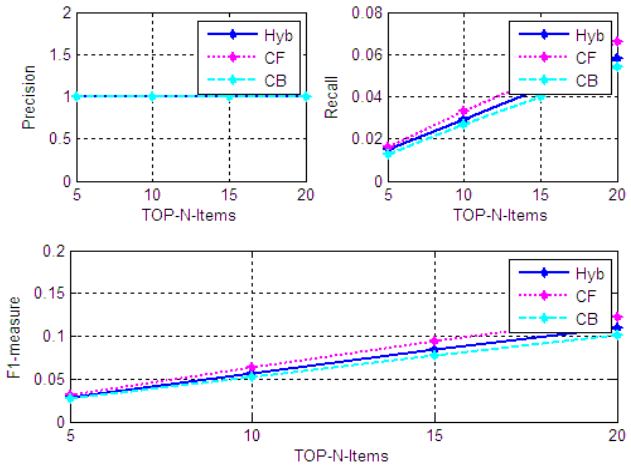


Figure 4.14: Performance de la recommandation basée prédiction

A partir de la *Figure 4.14*, on observe que le système fournit des recommandations avec une précision égale à 1 pour les trois recommandations, c'est-à-dire que tous les items générés sont pertinents pour l'utilisateur. Tandis que, le rappel et la mesure F1 augmentent avec chaque nombre N des items recommandés, respectivement. Où, la recommandation basée sur la prédiction hybride se situe au milieu entre la recommandation basée sur la prédiction à base du FC, et la recommandation basée sur la prédiction à base du contenu, avec des valeurs de performance très proches entre les trois recommandations. On observe aussi que la performance de la recommandation à base du FC est supérieure que les deux recommandations, ce qui prouve l'importance et l'efficacité de l'algorithme du filtrage collaboratif même que si ses résultats de prédictions n'étaient pas meilleurs que les deux autres prédictions.

4.5.3. Evaluation de la Recommandation Top-K Diversifiée basée Flou

Dans cette section, nous évaluons la performance de notre système de recommandation en trois parties. Tout d'abord, en testant la performance du système en termes de précision avec le schème de recommandation ordinaire (recommandation des items préférés dans le cluster le plus proche). Ensuite, en testant le taux de diversité obtenue dans les listes recommandées et comment elle évolue par rapport à la précision. Finalement, dans la dernière expérience nous allons tester l'efficacité de notre proposition d'utilisation d'une phase de prédiction avant le processus de

recommandation afin de réduire l'effet de la clairsemé de données et d'augmenter la précision des recommandations.

4.5.3.1.Performance générale de la Recommandation Top-k basée Flou

La performance de la recommandation Top-K hybride basée flou est comparée à celle de la recommandation ordinaire, où les Top-N items sont recommandés à partir du groupe le plus proche. Les performances du Rappel, Précision et la mesure F1 des recommandations générées sur l'ensemble de test avec différentes valeurs d'items à sélectionner N = [5, 10, 15, 20], sont présentées dans la *Figure 4.15*.

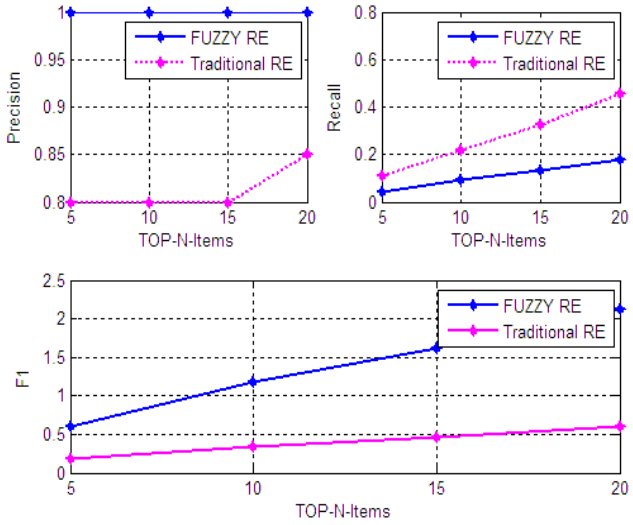


Figure 4.15: Performance de la Recommandation Top-K diversifiée basée flou Vs. Recommandation Top-N Ordinaire

Les résultats présentés dans la sous-figure en haut à droite de la *Figure 4.15*, montrent qu'avec N = 10 le rappel de la recommandation floue est d'environ 0,12, ce qui signifie que le système a une probabilité de 12% à recommander un item pertinent. Alors que, la recommandation ordinaire fournit un rappel égal à 0,22, ce qui signifie qu'environ 22% des items pertinents sont probablement recommandés.

En considérant maintenant les résultats de précision, dans la sous-figure en haut à gauche de la *Figure 4.15*, la recommandation floue a une précision de 100% pour toutes les valeurs des nombres d'items à recommander N, ce qui signifie que les items recommandés sont tous pertinents. Bien que dans la recommandation ordinaire, il ya 8 items pertinents présentés dans la liste du Top-10 avec une précision de 80% pour N=5 et N=10 items et elle augmente légèrement pour N=15 et N=20 items, où elle aboutit une précision de 85%.

Dans la partie inférieure de la *Figure 4.15*, nous pouvons voir clairement que la recommandation floue dépasse la recommandation ordinaire avec les quatre valeurs des items sélectionnés N.

En raison de la performance éprouvée de la recommandation floue par rapport à la recommandation ordinaire en termes de précision et de mesure F1, nous concluons qu'elle est plus efficace et plus pertinente.

Pour examiner plus, l'efficacité de la recommandation floue, sa performance est comparée à celle des recommandations générées dans les plus proches clusters flous trouvés. Les résultats sont présentés dans la *Figure 4.16*.

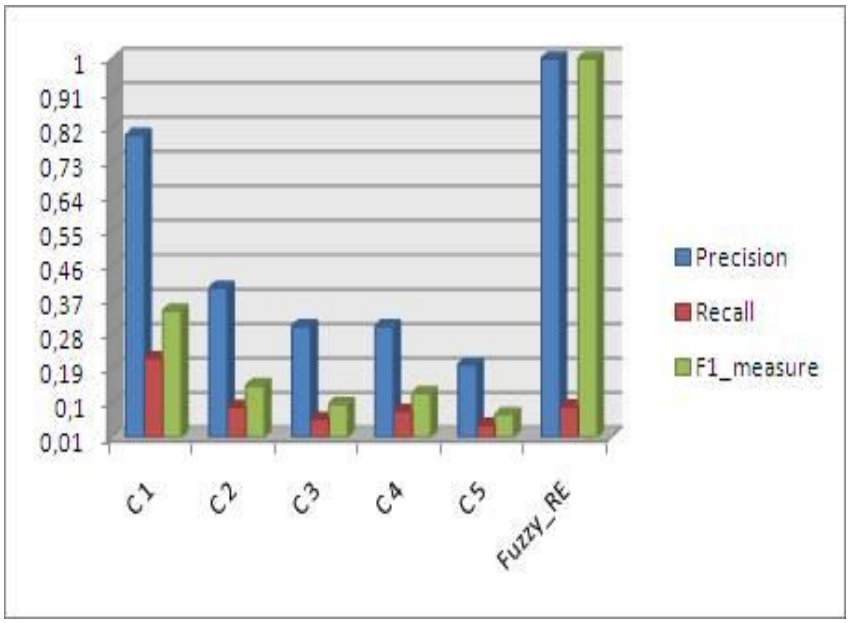


Figure 4.16: Comparaison des performances de la Recommandation Top-K

A partir de la *Figure 4.16*, il s'est avéré que la recommandation hybride a donné les meilleurs résultats, suivie de celle générée dans le plus proche cluster alors que la performance de la recommandation dans les autres plus proches clusters était assez pauvre. A noter que la performance dans les plus proches clusters diminue en fonction de leur similarité avec l'utilisateur actif (à partir du cluster le plus similaire C1 au moins similaire C5).

Concluant que la génération d'une liste de recommandation diversifiée pour les utilisateurs actifs est plus avantageuse, et plus intéressante pour eux que de recommander une liste des Top items générée à partir du groupe le plus similaire.

4.5.3.2.Diversité Vs. Performance de la Recommandation hybride basée Flou

Dans cette section, nous évaluons la diversité que notre approche de recommandation a prouvée en la comparant avec la précision, pour vérifier le compromis entre eux, c'est-à-dire vérifier si notre approche proposée augmente la diversité dans les listes de recommandation ou non, et est-ce-que cela provoque une grande perte dans la précision des recommandations ou non. Les résultats de comparaison sont présentés dans la *Figure 4.17.*

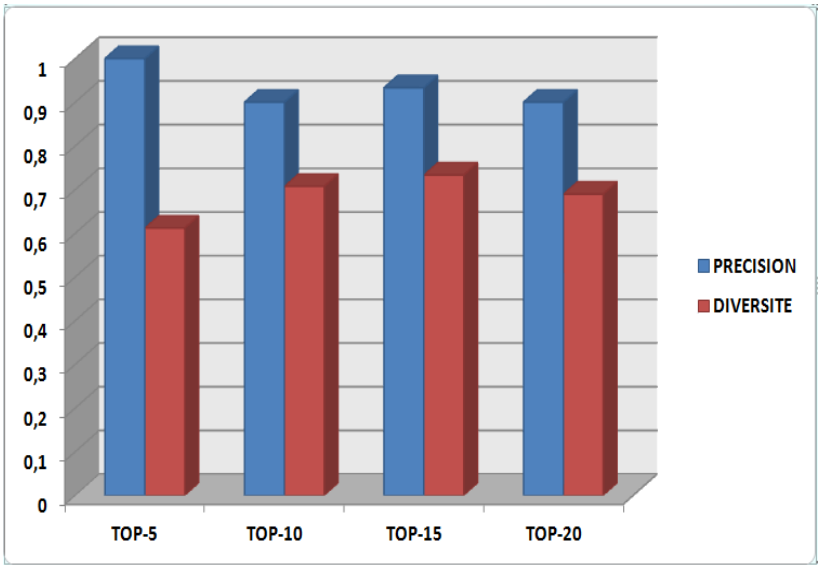


Figure 4.17: Diversité Vs. Performance de la Recommandation Top-K basée Flou

Les résultats présentés dans la *Figure 4.17* montrent que la recommandation Top-K hybride basée sur le flou augmente le taux de diversité avec le nombre d'items à recommander, en engendrant juste une petite perte en précision, ce qui confirme l'efficacité de la nouvelle approche de recommandation hybride basée flou à générer des recommandations diversifiées sans avoir perdre la précision du système.

4.5.3.3.Performance de la recommandation avant Vs. après la prédiction et Impact de la clairsemé de données

4.5.3.4. Nous allons comparer dans cette section, la performance de la recommandation hybride basée flou avec et sans la prédiction des évaluations manquantes pour l'utilisateur actif en termes de précision et de diversité, afin d'approuver notre proposition que la phase de prédiction avant la recommandation améliore la précision des recommandations et augmente la performance du système. Les résultats de comparaison entre les deux recommandations, recommandation directe sans la prédiction des évaluations (notée *REordin* dans la figure) et la

recommandation après la prédiction (notée *REBPred* pour recommandation basée prédiction), sont présentés dans la *Figure 4.18.*

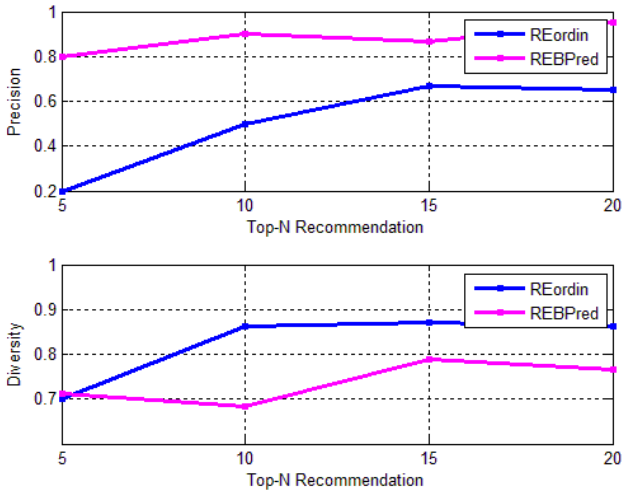


Figure 4.18: Comparaison de la performance de la Recommandation Top-K basée sur la prédiction

Les résultats présentés dans la *Figure 4.18*, montrent que la recommandation dans sa forme naturelle c.à.d. calcul des scores pour les items en utilisant un schème hybride (FC avec FBC) et recommandation des items ayant les plus grands scores a donné une précision trop dépassée par la recommandation proposée c.à.d. après la prédiction des évaluations manquantes, surtout pour le cas des Top-5 items. Tandis qu'elle a surpassé l'approche proposée en termes de diversité i.e. elle a généré plus d'items divers, mais avec des valeurs de diversité proches.

4.5.4. Discussion des résultats

Les études empiriques sur la base de données *Movielens* ont démontré que la méthode proposée est plus performante et efficace que la recommandation ordinaire à travers l'évaluation des performances du système avec les mesures MAE, Rappel, Précision et la mesure F1.

Les expérimentations ont prouvé que la nouvelle mesure de similarité a dépassé la mesure du CCP en termes de temps d'exécution et ça revient à l'utilisation des résultats obtenus après la factorisation matricielle pour choisir les plus proches groupes, et le calcul de la similarité juste pour un petit nombre d'utilisateurs ce qui réduit énormément le temps de calcul et confirme son efficacité. De plus, la recommandation diversifiée

montre de meilleurs résultats comparés à ceux de la recommandation ordinaire en termes de précision, rappel et mesure F1.

Les expériences sur différents ensembles de voisins sélectionnés pour la prédiction ont montré une augmentation de la performance avec l'augmentation du nombre d'utilisateurs à sélectionner, et cela revient à l'utilisation de ces utilisateurs dans la prédiction basée utilisateur, où le choix des voisins candidats joue un rôle crucial. Ce qui prouve que le système est capable de prendre en considération les ajouts des nouveaux utilisateurs à la base de données et qu'elle réduit l'effet du problème de l'évolutivité de données (*Scalability Problem*) dans les systèmes de recommandation.

De plus, l'hybridation entre le FC et le FBC a amélioré grandement la performance de la prédiction par rapport à la prédiction basée sur le FC, ce qui revient principalement à l'utilisation des contenus des items pour choisir les items les plus similaires aux préférences de l'utilisateur.

En outre, les résultats de comparaison entre l'approche de recommandation basée sur la prédiction par rapport à la recommandation ordinaire ont montré que la méthode proposée surpasse la recommandation ordinaire en termes de précision et offre moins de contenus divers. Ces résultats reviennent à la clairsemé des données dans la matrice d'évaluations, qui conduit à trouver peu de voisins similaires à l'utilisateur et à obtenir un ensemble insuffisant d'évaluations similaires par les utilisateurs similaires ou de produits similaires, ce qui réduit énormément la précision des recommandations, et influence sur la recherche des items pertinents pour l'utilisateur et donne des items divers. Nous confirmons donc à partir de ces résultats que l'approche proposée, qui consiste à prédire les évaluations manquantes pour l'utilisateur actif avant de lui générer le processus de recommandation, est appropriée pour l'amélioration des données dispersées et pour réduire l'effet de la dispersion des évaluations dans la matrice utilisateur-item sur la précision des recommandations. De plus, elle améliore la performance du système, tout en offrant des items divers.

A partir des résultats présentés dans cette section, visant à tester la validité de la nouvelle approche de recommandation basée flou, nous confirmons son efficacité et sa grande performance à rendre le système capable de s'adapter aux différents besoins de

l'utilisateur, en lui offrant différents items qui correspondent à ces différents choix, sans une grande perte dans la performance générale du système.

4.6. Expérimentation 2 : Evaluation de l'Approche de Diversité Hybride basée sur le Niveau de Similarité

Dans cette section, nous allons mener une série d'expériences afin de tester la performance de la nouvelle approche de diversité basée sur le niveau de similarité et son efficacité à générer des items divers pertinents pour l'utilisateur. Où, nous allons tester sa performance par rapport à l'approche hybride basée flou et par rapport à quelques méthodes de diversité citées dans la littérature.

4.6.1. Test de l'ensemble des items divers à sélectionner

Nous allons tester dans cette sous section la faisabilité de la proposition du choix de la catégorie la plus pertinente et la sélection des items pertinents de cette catégorie ayant les plus grands degrés d'intérêt pour être recommandés. Nous allons comparer la performance de notre proposition avec le cas où on sélectionne directement l'ensemble des items divers pertinents ayant les plus grands degrés d'intérêt.

Les expérimentations sont menées sur l'ensemble de test, avec K=15 voisins les plus proches à sélectionner dans la phase du FC, et λ=0.5 le paramètre de fusion pour la prédiction basée sur le voisinage (fusion entre prédiction basée utilisateur et basée item). La *Figure 4.19* présente les différents résultats de performance (F1 et Diversité) obtenus par les deux méthodes de sélection des items divers afin de générer les recommandations Top-N avec N= [5, 10, 15, 20].

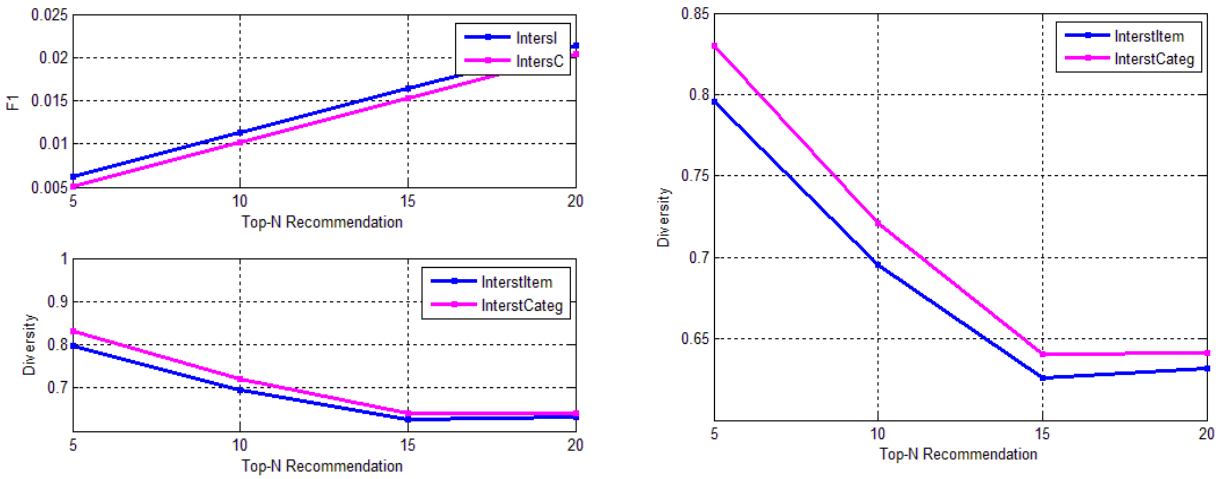


Figure 4.19 : Comparaison de la performance des deux méthodes de sélection des items divers

A partir de la *Figure 4.19*, on remarque que la méthode de sélection des items intéressants à partir de la catégorie la plus intéressante a dépassé la méthode de sélection des items divers ayant les plus grands degrés d'intérêt pour l'utilisateur en termes de performance. De plus, elle augmente la diversité dans les listes de recommandation par rapport à la sélection directe de différents items intéressants.

Ces résultats reviennent au fait que les items sélectionnés directement à partir de leur classement basé sur le degré d'intérêt, peuvent être trop dissimilaires aux préférences de l'utilisateur et inintéressants ce qui diminue la performance de la méthode, alors que la méthode de sélection de la catégorie la plus intéressante a augmenté la performance en sélectionnant des items divers et intéressants pour l'utilisateur. De plus, la méthode de sélection de la catégorie pertinente a surpassé aussi la méthode de sélection directe en termes de diversité, car elle génère les items dissimilaires aux préférences de l'utilisateur mais les plus importants.

4.6.2. Comparaison avec l'approche hybride basée flou

Pour démontrer l'efficacité de la nouvelle méthode de diversité, nous avons comparé ses résultats de performance (précision et diversité) avec l'approche hybride basée flou. L'utilisation des deux algorithmes sur l'ensemble de test de *Movilens* ont produit des recommandations diversifiées, dont les résultats sont présentés dans la *Figure 4.20*.

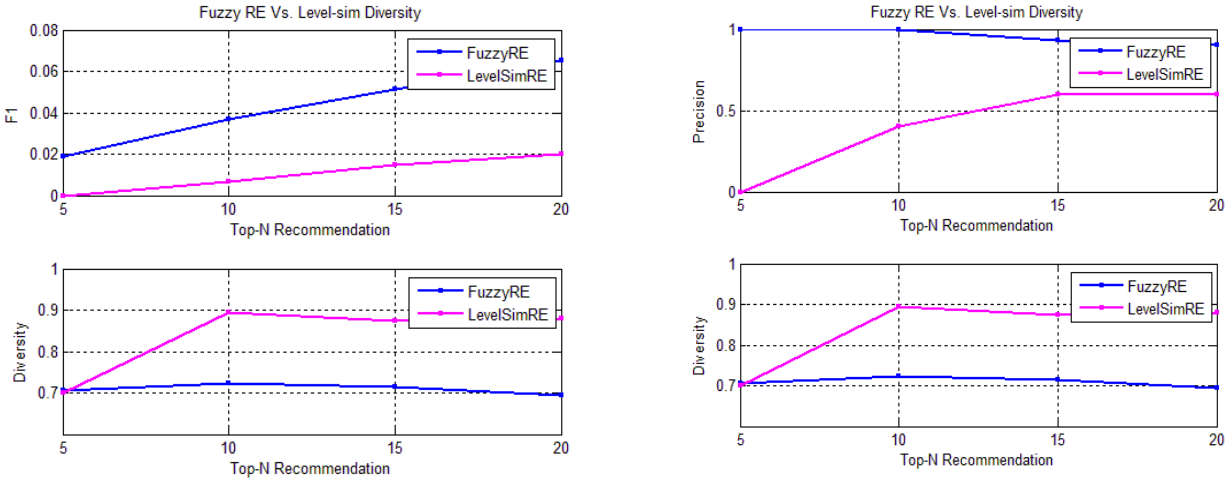


Figure 4.20 : Comparaison de la recommandation hybride floue avec la recommandation diversifiée basée sur le niveau de similarité

A partir des résultats présentés dans la *Figure 4.20*, l'algorithme de recommandation hybride floue surpasse l'algorithme de diversité basée sur le niveau de similarité en terme de performance (précision, rappel et F1) avec tous les Top-N items, N= [5, 10, 15, 20]. Cependant, l'algorithme de diversité basée sur le niveau de similarité dépasse l'algorithme hybride flou dans la diversité de la liste de recommandation.

4.6.3. Comparaison avec les méthodes de diversification de l'état de l'art

Dans cette sous section, nous allons tester l'efficacité de notre méthode de diversification, en comparant ses performances (précision et diversité) avec quelques méthodes citées dans la littérature, sachant : la méthode *long-tail* (popularité inversée), les méthodes de reclassement proposées dans **[ADO et KWO, 09]** valeur de note prédite inversée, évaluation moyenne, et *Likeability* Absolue, dont on a utilisé ces méthodes pour générer les items dissimilaires qui seront constitués la moitié de la liste à recommander. Pour tester la performance des méthodes utilisées pour la comparaison, nous avons généré une liste de recommandation avec Top-N items similaires par le processus de recommandation Top-N présenté dans le chapitre 3, sous section (3.2.9.2).

Ensuite, nous avons appliqué chaque méthode, séparément, sur l'ensemble des items dans le système et nous avons sélectionné un sous ensemble des items différent à chaque fois, tel que le nombre d'items à sélectionner est égal à la moitié du nombre N

des items à recommander. Ensuite, l'ensemble sélectionné des items divers est fusionné avec les (N/2) items classés en premier dans la liste similaire, pour constituer la liste finale de recommandation. Les résultats des expérimentations sont présentés dans les *Figures 4.21 et 4.22*.

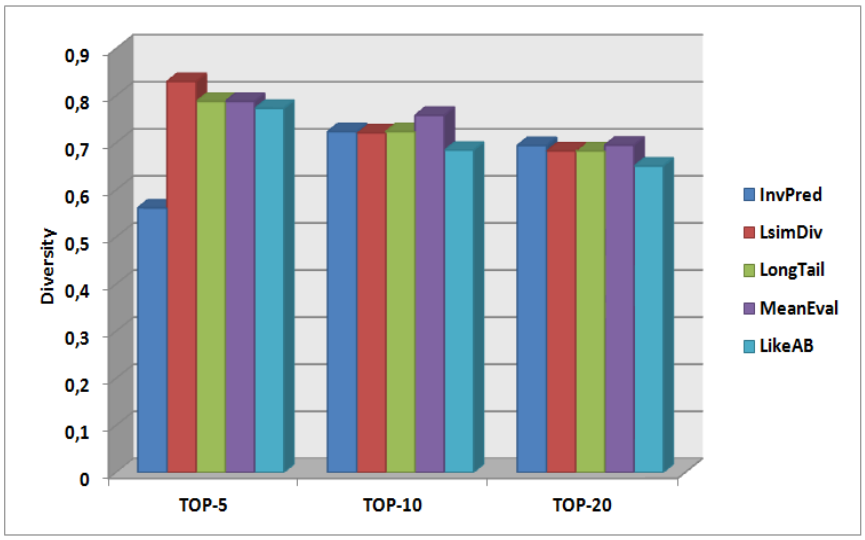


Figure 4.21 : Comparaison de la Diversité avec les méthodes de diversification de l'état de l'art

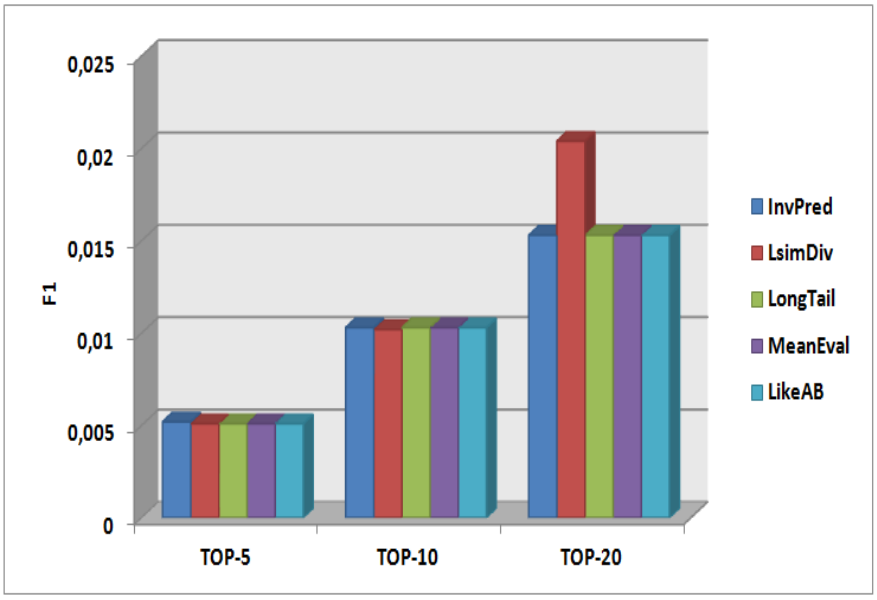


Figure 4.22: Comparaison de F1 avec les méthodes de diversification de l'état de l'art

La *Figure 4.21* montre que la méthode de diversification basée sur le niveau de similarité surpasse les autres méthodes en termes de diversité pour la recommandation Top-5 items. Puis, elle commence à diminuer avec Top-10 et Top-20, où elle coïncide avec les autres méthodes pour la recommandation Top-10 (les cinq méthodes ont généré le même taux de diversité dans la liste de recommandation), et elle prend la plus petite

valeur de diversité pour la recommandation Top-20. Notons que les deux méthodes (Prédiction inversée et *Long Tail*) ont donné la même diversité des listes de recommandation pour les trois ensembles d'items (Top-5, Top-10 et Top-20) et la méthode *Likeability* absolue a eu les plus petits taux de diversité pour Top-10 et Top-20.

Concernant maintenant la précision des recommandations générées par les différentes méthodes, présentée dans la *Figure 4.22*, nous observons que les cinq méthodes présentent des précisions similaires pour les recommandations Top-5 et Top-10, mais pour la recommandation Top-20, la diversité basée sur le niveau de similarité performe significativement mieux que les autres méthodes. Sachant que les autres méthodes de diversifications donnent la même performance pour les trois recommandations.

4.6.4. Discussion des résultats

Les expérimentations menées dans cette section ont prouvé la faisabilité de la méthode proposée. Où, les résultats ont confirmé notre hypothèse que la sélection de la catégorie dissimilaire ayant le plus grand degré d'intérêt augmente la diversité des recommandations en générant de nouveaux items dissimilaires aux préférences de l'utilisateur, et nous constatons que les items sélectionnés à partir de cette catégorie sont plus pertinents et plus importants pour l'utilisateur que les autres, parce que l'utilisateur ne sera pas sûrement intéressé par des items très dissimilaires appartenant à différentes catégories dissimilaires.

La comparaison avec les méthodes de diversification des recommandations citées dans la littérature ont montré que notre méthode agit d'une façon similaire avec ces méthodes dans le cas de N=10, alors qu'elle augmente la diversité pour d'autres (N=5 items recommandés), et elle réduit la diversité pour les grands nombres d'items recommandés (N=20). Cependant, l'évolution de la diversité fournie par la méthode basée sur le niveau de similarité engendre une augmentation dans les précisions des recommandations et surtout pour le cas du N=20.

C'est vrai que la méthode basée sur le niveau de similarité a donné des résultats similaires à ceux produits par les autres méthodes, mais l'avantage de notre méthode par rapport à celles utilisées dans la comparaison est qu'elle génère des items divers jamais vus par l'utilisateur mais pertinents pour lui en se basant sur leur degrés d'intérêt.

Tandis que les autres méthodes génèrent des items différents : les moins populaires, les moins aimés par les voisins, les items ayant les plus petites moyennes d'évaluations, ou bien qui ont eu les plus petites valeurs de prédiction, ce qui donne des items qui ne sont pas intéressants pour l'utilisateur ou qu'ils peuvent être un peu proches de ses préférences.

La comparaison des performances avec la méthode de recommandation diversifiée basée flou a prouvé que la méthode de diversification basée sur le niveau de similarité augmente significativement la diversité dans les listes de recommandation par rapport à la recommandation basée flou, ce qui est le but principal de notre approche. Tandis qu'elle réduit la précision des recommandations par rapport à la recommandation basée flou, ce qui constitue un fondement de base et confirme l'hypothèse de l'hybridation entre ces deux approches afin d'équilibrer le compromis entre la précision et la diversité dans les listes de recommandation.

Finalement, nous pouvons conclure que la méthode a donné de bonnes performances, en général, et elle est efficace pour générer des recommandations diversifiées pertinentes.

4.7. Expérimentation 3 : Evaluation de la Nouvelle Approche de Recommandation Hybride Diversifiée basée Flou et Niveau de Similarité

Nous allons tester dans cette section la performance de la nouvelle approche de recommandation hybride diversifiée qui combine entre l'algorithme du FC basé flou et la méthode de diversification basée sur le niveau de similarité. En premier lieu, nous allons tester l'efficacité de la méthode proposée pour la fusion. Ensuite, nous testons la performance de la nouvelle approche en comparant ses résultats avec les deux approches proposées.

4.7.1. Test de la méthode de fusion

Dans cette sous section, nous allons tester la façon avec laquelle les deux listes de recommandation sont fusionnées, en comparant les deux possibilités suivantes :

- La première (méthode proposée), en éliminant les *x* derniers items de la liste générée par la méthode du FC basé flou, où, *x* est égale au nombre d'items dans la liste générée par la méthode basée sur le niveau de similarité. Ensuite, la recommandation finale fusionne le reste de la liste du FC basé flou (N-*x*) avec la liste des items divers basée sur le niveau de similarité.
- La deuxième possibilité est l'agrégation des deux listes (fusion directe des deux listes), où le nombre des items dans la liste finale à recommander sera automatiquement plus que N.

Les expérimentations sur l'ensemble de test de *Movilens*, en considérant les mêmes valeurs trouvées déjà tout au long des tests pour les différents paramètres utilisés avec le nombre d'items à recommander N = [5 10 15 20], ont donné les résultats présentés dans la *Figure 4.23.*

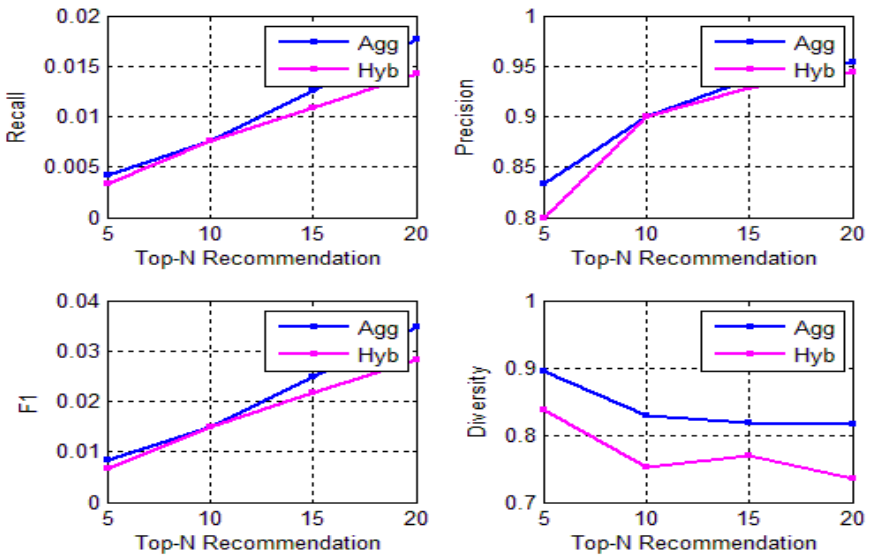


Figure 4.23: Comparaison des méthodes d'hybridation

A partir des résultats présentés dans la *Figure 4.23*, nous constatons que la méthode d'hybridation par agrégation des listes de recommandation a surpassé notre méthode proposée du côté de la performance et du côté de la diversité, sauf dans le cas où le nombre d'items à sélectionner est N=10, où elles se coïncident, ce qui prouve que même avec un nombre plus petits d'items notre approche est plus performante que la méthode par agrégation pour ce cas.

Le reste des résultats reviennent au fait que la méthode par agrégation génère un nombre d'items plus grand que notre méthode proposée, ce qui implique forcément une disparité dans les résultats entre eux.

4.7.2. Performance de la nouvelle approche hybride de diversité

Afin de prouver l'efficacité de la nouvelle approche hybride, nous avons mené des expériences sur l'ensemble de test de *Movilens*, et comparé ses performances et sa capacité à générer des listes de recommandation diversifiées avec l'approche hybride basée flou et la méthode de diversité basée sur le niveau de similarité. Les résultats sont présentés dans la *Figure 4.24*.

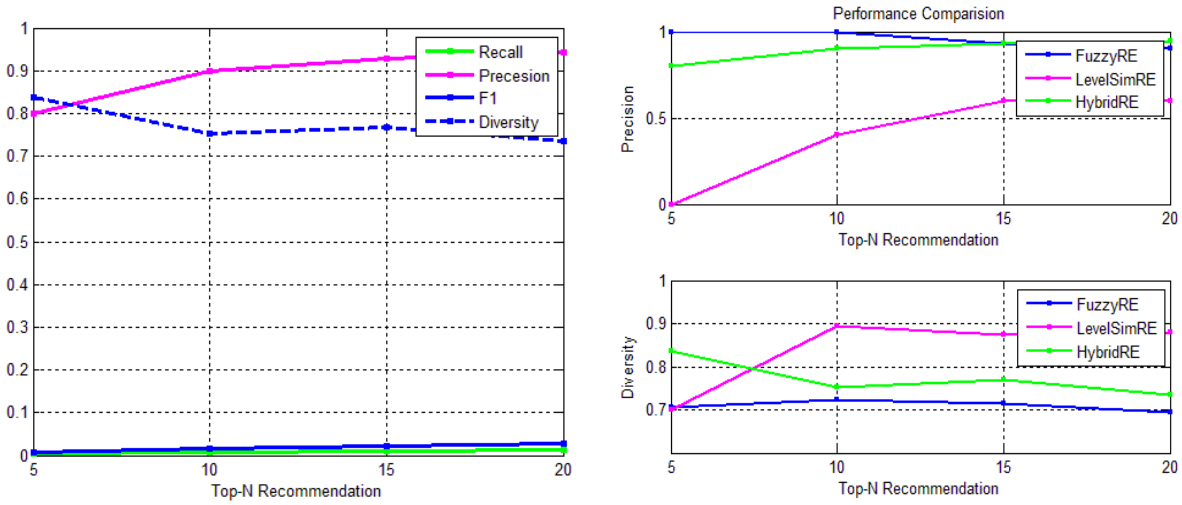


Figure 4.24 : Performance de la nouvelle approche hybride

Dans la sous-figure gauche de la *Figure 4.24*, les résultats de performance de la nouvelle méthode hybride diversifiée basée flou et niveau de similarité sont présentés, et on constate que la nouvelle méthode donne de bonne recommandation avec précision et diversité augmentées en même temps, ce qui était notre objectif pour l'hybridation.

Afin d'analyser encore plus la performance de la nouvelle approche hybride, nous avons comparé les résultats de la précision et de la diversité obtenus par les trois approches dans la sous-figure de droite de la *Figure 4.24.*

Nous voyons à partir des résultats que la méthode de recommandation hybride basée flou a amélioré la performance, tandis que le processus de diversité basée sur le niveau de similarité augmente la diversité dans la liste de recommandation. La méthode

hybride qui combine la recommandation basée flou avec la recommandation diversifiée basée sur le niveau de similarité donne une précision proche de celle produite par la recommandation basée flou en K = 5 et 10 items et similaire en K = 15 et 20 items. Alors que, la diversité produite par l'approche hybride est la plus élevée en K = 5 items, et se situe entre la méthode basée flou et la méthode basée sur le niveau de similarité en K = [10, 15, 20] items, respectivement, où elle augmente dans le cas de K = 15 items.

4.7.3. Discussion des résultats

A partir des résultats présentés, la nouvelle approche hybride a prouvé une amélioration de la diversité dans les listes de recommandation par rapport à la méthode basée flou tout en offrant une petite diminution dans la diversité par rapport à la méthode de diversité basée sur le niveau de similarité. D'un autre côté, elle a augmenté grandement la précision des recommandations par rapport à la méthode basée sur le niveau de similarité, où sa précision était proche de celle de la méthode basée flou avec N= [5, 10] items, similaire avec N=15, et plus grande avec N=20 items.

Donc, nous confirmons que la méthode hybride équilibre le compromis entre les deux critères de performance (précision et diversité), et qu'elle est efficace pour améliorer la diversité dans les listes de recommandation tout en conservant le gain de performance. Le test de performance des différentes approches avec des cas réels dans le domaine du *e-learning* est présenté dans la section suivante.

Partie II: ETUDE DE CAS (Scénario E-learning)

4.8. Intégration des Approches Proposées dans la Plateforme *Moodle*

Pour montrer comment les différentes approches proposées agissent dans un scénario concret, le domaine du *e-learning* est adressé pour montrer l'utilité du système de recommandation dans un scénario réel. Les différentes méthodes de diversité proposées : basée flou **[MAA et SER, 12a] [MAA et SER, 12b],** basée niveau de similarité et la troisième approche hybride qui les combine, ont été intégrées dans un système de gestion *e-learning* pour améliorer la personnalisation du contenu et faire

connecter les apprenants appropriés en fonction de leurs besoins individuels, leurs préférences et leurs objectifs d'apprentissage.

4.8.1. *Moodle* Dataset

Moodle[5] est une plateforme de logiciels *e-learning* des sources gratuites. En raison de l'absence des ensembles de données publiés pour l'apprentissage formel et non formel, nous avons créé un ensemble de données concrets en invitant les apprenants à participer à une application *Moodle*. Nous avons restreint la validation d'un sous-ensemble de cette base en sélectionnant seulement 22 apprenants, et 20 cours où ceux-ci sont regroupés en 6 catégories (voir *Figure 4.26*). Il est intéressant de noter que chaque cours de cette application peut être associé à une ou plus de catégories (par exemple, un cours sur le langage XML peut être associé à deux catégories différentes : Web sémantique et base de données).

Chaque apprenant dans le système peut évaluer un cours disponible avec un score dans l'intervalle [1,5] pour exprimer son opinion sur le contenu du cours.

Dans les systèmes d'apprentissage en ligne, il est clair que les apprenants sont intéressés non seulement avec des cours appartenant à leur spécialité étudiée (catégorie), mais aussi avec des cours diversifiés qui peuvent être intéressants aux profils des apprenants. Par exemple, les apprenants en informatique qui s'intéressent aux langages du Web sémantique peuvent être également intéressés par des cours sur les bases de données. Ainsi, les intérêts des étudiants avec ces modules sont donnés par des prédictions de notes.

4.8.2. Principales interfaces du système

Nous allons présenter dans cette section les principales interfaces de la plateforme *Moodle*, que nous avons installée et utilisée. On distingue principalement deux types d'interfaces attribuées à deux types d'entités : Administrateur et Utilisateur (Apprenant).

[5] www.moodle.org

4.8.2.1.Interfaces Administrateur

L'administrateur du système possède des activités et priorités autres que celles de l'utilisateur. Pour pouvoir accéder au système, l'administrateur doit se connecter en introduisant le nom d'utilisateur « Admin » et le mot de passe « Admin ». Ensuite, il peut accéder et faire un ensemble d'activités et fonctions : *Gestion des utilisateurs, gestion des cours, gestion des accès, et l'ajout des activités* :

- **Gestion des utilisateurs** : pour cette option, l'administrateur gère les différents comptes des utilisateurs du système.
- **Gestion des cours et catégories** : avec cette option l'administrateur peut gérer les cours en ajoutant de nouveaux cours, ou en organisant les cours dans les catégories par exemple.
- **Ajout d'une activité :** Une activité est un nom général pour un groupe de fonctions dans un cours *Moodle*. Habituellement, une activité est quelque chose que l'élève fera en interagissant avec d'autres étudiants et/ou l'enseignant.

  Il y a 14 activités différentes livrées en standard avec *Moodle*, qui peuvent être trouvées depuis le menu déroulant "Ajouter une activité" :

  - *Atelier* : permet l'évaluation par les pairs.
  - *Base de données* : permet aux participants de créer, maintenir et rechercher une banque de fiches.
  - *Chat* : permet aux participants d'avoir une discussion synchrone en temps réel.
  - *Consultation* : permet de recueillir des données auprès des élèves pour aider les enseignants à connaître leur classe et réfléchir sur leur propre enseignement. Les consultations sont prédéfinies (non modifiables).
  - *Devoir* : permet aux enseignants de noter et faire des commentaires sur des fichiers déposés par les étudiants, ou une réalisation faite en ligne ou hors ligne.
  - *Feedback* : permet de créer et réaliser des enquêtes afin de recueillir les commentaires.
  - *Forum* : permet aux participants d'avoir des discussions asynchrones.
  - *Glossaire* : permet aux participants de créer et de maintenir une liste de définitions, comme un dictionnaire.
  - *Leçon* : permet de délivrer du contenu de manière flexible, en suivant différents parcours programmables.

d'apprentissage conformes LTI et des activités sur d'autres sites Web.

- *Paquetage SCORM* : permet d'intégrer des paquets SCORM dans le contenu des cours.
- *Sondage* : permet à un enseignant de poser une question et donne un choix de réponses multiples.
- *Test* : permet à l'enseignant de concevoir et d'inclure des tests (quiz), qui peuvent intégrer les réponses correctes et/ou un feedback automatique.
- *Wiki* : une collection de pages web que n'importe qui peut créer ou modifier.

Avec l'option activités « activities » nous avons crée l'activité « rating ».

Les ressources constituent le contenu pédagogique de cours et peuvent se présenter sous plusieurs formes : un dossier, un fichier (word, excel, powerpoint, etc) déposé dans la zone de dépôt du cours, accessible par la fonctionnalité « Fichiers » du bloc administration (documents PDF, Word, Powerpoint, animations Flash, séquences vidéo, sons, etc ...). Cette zone de stockage est partagée par tous les enseignants du cours et inaccessible aux étudiants. Nous pouvons créer un dossier, déposer, supprimer, déplacer des fichiers, ou créer un archive compactée du cours.

- **Gestion des accès :** A chaque utilisateur (admin, enseignant ou étudiant, etc…) nous avons attribué des droits soit pour accéder au cours, ajouter des cours, faire un rating, poser des questions, etc.

Les figures suivantes présentent les différentes interfaces des activités de l'administrateur.

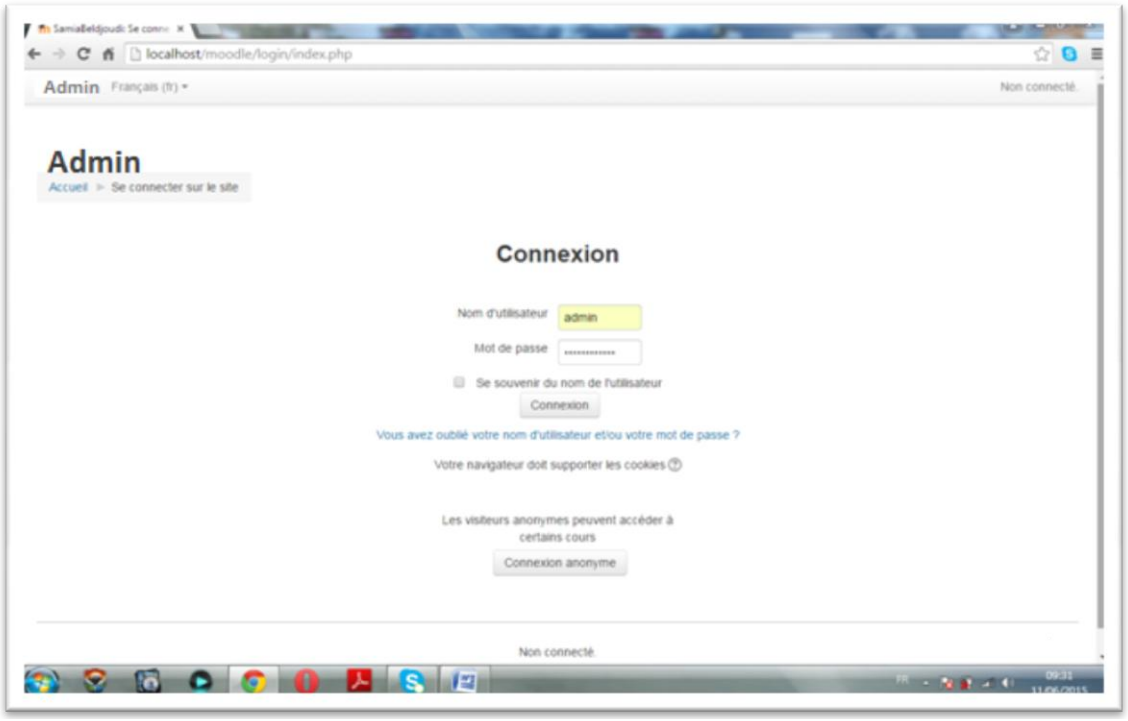


Figure 4.25 : Connexion de l'administrateur

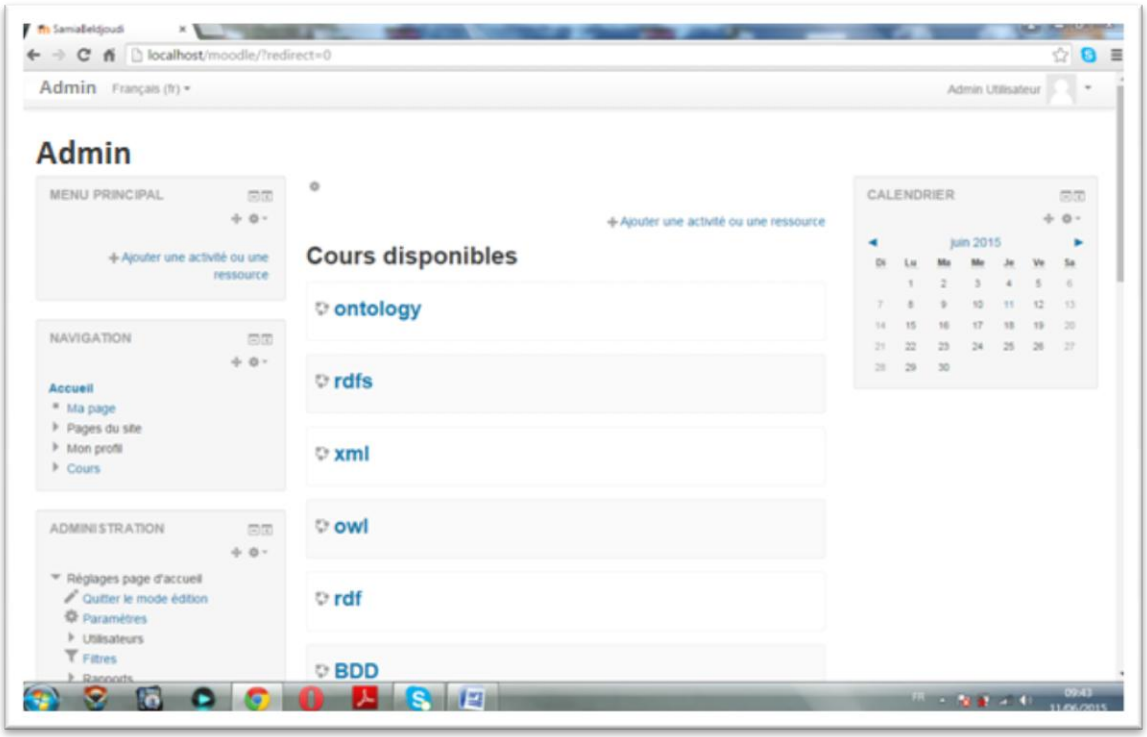


Figure 4.26 : Interface pour l'administrateur

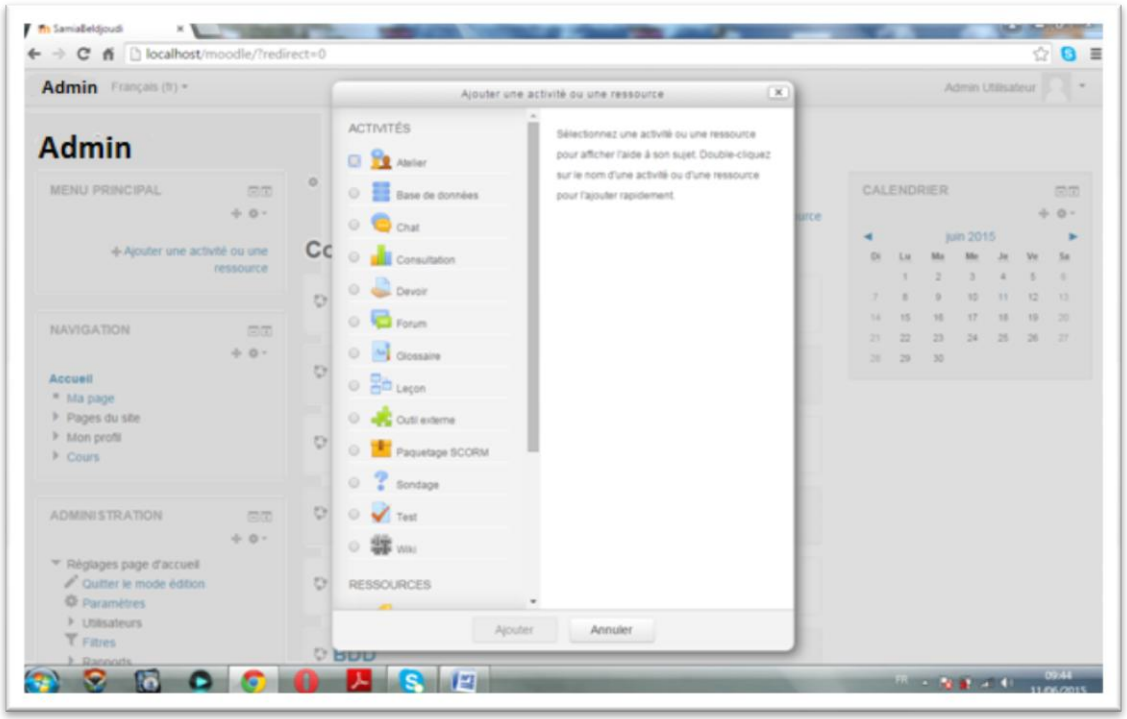


Figure 4.27 : Ajout d'une activité ou d'une ressource

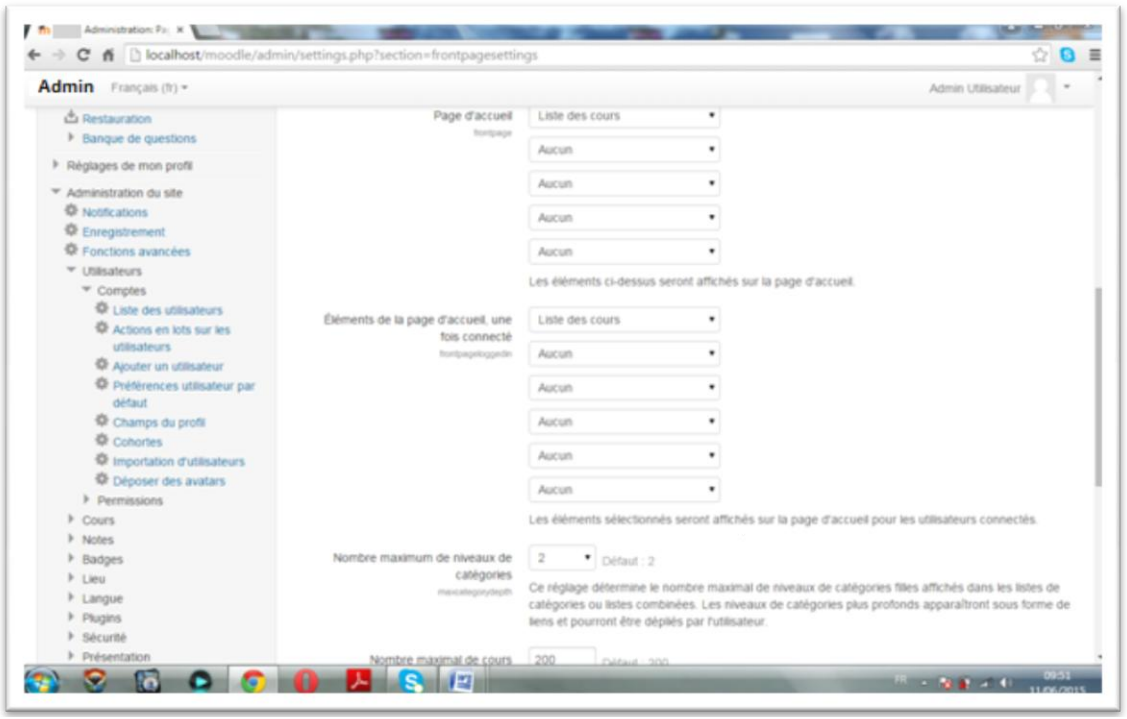


Figure 4.28 : Ajout d'une activité

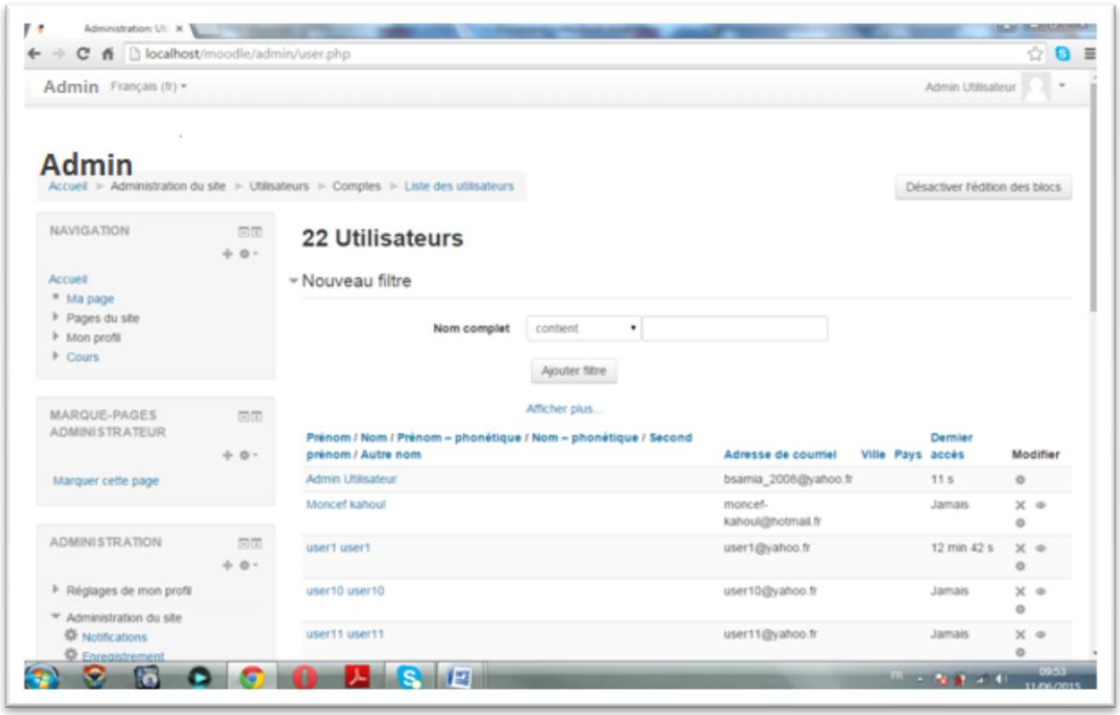


Figure 4.29 : Gestion des utilisateurs

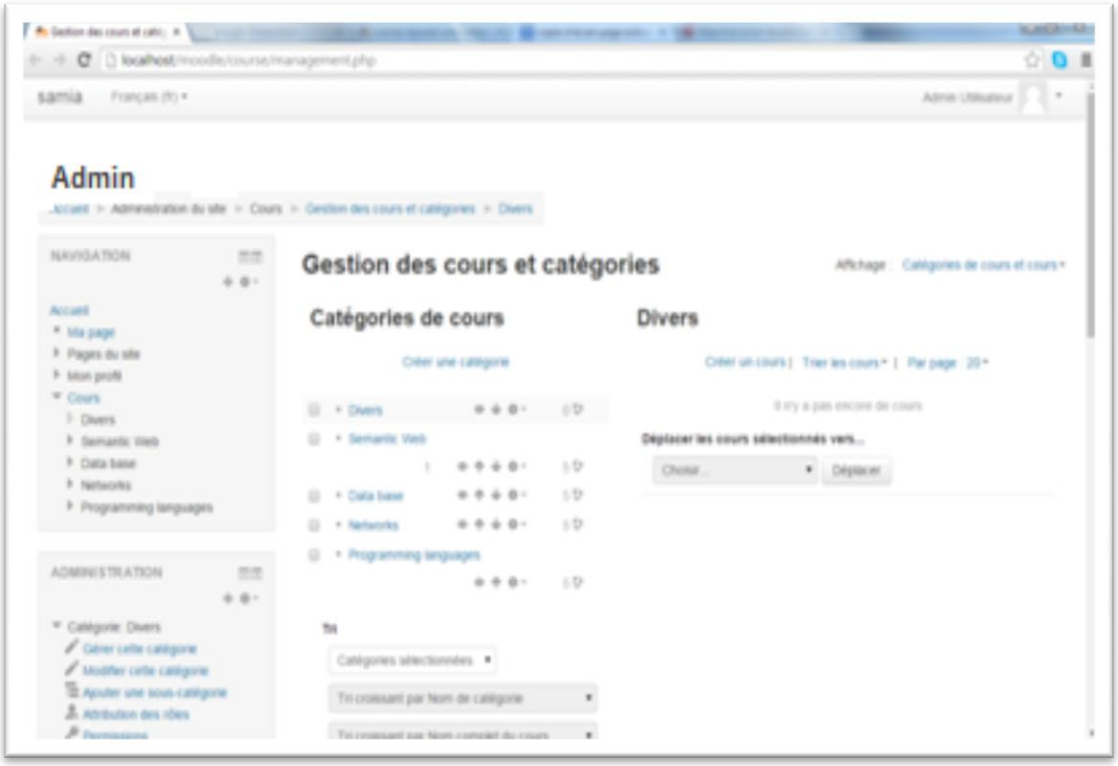


Figure 4.30 : Gestion des cours et catégories

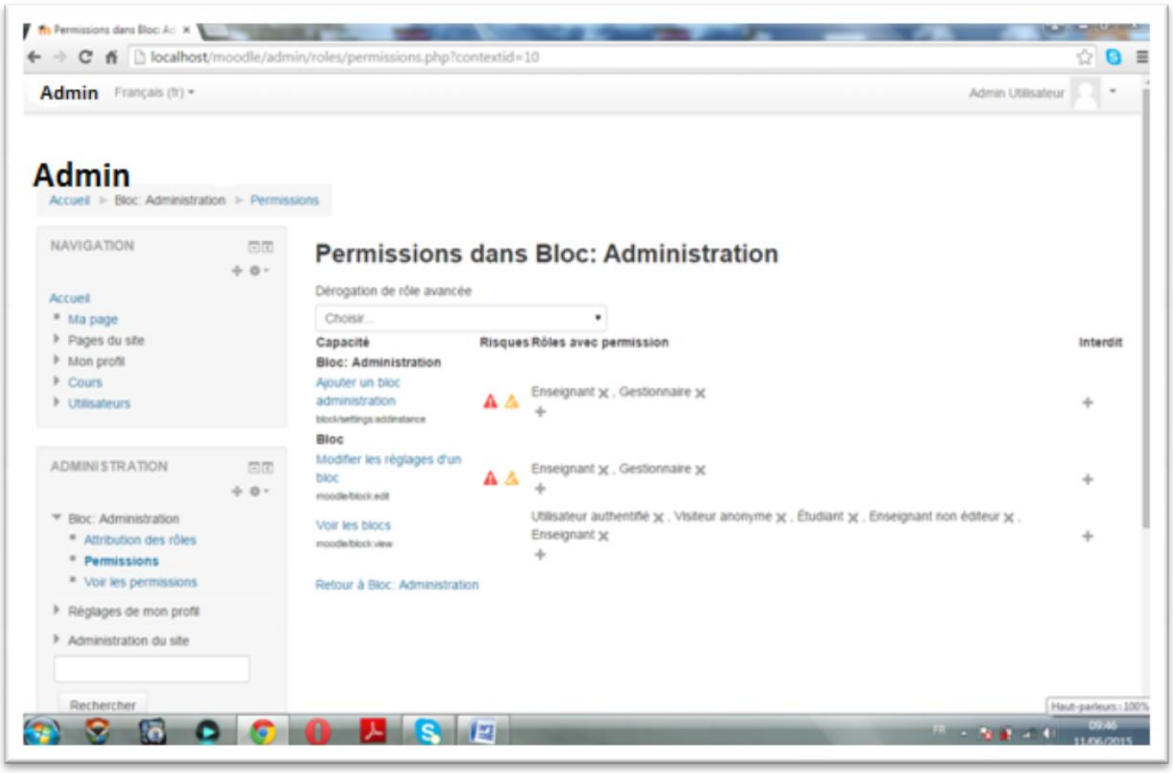


Figure 4.31 : Permission au bloc Administration

4.8.2.2.Interfaces Utilisateur (Apprenant)

Lors de sa première utilisation du système, un utilisateur doit tout d'abord s'inscrire pour créer un compte, en introduisant un nom d'utilisateur, un mot de passe et son adresse électronique. Ensuite, à chaque fois qu'il veut accéder au système, l'utilisateur se connecte en entrant son nom d'utilisateur et son mot de passe. Une liste des cours disponibles dans le système sera par la suite affichée à l'utilisateur sous forme de catégorie, ou chaque catégorie contient un ensemble de cours (ressources). L'utilisateur choisit la catégorie de son intérêt (terme général) à laquelle appartient le cours qu'il cherche, dans notre exemple le cours Ontologie est choisi.

Pour chaque cours, l'utilisateur aura une option « Rating » qui lui permet d'accéder à l'interface d'évaluation du cours pour exprimer son avis à propos du cours sous forme de vote de 1 à 5.

Lorsque l'apprenant rejoint le système dans un premier temps, il cherche les cours disponibles dans la catégorie désirée (web sémantique, les langages de programmation ... etc.). L'apprenant montre le cours qu'il veut apprendre (par exemple: Owl, Java, C ++

... etc), ce terme général contient une liste de sujets (par titre), puis il consulte les sous-thèmes (cours et exercices constructifs du module algorithmique) et il fait une évaluation sur le cours appris.

Les différentes interfaces de l'utilisateur sont présentées ci-dessous.

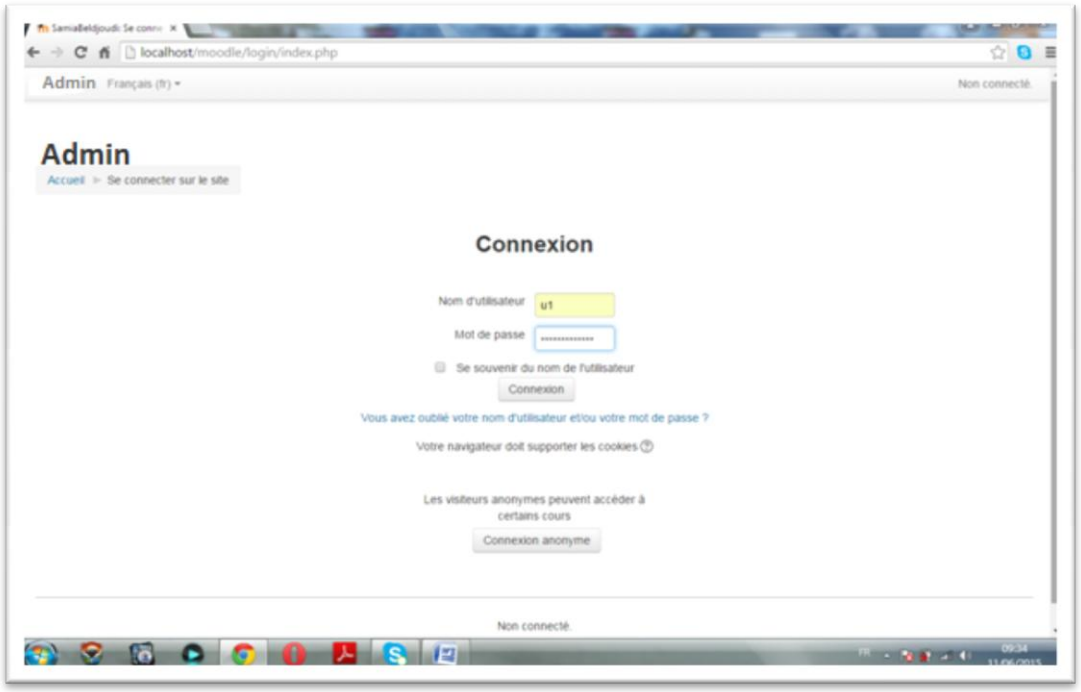


Figure 4.32 : Connexion utilisateur

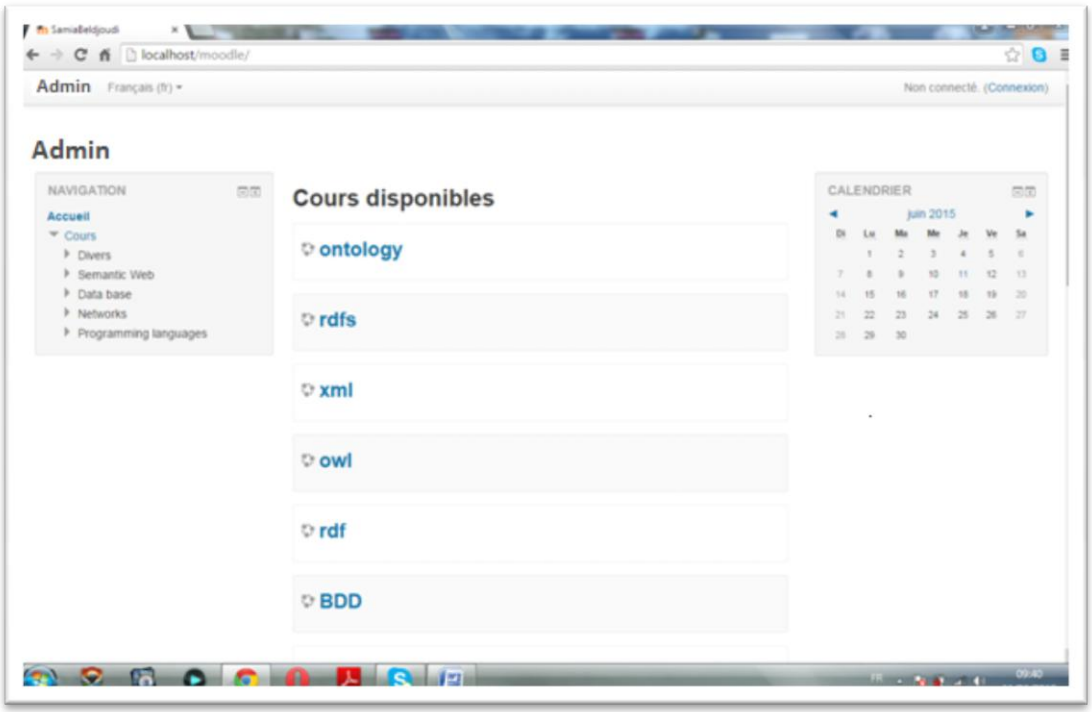


Figure 4.33 : Liste des cours disponibles pour l'apprenant

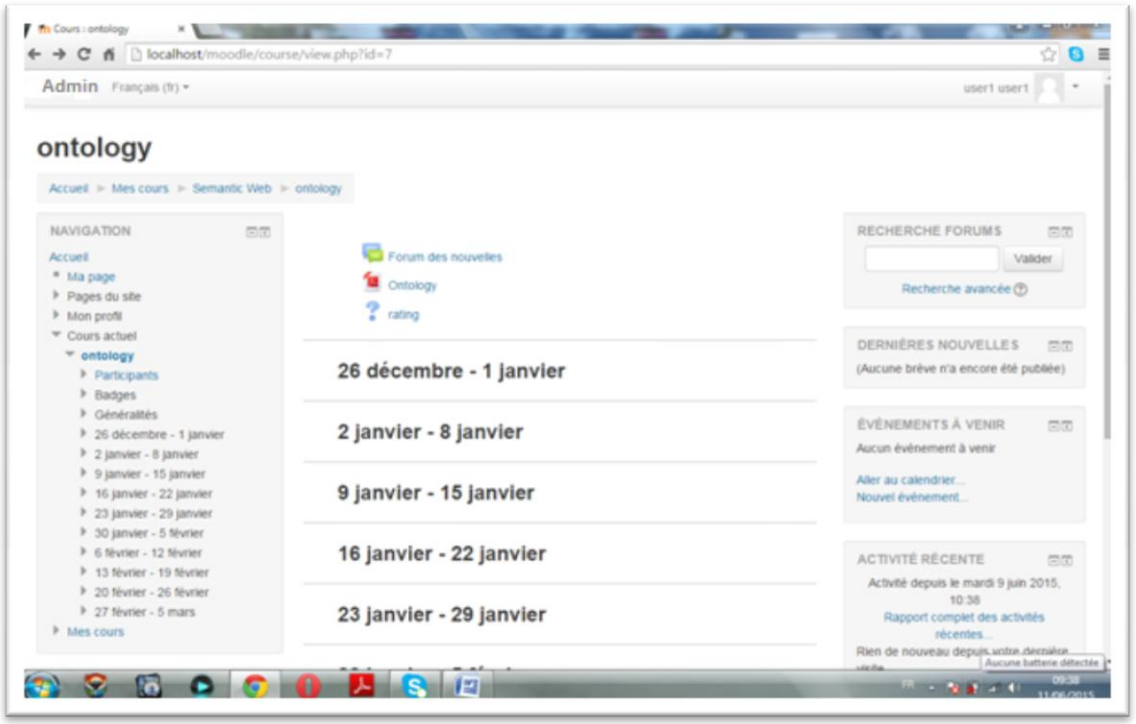
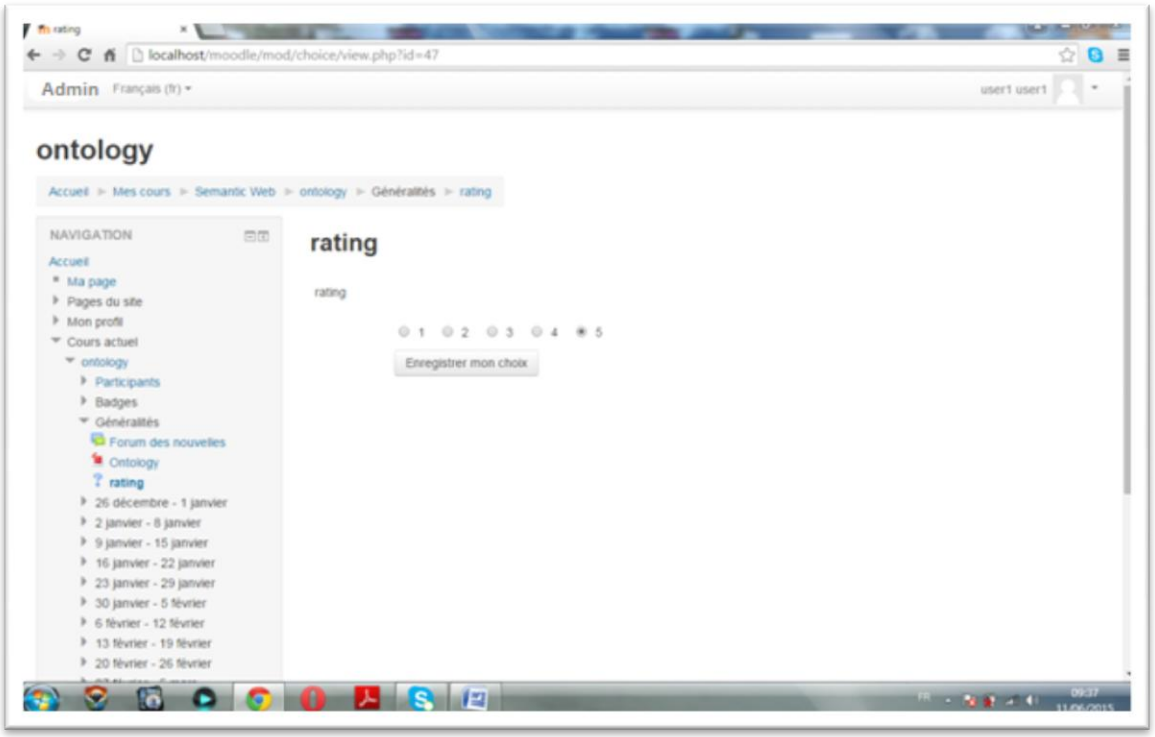


Figure 4.34 : Consultation du cours Ontologie



4.35: Evaluation du cours Ontologie

Pour quitter le système, l'administrateur et l'utilisateur doivent se déconnecter en choisissant l'option « Déconnexion » du menu « Mon compte » présentée dans la *Figure 4.36*.

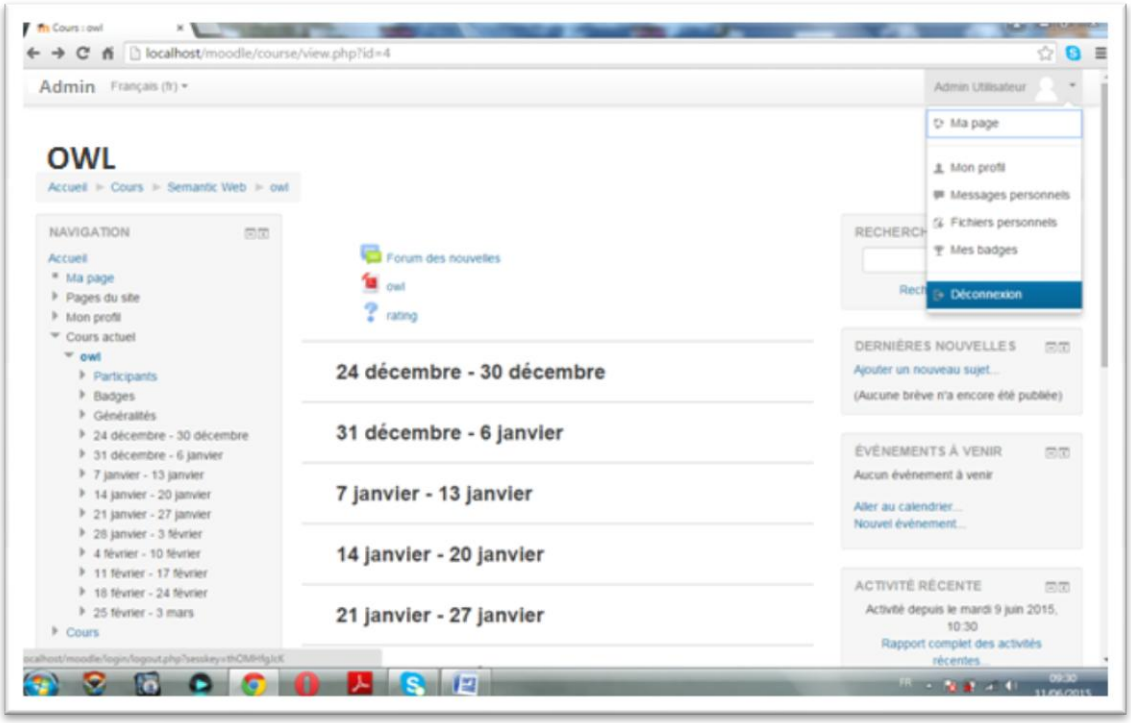


Figure 3.36 : Déconnexion

4.8.3. Évaluation et Résultats

L'application des différentes approches sur l'ensemble de données du *Moodle* a donné les résultats suivants présentés ci-dessous.

Les résultats de performance pour l'approche hybride basée flou sont présentés dans la *Figure 4.37*.

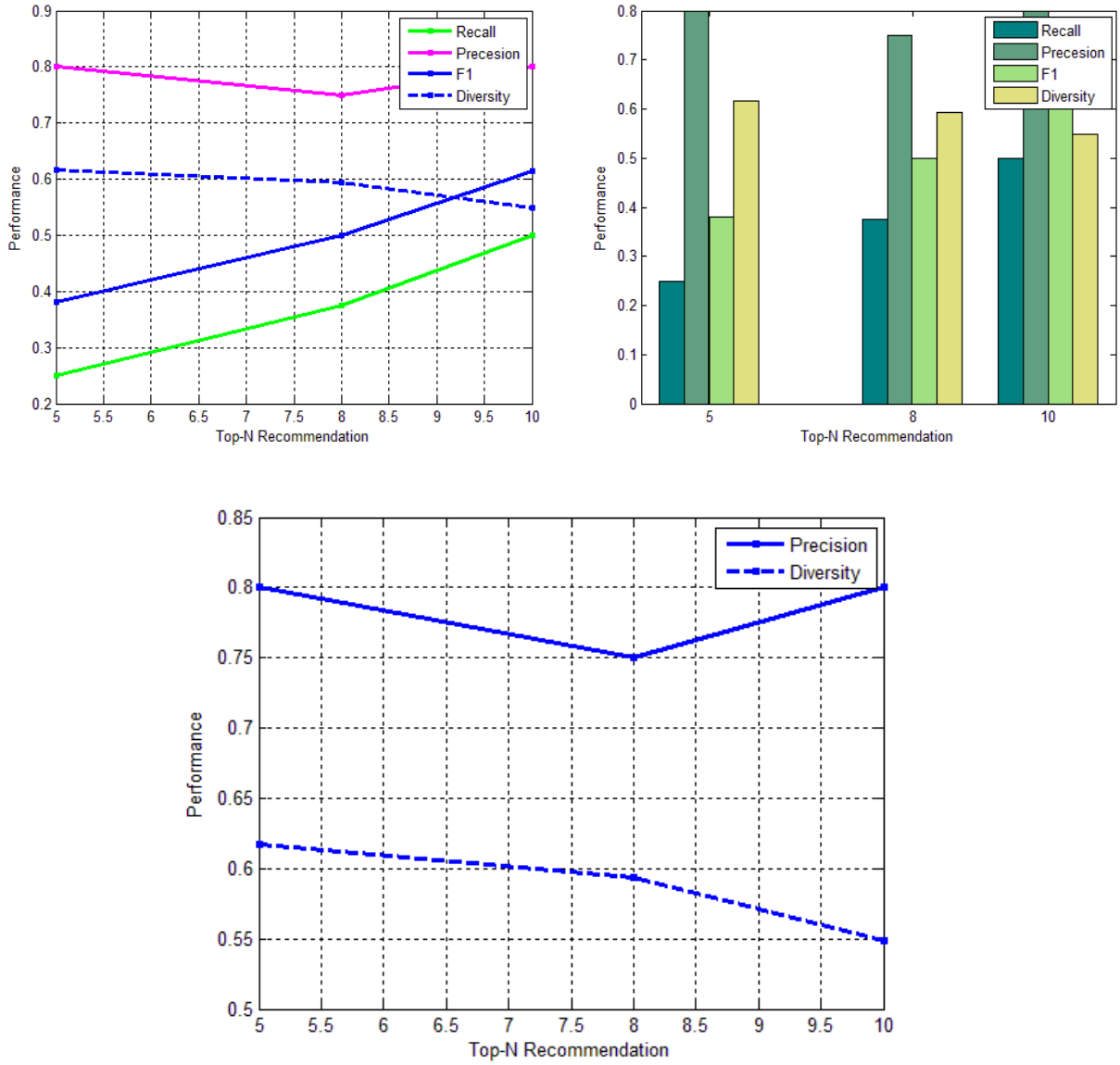


Figure 4.37 : Performance de la Recommandation basée flou sur le Dataset Moodle

La *Figure 4.37* montre la performance de l'approche de recommandation hybride diversifiée basée sur le flou. Où, la performance du système est élevée dans le de Top-5 items recommandés avec précision = 0,8 et diversité = 0,62. Ensuite, la précision se diminue avec Top-8 items jusqu'à 0,75 et elle augmente une autre fois pour le cas de 10 items recommandés et elle prend la valeur 0,8. Cependant, la diversité diminue légèrement avec le nombre des articles recommandés, où elle éteint les valeurs 0,6 et 0,55 pour Top-8 et Top-10, respectivement, ce qui prouve l'efficacité de l'algorithme pour générer des articles diversifiés et pertinents à l'apprenant actif.

Les résultats de l'application de la méthode de diversité basée sur le niveau de similarité sur l'ensemble de données de *Moodle* sont présentés dans la *Figure 4.38*.

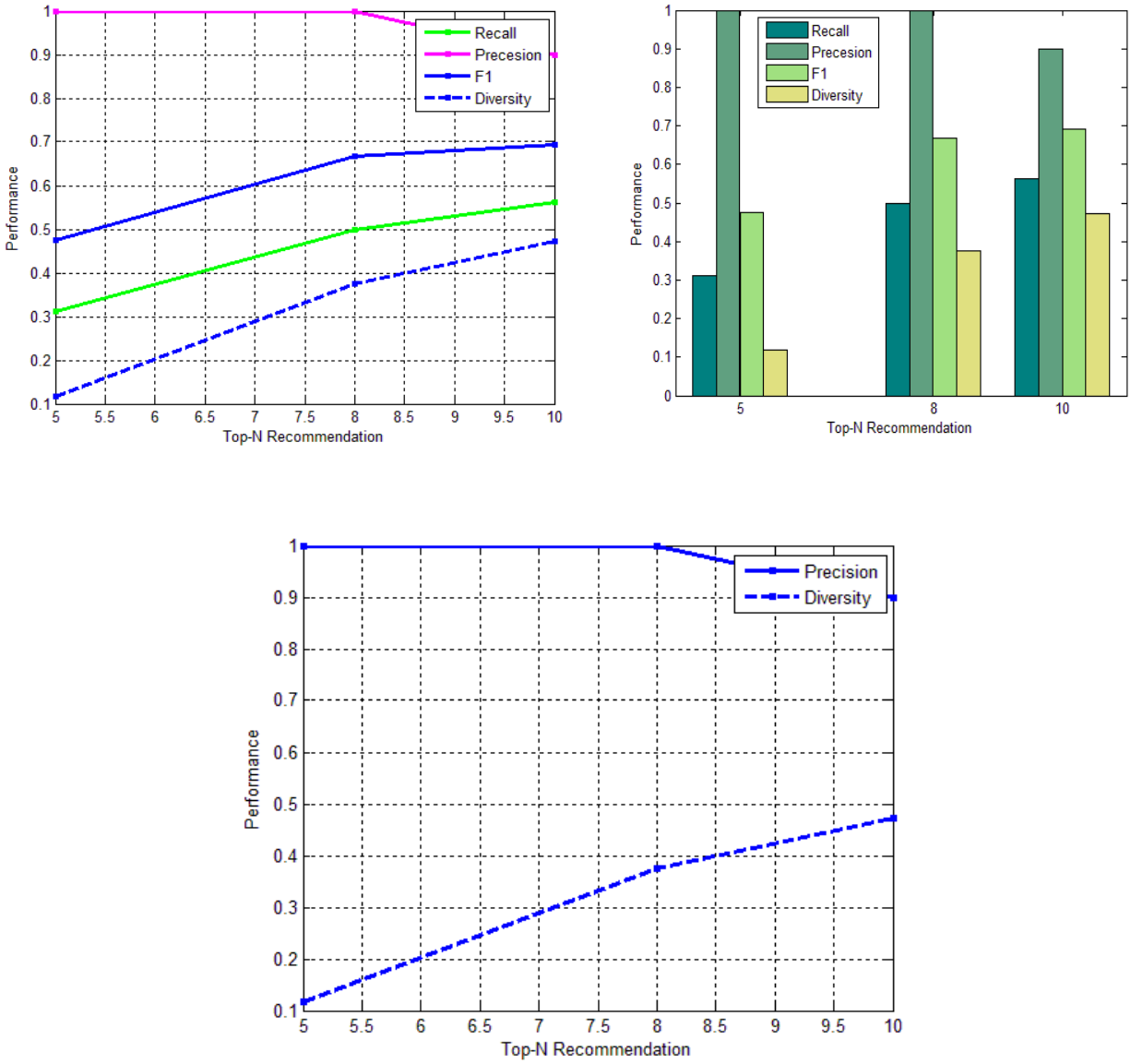


Figure 4.38. Performance de la Recommandation Diversifiée basée sur le Niveau de Similarité sur le Dataset Moodle

Dans la *Figure 4.38*, nous montrons l'évolution de la performance de la recommandation (rappel, précision, métrique F1) avec la diversité, et dans la partie en bas de la figure, nous avons considéré juste la précision et la diversité pour montrer clairement leur évolution avec le nombre des articles recommandés.

La *Figure 4.38* montre que la précision de l'algorithme était maximale dans le cas de Top-5 et Top-8 items recommandés et elle s'est dégradée légèrement avec une valeur de 0,9 avec Top-10 items. D'un autre côté, la diversité obtenue avec cette méthode augmente avec le nombre des items recommandés et elle atteint presque 0,5 pour Top-

10 items, ce qui prouve l'efficacité de la méthode à augmenter la diversité dans les listes de recommandation sans une grande perte en précision.

Dans la *Figure 4.39* nous comparons les deux algorithmes en termes de performance et de diversité.

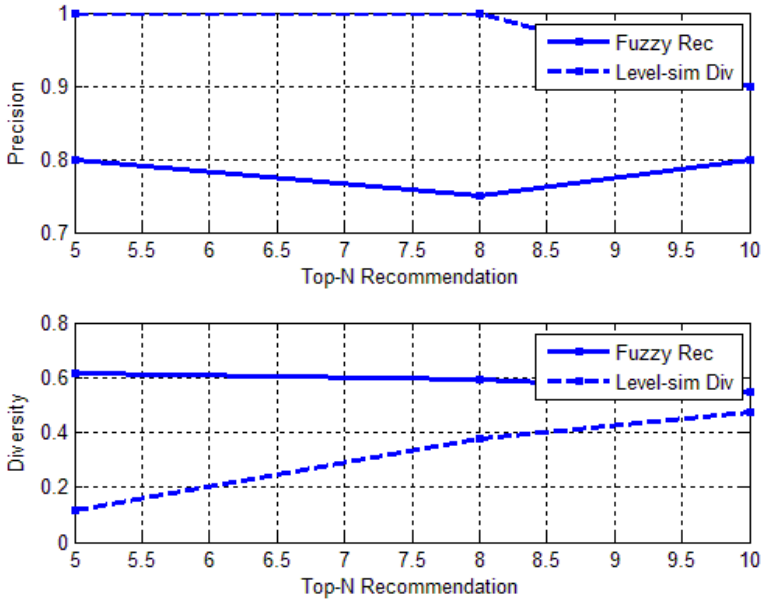


Figure 4.39. Recommandation basée Flou Vs. Recommandation diversifiée basée sur le niveau de similarité sur le Dataset Moodle

Les expériences sur le même ensemble de données ont montré que l'approche de diversité basée sur le niveau de similarité surpasse l'approche de recommandation basée flou en termes de précision, alors que la méthode basée flou à augmenter la diversité dans les listes de recommandation plus que la méthode de diversité basée sur le niveau de similarité.

Pour améliorer la précision et la diversité des recommandations en même temps, nous avons appliqué l'approche hybride qui fusionne les deux approches. Les résultats sont présentés dans la *Figure 4.40*.

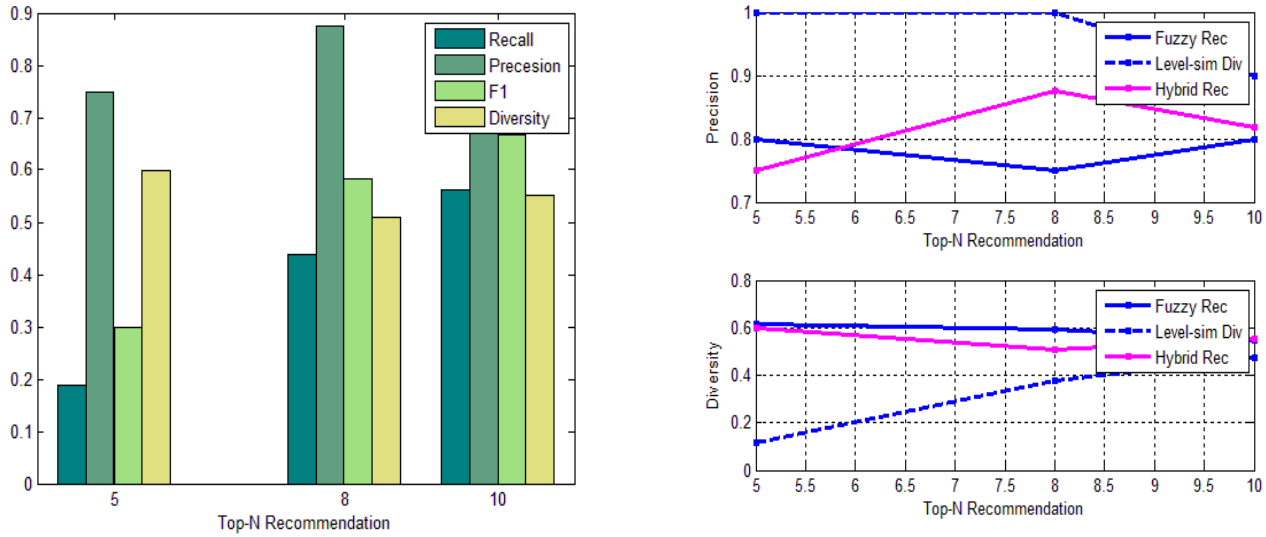


Figure 4.40: Performance de la nouvelle approche hybride sur le Dataset Moodle

Dans la sous-figure gauche de la *Figure 4.40*, nous présentons la performance générale de la nouvelle approche hybride sur l'ensemble de données *Moodle* (Précision, Rappel, F1 et Diversité), où il a été démontré que la précision, le rappel et la mesure F1 augmentent avec le nombre des items recommandés. Tandis que la diversité atteint sa plus grande valeur 0,6 dans le cas de Top-5 items, puis elle se dégrade dans les autres cas, mais elle prend une grande valeur avec Top-10 items qu'avec Top-8 items.

La sous-figure droite de la *Figure 4.40*, présente une comparaison avec l'approche de diversité basée sur le niveau de similarité et l'approche de recommandation basée flou, en termes de précision et de diversité où la précision de la nouvelle approche hybride est plus faible que les deux approches avec K = 5 articles et elle augmente ensuite avec K = 8 et K = 10 articles, respectivement, et elle atteint une valeur moyenne comprise entre celles de l'approche basée sur le niveau de similarité et de l'approche basée flou dans les deux recommandations (Top-8 et Top-10 articles).

Outre, la diversité dans la liste de recommandation générée par la nouvelle approche de recommandation hybride est supérieure à celle de l'approche basée sur le niveau de similarité, et plus proche de la diversité fournie par la méthode basée flou avec $K = 5$ et $K = 8$ articles, respectivement. Dans le cas de la recommandation des Top-10 articles, la diversité générée par l'approche hybride est égale à celle de l'approche basée flou et plus élevée que l'approche de diversité basée sur le niveau de similarité. Cette égalité

revient à la diminution de la diversité fournie par la méthode basée flou et l'augmentation de la diversité fournie par la méthode basée sur le niveau de similarité.

4.8.4. Discussion

A partir de ces résultats présentés ci-dessus, nous avons vu que la recommandation basée flou a augmenté la diversité des articles recommandés alors que la méthode basée sur le niveau de similarité a augmenté la précision dans les listes de recommandation. Par conséquent, la recommandation hybride a donné une précision proche de celle de la méthode basée sur le niveau de similarité et une diversité proche de celle de la méthode basée flou. Donc, nous confirmons que la nouvelle approche hybride est efficace et faisable du faite qu'elle a donné une meilleure performance que les deux approches proposées en termes des deux critères: précision et diversité dans la même recommandation, et nous avons observé clairement qu'elle a amélioré la précision et la diversité.

4.9. Conclusion

Nous avons présenté dans ce chapitre une étude expérimentale des différentes propositions, afin de valider leurs performances et prouver leurs efficacités de générer des recommandations ayant du contenu diversifié et pertinent qui correspond aux différents choix et préférences de l'utilisateur. Les résultats étaient satisfaisants et répondent aux deux critères d'évaluation, précision et diversité, car elles génèrent des recommandations diversifiées avec amélioration de la précision par rapport à la recommandation traditionnelle. Néanmoins, ces résultats restent en besoin d'amélioration et de perfectionnement ce qui ouvre la voie vers quelques perspectives présentées dans la conclusion générale.

CONCLUSION GENERALE

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 leur besoins. Comme tout autre domaine, les systèmes de recommandation ont leur problèmes qui réduisent leur efficacité, où l'un de ces problèmes est le problème de la stabilité de ces systèmes par rapport au profil dynamique de l'utilisateur, ce qui conduit à une stabilisation dans les contenus recommandés aux utilisateurs et par la suite à des recommandations similaires conduisant à ennuyer les utilisateurs. Du moment où, la diversité dans les listes de recommandation est devenue un critère important pour juger la qualité des recommandations et la faisabilité du système en termes de satisfaction des utilisateurs.

L'objectif de notre travail, dans le domaine des systèmes de recommandation, était la conception de nouvelles approches de recommandation hybrides qui permettent la génération de recommandation diversifiée aux utilisateurs, en prenant en compte le maintien de la précision des recommandations, afin de rendre le système plus adaptatif au profil dynamique des utilisateurs, tout en considérant l'incertitude et l'imprécision dans leurs préférences et goûts.

Dans une première contribution, une nouvelle approche hybride basée sur le flou a été présentée pour combler les lacunes des systèmes de recommandation telles que la clairsemé et l'évolutivité de données, l'incertitude et l'imprécision dans les préférences des utilisateurs et la stabilité des contenus délivrés.

L'incertitude et l'ambiguïté dans les préférences des utilisateurs ont été prises en compte en affectant les utilisateurs à différents groupes les plus proches en utilisant un nouveau schème du filtrage collaboratif hybride flou. Le nouveau schème utilise une hybridation entre la méthode FCM et la méthode de factorisation en matrice non négative. Ensuite, la sélection des voisins pertinents se fait en choisissant les plus proches voisins flous à partir des plus proches clusters flous en appliquant les deux algorithmes FKNP et FKNN où l'algorithme de sélection des voisins utilise une nouvelle mesure similarité. La nouvelle mesure de similarité calcule la similarité entre l'utilisateur actif et les utilisateurs appartenant aux plus proches clusters en utilisant les degrés d'appartenance aux groupes trouvés par les résultats de factorisation.

Le problème de la clairsemé de données a été adressé par une nouvelle méthodologie qui consiste à précéder la phase de recommandation par une phase de prédiction. Cette dernière, combine les deux prédictions basée FC flou et basée contenu, afin d'améliorer la précision des prédictions en incluant le contenu des items dans le processus de recommandation.

De plus, la combinaison entre la prédiction basée sur le contenu avec celle basée sur le FC, a amélioré énormément la précision de la prédiction basée sur le FC flou. Le processus de sélection du voisinage flou utilisant une nouvelle mesure de similarité a réduit énormément le temps de calcul et de la sélection des voisins, ce qui augmente la performance générale du système. De plus, l'approche a dépassé la recommandation ordinaire en termes de précision et de diversité du contenu qui convient au différents besoins et préférences de l'utilisateur.

Afin de répondre aux divers besoins et choix de l'utilisateur, le nouveau schème de recommandation proposé vise à incrémenter la diversité dans les listes de recommandation en suggérant aux utilisateurs différents items à partir de leurs clusters les plus proches selon leurs degrés d'appartenance. D'où la diversité a été augmentée dans les listes de recommandation par rapport à la recommandation ordinaire.

Les expérimentations ont prouvé l'efficacité de l'approche en recommandant des items qui correspondent aux différents choix et préférences des utilisateurs, où le nouveau schème de recommandation basé sur la prédiction a dépassé le processus de recommandation ordinaire en termes de précision à cause de la réduction de la clairsemé de données.

Dans une deuxième proposition nous avons voulu augmenter la diversité encore plus, mais en incorporant la notion de la nouveauté pour offrir des items divers non vus déjà par l'utilisateur ni son voisinage. La nouvelle approche utilise une formule basée sur le niveau de similarité entre l'utilisateur et les items divers pour calculer leurs degrés d'intérêt, afin de juger leur pertinence pour l'utilisateur. Le processus de diversification basée sur le niveau de similarité a été fusionné avec un processus du FC ordinaire, pour générer une liste avec des items similaires et divers en même temps.

Les expériences ont prouvé que la nouvelle méthode augmente la diversité dans les listes de recommandation par rapport à la première approche, mais avec moins de performance. De plus, elle a donné une performance proche à celles des méthodes de diversification citées dans la littérature, où elle les a surpassées en termes de diversité dans des cas, et en performance dans d'autres. Ainsi, la nouvelle méthode est plus avantageuse que les autres méthodes du faite qu'elle génère des items divers qui sont pertinents pour l'utilisateur contrairement aux autres méthodes.

Une troisième méthode hybride, qui combine entre le processus du FC basé flou et la méthode de diversification basée sur le niveau de similarité, est présentée afin de profiter de leurs avantages et améliorer la performance générale du système, en augmentant la diversité dans les listes de recommandation tout en conservant leurs précisions, et ce qui a été prouvé à travers les expérimentations.

A la fin, nous pouvons dire que les différentes approches proposées dans cette thèse ont achevé les buts visés pour lesquels ont été conçues, en recommandant des items divers et pertinents pour les utilisateurs qui correspondent à leurs choix et intérêts, et de rendre le système plus adaptatif à leurs besoins, et d'aboutir la satisfaction des utilisateurs ce qui est le but principal de tout système de recommandation. Cependant, nos approches restent un peu restreintes ce qui ouvre la voie vers un nombre de perspectives afin de les perfectionner encore plus.

L'approche hybride basée flou nécessite beaucoup de paramètres pour la fusion des différentes prédictions et méthodes présentées que nous avons trouvés expérimentalement, d'où ils peuvent être calculés avec les techniques de l'IA et les méthodes de l'apprentissage automatique, pour en trouver les bonnes valeurs conduisant à de bonnes performances.

C'est vrai que la diversité des contenus générés à l'utilisateur constitue un critère important pour valider l'importance et la pertinence des recommandations, mais les autres nouveaux critères d'évaluation des systèmes de recommandation (cités dans le Chapitre 1, section 1.11) jouent un rôle crucial pour atteindre la satisfaction des utilisateurs ; c'est la raison pour laquelle nous cherchons à les prendre en considération dans les travaux prochains.

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 des approches proposées dans d'autres domaines utilisant les systèmes de recommandation outre que la recommandation des films, comme la recommandation dans les réseaux sociaux pour en tester leur validité.

RÉFÉRENCES BIBLIOGRAPHIQUES

[ABB et al., 13] Abbar, S., Amer-Yahia, S., Indyk, P., & Mahabadi, S. (2013, May). Real-time recommendation of diverse related articles. *In Proc. of the 22nd Inter.Conf. on WWW* (pp. 1-12).

[ABD et HAI, 98] Abdul-Rahman, A., & Hailes, S. (1998, January). A distributed trust model. In*Proceedings of the 1997 workshop on New security paradigms* (pp. 48-60). ACM.

[ADO et al., 07] Adomavicius, G., Kamireddy, S., & Kwon, Y. (2007). Towards more confident recommendations: Improving recommender systems using filtering approach based on rating variance. In *Proc. of the 17th Workshop on Information Technology and Systems*.

[ADO et KWO, 09] Adomavicius, G., & Kwon, Y. (2009, December). Toward more diverse recommendations: Item re-ranking methods for recommender systems. *In Workshop on ITS.*

[ADO et KWO, 11] Adomavicius, G., & Kwon, Y. (2011, October). Maximizing aggregate recommendation diversity: A graph-theoretic approach. *In Proc. of the 1st Inter. Workshop on Novelty and Diversity in Recommender Systems (DiveRS 2011)* (pp. 3-10).

[ADO et KWO, 12] Adomavicius, G., & Kwon, Y. (2012). Improving aggregate recommendation diversity using ranking-based techniques *IEEE Trans. on KDE,* 24(5), 896-911.

[ADO et TUZ, 05] Adomavicius G. & Tuzhilin A., (2005). Towards the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. In *IEEE Trans. on KDE*, 17(6), pp. 734-749.

[AIM et al., 06] Aimeur E., Brassard G., Fernandez J. M., Mani Onana F. S., (2006). Privacy-preserving demographic filtering, *Proc. of the ACM Sym. on Applied computing*, pp. 872 – 878.

[ALE et al., 15] Alexandridis, G., Siolas, G., & Stafylopatis, A. (2015). Accuracy Versus Novelty and Diversity in Recommender Systems: A Nonuniform Random Walk Approach. *In Recommendation and Search in Social Networks*, pp. 41-57. Springer International Publishing.

[ALL 96] Allan J. (1996) Incremental Relevance Feedback for Information Retrieval. *Proc. of SIGIR'96 Zurich, Switzerland, ACM Press*, pp. 270-278.

[ARD et al., 03] L. Ardissono, C. Gena, P. Torasso, F. Bellifemine, A. Chiarotto, A. Difino, B. Negro. (2003). Personalized Recommendation of TV Programs. *LN. in AI n. 2829. Advances in AI, Italy,* pp. 474-486.

[AYT et KAR 14] Aytekin, T., & Karakaya, M. Ö. (2014). Clustering-based diversity improvement in top-N recommendation. *J. of Intelligent Information Systems*, 42(1), pp.1-18.

[BAL et RIC, 08] Baltrunas L. and Ricci F., (2008). Locally Adaptive Neighborhood Selection for Collaborative Filtering Recommendations. *Proc. 5th Inter. Conf. AH 2008 , Germany*, 5149, pp. 22-31.

[BAL et SHO, 97] Balabanovic M. and Shoham. Fab Y., (1997). Content-based, Collaborative recommendation. *Communications of the ACM*, Vol. 40, No. 3, pp.66-72.

[BEL et al. 07] Bell R. M., Koren Y. and Volinsky C., (2007). Modeling Relationships at Multiple Scales to Improve Accuracy of Large Recommender Systems. In *Proc. of the 13th ACM SIGKDD, Inter. Conf. On Knowledge.*

[BEL et CRO, 92] Belkin N.J., Croft W.B. (1992, December). Information filtering and information retrieval: two sides of the same colin? , *Communication of the ACM*, 5(12), pp. 29-38.

[BEL et KOR, 07] Bell, R. M., & Koren, Y. (2007, October). Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In the *7th IEEE ICDM 2007.* (pp. 43-52). IEEE.

[BER 02] Berti-Equille L, (2002). Annotaion et recommandation collaboratives des documents selon leur qualité, *RSTI série ISI-NIS Recherche de filtrage d'information*, 7(12), pp. 125-155.
[BEZ, 73] Bezdek J. C., (1973). Fuzzy mathematics in pattern classification. *ph_D. dissertation.* Cornell University Ithaca, NY.
[BHA 99] Bhargava, H. K., Sridhar, S. and Herrick, C. (1999). Beyond Spreadsheets: Tools for Building Decision Support Systems. *IEEE Computer*, 32(3), pp. 31-39.
[BIL et PAZ, 00] Billsus D. & Pazzani M., (2000). User Modeling for Adaptive News Access. In *User-Modeling and User-Adapted Interaction*, Vol.10, No. 2-3, pp 147-180.
[BIL et PAZ, 99] Billsus, D., and Pazzani, M. J. (1999). A Hybrid User Model for News Classification, *In Kay J., User Modeling Proc. of the 7th Inter. Conf., Springer-Verlag*, pp. 99-108.
[BOB et al., 10] Bobadilla, J., Serradilla, F., & Bernal, J. (2010). A new collaborative filtering metric that improves the behavior of recommender systems. *Knowledge-Based System*s, 23(6), pp. 520-528.
[BOB et al., 11] Bobadilla, J., Hernando, A., Ortega, F., & Bernal, J. (2011). A framework for collaborative filtering recommender systems. *ESA*, 38(12), pp. 14609-14623.
[BOB et al., 11a] Bobadilla, J., Ortega, F., Hernando, A., &Alcalá, J. (2011). Improving collaborative filtering recommender system results and performance using genetic algorithms. *KBS*, 24(8), pp.1310-1316.
[BOB et al., 12a] Bobadilla, J., Hernando, A., Ortega, F., & Gutiérrez, A. (2012). Collaborative filtering based on significances. *Information Sciences*, 185(1), 1-17.
[BOB et al., 12b] Bobadilla, J., Ortega, F., & Hernando, A. (2012). A collaborative filtering similarity measure based on singularities. *Information Processing & Management*, 48(2), 204-217.
[BOB et al., 13] Bobadilla, J., Ortega, F., Hernando, A., & Gutiérrez, A. (2013). Recommender systems survey. *In Knowledge-Based Systems*, Vol. 46, 109-132.
[BOI et al., 11] Boim, R., Milo, T., & Novgorodov, S. (2011, October). Diversification and refinement in collaborative filtering recommender. *In Proc. of the 20th ACM Inter. Conf. on IKM*, pp. 739-744. ACM.
[BON et al., 06] Bonhard, P., Harries, C., McCarthy, J., & Sasse, M. A. (2006, April). Accounting for taste: using profile similarity to improve recommender systems. In *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems* (pp. 1057-1066). ACM.
[BOT 03] Bottraud, J. C., Bisson, G., & Bruandet, M. F. (2003, August). An adaptive information research personal assistant. In *proc. of the workshop on AIIAMC*.
[BOU 05] Bouzghoub M., Kostadinov D., (2005). Personnalisation de l'information : Aperçu de l'état de l'art et définition d'un modèle flexible de profils, *CORIA'05*, 5, pp. 201-218, France.
[BOU 99] Boughanem, M., Chrisment, C., & Soulé-Dupuy, C. (1999). Query modification based on relevance back-propagation in an ad hoc environment. *IP&M*, *35*(2), 121-139.
[BOU et al., 13] Bouneffouf, D., Bouzeghoub, A., & Ganarski, A. L. (2013, January). Risk-aware recommender systems. In *Neural Information Processing* (pp. 57-65). Springer Berlin Heidelberg.
[BRE et al., 98] Breese J. S., Heckerman D., and Kadie C., (1998). Empirical analysis of predictive algorithms for collaborative filtering, In *Proc. Of the 14th Conf. on UAI*, pp. 43–52.
[BRI et KEL, 02] Bridge, D., & Kelleher, J. (2002). Experiments in sparsity reduction: Using clustering in collaborative recommenders. *In AICS*, pp. 144-149. Springer
[BRI et PAG, 98] Brin S., Page L. (1998). The anatomy of a large-scale hyper textual Web search engine, *Computer Networks and ISDN Systems*, 30(1-7), pp 107-117.

[BUR 02] Burke R., (2002): Hybrid recommender systems: Survey and experiments. In *User Modeling and User Adapted Interaction*, 12(4), pp. 331-370.
[BUR 96] Burke, R., Hammond, K. & Cooper, E. (1996). Knowledge-based navigation of complex information spaces. *In Proc. of the 13th Nat. Conf. on AI*, pp. 462-468. Menlo Park, CA: AAAI Press.
[BUR 97] Burke, R., Hammond, K., and Young, B. (1997). The FindMe Approach to Assisted Browsing. *IEEE Expert*, 12(4), pp. 32-40.
[BUR et al., 06] Burke, R., Mobasher, B., Williams, C., & Bhaumik, R. (2006). Detecting profile injection attacks in collaborative recommender systems. *In the 3rd IEEE Inter. Conf. on E-Commerce Technology*, pp. 23.
[BUR et al., 11] Burke, R., O'Mahony, M. P., & Hurley, N. J. (2011).). Robust collaborative recommendation. In *Recommender Systems Handbook* (pp. 805-835). Springer US.
[**CAN 02]** Canny J., (2002). Collaborative filtering with privacy via factor analysis. In *Proc. the 25th ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 238–245.
[CAS et al., 11] Castells, P., Vargas, S., & Wang, J. (2011). Novelty and Diversity Metrics for Recommender Systems: Choice, Discovery and Relevance. *Inter. Workshop on DDR at the 33rd ECIR.*
[CAS et al., 13] Castagnos, S., Brun, A., & Boyer, A. (2013, April). Utilité et perception de la diversité dans les systèmes de recommandation. *In CORIA'13, 10ème Conf. en Recherche d'Information et Applications*.
[CAT et DEN, 03] Berrut C. and Denos N., (2003). Filtrage collaboratif, in Assistance intelligente à la recherche d'informations, *Hermes - Lavoisier, chapter 8*, pp. 30.
[CEL et HER, 08] Celma, O., Herrera, P. (2008). A new approach to evaluating novel recommendations. *In Proc. of the ACM RecSys'2008, New York,* pp. 179–186.
[CHA et al., 08] Chandrasekaran K., Gauch S., Lakkaraju P., & Luong H. P., (2008). Concept-Based Document Recommendations for CiteSeer Authors. *In Proc. Of the 5th Inter. Conf.,* 5149, pp. 83-92.
[CHE et al., 09] Chen G., Wang F., Zhang C., (2009). Collaborative Filtering Using Orthogonal Nonnegative Matrix Tri-factorization. *JIPM*, Vol. 45.
[CHE et PU, 04] Chen, L., & Pu, P. (2004). *Survey of preference elicitation methods* (No. EP-REPORT-52659).
[CHI et al., 05] Chirita, P. A., Nejdl, W., & Zamfir, C. (2005, November). Preventing shilling attacks in online recommender systems. In *Proc. of the 7th Ann. ACM Inter. Workshop on WIDM*, pp. 67-74.
[CHO and SUH, 13] Choi, K., & Suh, Y. (2013). A new similarity function for selecting neighbors for each target item in collaborative filtering. *JKBS*, 37, pp.146-153.
[CHO et al., 02] Cho Y. H., Kim S.H., Kim J.K. (2002). A personalized recommender system based on web usage mining and decision tree induction, *ESA*, 23(3), pp. 329-342.
[CHR et al., 07] Christakou, C., Vrettos, S., & Stafylopatis, A. (2007). A hybrid movie recommender system based on neural networks. *Inter. Journal on Artificial Intelligence Tools*, 16(05), 771-792.
[CHU et al., 04], Chu M., Diele F, Plemmons R, Ragni S (2004) Optimality, computation, and interpretations of nonnegative matrix factorizations. *SIAM J Matrix Anal*, Vol. 4, pp.8030.
[CLA et al., 99] Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D. and Sartin, M., (1999). Combining Content-Based and Collaborative Filters in an Online Newspaper. *SIGIR '99 Workshop on Recommender Systems: Algorithms and Evaluation*. Berkeley, CA.
[CLE et al., 13] Cleger-Tamayo, S., Fernández-Luna, J. M., Huete, J. F., & Tintarev, N. (2013). Being confident about the quality of the predictions in recommender systems. In *Advances in Information Retrieval*, pp. 411-422. Springer Berlin Heidelberg.

[CRA et al., 08] Cramer, H., Evers, V., Ramlal, S., Van Someren, M., Rutledge, L., Stash, N., & Wielinga, B. (2008). The effects of transparency on trust in and acceptance of a content-based art recommender. *User Modeling and User-Adapted Interaction*, *18*(5), 455-496.
[CRO 93] Croft W. B. (1993). Knowledge-based and Statistical approaches to Text Retrieval. *IEEE EXPERT*, 8(2), pp. 8-12.
[CRU 99] Lainé-Cruzel, (1999). Filtrer une information exploitable, *Bulletin des bibliothèques de France*, Vol. 5, pp.60-65.
[DES and KAR 04] M. Deshpande and G. Karypis, (2004). Item-based top-n recommendation algorithms, *ACM Trans. Inf. Syst.*, 22(1), pp. 143–177.
[DES et KAR, 11] Desrosiers, C., & Karypis, G. (2011). A comprehensive survey of neighborhood-based recommendation methods. *In Recommender systems handbook*, pp. 107-144. Springer US.
[DIN et al., 05], Ding, C.; He, X.; and Simon, H. (2005).On the equivalence of nonnegative matrix factorization and spectral clustering. *In Proc. SIAM Data Mining Conf.*
[DIN et al., 06] Ding C., Li T., Peng W., and Park H., (2006). Orthogonal nonnegative matrix tri-factorizations for clustering. In *Proc. of the 12th ACM Conf. on KDDM*, pp. 126-135.
[FAN et LIU 03] Fang K. & Liu C.Y., (2003). Recommendation System Using Fuzzy C-Means Clustering. *Book of Information Technology and Organizations: Trends, Issus, Challenges and Solutions*, Vol.1.
[FRA et al., 06] Frankowski, D., Cosley, D., Sen, S., Terveen, L., & Riedl, J. (2006, August). You are what you say: privacy risks of public mentions. In *Proc. of the 29th Ann. Inter. ACM SIGIR Conf. on Research and development in information retrieval*, pp. 565-572. ACM.
[FRE et al., 03] Freund, Y., Iyer, R., Schapire, R. E., & Singer, Y. (2003). An efficient boosting algorithm for combining preferences. *The Journal of machine learning research*,*4*, pp. 933-969.
[GAN et GAR, 03] Ganesan P, Garcia-molina H., (2003). Expoiting Hierarchical Domain Structure to Compute Similarity, *ACM Transaction on System*, 21(1), pp 64-93.
[GE et al., 10] Ge, M., Delgado B. C., & Jannach, D. (2010). Beyond accuracy: evaluating recommender systems by coverage and serendipity. *In Proc. of the 4th ACM RecSys'10*, pp. 257-260.
[GEO et MER, 05] George, T., & Merugu, S. (2005, November). A scalable collaborative filtering framework based on co-clustering. *In Data Mining, 5th IEEE Inter. Conf. on*, pp. 4.
[GEO et TSA , 10] Georgiou, O., & Tsapatsoulis, N. (2010). The importance of similarity metrics for representative users identification in recommender systems. *In AIAI*, pp. 12-21. Springer.
[GHA et al., 13] Ghanghas, V., Rana, C., & Dhingra, S. (2013). Diversity in Recommender System. *In International Journal of Engineering Trends and Technology (IJETT)*, 4(6), pp 2344-2348.
[GOL 92] Goldberg D., Nicols D.,Oki B.M., and Terry D. (1992). Using collaborative filtering to weave an information tapestry, *Communication of the ACM*, 35(12), pp. 61-70.
[GOL et al., 01] Goldberg K., Roeder T., Gupta D., and Perkins C., (2001), Eigentaste, a constant time collaborative filtering algorithm. In *Information Retrieval Journal*, Vol. 4, No. (2), pp.133–151.
[GON, 10] Gong, S. (2010). A collaborative filtering recommendation algorithm based on user clustering and item clustering. *Journal of Software*, 5(7), pp. 745-752.
[GOO 99] Good N., Schafer J. B., Konstan J. A., Borchers A., Sarwar B., Herlocker J., Riedl J., (1999). Combining collaborative filtering with personal agents for better recommendations, *In Proc. of the 6th Nat. Conf. on AI, Orlando, USA*, pp. 439- 446.
[GRA 98] Grabtree, I. and Soltysiak, S. (1998). Identifying and Tracking Changing Interests. *International Journal of Digital Libraries*, Vol. 2, pp. 38-53.

[GU et al., 10] Gu Q., Zhou J., Ding C., (2010). Collaborative Filtering: Weighted Nonnegative Matrix Factorization Incorporating User and Item Graphs. *The 10th SIAM Inter. Conf. on Data Mining*, pp.199-210.
[GUR et al., 13] Gurini, D. F., Gasparetti, F., Micarelli, A., & Sansonetti, G. (2013). A Sentiment-Based Approach to Twitter User Recommendation. *In RSWeb@ RecSys*.
[HAM 89] Hammond, K. (1989). Case-based Planning: Viewing Planning as a Memory Task. *Boston, MA: Academic Press*.
[HER et al., 02] Herlocker, J., Konstan, J. A., & Riedl, J. (2002). An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms.*IR*, *5*(4), pp. 287-310.
[HER et al., 04] Herlocker J., Konstan J., Terveen L., and Riedl J., (2004). Evaluating Collaborative Filtering Recommender Systems. *ACM Transactions on Information Systems*, 22(1), pp. 5-53.
[HER et al., 99] Herlocker J. L., Konstan J. A., Borchers A., and Riedl J., (1999). An algorithmic framework for performing collaborative filtering. In *Proc. of the Ann. Inter. ACM SIGIR Conf. on Dev. and Inf. Ret., SIGIR*.
[HOF 04] Hofmann T., (2004). Latent semantic models for collaborative filtering. *ACM* Transaction *Information Systems*, 22(1), pp. 89–115.
[HOF et PUZ 99] Hofmann, T., & Puzicha, J. (1999, July). Latent class models for collaborative filtering. *In IJCAI*, Vol. 99, pp. 688-693.
[HOW et FOR, 08] Howe, A. E., & Forbes, R. D. (2008, October). Re-considering neighborhood-based collaborative filtering parameters in the context of new data. In *Proc. of the 17th ACM Conf. on Information and knowledge management*, pp. 1481-1482. ACM.
[HOY 02] Patrik O. Hoyer, (2002). Non-negative sparse coding. In Neural Networks for Signal Processing XII (Proc. IEEE Workshop on Neural Networks for Signal Processing), pp. 557–565.
[HOY 04] Patrik O. Hoyer, (2004). Non-negative matrix factorization with sparseness constraints. *Journal of Machine Learning Research*, 5, pp. 1457–1469.
[HU et al., 08] Hu, Y., Koren, Y., & Volinsky, C. (2008, December). Collaborative filtering for implicit feedback datasets. *In Proc.of the 8th IEEE Inter. Conf. On Data Minin. ICDM'08*pp. 263-272.
[HUA 11] Huang, S. L. (2011). Designing utility-based recommender systems for e-commerce: Evaluation of preference-elicitation methods. *ECRA*, *10*(4), 398-407.
[HUR et ZHA, 11] Hurley, N., & Zhang, M. (2011). Novelty and diversity in top-N recommendation--analysis and evaluation. *ACM Transactions on Internet Technology*, 10(4), pp.14.
[IAQ et al., 08] Iaquinta L., Gemmis M., Lops P., Semeraro G. (2008). Introducing serendipity in a content-based recommender system. *The 8th Inter. Conf. on Hybrid Intelligent Systems*, pp 168-174.
[JEO et CHO, 09] Jeong, B., Lee, J., & Cho, H. (2009). User credit-based collaborative filtering. *Expert Systems with Applications*, 36(3), 7309-7312.
[JIN et al., 03] Jin, R., Si, L., Zhai, C., & Callan, J. (2003). Collaborative filtering with decoupled models for preferences and ratings. In *Proc. of the 12th Inter. Conf. on IKM*. pp. 309-316. ACM.
[JIN et al., 04] R. Jin, J. Y. Chai, and L. Si, (2004). An automatic weighting scheme for collaborative filtering, in *Proc. of SIGIR*.
[JIN et ZHA, 03] Jin, R., Si, L., Zhai, C. (2003). Preference-based graphic models for collaborative filtering. *In Proc. Of the 19th Ann. Conf. on Uncertainty in AI (UAI-03)*, pp. 329–33.
[JON 10] Jones N., User Perceived Qualities and Acceptance of Recommender Systems, *Phd thesis, Ecole Polytechnique Fédérale De Lausanne*, 2010

[JON et PU, 08] Jones, N., & Pu, P. (2008). User Acceptance Issues in Music Recommender Systems. (*No. HCI-REPORT-2009-001*).
[KAL, 14] Kaleli, C. (2014). An entropy-based neighbor selection approach for collaborative filtering. *Knowledge-Based Systems*, Vol. 56, 273-280.
[KEL et al., 85] Keller J. M., Gray M. R., Givens J. A., JR., (1985). A Fuzzy Nearest Neighbour Algorithm. In *IEEE Transaction on Systems, Man and Cybernetics*, Vol.15, pp. 580-585.
[KEN 48] Kendall, M. G. (1948). A new measure of rank correlation. Biometrika, pp. 81-93.
[KIM and AHN 08] Kim K. and Ahn J., (2008). A recommender system using GA K-means clustering in an online shopping market. *Expert systems with applications*, *34*(2), 1200-1209.
[KIM and PAR, 07] Kim H. and Park H., (2007). Sparse Non-negative Matrix Factorizations via Alternating Non-negativity-constrained Least Squares for Microarray Data Analysis. *Bioinformatics*, 23(12), pp.1495-1502.
[KIM and PAR, 08] Kim H. and Park H., (2008). Nonnegative Matrix Factorization Based on Alternating Non-negativity-constrained Least Squares and the Active Set Method. *SIAM J. on Matrix Analysis and Applications (SIMAX)*, 30(2), pp.713-730.
[KOL 93] Kolodner, J. (1993). Case-based reasoning. *San Mateo, CA: Morgan Kaufmann*.
[KOR 08] Koren Y., (2008). Factorization Meets the Neighbourhood: a Multifaceted Collaborative Filtering Model. In *Proc. Of the 14th ACM SIGKDD Inter. Conf. on KDDM*, pp. 424-436.
[KOR et al., 09] Koren Y., Bell R. and Volinsky C., (2009). Matrix Factorization Techniques for Recommender Systems. *IEEE Computer semi*, 42(8), pp. 30–37.
[KOY 02] Koychev I., (2002). Tracking Changing User Interests through Meta-Leaning of Context, Adaptive hypermedia and adaptive Web-based systems. *In the 2nd Inter. Conf., Springer, LNCS 2347*.
[KRU 97] Krulwich B., (1997). Lifestyle Finder: Intelligent user profiling using large-scale demographic data, *AI Magazine*, 18(2), p. 37-45.
[KAM et al., 05] Kamahara, J., Asakawa, T., Shimojo, S. and Miyahara, H. (2005). A community-based recommendation system to reveal unexpected interests. *11th Inter. MM Conf.*, pp. 433 – 438.
[KAR 01] Karypis, G. (2001, October). Evaluation of item-based top-n recommendation algorithms. In *Proc. of the 10th Inter. Conf. on Information and knowledge management*, pp. 247-254. ACM.
[KAW et al., 07] Kawai, Y., Kumamoto, T., & Tanaka, K. (2007, January). Fair News Reader: Recommending news articles with different sentiments based on user preference. *In Knowledge-Based Intelligent Information and Engineering Systems*, pp. 612-622. Springer Berlin Heidelberg.
[KEL et TEE, 03] Kelly, D., & Teevan, J. (2003, September). Implicit feedback for inferring user preference: a bibliography. *In ACM SIGIR Forum* , Vol. 37(2), pp. 18-28. ACM.
[KIM et AHN, 08] Kim, K. J., & Ahn, H. (2008). A recommender system using GA K-means clustering in an online shopping market. *Expert systems with applications*.34(2), pp. 1200-1209.
[KIM et YAN, 05] Kim, T. H., & Yang, S. B. (2005). An effective recommendation algorithm for clustering-based recommender systems. *Advances in AI*, pp. 1150-1153. Springer Berlin Heidelberg.
[KIM et YAN, 07] Kim, T. H., & Yang, S. B. (2007). An effective threshold-based neighbor selection in collaborative filtering. pp. 712-715. Springer Berlin Heidelberg.
[KOR 10] Koren, Y. (2010). Factor in the neighbors: Scalable and accurate collaborative filtering. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, *4*(1), pp.1.

[KOY et SCH, 00] Koychev, I., & Schwab, I. (2000, May). Adaptation to drifting user's interests. *In Proceedings of ECML2000 Workshop: Machine Learning in New Information Age* (pp. 39-46).
[KUZ 14] Kuzelewska, U. (2014). Clustering Algorithms in Hybrid Recommender System on MovieLens Data. *Studies in Logic, Grammar and Rhetoric*, 37(1), pp.125-139.
[LAM et al., 96] Lam W., Mukhopadhyay S., Mostafa J. and Palakal M. (1996). Detection of shifts in user interests for personalized information filtering. *Proc. of SIGIR'96, Switzerland*, pp. 317-325.
[LAM et RIE., 04] Lam, S. K., & Riedl, J. (2004, May). Shilling recommender systems for fun and profit. In *Proc. of the 13th inter. Conf. on World Wide Web*, pp. 393-402. ACM.
[LAN 05.a]. Langville A. N. (2005) Algorithms for the nonnegative matrix factorization in text mining. *Slides from SAS Meeting*.
[LAN 05.b]. Langville A. N. (2005). Experiments with the nonnegative matrix factorization and the reuters10 dataset. *Slides from SAS Meeting*.
[LAN et al., 06] Langville A., Meyer C., Albright R., Cox J. and Duling D., (2006). Algorithms, Initializations, and Convergence for the Nonnegative Matrix Factorization. *In the 12th ACM SIGKDD Inter. Conf. on Knowledge Discovery and Data Mining.*
[LAT et al., 10] Lathia, N., Hailes, S., Capra, L., & Amatriain, X. (2010). Temporal diversity in recommender systems. *In Proc. of the 33th Inter. ACM SIGIR Conf. on RDIR*, pp. 210-217. ACM.
[LEE and SEU 99] Daniel D. Lee and H. Sebastian Seung. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature*, Vol. 401, pp. 788–791.
[LEE and SEU, 01] Lee D. and Seung H., (2001). Algorithms for Non-Negative Matrix Factorization. *Advances in Neural Information Processing Systems*, Vol. 13, pp.556–562.
[LEE et al., 08] Lee, T. Q., Park, Y., & Park, Y. T. (2008). A time-based approach to effective recommender systems using implicit feedback. *ESA*, 34(4), pp. 3055-3062.
[LEE et al., 14] Lee, W. J., Oh, K. J., Lim, C. G., & Choi, H. J. (2014). User profile extraction from Twitter for personalized news recommendation. *In ICACT'16,* pp. 779-783. IEEE.
[LEU et al., 07] Leung, C. W. K., Chan, S. C. F., & Chung, F. L. (2007, November). Applying cross-level association rule mining to cold-start recommendations. *InProc. of the 2007 IEEE/WIC/ACM Inter. Conf. on Web Intelligence and Intelligent Agent Technology-Workshops*, pp. 133-136.
[LIN et al. 03] Linden G., Smith B., and York J., (2003). Amazon.com recommendations: Item-to-item collaborative filtering. In *Proceedings of IEEE Internet Computing*, pp. 76–80.
[LIU et al., 14] Liu, H., Hu, Z., Mian, A., Tian, H., & Zhu, X. (2014). A new user similarity model to improve the accuracy of collaborative filtering. *Knowledge-Based Systems*, Vol. 56, pp. 156-166.
[LIU et PED, 10] Liu, J., Dolan, P., & Pedersen, E. R. (2010, February). Personalized news recommendation based on click behavior. *In Proc. of the 15th Inter. Conf. on IUI*, pp. 31-40. ACM.
[LÜ et al., 12] Lü, L., Medo, M., Yeung, C. H., Zhang, Y. C., Zhang, Z. K., & Zhou, T. (2012). Recommender systems. *Physics Reports*, 519(1), pp.1-49.
[LUO et al. 13] Luo, X., Xia, Y., & Zhu, Q. (2013). Applying the learning rate adaptation to the matrix factorization based collaborative filtering. *Knowledge-Based Systems*, Vol. 37, pp. 154-164.
[MA et al., 07] Ma, I. King, M. R. Lyu, (2007), Effective messing data prediction for collaborative filtering, in *Proc. Of the 30th Inter. ACM SIGIR Conf. on RDIR*, pp. 39-46.

[MAA et SER, 10] M. Maâtallah and H. Seridi, (2010). A Fuzzy Hybrid Recommender System. *In the Proc. Of the 1st Inter. Conf. on Machine and Web Intelligence* (ICMWI'10), pp. 258-263.
[MAA et SER, 12a] M. Maâtallah and H. Seridi, (2012). Enhanced Collaborative Filtering to Recommender Systems of Technology Enhanced Learning, *In the Proc. Of the 4th International Conference on Web and Information Technology*, ICWIT'2012, pp. 129-138.
[MAA et SER, 12b] M. Maâtallah and H. Seridi, (2012). Multi-Context Recommendation in Technology Enhanced Learning, *In the Proc. Of the 11th Inter. Conf. on Intelligent Tutoring System*, ITS'2012, LNCS 7315, Intelligent Tutoring Systems, pp. 720-721.
[MAA et SER, 15] M. Maatallah and H. Seridi-Bouchelaghem, (2015). A Fuzzy Hybrid Approach to Enhance the Diversity in TOP-N Recommendations. *Int. J. Business Information Systems*, 19(14), pp505-530.
[MAH et RIC, 07] Mahmood, T., & Ricci, F. (2007, August). Learning and adaptivity in interactive recommender systems. *In Proc. of the 9th inter. Conf. on Electronic commerce*, pp. 75-84. ACM.
[MAH et RIC, 09] Mahmood, T., & Ricci, F. (2009). Improving recommender systems with adaptive conversational strategies. In *Proc. of the 20th ACM conf. on Hypertext and hypermedia*. pp. 73-82.
[MAS et BHA, 04] Massa, P., & Bhattacharjee, B. (2004). Using trust in recommender systems: an experimental analysis. In *Trust Management*, pp. 221-235. Springer Berlin Heidelberg.
[MAS et AVE, 07] Massa, P., & Avesani, P. (2007, October). Trust-aware recommender systems. In *Proc. of the 2007 ACM conf. on Recommender systems*, pp. 17-24. ACM.
[MAZ 13] Mazurowski, M. A. (2013). Estimating confidence of individual rating predictions in collaborative filtering recommender systems. *Expert Systems with Applications*, *40*(10), pp.3847-3857.
[McN et al., 03] McNee, S. M., Lam, S. K., Guetzlaff, C., Konstan, J. A., & Riedl, J. (2003). Confidence displays and training in recommender systems. In*Proc. INTERACT* , Vol. 3, pp. 176-183.
[MCN et al., 06] Mcnee S., Riedl J and Konstan J. (2006). Accurate is not always good: How Accuracy Metrics have hurt Recommender Systems. *In Conf. on Human Factors in Computing Systems, Quebec, Canada.* pp. 1-5.
[MCS 02] McSherry, D. (2002). Diversity-conscious retrieval. *In Advances in Case-Based Reasoning* , pp. 219-233. Springer Berlin Heidelberg.
[MEH et NEJ 08]Mehta, B., & Nejdl, W. (2008,). Attack resistant collaborative filtering. In *Proc. of the 31st ann. inter. ACM SIGIR conf. on Research and development in information retrieval*, pp. 75-82.
[MEL et al., 02] Melville P., Mooney R. J., Nagarajan R., (2002). Content-Boosted Collaborative Filtering for Improved Recommendations, *Proc. of the 80th Nat. Conf. on Artif. Intel., AAAI'02*, Canada, pp. 187-192.
[MID et al., 02] 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*.
[MID et al., 04] Middleton S. E., Shadbolt N. R., De Roure D. C., Ontological user profiling in Recommender systems, *ACM Transactions on Information Systems,* 22(1), pp.54-88.
[MIT et al., 94] Mitchell, T., Caruana, R., Freitag, D., McDermott, J. and Zabowski, D. (1994). Experience with a Learning Personal Assistant. *Communications of the ACM*, 37(7), pp. 81-91.

[MOB et al., 07] Mobasher, B., Burke, R., Bhaumik, R., & Williams, C. (2007). Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness. *ACM Transactions on Internet Technology (TOIT)*, *7*(4), 23.
[MON et al., 03] Montaner M., López B., De La Rosa J. L., (2003). A Taxonomy of Recommender Agents on the Internet, *Artificial Intelligence Review*, Vol. 19, pp.285-330.
[MOO et ROY, 99] Mooney R. J. and Roy L. (1999). Content-based book recommendation using learning for text categorization. In *Proc; of the SIGIR Workshop on Recommender Systems: Algorithms and Evaluation.*
[MUR et al., 08] Murakami, T., Mori, K., & Orihara, R. (2008). Metrics for evaluating the serendipity of recommendation lists. In *New Frontiers in Artificial Intelligence*, pp. 40-46. Springer.
[NGU et al., 05] Nguyen A., Denos A., Berrut C., (2005). Cartes de communautés pour l'adaptation interactive de profils dans un système de filtrage d'information, *Actes du XXIIIème Congrès INFORSID,* pp. 253-268.
[NGU et al., 06] Nguyen A. and Denos N.and Berrut C., (2006). Modèle d'espaces de communautés basé sur la théorie des ensembles d'approximation dans un système de filtrage hybride, *Conf. en Recherche Information et Applications (CORIA*), Lyon, France, pp. 303-314.
[NGU et al., 06a] Nguyen A., Denos N., Berrut C., (2006). Exploitation des données "disponibles à froid" pour améliorer le démarrage à froid dans les systèmes de filtrage d'information, *in INFORSID '06*, pp. 81-95.
[NGU et al., 06b] Nguyen A., Denos N., and Berrut C., (2006). Modèle des espaces de communautés orienté vers la diversité des recommandations pour les systèmes de filtrage, *Conférence en recherche information et applications No3*, Lyon, France, 6(2), pp.125-150.
[NGU et SHU, 13] Nguyen, J., & Shu, M. (2013). Content-boosted matrix factorization techniques for recommender systems. *Statistical Analysis and Data Mining*, *6*(4), pp. 286-301.
[NOU et al., 97] : Nouali O.; Benmeziane S.; Djaid M.; Sidi-Boumediane S. (1997). Filtrage de l'information, Laboratoire Intelligence Artificielle, *CERIST*, 7(2), pp. 27-35.
[O'DO et SMY, 05] O'Donovan, J., & Smyth, B. (2005, January). Trust in recommender systems. In *Proc. of the 10th international conference on Intelligent user interfaces*, pp. 167-174. ACM.
[O'MA et al., 04] O'Mahony, M., Hurley, N., Kushmerick, N., & Silvestre, G. (2004). Collaborative recommendation: A robustness analysis. *ACM Transactions on Internet Technology*, *4*(4), pp.344-377.
[O'MA et al., 05] O'Mahony, M. P., Hurley, N. J., & Silvestre, G. C. (2005, July). Recommender systems: Attack types and strategies. In *AAAI* , pp. 334-339.
[OAR et KIM, 98] Oard, D. W., & Kim, J. (1998, July). Implicit feedback for recommender systems. *In Proc. of the AAAI workshop on recommender systems*, pp. 81-83.
[OCO et HER, 99] O'Connor, M., & Herlocker, J. (1999, August). Clustering items for collaborative filtering. *In Proc. of the ACM SIGIR workshop on recommender systems*, Vol. 128.
[OH et al., 11] Oh, J., Park, S., Yu, H., Song, M., & Park, S. T. (2011). Novel recommendation based on personal popularity tendency. In *Inter. IEEE Conf. on Data Mining (ICDM)*, pp. 507-516. IEEE.
[OKU et HAT, 12] Oku, K., & Hattori, F. (2012, September). User Evaluation of Fusion-based Approach for Serendipity-oriented Recommender System. In *Workshop on Recommendation Utility Evaluation: Beyond RMSE (RUE 2011)*, pp. 39.
[PAA 97] Pentti Paatero. (1997). Least squares formulation of robust non-negative factor analysis. *Chemometrics and Intelligent Laboratory Systems*, Vol. 37, pp.23–35.

[PAA and TAP, 94] Paatero P. and Tapper U., (1994). Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, Vol. 5, pp. 111–126.
[PAR et al., 12] Park, D. H., Kim, H. K., Choi, I. Y., & Kim, J. K. (2012). A literature review and classification of recommender systems research. *ESA*, *39*(11), pp. 10059-10072.
[PAR et SHE, 14] Prabhat K., Sherry C., (2014). An Efficient Recommender System using Hierarchical Clustering Algorithm. *In JCSTTechnology* (IJCST), 2(4), pp. 1-6.
[PAU et al. 06] Pauca, V. P., Piper, J., & Plemmons, R. J. (2006). Nonnegative matrix factorization for spectral data analysis. *Linear algebra and its applications*, *416*(1), pp. 29-47.
[PAZ 99] Pazzani M., (1999). A framework for collaborative, content-based and demographic filtering, *Artificial Intelligence Review*, 13(5), pp. 393-408.
[PAZ et BIL, 07] Pazzani M., Billsus D., (2007). Content-based Recommendation Systems. *In The Adaptive Web: Methods and Strategies of Web Personalization, LNCS,* Vol. 4321. Springer
[PER et al., 04] Perik, E., De Ruyter, B., Markopoulos, P., & Eggen, B. (2004). The sensitivities of user profile information in music recommender systems. *Proc. of Private, Security, Trust*, pp.137-141.
[PEN et al., 00] Pennock, D. M., Horvitz, E., Lawrence, S., & Giles, C. L. (2000, June). Collaborative filtering by personality diagnosis: A hybrid memory-and model-based approach. *In Proc. of the Sixteenth conference on Uncertainty in artificial intelligence,* pp. 473-480.
[PHE et SMY, 09] Phelan, O., McCarthy, K., & Smyth, B. (2009, October). Using twitter to recommend real-time topical news. *In Proc. of the 3th ACM Conf. RecSys'2009,* pp. 385-388. ACM.
[PIA et al., 09] Piao, C. H., Zhao, J., & Zheng, L. J. (2009). Research on entropy-based collaborative filtering algorithm and personalized recommendation in e-commerce. *Service Oriented Computing and Applications*, 3(2), pp. 147-157.
[PIL et al., 10], Pilászy, I., Zibriczky, D., & Tikk, D. (2010). Fast als-based matrix factorization for explicit and implicit feedback datasets. *In Proc. of the 4th ACM Conf. RecSys'2010,* pp. 71-78. ACM.
[PIT , 09] Pitsilis, G. (2009). Trust-enhanced Recommender Systems for efficient on-line collaboration. In *Trust Management III*, pp. 30-46. Springer Berlin Heidelberg.
[PIT et MAR, 08] Pitsilis, G., & Marshall, L. F. (2008). Modeling trust for recommender systems using similarity metrics. In *Trust Management II*, pp. 103-118. Springer US.
[POR et al., 09] Porcel, C., López-Herrera, A. G., & Herrera-Viedma, E. (2009). A recommender system for research resources based on fuzzy linguistic modeling. *Expert Systems with Applications*, 36(3), pp. 5173-5183.
[PRE et al., 13] Premchaiswadi, W., Poompuang, P., Jongswat, N., & Premchaiswadi, N. (2013, July). Enhancing Diversity-Accuracy Technique on User-Based Top-N Recommendation Algorithms. *In Computer Software and Applications Conference Workshops, 2013 IEEE 37th Annual*, pp. 403-408.
[PU et CHE, 06] Pu, P., & Chen, L. (2006, January). Trust building with explanation interfaces. In *Proc. of the 11th inter. Conf. on Intelligent user interfaces*, pp. 93-100. ACM.
[RAM et al., 01] Ramakrishnan, N., Keller, B. J., Mirza, B. J., Grama, A. Y., & Karypis, G. (2001). Privacy risks in recommender systems. *IEEE Internet Computing*, *5*(6), pp.54-62.
[RAS et al., 02] Rashid A., Albert I., Cosley, D., Lam S. K., Mcnee S. M., Konstan J. A., Riedl J., (2002). Getting to Know You: Learning New User Preferences in Recommender Systems, *Proc. of the 7th Inter. Conf. on Intelligent User Interfaces, IUI'02*, San Francisco, California, USA, pp. 127-134.

[REN et al., 09] Rendle, S., Freudenthaler, C., Gantner, Z., & Schmidt-Thieme, L. (2009). BPR: Bayesian personalized ranking from implicit feedback. *In Proc. of the 25th Conf. on UAI*. pp. 452-461.

[REN et al., 11] Ren G., Long T., Juebo W., (2011). A Novel Recommender System Based on Fuzzy Set and Rough Set Theory. *Advances in Information Sciences and Service Sciences*. 3(4).

[RES and VAR, 97] Resnick P., Varian H.R., (1997). *Recommender Systems, Communications of the ACM*, 40(3), pp. 56-58.

[RES et al., 94] Resnick P., Iacovou N., Suchak M., Bergstrom P., and Riedl J., (1994). GroupLens: An open architecture for collaborative filtering of netnews. *In Proc. Of the ACM Conf. on Computer Supported Cooperative Work*, USA, pp. 175–186.

[RIE et SCH, 89] Riesbeck, C., & Schank, R. C. (1989). Inside Case-Based Reasoning. *Hillsdale, NJ: Lawrence Erlbaum*.

[RIJ 79] Rijsbergen C.J.V, (1979). Information Retrieval. *Second edition, Butterworks*.

[SAI et al., 12] Said, A., Kille, B., Jain, B. J., & Albayrak, S. (2012). Increasing diversity through furthest neighbor-based recommendation. *Proceedings of the WSDM*, Vol. *12*.

[SAL et al., 13] Salehi, M. and Kamalabadi I. N., Ghaznavi M. B. (2013). Attribute-based Collaborative filtering using genetic algorithm and weighted C-means algorithm. *IJBIS*, 13(3), pp. 265 – 283.

[SAL et KAM, 13] Salehi, M. and Kamalabadi I. N. (2013). A hybrid recommendation approach based on attributes of products using genetic algorithm and naive Bayes classifier. *IJBIS*, 13(4), pp. 381 – 399.

[SAL et MNI, 08], Salakhutdinov R. & Mnih A., (2008). Probabilistic matrix factorization. In *Advances in Neural Information Processing Systems*, Vol. 20, pp. 1257-1264, Cambridge, MA.

[SAL 13] Salehi, M. (2013). An effective recommendation based on user behaviour: a hybrid of sequential pattern of user and attributes of product. *IJBIS*, 14(4), pp. 480-496.

[SAN et al., 10] Santos, R. L., Macdonald, C., & Ounis, I. (2010). Exploiting query reformulations for web search result diversification. *In Proc. of the 19th intern. conf. on WWW*, pp. 881-890. ACM.

[SAR et al., 01] Sarwar B. M., Karypis G., Konstan A., and Riedl J., (2001). Item-based collaborative filtering recommendation algorithms. *In Proc. of the 10th Inter. Conf. on WWW*, pp. 285–295.

[SAR et al., 02] Sarwar, B. M., Karypis, G., Konstan, J., & Riedl, J. (2002, December). Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering. In *Proc. of the 5th Inter. Conf. on computer and information technology*, Vol. 1.

[SAR et al., 02a] Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2002). Incremental singular value decomposition algorithms for highly scalable recommender systems. In *5th Inter. Conf. on CIS*, pp. 27-28.

[SCH et al., 01] Schein A. I., Popescul A., Ungar L. H. (2001). Generative Models for Cold-Start Recommendations. *In Proc. of the SIGIR Workshop on Recommender Systems*, USA.

[SCH et al., 01a] Schwab I., Kobsa A., Koychev I., (2001). Learning User Interests through Positive Examples Using Content Analysis and Collaborative Filtering. *Internal Memo, GMD, Germany*.

[SCH et al., 02] Schein A. I., Popescul A., and Ungar L. H., (2002). Methods and Metrics for Cold Start Recommendations, *Proc. of the 25th Inter. ACM SIGIR Conf. on Research and Development in Information Retrieval* (SIGIR 2002), pp. 253-260. Tampere, Finland.

[SCH et al., 99] Schafer J.B., Konstan J., and Reidl J., (1999). Recommender Systems in Ecommerce. *Proc. of the 1st ACM Conf. on Electronic Commerce*, Denver, CO, pp. 158 -166.

[SCH et FAL, 07] Schickel-Zuber, V., & Faltings, B. (2007, August). Using hierarchical clustering for learning theontologies used in recommendation systems. *In Proc. of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 599-608. ACM.

[SCH et SIN, 98] Schapire, W. W. C. R. E., & Singer, Y. (1998, October). Learning to order things. In *Proc. of the 1997 Conf. on Advances in Neural Information Processing Systems*, 10, pp. 451.

[SEM et al., 12] Seminario, C. E., & Wilson, D. C. (2012, May). Robustness and accuracy tradeoffs for recommender systems under attack. In *25th Inter. FLAIRS Conf.*

[SHA 05] Shahnaz F. (2005). A clustering method based on nonnegative matrix factorization for text mining. *Master's thesis, University of Tennessee*, Knoxville.

[SHA et al., 06] Shahnaz F., Berry M.W., Pauca V. P., and Plemmons R. J.. (2006). Document Clustering Using Nonnegative Matrix Factorization. *IPM*, 42(2), pp. 373–386.

[SHA et al., 07] Shani G., Meisles A. and Gleyzer Y., (2007). A Stereotypes-Based Hybrid Recommender System for Media Items. *American Association for Artificial Intelligence, Workshop on Recommender Systems*, pp. 76-83.

[SHA et al., 08] Shani, G., Chickering, D.M., Meek, C. (2008). Mining recommendations from the web. *In Proc. of the 2008 ACM Conf. RecSys'08*, pp. 35–42.

[SHA et al., 97] Shapira B., Shoval P., Hannani U., (1997). Stereotypes in Information Filtring Systems. *Information Proceedings 1 Management*, 33(3), pp. 273-287.

[SHA et GUN, 11] Shani, G., & Gunawardana, A. (2011). Evaluating recommendation systems. In*Recommender systems handbook*, pp. 257-297. Springer US.

[SHA et LU, 11] Shambour, Q., & Lu, J. (2011). A hybrid trust-enhanced collaborative filtering recommendation approach for personalized government-to-business e-services. *IJIS*, 26(9), pp. 814-843.

[SHA et MAE, 95] Shardanand, U., & Maes, P. (1995). Social information filtering: algorithms for automating "word of mouth". *In Proc. of the SIGCHI Conf. on HFCS*, pp. 210-217.

[SHA et SUM, 13] Sharma, S. K., & Suman, U. (2013). A framework of hybrid recommender system for web personalisation. *Journal of Business Information Systems*, 13(3), pp. 284-316.

[SHE et MIR, 09] McSherry, F., & Mironov, I. (2009). Differentially private recommender systems: building privacy into the net. In *Proc. of the 15th ACM SIGKDD Inter. Conf. on KDDM*, pp. 627-636.

[SHI et HAN, 14] Shi, Y., Larson, M., & Hanjalic, A. (2014). Collaborative filtering beyond the user-item matrix: A survey of the state of the art and future challenges. *ACM Computing Surveys (CSUR)*, 47(1), pp. 3.

[SHI et KUL, 11] Shinde S. K. and Kulkarni U.V., (2011). Hybrid Personalized Recommender System Using Modified Fuzzy C-Means Clustering Algorithm. *IJAE*, 1(4), pp. 88-99.

[SMY et McC, 01] Smyth, B., & McClave, P. (2001). Similarity vs. diversity. *In Case-Based Reasoning Research and Development*, pp. 347-361. Springer Berlin Heidelberg.

[SRE et al., 04] Srebro, N., Rennie, J., & Jaakkola, T. S. (2004). Maximum-margin matrix factorization. *In Advances in neural information processing systems*, pp. 1329-1336.

[SRE et al., 05], Srebro N., Rennie J.D.M., and Jaakkola T.S., (2005). Maximum-margin matrix factorization. *Advances in Neural Information Processing Systems,* Vol. 17, pp. 1329-1336.

[SRE et JAA, 03]Srebro, N., & Jaakkola, T. (2003, August). Weighted low-rank approximations. *In ICML*, Vol. 3, pp. 720-727.

[SU et KHO, 09] Su X. and Khoshgoftaar T. M., (2009). A Survey of Collaborative Filtering Techniques. In *Advances in Artificial Intelligence*, pp. 1-20.

[SUN et al., 12] Sun, H. F., Chen, J. L., Yu, G., Liu, C. C., Peng, Y., Chen, G., & Cheng, B. (2012). JacUOD: A new similarity measurement for collaborative filtering. *JCST*, 27(6), pp. 1252-1260.

[TAK et al., 08], Takacs G., Pilàszy I., Németh B. and Tikk D., (2008). Matrix Factorization and Neighbour based Algorithms for the Netflix Prize Problem. In *Proc. of 2nd ACM Conf. RecSys*, pp.267–274.

[TAK et al., 09] Takács, G., Pilászy, I., Németh, B., & Tikk, D. (2009). Scalable collaborative filtering approaches for large recommender systems. *The JMLR*, Vol. *10*, pp. 623-656.

[TER et MEI 10] Terán L. & Maier A., (2010). A Fuzzy Recommender System for eElections. In Proc. *of the 1st inter. Conf. on EGOVIS, LNCS*, Vol. 6267, Springer, Heidelberg, pp. 62-76.

[TOW et QUI, 00] Towle B., Quinn C., (2000). Knowldge Bsed Recommender Systems Using Explicit User Models, *In KBEM, AAAI Technical Report WS-00-04*, pp. 74-44.

[TRE et JAR, 12], Treerattanapitak K., and Jaruskulchai C., (2012). Exponential fuzzy C-means for collaborative Filtering. *Journal of Computer Science and Technology*, 27(3), pp.567-576.

[UPE 94] Upendre,S. (1994), Social information filtering for music recommendation, Technical Report TR-94-04, MIT EECS, Learning and Common Sense Group, MIT Media Laboratory;

[VAR et CAS, 11] Vargas, S., & Castells, P. (2011). Rank and relevance in novelty and diversity metrics for recommender systems. *In the 5th ACM Conf. RecSyst'11*, pp.109-116. ACM.

[VAR et CAS, 13] Vargas, S., & Castells, P. (2013, May). Exploiting the diversity of user preferences for recommendation. *In Proc. of the 10th Conf. on ORAIR*, pp. 129-136.

[WAN et al. 06] Wang J., de Vries A. P., Reinders M.J.T., (2006). Unifying User-based and Item-based Collaborative Filtering Approaches by Similarity Fusion. In *Proc. Of the 29th Inter. ACM SIGIR Conf. on Research and Development in Information Retrieval*, USA, pp. 501-508.

[WAN et al., 08] Wang, J., De Vries, A. P., & Reinders, M. J. (2008). Unified relevance models for rating prediction in collaborative filtering. *ACM Trans. on Information Systems (TOIS)*, 26(3), pp. 16.

[WAN et al., 10], Wang J., Zhang N. J., Yin J., (2010). Collaborative filtering recommendation based on fuzzy clustering of user preferences. *In 7th Inter. Conf. on FSKD*, Vol. 5, pp. 1946–1950.

[WAN et KAO, 13] Wang, J. Y., & Kao, H. Y. (2013). RSOL: A trust-based recommender system with an opinion leadership measurement for cold start users. In *IRT*, pp. 500-512. Springer

[WAN et YIN 13] Wang, J., & Yin, J. (2013). Combining user-based and item-based collaborative filtering techniques to improve recommendation diversity. In 6th *Inter. Conf. on BMEI*, pp. 661-665.

[WEB 01]: Webb, G. Pazzani, M. and Billsus, D. (2001). Machine Learning for user modeling". *User Modeling and User-Adaptive Interaction*, Vol. 11, pp. 19-29.

[WEN et al. 08] Weng L. T., Xu Y., Li Y., Nayak R., (2008). Exploiting Item Taxonomy for Solving Cold-Start Problem in Recommendation Making. In *the 20th IEEE Inter. Conf.*, Vol. 2, pp. 113-120.

[WIT et al., 09] Witten DM., Tibshirani R., and Hastie T., (2009). A Penalized Matrix Decomposition with applications to sparce principal components and canonical correlation analysis. *Biostatistics*, pp. 515-534.

[WU and LI, 08] Wu J. and Li T., (2008).A modified fuzzy C-means algorithm for collaborative filtering. In *Proc. of the 2nd KDD Workshop on Large-Scale Recommender Systems and the Netflix Price Competition*, ACM New York.
[XU et WUN, 05] Xu, R., & Wunsch, D. (2005). Survey of clustering algorithms. *Neural Networks, IEEE Transactions on*, 16(3), pp.645-678.
[XUE et al., 05] Xue G.R., Lin C., Yang Q., Xi W., Zeng H.J., Yu Y., and Chen Z., (2005). Scalable collaborative filtering using cluster-based smoothing. In *Proc. of the ACM SIGIR Conf.*, pp. 114, 121.
[YAG 03] Yager, R. R. (2003). Fuzzy logic methods in recommender systems. *Fuzzy Sets and Systems*, 136(2), pp.133-149.
[YAN 14] Yan, S. (2014). A Collaborative Filtering Recommender Approach by Investigating Interactions of Interest and Trust. In *Knowledge Engineering and Management* , pp.173-188. Springer
[YAT et NET, 99] Baeza-Yates R., Ribeiro-Neto B. 1999. Modern Information Retrieval, 1st edition, Addison Wesley.
[YIL et KRI, 08] Yildirim H., S.Krishnamoorthy M. S., (2008). A random walk method for alleviating the sparsity problem in collaborative filtering, in *Proc. of RecSys'2008, pp*. 131-138.
[YIN et PEN, 12]Yin, C. X., & Peng, Q. K. (2012). A careful assessment of recommendation algorithms related to dimension reduction techniques. *KBS*, Vol. 27, pp. 407-423.
[YOO et CHO, 09] Yoo, J., & Choi, S. (2009, April). Probabilistic matrix tri-factorization. *InAcoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE Inter. Conf. on*, pp. 1553-1556.
[ZHA 08] Zhang, F. (2008, October). Research on recommendation list diversity of recommender systems. *In Inter. Conf. on Management of e-Commerce and e-Government*, pp. 72-76.
[ZHA 09] Zhang, M. (2009, October). Enhancing diversity in top-n recommendation. *In Proc. of the 3rd ACM Conf. on Recommender systems*, pp. 397-400. ACM.
[ZHA 13] Zhang, L. (2013). The Definition of Novelty in Recommendation System. *Journal of Engineering Science and Technology Review*, *6*(3), pp. 141-145.
[ZHA et al., 06] Zhang, S., Wang, W., Ford, J., & Makedon, F. (2006, April). Learning from Incomplete Ratings Using Non-negative Matrix Factorization. *In SDM*, pp. 549-553.
[ZHA et al., 11] Zhang, F., Liu, H., & Cui, Y. (2011). Rating Information Entropy for Cold-start Recommendation. *JICS,* 8(1), pp.16-22.
[ZHA F. 09] Zhang, F. (2009). Improving recommendation lists through neighbor diversification. *In IEEE Inter. Conf. on* Intelligent Computing and Intelligent Systems,Vol. 3, pp. 222-225. IEEE
[ZHE et al., 13] Zheng, L., Li, L., Hong, W., & Li, T. (2013). PENETRATE: Personalized news recommendation using ensemble hierarchical clustering. *Expert Syst. with App.,* 40(6), pp. 2127-2136.
[ZHU et GON, 09] Zhu, R., & Gong, S. (2009). Analyzing of collaborative filtering using clustering technology. *In Computing, Communication, Control, and Management*, Vol. 4, pp. 57-5, IEEE
[ZIE et al., 04] Ziegler, C. N., Lausen, G., & Schmidt-Thieme, L. (2004,). Taxonomy-driven computation of product recommendations. *In Proc. of the 30th ACM Inter. Conf. on IKM*, pp. 406-415.
[ZIE et al., 05] Ziegler, C. N., McNee, S. M., Konstan, J. A., & Lausen, G. (2005, May). Improving recommendation lists through topic diversification. *Proc. of the 14th inter. Conf. on WWW*, pp. 22-32.

[ZIE et al., 05] Ziegler, C.N., McNee, S.M., Konstan, J.A., Lausen, G. (2005). Improving recommendation lists through topic diversification. *In: WWW 05,* pp. 22–32.
[ZIE et al., 09] Ziegler, C. N., Lausen, G., & Georges-Köhler-Allee, G. (2009). Making Product Recommendations More Diverse. *IEEE Data Eng. Bull.*, 32(4), pp. 23-32.

ANNEXE A

METHODES DE CLASSIFICATION NON SUPERVISEES (FCM ET FACTORISATION EN MATRICE NON NEGATIVE)

D'un point de vue général, les méthodes de classification ont pour but de regrouper les éléments d'un ensemble $X = \{X^1, ..., X^n, ..., X^N\}$ en un nombre *K* optimal de classes selon leurs ressemblances [**LUR 03**].

En effet, les objets sont souvent répertoriés par rapport à des catégories ou des classes auxquelles ils sont censés appartenir. Cette appartenance est, la plupart du temps, vague et/ou graduelle.

De manière générale, les problèmes de classification s'attachent à déterminer des procédures permettant d'associer une classe à un objet (individu).

Ces problèmes se déclinent essentiellement en deux variantes selon Bezdek [**BEZ 93**] : la classification dite " supervisée " et la classification dite " non supervisée ".

La classification, supervisée ou non, en tant que discipline scientifique, n'a été automatisée et massivement appliquée [**KHO 97**]. Comme la plupart des activités scientifiques, l'essor des différentes techniques de classification a largement bénéficié de l'avènement et du perfectionnement des outils informatiques. De nos jours, la classification est une démarche qui est appliquée dans d'innombrables domaines. Un autre nom possible pour cette branche de la recherche est *la typologie*, et la science qui lui est associée est *la taxonomie*.

Les méthodes de classification ont pour but de regrouper les éléments d'un ensemble X, de nature quelconque, en un nombre restreint de classes. La qualité de la classification peut être jugée sur la base des deux critères suivants :

- Les classes générées doivent être les plus différentes possibles les unes des autres vis-à-vis de certaines caractéristiques,

- chaque classe doit être la plus homogène possible vis-à-vis de ces caractéristiques.

L'intérêt des méthodes de classification non supervisées est qu'elles ne nécessitent aucune base d'apprentissage et par là même aucune tâche préalable d'étiquetage manuel n'est requise. Les algorithmes non supervisés les plus répandus tendent à minimiser une fonction coût, dépendant de la distance de chaque item aux prototypes (ou noyaux) des classes. Le prototype d'une classe étant un point connu dont l'appartenance à la classe est garantie et où chaque item est assigné à la classe qui lui est la plus proche. Selon la certitude de la classification que nous voulons obtenir, et la relation entre les classes, nous pourrons distinguer plusieurs méthodes de classification :

A.1. L'algorithme des C-moyennes floues (Fuzzy C-means FCM)

Fuzzy c-means est une méthode de classification non supervisée développée par Dunn en 1973 **[DUN 73]** et améliorée par Bezdek en 1981 **[BEZ 81]** qui a atteint la solution d'optimisation alternée de classification floue en améliorant la performance de partitionnement du Fuzzy c-means. Avec la classification floue, une partition floue est générée basée sur le principe qu'il n'est pas nécessaire d'affecter une classification définitive à chaque point de l'ensemble de données, en effet, le Fuzzy c-means permet à un point d'appartenir à différentes classes.

L'algorithme des C-moyennes floues [**BEZ 81**], [**BEZ 87**] effectue une optimisation itérative en évaluant de façon approximative les minimums d'une fonction d'erreur. Il existe toute une famille de fonctions d'erreur associées à cet algorithme qui se distinguent par des valeurs différentes prises par un paramètre réglable, *m*, appelé indice de flou « fuzzy index » et qui détermine le degré de flou de la partition obtenue. Les FCMs sont un cas particulier d'algorithmes basés sur la minimisation d'un critère ou d'une fonction objective.

Dans ce formalisme, chaque objet x_i est affecté de façon « fractionnaire » c'est-à-dire qu'il se partage suivant les différents clusters $\{C_1,\ldots, C_t\}$ relativement à une famille de fonctions d'appartenance $\{u_j(x_i)\}$ *j=1,….,t* du vecteur x_i à la classe *j*.

Le but de l'algorithme FCM est alors non seulement de calculer les centres des classes mais aussi l'ensemble des degrés d'appartenance des vecteurs aux classes.

A.1.1. Présentation de l'algorithme FCM

A.1.1.1. Formulation du problème

Si u_{ij} est le degré d'appartenance de x_j à la classe *i*, la matrice *U*= [u_{ij}] de taille *CxN*, est appelée matrice de *C*-partition floue si et seulement si elle satisfait :

$$(\forall\, i \in \{1..\,C\})\ (\forall\, j \in \{1..\,N\})\ u_{ij} \in [0,1], \tag{A.1}$$

$$\forall\, j \in \{1,..,N\}\ \sum_{i=1}^{C} u_{ij} = 1, \tag{A.2}$$

$$0 < \sum_{j=1}^{N} u_{ij} < N\,, \tag{A.3}$$

Bezdek a montré dans [**BEZ 81**] que le problème de partition de *X* en *C* classes floues pouvait être formulé comme la minimisation d'une fonction *J(B,U,X)* définie par :

$$J(B,U,X) = \sum_{i=1}^{C}\sum_{j=1}^{N} (u_{ij})^m . d^2(x_j, b_i) \tag{A.4}$$

$$\underline{\text{St.}}\ \sum_{i=1}^{C} u_{ij} = 1\ ,0 < \sum_{j=1}^{N} u_{ij} < N, \tag{A.5}$$

Où, *m*>1 est un paramètre qui contrôle le degré du flou de la partition résultante. Il a été suggéré que la meilleur valeur pour ce paramètre est m = 2 [**Gao et al.]**. $d^2(x_j, b_i)$ est une distance du vecteur x_j au prototype b_i. La contrainte (A.3) indique que les clusters vides ne sont pas autorisés.

A.1.1.2. Obtention de U et B

Les solutions au problème de minimisation précédent sont obtenues par une méthode lagrangienne. Plus précisément, si $H(x_j)$ est défini pour chaque vecteur x_j par :

$$H(x_j) = \sum_{i=1}^{C} u_{ij}^m d^2(x_j, b_i) - \alpha\left(\sum_{i=1}^{C} u_{ij} - 1\right), \alpha > 0, \tag{A.6}$$

L'annulation des dérivées partielles par rapport à u_{ij} et α donne :

$$\frac{\partial H(x_j)}{\partial \alpha} = 0\ \ et\ \frac{\partial H(x_j)}{\partial u_{ij}} = m . u_{ij}^{m-1} . d^2(x_j, b_i) - \alpha = 0 \tag{A.7}$$

De telle sorte que, si $d^2(x_j, b_i) \neq 0$:

$$u_{ij} = \left[\frac{\alpha}{m.d^2(x_j,b_i)} \right]^{\frac{1}{m-1}} \quad \text{(A.8)}$$

Avec

$$\sum_{i=1}^{c} u_{ij} = \left[\frac{\alpha}{m} \right]^{\frac{1}{m-1}} \sum_{i=1}^{C} \left[\frac{1}{d^2(x_j,b_i)} \right]^{\frac{1}{m-1}} = 1 \quad \text{(A.9)}$$

Et finalement

$$u_{ik} = \left[\sum_{k=1}^{C} \left(\frac{d^2(x_j,b_i)}{d^2(x_j,b_k)} \right)^{\frac{2}{(m-1)}} \right]^{-1} \quad \text{(A.10)}$$

Si $d^2(x_j, b_i) = 0$; x_j est un prototype b_i et $u_{ij} = 1$ avec $u_{kj} = 0$, k ≠ i .

Pour C et X fixés, l'annulation des dérivées partielles H' (B,g) par rapport à une direction quelconque g de B donne enfin :

$$b_i = \frac{\sum_{k=1}^{n} u_{ik}^m . x_k}{\sum_{k=1}^{n} u_{ik}^m} \quad \text{(A.11)}$$

L'algorithme des *C*-moyennes floues (FCM) consiste alors en l'application itérée de (A.10) et (A.11) jusqu'à stabilité des solutions. Le critère d'arrêt des itérations, définissant cette stabilité, peut par exemple consister en l'étude de la norme de la matrice *U* ou en la stabilité des centres de classe sur deux itérations successives.

A.1.1.3.L'algorithme FCM

Les principales étapes de l'algorithme FCM sont citées dans **l'Algorithme A.1**.

Algorithme A.1. Algorithme FCM

Étape 1: *Initialisation:* Fixer *K*, *m* et choisir n'importe quelle métrique norme de produit pour le calcul de $d^2(x_i, c_k)$.

Choisir au hasard *K* échantillons comme centroïdes initiaux $c_k^{(0)}$et ensuite former des partitions de tous les autres échantillons autour de ces centroïdes pour obtenir la matrice de partition initiale $U^{(0)}= [u_{ik}]$, *k=1……K* et *i= 1……N.* à l'étape *l*, *l*=1,2,….., exécuter les étapes suivantes :

Étape 2: Calcul des centroïdes $c_k^{(l)}$:

$$c_k^{(l)} = \frac{\sum_{i=1}^{N}(u_{ki}^{(l-1)})^m x_i}{\sum_{i=1}^{N}(u_{ki}^{(l-1)})^m}; k = 1,2, \dots, K \qquad \text{(A.12)}$$

Étape 3: Calcul des valeurs d'appartenance $u_{ik}^{(l)}$:

$I_i = \{\frac{k}{1} \leq k \leq K;\ d^2\left(x_i, c_k^{(j)}\right) = 0\}$

$\tilde{I}_i = \{1,2, \dots I_i\}$

$$\boldsymbol{u_{ki}^{(l)}} = \begin{cases} \dfrac{1}{\sum_{s=1}^{K}\left[\frac{d^2(x_i, c_k^{(l)})}{d^2(x_i, c_s^{(l)})}\right]^{\frac{1}{(m-1)}}} & \textbf{Si } \boldsymbol{I_i = \emptyset} \\ \mathbf{0} & \forall \boldsymbol{i} \in \boldsymbol{\tilde{I}_i} \\ \dfrac{\mathbf{1}}{|\boldsymbol{I_i}|} & \forall \boldsymbol{i} \in \boldsymbol{I_i} \end{cases} \qquad \text{(A.13)}$$

Étape 4: Répéter les étapes **2** et **3** jusqu'à stabilisation, ie. $\|U_{(l)} - U_{(l-1)}\| \leq \varepsilon,\ l > 1$.

Les équations (A.12) et (A.13) sont appliquées en alternance, jusqu'à ce que les prototypes des clusters se stabilisent et l'algorithme s'arrête, c'est à dire l'erreur entre deux valeurs consécutives de la matrice de contraintes de partition floue *U* sera plus petite qu'un niveau spécifié a priori qui est la valeur du paramètre ε, avec 0< ε <1.

A la fin, le partitionnement flou est réalisé et tous les points sont répartis en clusters selon l'adhésion maximale u_{ik}. La convergence de la FCM a été prouvée dans [**BEZ 81**].

A.1.2. Les limites du FCM

Un grand nombre d'algorithmes est dérivé à partir de l'algorithme FCM, utilisant des métriques différentes ou des prototypes de dimensions supérieures. Citons les FcV (fuzzy c-variétés), FcE (fuzzy c-elliptotypes), Le FCM et ses algorithmes dérivés souffrent de plusieurs inconvénients cités dans **[SEM et al., 07]:**

- Les degrés d'appartenance sont des degrés relatifs. Autrement dit, l'appartenance d'un individu à une classe dépend de l'appartenance de cet individu aux autres classes. Les fonctions d'appartenance construites sont donc interdépendantes. De plus, les estimations des centres des classes ne correspondent pas aux centres réels ou typiques.

- Les points ambigus peuvent avoir des valeurs d'appartenance élevées et ils peuvent affecter de façon significative l'estimation des centres des classes.

- Ces algorithmes modélisent dans la phase de classification l'ambiguïté entre les classes à partir de règles de décisions floues fixées a priori qui peuvent alors affecter la position des centres [**MEN 98**].

A.2. La Factorisation en Matrice Non négatives (Nonnegative Matrix Factorization)

La méthode de factorisation matricielle est une méthode utilisée en Analyse en Composantes Principales (ACP) par l'application de la Décomposition en Valeurs Singulières (SVD) d'une matrice, afin d'obtenir des facteurs orthogonaux à faible rang. La contrainte d'orthogonalité des matrices des facteurs a été par la suite remplacée par la contrainte de non négativité, dans une nouvelle méthode de décomposition proposée par **[PAA et TAP, 94]** et **[LEE et SEU, 99]** afin d'en simplifier l'interprétation des résultats de la factorisation, cette méthode est la méthode de factorisation en matrices non négatives (*Nonnegative Mtarix Factorization* NMF). Cette technique a été largement utilisée dans de très nombreux domaines, plus précisément dans le domaine des systèmes de recommandation, afin de réduire la clairsemée des très grandes matrices creuses, et de factoriser les matrices contenant différents types d'entités.

La factorisation en matrices non négatives a reconnu ses premiers succès avec les contributions de Lee *et al.* **[LEE et al., 99].** Il s'agit d'envisager à des factorisations approchées d'une matrice sous les contraintes de positivité des facteurs trouvés.

Étant donné une matrice non négative X, la factorisation en matrice non négative vise à trouver deux matrices non négatives à faible rang W et H qui minimisent le problème d'optimisation non linéaire suivant:

$$\min \ ||X_{p,n} - W_{p,k}H_{k,n}||_F^2 \qquad \text{(A.14)}$$
$$\text{s.t.} \qquad W \geq 0, H \geq 0$$

Où, $X \in R_+^{p \times n}$, $W \in R_+^{P \times k}$ et $H \in R_+^{k \times n}$ ($R_+^{P \times k}$ et $R_+^{k \times n}$ sont des ensembles de matrices à $p \times k$ et $k \times n$ dont leurs éléments sont tous non négatifs). En général, le rang des matrices W et H est un rang beaucoup plus faible que celui de la matrice X, i.e. $k << min\ (p, n)$, ce paramètre doit être réglé par l'utilisateur.

L'indice F désigne la norme matricielle de Frobenius, qui est la norme utilisée souvent pour mesurer l'erreur entre la matrice originale X et son approximation à faible rang WH, mais il y en a aussi d'autres fonctions coûts qui sont très utilisées en particulier celle utilisant la divergence de Kullback-Leibler.

Ainsi, NMF peut être considérée comme une factorisation spéciale en matrice à faible rang car elle impose la contrainte de non-négativité des facteurs W et H, i.e. tous les éléments de W et H doivent être égaux ou supérieurs à zéro.

A.2.1. Algorithmes de résolution

Plusieurs algorithmes ont été proposés dans la littérature afin de résoudre la fonction objective (A.14) **[BER et al., 07]**, avec différentes versions dépendantes de différentes méthodes d'initialisation et des contraintes ajoutées comme la contrainte de la parcimonie par exemple, dont les trois principaux sont:

- La règle de mise à jour multiplicative (MUR),
- L'algorithme des moindres carrés alternés (ALS),
- L'algorithme de la descente du gradient (DG).

A.2.1.1. La règle de mise à jour multiplicative MUR ***[LEE et al., 99]***

La méthode de minimisation la plus utilisée est la méthode multiplicative de mise à jour de **[LEE et al., 99],** qui constitue un bon compromis entre la rapidité et la facilité d'implémentation.

La mise à jour des facteurs W et H s'opère avec un coefficient de mise à jour multiplicatif qui dépend des données de X et des matrices W et H comme suit:

$$H_{k,n} \leftarrow H_{k,n} \frac{(W^T X)_{k,n}}{(W^T WH)_{k,n}} \quad \text{(A.15)}$$

$$W_{p,k} \leftarrow W_{p,k} \frac{(XH^T)_{p,k}}{(WHH^T)_{p,k}} \quad \text{(A.16)}$$

A.2.1.2. La méthode des Moindres Carrés Alternés ALS ***[LEE et al., 99]***

Il s'agit de trouver les deux facteurs W et H en fixant W (respectivement H) à chaque fois pour trouver leur valeurs optimales. La formulation mathématique du problème à l'itération *r+1* s'exprime en fonction des résultats à l'itération *r* :

$$H^{r+1} = argmin_{H \geq 0} ||X - W^r H||_F^2 \quad \text{(A.17)}$$

$$W^{r+1} = argmin_{W \geq 0} ||X - W H^r||_F^2 \quad \text{(A.18)}$$

La contrainte de la non négativité est ajoutée en mettant à zéro toutes les valeurs négatives dans les matrices W et H à chaque étape, en appliquant l'algorithme suivant:

Algorithme A.2. Algorithme ALS pour NMF

Entrée: *p,k*
Sortie: *W et H*
Début
8. W = *rand*(*p*,*k*);
9. Pour *i* = 1 : *maxiter* faire,
10. (Als) Résoudre H : W'WH = W'X,
11. (nonnég) Mettre tous les éléments négatifs dans H à 0.
12. (Als) Résoudre W: HH'W'= HX',
13. (nonnég) Mettre tous les éléments négatifs dans W à 0.
14. Fin Pour
Fin

A.2.1.3. La Descente du Gradient Stochastique

La méthode de la descente du gradient stochastique est une méthode d'optimisation de la descente de gradient pour minimiser une fonction objective qui est écrit comme une

somme de fonctions différentiables. Pour chaque cas de formation donné, le système prédit la valeur $\hat{r}_{u,i}$ et calcule l'erreur de prédiction associée comme suit :

$$e_{u,i} = r_{u,i} - w_u h_i \quad \text{(A.19)}$$

La règle d'apprentissage est :

$$h_i = h_i + \gamma.(e_{u,i}.w_u - \gamma.h_i) \quad \text{(A.20)}$$

$$w_u = w_u + \gamma.(e_{u,i}.h_i - \gamma.w_u) \quad \text{(A.21)}$$

A.1.3. Avantages et inconvénients

La méthode NMF possède un nombre d'avantages ainsi qu'inconvénients par rapport aux algorithmes de réduction de la dimension, en particulier la méthode SVD.

- Faible densité et positivité : la factorisation non négative donne des facteurs à faible densité et implique la contrainte de non négativité sur leur contenus, et elle maintient ces propriétés de la matrice d'origine.
- La réduction de stockage : les facteurs sont rares, ce qui se traduit également par une application plus facile de nouvelles données. En raison des contraintes de non-négativité, la NMF produit une représentation basée pièces additives de données. Une conséquence de ceci est que les facteurs W et H sont généralement de nature dispersée, économisant ainsi beaucoup de stockage par rapport aux facteurs denses de la SVD.
- L'interprétation : La NMF a aussi des avantages impressionnants en termes d'interprétation de ses facteurs, où les vecteurs de base correspondent naturellement aux propriétés conceptuelles de données, qui est aussi une conséquence des contraintes de non-négativité.

Bien sûr, la méthode NMF possède ses inconvénients aussi :

- Autres factorisations populaires, en particulier la SVD, ont des points forts concernant l'unicité et le calcul robuste, ce qui n'est pas le cas avec la méthode NMF qui ne donne pas un minimum global unique.
- Le problème d'optimisation de la fonction objective est convexe en W ou H, mais pas dans les deux, ce qui signifie que les algorithmes peuvent seulement garantir la convergence vers un minimum local **[LAN et al., 06].** Dans la pratique, les utilisateurs utilisant la méthode NMF comparent souvent le minimum local de

plusieurs points de départ différents, en utilisant les résultats du meilleur minimum local trouvé.

- L’obtention de différents facteurs NMF produits, non seulement par l’utilisation de différents algorithmes NMF, mais aussi en utilisant le même algorithme NMF avec des paramètres légèrement différents.
- La faiblesse la plus grave de la NMF est son problème de convergence, du faite qu’elle ne donne pas une factorisation unique contrairement à la SVD qui donne une unique factorisation.
- Les différents algorithmes NMF peuvent converger vers différents minimas locaux (et même cette convergence aux minima locaux ne sont pas garanties), ce qui rend la méthode d’'initialisation de l'algorithme critique pour atteindre la convergence.

ANNEXE B

MOVIELENS DATASET

Les ensembles de données *MovieLens* ont été recueillies par le Projet de recherche *GroupLens*[6] à l'Université du Minnesota. Cet ensemble de données comprend 100.000 notes dans une échelle de (1-5) à partir de 943 utilisateurs sur 1682 films. Où, chaque utilisateur a évalué au moins 20 films.

Les données ont été collectées à travers le site *MovieLens* (*movielens.umn.edu*) au cours de la période de sept mois à partir du 19 Septembre 1997 au 22 Avril, 1998. Ces données ont été nettoyées, où les utilisateurs qui avaient moins de 20 évaluations ou qui n'ont pas eu des informations démographiques complètes ont été retirés de cet ensemble de données. Cet ensemble de données a été utilisé pour la première fois par Herlocker et al., **[HER et al., 99]**

B.1. INFORMATIONS SUR LE PROJET DE RECHERCHE GroupLens

Le projet de recherche *GroupLens* est un groupe de recherche au Département d'informatique et de génie de l'Université de Minnesota. Les Membres du projet de recherche *GroupLens* sont impliqués dans de nombreux projets de recherche liés aux domaines du filtrage d'information, filtrage collaboratif, et les systèmes de recommandation. Le projet est dirigé par les professeurs *John Riedl* et *Joseph Konstan*. Le projet a commencé à explorer le filtrage collaboratif automatisé en 1992, mais est surtout connu pour son vaste essai mondial d'un système de filtrage collaboratif automatisé pour les nouvelles Usenet en 1996. La technologie développée dans le procès Usenet a formé la base pour la formation des perceptions nettes, Inc., qui a été fondée par des membres de recherche *GroupLens*. Depuis, le projet a élargi son champ d'application à la recherche de solutions globales de filtrage de l'information,

[6] http://www.grouplens.org/

l'intégration dans les méthodes basées sur le contenu ainsi que l'amélioration de la technologie actuelle de filtrage collaboratif.

De plus amples informations sur le projet *GroupLens* recherche, y compris les publications de recherche, peut être trouvé sur le site Web suivant: *http://www.grouplens.org/*

GroupLens recherche exploite actuellement un recommandeur de film basé sur le filtrage collaboratif:

http://www.movielens.org/

B.2. Description détaillée des fichiers de données

La base de données *MoviLens* contient plusieurs fichiers pour représenter les différents ensembles de données et pour décrire les films, les utilisateurs, la relation entre utilisateurs et items, ainsi que les scripts de génération des différents ensembles de données. Nous présentons dans les sections suivantes une description détaillée des différents fichiers, ceux disponibles avec la base et ceux que nous avons créés et ajouté afin d'accomplir les expérimentations.

B.1.1 Les fichiers disponibles avec la base

ml-data.tar.gz : Fichier compressé de type .tar qui contient toute la base de données téléchargée.

u.data : L'ensemble de données *u.data* est l'ensemble plein, c.à.d. l'ensemble qui contient les 100 000 évaluations des 943 utilisateurs sur les 1682 films, où, chaque utilisateur a évalué au moins 20 films. Dans cet ensemble, les utilisateurs et les articles sont numérotés consécutivement à partir de 1, mais les relations entre les utilisateurs et les items sont arrangés dans l'ordre de leur collecte. La relation entre un utilisateur et un item dans l'ensemble de données *u.data* est représentée avec une tab séparée comme suit :

ID utilisateur | Id item | Note | Horodatage (timestamp)

Un exemple de cet ensemble de données est le suivant :

1 1 5 874965758 : où, cette ligne indique que l'utilisateur 1 a évalué l'item 1 avec une note (vote) égale à 5 au moment 874965758s, qui est le temps de l'évaluation en secondes depuis le 1/1/1970.

u.info : Ce fichier contient le nombre d'utilisateurs, d'items, et de notes dans l'ensemble de données *u.data*, et il est constitué de 3 lignes:

943 users

1682 items

100000 ratings

u.item : Ce fichier contient les informations sur les items (films), représentés sous forme de tab séparée comme suit:

Film id | *Titre du film* | *date de diffusion* | *date de sortie de la vidéo* | *IMDb URL* | *inconnu* | *Action* | *Aventure* | *Animation* | *Enfant* | *Comédie* | *Crime* | *Documentaire* | *Drama* | *Fantaisie* | *Film-Noir* | *Horreur* | *Musique* | *Mystère* | *Romance* | *science Fiction* | *Thriller* | *guerre* | *Western* |

Où, les identifiants des films sont ceux utilisés dans la base *u.data*. Comme exemple nous citons la description du premier film *i1*:

1|Toy Story (1995)|01-Jan-1995||http://us.imdb.com/M/title-exact?Toy%20Story%20(1995)|0|0|0|1|1|1|0|0|0|0|0|0|0|0|0|0|0|0|0

Où, les 19 derniers champs présentent les genres des films, avec *i1* indique que le film est de ce genre, un 0 indique qu'il n'est pas; et les films peuvent être en plusieurs genres à la fois, ce qui est évidemment le cas avec le film i1 qui appartient à 3 genres.

u.genre : Ce fichier contient la liste des genres des films disponibles et qui correspondent aux genres dans la base *i.item* avec le même ordre.

unknown|0

Action|1

Adventure|2

Animation|3

Children's|4

Comedy|5

Crime|6

Documentary|7

Drama|8

Fantasy|9

Film-Noir|10

Horror|11

Musical|12

Mystery|13

Romance|14

Sci-Fi|15

Thriller|16

War|17

Western|18

u.user: Ce fichier contient les données démographiques sur les utilisateurs, sous forme d'une tab séparée de données démographiques simples pour les utilisateurs comme suit :

ID utilisateur | *âge* | *genre* | *profession* | *code postal*

Où, les identifiants des utilisateurs (*ID utilisateur*) sont ceux utilisés dans l'ensemble de données *u.data*.

A titre d'exemple, l'utilisateur ayant l'identifiant 7, qui est un homme âgé de 57 ans et qui travaille comme administrateur avec un code postal 91344, est représenté dans la base comme suit:

7|57|M|administrator|91344

u.occupation : Ce fichier contient la liste des professions des différents utilisateurs : *administrator, artist, doctor, educator, engineer, entertainment, executive, healthcare, homemaker, lawyer, librarian, marketing, none, other, programmer, retired, salesman, scientist, student, technician, writer.*

u1.base, u1.test, u2.base, u2.test, u3.base, u3.test, u4.base, u4.test, u5.base, u5.test : Les ensembles de données *u1.base* et *u1.test* jusqu'à *u5.base* et *u5.test* sont des sous ensembles de données de *u.data* avec 80% de données dans *.base* et 20% dans *.test*.

Les ensembles de u1, ..., u5 sont des ensembles de test disjoints et ils sont construits pour être utilisés dans la validation croisée. Ces ensembles de données peuvent être générés à partir *u.data* par le fichier *mku.sh*.

ua.base, ua.test, ub.base, ub.test : Les ensembles de données *ua.base*, *ua.test*, *ub.base* et *ub.test* divisent l'ensemble de données *u.data* en un ensemble d'entrainement et un ensemble de test, avec 10 notes par utilisateur dans les ensembles de test qui sont disjoints. Les quatre ensembles de données peuvent être générés à partir *u.data* par le fichier *mku.sh*.

mku.sh : Ce fichier contient un script pour générer tous les ensembles de données de *u.data*.

allbut.pl : Ce fichier contient le script qui génère les ensembles d'apprentissage et de test, avec contrainte de sélection de seulement *n* évaluations des utilisateurs dans les données d'apprentissage.

```
#!/usr/local/bin/perl
# get args
if (@ARGV < 3) {
  print STDERR "Usage: $0 base_name start stop max_test [ratings ...]\n";
  exit 1;
}
$basename = shift;
$start = shift;
$stop = shift;
$maxtest = shift;
# open files
open( TESTFILE, ">$basename.test" ) or die "Cannot open $basename.test for
writing\n";
open( BASEFILE, ">$basename.base" ) or die "Cannot open $basename.base for
writing\n";
# init variables
$testcnt = 0;
while (<>) {
  ($user) = split;
  if (! defined $ratingcnt{$user}) {
        $ratingcnt{$user} = 0;
  }
  ++$ratingcnt{$user};
  if (($testcnt < $maxtest || $maxtest <= 0)
  && $ratingcnt{$user} >= $start && $ratingcnt{$user} <= $stop) {
        ++$testcnt;
        print TESTFILE;
  }
  else {
        print BASEFILE;
  }
}
```

B.1.2 Les fichiers ajoutés:

Dans cette section, nous présentons une description des fichiers que nous avons créés et ajoutés à la base de données *MovieLens* pour être utilisés dans les expérimentations.

u1.data: L'ensemble de données organisé de la base *u.data,* en commençant par l'utilisateur *u1* jusqu'à l'utilisateur *u943,* où toutes les évaluations faites par le même utilisateur sont arrangées consécutivement.

i1.item: L'ensemble de données des items que nous avons sélectionné à partir de l'ensemble *i.item*. Cet ensemble contient tous les items mais seulement les 19 colonnes correspondantes aux genres des items sont prises en comptes pour représenter les items comme suit :

$Film\ id\ |g_1|g_2|.....|g_{19}$

Donc, la liste des caractéristiques utilisées pour représenter l'item **i1** devient comme suit :

1| 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0

DATABASE : la matrice d'évaluations utilisateur-item que nous avons générée à partir de l'ensemble de données *u1.data* en utilisant le fichier créé *loaddataset*.

Loaddataset : Ce fichier contient un script pour créer la matrice d'évaluation utilisateur-item en transformant les données incluses dans *u1.data*, en une matrice qui contient l'ensemble des utilisateurs en lignes et les items en colonnes, et les évaluations faites par les utilisateurs aux items sont mises dans les cases correspondantes à V(u,i).

```
clear all,close all,clc;
[name,path,n] = uigetfile;
   if n
      M=load(strcat(path,name));
      [Mdim,Mcol]=size(M);
      for i=1:Mdim
         V(M(i,1),M(i,2))=M(i,3);
      end
   end
 save('DATABASE','V');
```

Sparcity_20, Sparcity_40, Sparcity_60 et Sparcity_80 : des sous ensembles de données sélectionnés aléatoirement à partir de la matrice d'évaluation V contenant 20%,40%,60% et 80%, respectivement, en utilisant le code suivant :

```
clear all,close all,clc;
load('DATABASE','V');
% N % de users
n = 20;
[N,M] = size(V);
N1 = floor((n*N)/100);
if N1==0, N1+1;end
V1 = V(1:N1,:);
save(strcat('base_',num2str(n)),'V1','n');
```

À chaque fois qu'on veut sélectionner un sous ensemble on spécifie juste le nombre *n* des utilisateurs.