Étude d'interprétabilité de la factorisation de matrice pour les systèmes de recommandation

—— Master IASD : Datascience Lab. ——

Laurene BOUSKILA - Julien SENTUC - Matthieu SERFATY - Mathurin VIDEAU

21 Octobre 2020

1 Introduction

Les systèmes de recommandation sont des systèmes permettant de donner une recommandation personnalisée pour chaque individu. On peut lui trouver de nombreuses applications, notamment dans le commerce en ligne ou encore dans la musique. Mais le terrain le plus étudié reste celui de la recommandation de film.

Dans la littérature, on peut distinguer deux types d'approches:

- Basée sur le contenu, chaque objet et utilisateur sont décrits par des "descripteurs" qui les caractérisent. Ici, on cherche à mettre en relation ces descripteurs pour établir les préférences des utilisateurs.
- Basée sur le filtrage collaboratif, cette approche utilise une matrice R dans laquelle les notes de chaque utilisateur u et i sont consignées. De cette manière, on recherche une corrélation entre les utilisateurs pour déterminer leurs goûts. C'est particulièrement cette approche qui nous intéresse et qui remporte le plus de succès ces dernières années.

L'algorithme le plus basique et intuitif utilisé dans le domaine du filtrage collaboratif est basé sur les K plus proches voisins (KNN). Pour recommander un objet à un utilisateur u, l'algorithme va au préalable trouver d'autres utilisateurs ayant des goûts similaires à u, appelé voisin de u, et recommande ensuite à u les objets ayant obtenu une "bonne" note par ces voisins.
Grâce au déroulement très intuitif de l'algorithme, il est très facile de justifier l'objet recommandé. Le problème de cette méthode est qu'elle est très difficile à mettre en oeuvre dès lors que le volume de données est conséquent.

Pour remédier à ce problème, des algorithmes plus modernes appliquent la factorisation de matrice. Cette opération de factorisation a pour but de synthétiser le comportement des utilisateurs à l'aide de K facteurs latents. Ces facteurs sont appris lors d'un processus d'entraînement. Malheureusement, il est difficile d'attribuer une signification précise à ces facteurs; si bien que les prédictions produites sont inintelligibles. On perd ainsi la propriété intuitive des KNN.

C'est pourquoi, dans cette étude, on cherche à redonner du sens à ces facteurs latents, de telle sorte que : si le système de recommandation prédit qu'un utilisateur u va apprécier un objet i, alors il existe un groupe d'utilisateurs ayant les mêmes goûts que u et qui, de façon similaire, ont noté de manière positive l'objet i. Nous nous intéresserons à cet aspect au travers de trois approches: la factorisation de matrice classique (Baseline), la factorisation de matrice non-négative (NMF) et la factorisation de matrice à partir d'un modèle bayésien.
Chacune de ces expérimentations sera testée à partir de deux configurations expérimentales. La première consiste en la mise en place d'un jeu de données synthétique simple, dans lequel des groupes d'utilisateurs sont introduits et donc connus afin de pouvoir tester l'interprétabilité de ces méthodes. La seconde comportera l'application de ces méthodes sur un dataset réél : "Movielens 100k".

2 Méthodes sélectionnées

Notre projet consiste en l'analyse de la factorisation de matrice comme solution du "collaborative filtering". Notre principale investigation se portera sur l'interprétation des facteurs latents des coefficients des matrices U et I.

Rappelons que le but mathématique de l'expérience et de représenter la matrice de notation R en une multiplication de matrices de dimensions inférieures U et I.

Soit: $R = I \times U^T$

- $R(m \times n)$ matrice de notation - utilisateurs/objets
- $U(m \times k)$ matrice utilisateurs/caractéristiques
- $I(k \times n)$ matrice caractéristiques/objets

Dans chaque approche, les matrices U et I sont apprises de manière à reconstruire les termes connus de R. La différence principale entre chaque approche réside dans les contraintes appliquées sur U et I.

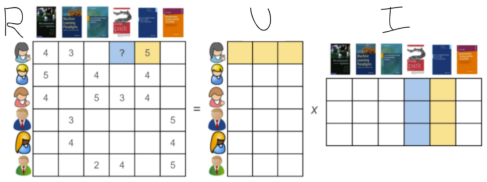

Figure 1: Matrix Decomposition

2.1 Factorisation de matrice (Baseline)

Cette approche formule la factorisation de matrice comme un problème d'optimisation à partir de la fonction de coût suivante:

$$C(U, I) = ||R - IU^T||_F^2 + \lambda||U||_F^2 + \mu||U||_F^2$$

Les paramètres λ et μ sont des hyper-paramètres pour la régularisation. La différence entre les matrices R et IU^T quantifie la *similarité* entre les deux matrices. On peut remarquer que l'optimisation de ce problème est non-convexe, mais le devient lorsque l'on fixe U ou I.

L'optimisation peut s'effectuer de différentes façons; les deux méthodes implémentées dans cette étude sont la descente de gradient et l'ALS (*Alternate Least Square*). Néanmoins, elles suivent un schéma similaire que l'on peut illustrer de la manière suivante :

1. Les matrices U et I sont initialisées aléatoirement
2. Elles sont ensuite mises à jour de façon itérative jusqu'à obtenir une certaine convergence. À noter que les matrices sont mises à jours uniquement à partir des notes connues de R. Toutes les notes inconnues, représentées par 0, sont donc occultées lors de l'apprentissage.

L'étape 1 est importante du fait de la nature non-convexe du problème. Ainsi, différentes initialisations ont été testées. Ces variations d'initialisation peuvent donner lieu à des variations de performance significatives.

L'étape 2 dépend de la méthode que l'on souhaite utiliser, nous allons décrire en détail celle de la descente du gradient :
Dans cet algorithme, les matrices U et I sont itérées dans la direction opposée à la descente de gradient selon les équations suivantes:

$$I_{t+1} = I_t - \eta_t \frac{\partial C}{\partial I}(U_t, I_t), U_{t+1} = U_t - \xi_t \frac{\partial C}{\partial U}(U_t, I_t)$$

En effet, l'itération des matrices U et I se fait au travers des hyper-paramètres, η et ξ qui correspondent aux taux d'apprentissage respectifs des matrices U et I.
On notera que le choix de ces taux est important car un taux trop élevé pourrait résulter en un pas d'erreur trop grand (et donc on pourrait "manquer" le minimum de la fonction de coût) tandis qu'une valeur trop petite impliquerait une convergence plus lente.

Bien que la mise à jour de U et I à partir de la méthode ALS soit différente et donne une convergence plus rapide, l'entraînement par descente du gradient a été privilégiée au sein de cette étude car elle est aussi utilisée dans la NMF. Nous verrons également plus tard qu'elle donne de meilleurs résultats sur le jeux de données réel: "Movielens 100k".

2.2 Factorisation de matrice non-négative

La factorisation de matrice non-négative (FMN) a été initialement introduite en traitement d'images pour la reconnaissance des visages, car elle permettait d'identifier les différentes parties du visage (yeux,bouche,nez,...). Pour le filtrage collaboratif de films, elle permet d'identifier les films dont les notes sont fortement corrélées selon un groupe d'utilisateurs. La FMN permet donc d'obtenir une explication des notes qu'elle prédit. On essaye ainsi de donner du sens aux facteurs latents, ce sont les k éléments introduits lors de la réduction de dimension. La FMN reprend le principe de la factorisation de matrice classique à laquelle on ajoute la contrainte que les éléments des deux matrices soient tous positifs. Un article (film), est donc une addition pondérée de vecteurs de bases positifs, ce qui correspond à l'intuition d'une décomposition par partie :

$$x_{ij} = [w_{i1}, w_{i2}, ..., w_{ik}] \times [h_{j1}, h_{j2}, \ldots, h_{jk}]^T = \sum_{p=1}^{k} w_{ip} \times h_{pj}$$

Dans cette équation, la matrice W représente les poids et la matrice H représente les vecteurs de base (les composants). Il est également possible de considérer les poids à gauche et les composants à droite. Ces deux modèles sont d'ailleurs équivalents. Le premier recherche des utilisateurs fictifs représentatifs dont les goûts représentent une certaine communauté, le second recherche des communautés d'utilisateurs ayant les mêmes goûts. Dans cette étude, nous utilisons ce dernier.
Le travail réalisé pour la partie FMN se base sur les travaux de [2] et [3] pour l'interprétation des recommandations. Dans notre cas, la matrice de notes est incomplète (tous les utilisateurs ne notent pas tous les films), elle contient d'ailleurs peu de notes. Deux méthodes sont utilisées et comparées dans cette étude pour répondre à ce problème : EM-based (Expectation–Maximization) et WNMF (Weighted Non-negative Matrix Factorization). Dans cette section nous prenons $V = U^T$. On cherche donc I et V, tel que $R \approx I \times V$. On note ainsi la matrice obtenue $\bar{R}$.

2.2.1 EM procédure

L'algorithme EM permet de trouver les paramètres de vraisemblance maximale d'un modèle lorsque les données sont incomplètes. Il se décompose en deux étapes. La première est la phase "Expectation"; elle consiste à combler la matrice de notation afin d'en obtenir une complète. À la première itération, la matrice est complétée pour chaque article avec sa note moyenne. Pour les itérations suivantes, on reprend les valeurs du modèle de l'étape, c'est à dire $I^t \times V^t$. La seconde phase "Maximization", tente de trouver I et V minimisant la norme de Frobenius pondérée entre $\bar{R}$ et $I \times V$ avec de la FMN. Les formules de mises à jour qui en découlent sont pour I_{ij} et V_{ij}:

$$I_{ij}^{t+1} = I_{ij}^t \times \frac{(\bar{R}V^T)_{ij}}{(IVV^T)_{ij}}, V_{ij}^{t+1} = V_{ij}^t \times \frac{(I^T\bar{R})_{ij}}{(I^TIV)_{ij}}$$

On effectue ainsi une FMN à chaque itération de l'algorithme. L'algorithme EM consiste ainsi en N itérations de ces deux phases.

2.2.2 WNMF

Une alternative à l'approche ci-dessus est la "weighted non-negative matrix factorization". Elle consiste à maximiser la log-vraisemblance des données observées. Cela revient à minimiser $\sum_{ij}(\bar{R}_{ij} - (IV)_{ij})^2$. Avec cette méthode, on obtient les formules de mise à jour suivantes :

$$I_{ij}^{t+1} = I_{ij}^t \times \frac{((W \otimes \bar{R})V^T)_{ij}}{(I^T(W \otimes (IV)))_{ij}}$$

$$V_{ij}^{t+1} = V_{ij}^t \times \frac{((U^T(W \otimes \bar{R}))_{ij}}{(I^T(W \otimes (IV)))_{ij}}$$

Cette méthode ne nécessite qu'un tour de changements pour I et V et par conséquent, est plus rapide que l'algorithme EM.

2.3 Factorisation de matrice par approche bayésienne

Dans cette section nous allons voir une autre méthode de factorisation de matrice qui est basée sur une approche bayésienne. Cette méthode est très semblable aux k plus proches voisins (KNN) en terme d'interprétabilité. Le but est de retrouver des groupes d'utilisateurs ayant les mêmes goûts à partir d'une matrice de rating M. Avec ce type de factorisation de matrice, on espère donner du sens aux facteurs latents obtenus dans les matrices U et I, respectivement liées aux utilisateurs et aux items que les utilisateurs notent.

Tout comme la factorisation de matrice classique, on définit un nombre K de groupes d'utilisateurs que l'on souhaite retrouver dans les matrices U et I. Ainsi, avec cette méthode dans la matrice U on souhaite regrouper les probabilités qu'un utilisateur appartienne à un groupe k. On note $a_{u,k}$, la probabilité qu'un utilisateur u appartienne à un groupe k. Quant aux facteurs latents de la matrice I, ils représentent la probabilité qu'un utilisateur u d'un groupe k aime un item i que l'on note $b_{i,k}$. Ici on voit bien que, contrairement à la factorisation de matrice classique, on peut facilement interpréter les coefficients des matrices U et I. Tous les coefficients de ces matrices sont donc compris entre 0 et 1. De plus, pour la matrice U on impose que les probabilités $a_{u,k}$ déterminées pour un utilisateur u, respectent l'égalité suivante:

$$\sum_{k=1}^{K} a_{u,k} = 1$$

Notez que cela n'empêche pas le fait qu'un utilisateur appartienne à plusieurs groupes.

À partir de ces deux matrices U et I on peut déterminer $p_{u,i}$, la probabilité qu'un utilisateur u aime un item i. Finalement, à partir de cette probabilité $p_{u,i}$ on peut estimer le rating de tous les utilisateurs u pour tous les items i. Dans ce cas, on considère que les notes des utilisateurs peuvent prendre des valeurs entières comprises entre 0 et 5 : 0 s'il n'a pas du tout aimé l'item et 5 s'il l'a beaucoup aimé. Ce rating $q_{u,i}$ se définit de la façon suivante:

$$q_{u,i} = \begin{cases} 1 & \text{si } 0 < p_{u,i} < 0,2 \\ 2 & \text{si } 0,2 \leq p_{u,i} < 0,4 \\ 3 & \text{si } 0,4 \leq p_{u,i} < 0,6 \\ 4 & \text{si } 0,6 \leq p_{u,i} < 0,8 \\ 5 & \text{si } 0,8 \leq p_{u,i} \leq 1 \end{cases}$$

Les estimations de notes sont donc obtenues à partir des probabilités $a_{u,k}$ et $b_{i,k}$. Contrairement à la NFM ici on se base uniquement sur un modèle probabiliste baysien pour calculer les facteurs latents en représentant les probabilités d'appartenance aux groupes.

3 Protocole d'expérimentation : Expérience Contrôlée

Pour les prochaines méthodes de factorisation de matrices, nous allons utiliser un jeu de données construit. Le but étant, à cette étape du projet, de contrôler l'expérience et les résultats en observant les prédictions de notre algorithme sur cet ensemble maîtrisé.

En effet, notre base de données contient des groupes caractéristiques d'utilisateurs qui seront "cachés" parmi le reste des données. La finalité est d'observer et de contrôler la cohérence de nos résultats ainsi que la logique de nos algorithmes et en fonction de ces derniers, retrouver ou non un groupe caché.

3.1 Génération des données

La base de données a été constituée de manière simple afin d'obtenir des résultats test.
Sa génération se fait à partir de quatre paramètres simples : N le nombre d'objets, M le nombre d'utilisateurs, G le nombre de groupes ou leur taille et r le nombre de notes.

Dans cette base de données simple, une partition des objets est réalisée formant G groupes distincts. À chacun de ces ensembles d'objets, on peut attribuer un numéro. Ce numéro de groupe est attribué aléatoirement de manière uniforme à chacun des utilisateurs et correspond au groupe d'objet apprécié par le groupe. Chaque utilisateur va ensuite noter de manière positive un certain nombre d'objets de son groupe en leur donnant une note de 5 et une autre partie des objets hors du groupe avec une note de 1. Ce nombre

d'objet noté est paramétré par r. On peut remarquer que pour la factorisation classique et la NMF, si les utilisateurs n'ont que des éléments positifs à 5, le produit de U et I comblera les notes inconnues, uniquement par des 5. De ce fait, aucun groupe ne sera trouvé sous ces conditions.

Ainsi, tous les groupes d'utilisateurs et groupes d'objets sont connus. Nous pouvons les comparer à ceux trouvés pour les différents algorithmes.

3.2 Résultats et simulations sur les différentes méthodes

3.2.1 factorisation de matrice : test d'interprétabilité

Les factorisations de matrice classique sont très difficiles à interpréter du fait que les facteurs latents peuvent prendre n'importe quelles valeurs, y compris négatives. Or cela n'a beaucoup de sens d'avoir une corrélation négative par rapport à un autre groupe. On ne peut donc pas donner de sens direct à la valeur de k ni aux valeurs de U et de I. Plusieurs tests ont été menés sur différents jeux de données synthétiques avec un nombre d'utilisateurs, d'objets et de groupes variable. Si l'on affiche les facteurs latents sur le plan par l'intermédiaire d'un PCA, on retrouve les groupes de manière assez claire pour une matrice très complète (Fig.2). Cependant, dès lors que la matrice possède environ moins des trois-quarts de ces valeurs, le résultat devient inintelligible(Fig.3).

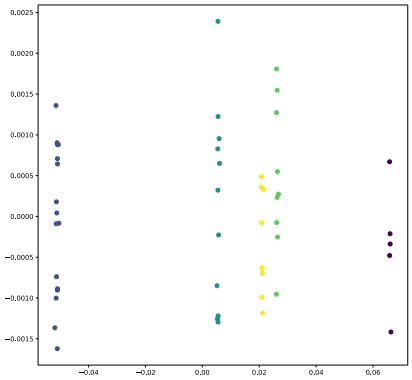

Figure 2: PCA sur 5 facteurs latents, matrice complète

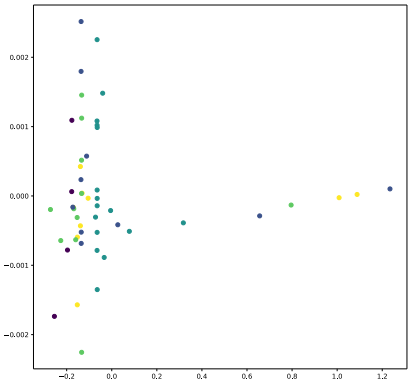

Figure 3:
PCA sur 5 facteurs latents, matrice incomplète

3.2.2 Factorisation de matrice non-négative : test d'interprétabilité

Les tableaux ci-dessous présentent les résultats de la FMN pour l'identification des groupes, en connaissant le k à l'avance, selon différentes configurations de: nombre de groupes, nombre d'utilisateurs, nombre de films, et pourcentages de notes.

Tableau 1 : Petit dataset (10 utilisateurs, 10 films, 3 groupes, 50% de notes)

Groupes	1	2	3
Utilisateurs réels	6	1,2,4,7,8,9	0,3,5
Utilisateurs trouvés	6	1,2,4,7,8,9	0,3,5

Tableau 2 : Dataset moyen peut de notes (20 utilisateurs, 20 films, 8 groupes)

Groupes	1	2	3	4	5	6	7	8
Utilisateurs réels	9,12,17	7,18,19	1,3,14	8	0,4,5,11	6,10	2	13,15,16
Utilisateurs trouvés	9,12	4,7	1,3	8,13,18	0,5	6,10,11,14	2,19	15,16,17

Tableau 3 : Dataset moyen plus de notes (20 utilisateurs, 50 films, 5 groupes)

Groupes	1	2	3	4	5
Utilisateurs réels	8,10,16,19	3,4,7,14,18	2,12,15	11,13	0,1,5,6,9,17
Utilisateurs trouvés	8,10,16,18	3,4,7,14	2,12,15	11,13	0,1,5,6,9,17,19

On remarque que pour un grand nombre de groupes, il est nécessaire d'avoir plus de notes pour identifier plus précisément les groupes. On notera également que les groupes ne sont pas trouvés parfaitement, à l'exception du petit dataset. Nous avons également éssayé de trouver le k du dernier dataset en évaluant la MAE pour différents k entre 2 et 8. Deux groupes de k sont ainsi identifiés les k pairs et les k impairs. Les k impairs présentent de meilleurs résultats.

Tableau 4 : Dataset 3 MAE pour différentes valeurs k

k	2	3	4	5	6	7	8
MAE	1.242356	0.984905	1.242356	0.984679	1.242356	0.984659	1.242356

3.2.3 Factorisation par approche bayésienne : test d'interprétabilité

Malheureusement pour cette approche nous n'avons pas réussi à mettre à place l'algorithme pour tester le modèle probabiliste sur nos propres données. Nous utilisons donc les exemples et résultats du papier [1]. On rappelle que cette méthode permet de calculer les probabilités $a_{u,k}$ et $b_{k,i}$, respectivement la probabilité qu'un utilisateur u appartienne à un groupe k d'utilisateur et la probabilité qu'un utilisateur du groupe k aime un item i. On utilise la matrice de rating suivante pour simuler la prédiction :

	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}	I_{11}	I_{12}	I_{13}	I_{14}	I_{15}
U_1	5	5	5	5	5	•	1	•	•	•	1	•	•	•	•
U_2	5	5	5	5	5	•	•	1	•	•	•	•	•	•	•
U_3	5	5	•	5	5	•	•	•	•	•	•	3	•	•	•
U_4	•	•	1	•	•	5	5	5	5	5	•	1	•	2	•
U_5	•	3	•	•	2	5	5	•	5	5	•	2	•	4	•
U_6	•	1	•	•	•	5	5	5	5	5	•	•	•	•	•
U_7	•	•	•	4	•	•	•	•	1	•	5	4	5	5	5
U_8	•	1	•	•	•	•	4	•	•	•	5	5	5	4	5
U_9	•	•	•	•	•	3	•	•	3	•	5	4	5	5	5
U_{10}	5	5	5	5	5	1	5	2	1	5	•	•	•	•	•
U_{11}	•	1	•	•	•	5	5	5	5	5	•	•	5	•	•

Figure 4: Exemple de matrice de rating

Ici la matrice de rating est composée de 11 utilisateurs et de 15 items. Dans un premier temps on peut voir que plusieurs groupes se dégagent très clairement. Le groupe 1 avec $\{U_1, U_2, U_3\}$, le groupe 2 avec $\{U_4, U_5, U_6\}$ et le groupe 3 avec $\{U_7, U_8, U_9\}$. Les utilisateurs U_{10} et U_{11} semblent avoir des goûts un peu différents. On décide donc de fixer K à 3 pour retrouver les trois groupes d'utilisateurs.

	G_1	G_2	G_3
U_1	**0.98**	0.01	0.01
U_2	**0.98**	0.01	0.01
U_3	**0.98**	0.01	0.01
U_4	0.01	**0.98**	0.01
U_5	0.01	**0.98**	0.01
U_6	0.01	**0.98**	0.01
U_7	0.01	0.01	**0.98**
U_8	0.01	0.01	**0.98**
U_9	0.01	0.01	**0.98**
U_{10}	**0.86**	**0.13**	0.01
U_{11}	0.01	**0.98**	0.01

Figure 5: Prédiction de la matrice U avec l'approche probabiliste

On voit que l'on retrouve très facilement les trois groupes que nous avions prévu d'avoir avant de faire la prédiction avec de fortes probabilités. Il est plus facile de donner du sens à ces résultats plutôt qu'à ceux

que l'on pourrait obtenir avec une factorisation de matrice classique. Par exemple, on peut dire suite à cette prédiction que l'utilisateur U_1 appartient au groupe 1 avec une probabilité $a_{1,1} = 0,98$. On note que l'utilisateur U_{10} a une petite probabilité d'appartenir à deux groupes différents. Ce qui est lié au fait qu'il a des goûts différents du reste des autres utilisateurs.

	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}	I_{11}	I_{12}	I_{13}	I_{14}	I_{15}
G_1	0.9	0.9	0.9	0.9	0.9	0.3	0.3	0.3	0.3	0.7	0.3	0.6	0.5	0.5	0.5
G_2	0.5	0.4	0.4	0.5	0.4	0.9	0.9	0.9	0.9	0.9	0.5	0.4	0.7	0.6	0.5
G_3	0.5	0.3	0.5	0.7	0.5	0.6	0.7	0.5	0.4	0.5	0.9	0.8	0.9	0.8	0.9

Figure 6: Prédiction de la matrice I avec l'approche probabiliste

Enfin on peut obtenir la prediction de la matrice de rating à partir de ces deux matrices U et I en suivant les règles énoncé dans la section 2.3.

4 Application sur des données réelles : "Movielens 100k"

Dans cette partie, nous allons appliquer notre algorithme précédemment contrôlé par une génération spécifique et maîtrisée de la base de données, sur une base de données générale appelée: MovieLens.

En d'autres termes, en observant les garanties expérimentales du data set contrôlé, parvient-on à retrouver les groupes caractéristiques lorsque l'expérience est appliquée dans un cas réel ?
C'est ce que nous allons essayer de savoir au travers des analyses qui suivent.

En outre nous pouvons comparer les performances des différents algorithmes à partir de la RMSE (*Root mean square error*) et de la MAE=

4.1 Présentation du jeux de données MovieLens

La base de données met en relation des utilisateurs et la note qu'ils ont attribuée à chaque film. Elle est constituée de 600 utilisateurs et de 9000 films pour un total de 100 000 notes. On peut tout d'abord tenter d'avoir un premier aperçu général de la base de données en observant quelques chiffres. Tout d'abord, la matrice semble très creuse, elle est composée à 98% de zéros. Ce qui justifie le fait qu'elle puisse être synthétisée par des matrices de rang inférieur. Regardons plus en détail comment se répartissent les notes.

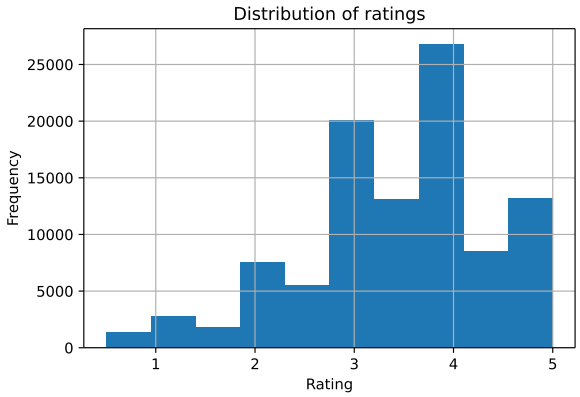

Figure 7: Fréquence d'apparition des notes

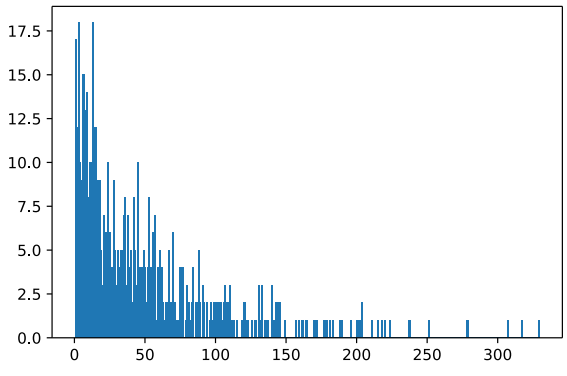

Figure 8:
Nombre de notes (x) par utilisateur (y)

De manière générale, on peut voir que les utilisateurs notent majoritairement de façon positive et très peu de notes négatives sont perceptibles (Fig.7). Cela laisse sous-entendre qu'ils notent plus souvent les films

qui leur plaisent plutôt que ceux qu'ils n'ont pas apprécié. La Fig.8, quant à elle, nous fait remarquer que la moitié des utilisateurs ont attribué au moins 34 notes, dont 17 d'entre eux n'ont attribué qu'une seule note.

4.2 Résultats empiriques

4.2.1 Résultats empiriques de la factorisation de matrice

En premier lieu, nous avons estimé les paramètres optimaux pour l'algorithme. Le paramètre le plus important étant k, qui contrôle le nombre de valeurs latentes. Ainsi en appliquant l'algorithme avec différentes valeurs de k, λ, μ, et le bon taux d'apprentissage (*learning rate*), on obtient la courbe de droite (Fig.10)

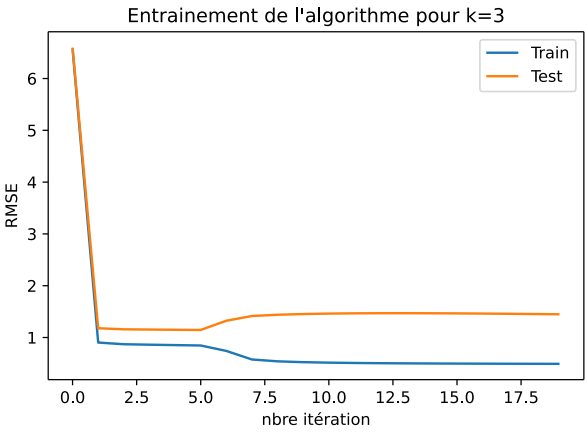

Figure 9: ALS, courbe d'apprentissage

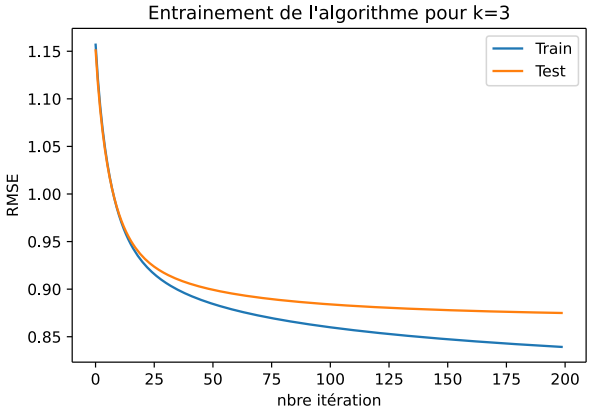

Figure 10:
Descente de gradient, courbe d'apprentissage

Les valeurs de RMSE ont été obtenues par "cross-validation" avec un nombre de plis égal à quatre. Les barres d'erreur sont représentées pour un intervalle de confiance de 95%. Sur le graphique (Fig.11) on peut voir qu'une valeur de K comprise entre 2 et 5, donne des résultats équivalents et a la meilleur valeur de RMSE possible.

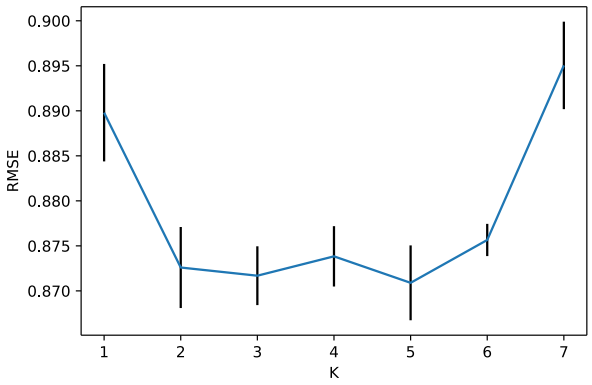

Figure 11: fig:test1
Variation de la RMSE en fonction de K

En comparant l'ALS(Fig.9) et la descente de gradient(Fig.10), on se rend compte que l'ALS converge beaucoup plus vite.

Cependant, sur ce jeux de données relativement petit, l'ALS semble sur-apprendre très rapidement, ce qui provoque une plus mauvaise RMSE qu'avec la descente de gradient.

Enfin, on peut s'attarder sur l'interprétabilité des matrices U et I.
Si nous essayons de regarder les points aussi bien de U que de I dans le plan, nous notons qu'il est très difficile d'en déduire un quelconque motif ...

Cependant, lorsque l'on s'attarde sur la similarité entre les films à l'aide de la similarité cosinus : nous observons une certaine logique seulement pour quelques recommandations spécifiques envers des films particuliers, ce qui rend la tâche d'autant plus difficile.

Voici, en exemple, les films identifiés comme similaires pour 'Rudolph, the Red-Nosed Reindeer' film de Noël pour enfant et 'Shining' un film d'horreur :

Rudolph, the Red-Nosed Reindeer	The Shining
Christmas Carol : Adventure|Animation|Children	Touch of Evil : Crime|Film-Noir|Thriller
The Night Before : Comedy	Bonnie and Clyde : Crime|Drama
London (2005) : Drama	Gorillas in the Mist : Drama
Chaser, The (Chugyeogja): Crime|Drama|Thriller	Cloud Atlas : Drama|Sci-Fi|IMAX

On peut voir que les deux premiers films semblent tenir une certaine logique, la suite semble être bien plus cryptique.

4.2.2 Résultats empiriques de la factorisation de matrice non-négative

Dans cette section est utilisée la MAE (Mean Absolute Error), afin de comparer les résultats obtenus avec ceux des travaux existants pour la FMN [2] [3].

$$MAE(R, \hat{R}, T) = \frac{\sum_{(i,u)\in T} |R_{iu} - \hat{R}_{iu}|}{|T|}$$

Nous avons dans un premier temps évalué les paramètres optimums pour l'algorithme EM. Le premier paramètre évalué est k (réduction de dimension). On retrouve le même k optimal que pour la factorisation de matrice classique: 3. La valeur de MAE initiale augmente avec le k car il y a plus d'élément sommés.

Différentes matrices de départ ont ensuite été testées pour I et V : Une matrice composée que de 1, et des matrices composées de valeurs aléatoires supérieures à 0 et inférieures ou égales à v_{max} pour $v_{max} \in \{1, 2, 2.5, 3\}$.

Tableau 4 : Valeurs de MAE après 1000 itérations de EM

k	1	2	3	4	5	6	7	8	9	10
MAE	0.660343	0.62306	0.62121	0.62308	0.62455	0.62555	0.62628	0.62685	0.62730	0.62767

La meilleure valeur obtenue est $v_{max} = 2$, Il semblerait que plus v_max se rapproche de la note moyenne (2.25), meilleurs sont les résultats.
Comme évoqué dans la présentation des méthodes EM et WNMF, la première est plus lente que la seconde car elle doit optimiser I et V sur plusieurs itérations, tandis que la EM doit optimiser I et V à chaque étape. Néanmoins, l'algorithme EM présente de biens meilleurs résultats pour la MAE.

En effet, la WMNF est très dépendante des valeurs initiales de I et V. Une idée présenté chez [Zhang et al. 2006] est la combinaison des deux méthodes afin d'obtenir un algorithme rapide et obtenant de bons résultats. L'algorithme EM est utilisé sur quelques itérations, afin d'obtenir un I et V approchés qui seront ensuite utilisés dans la WNMF. La WNMF obtient ainsi de meilleurs résultats.

Enfin, nous avons comparé les résultats de la NMF avec la factorisation de matrice classique afin d'observer la perte en prédiction par rapport au gain en interprétabilité.

Tableau 6 : MAE et RMSE

Méthodes	FM	FMN
MAE	0.66472	0.805123
RMSE	0.86501	1.1337

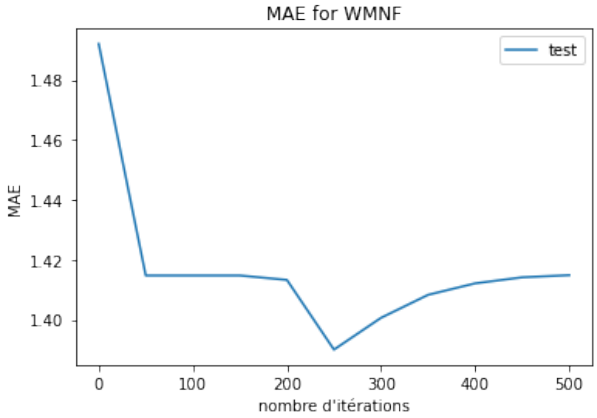

Figure 12: score MAE de la WMNF sur 1000 itérations

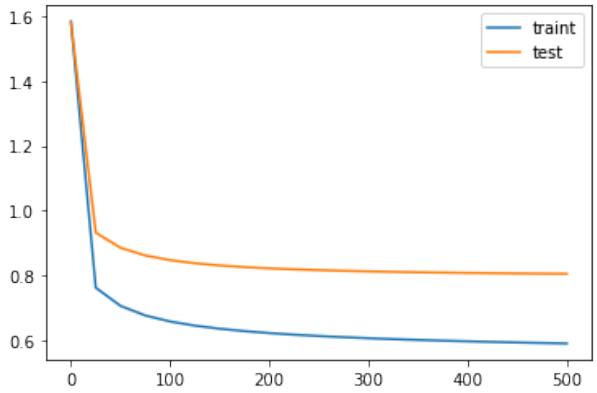

Figure 13: score MAE de la EM sur 1000 itérations

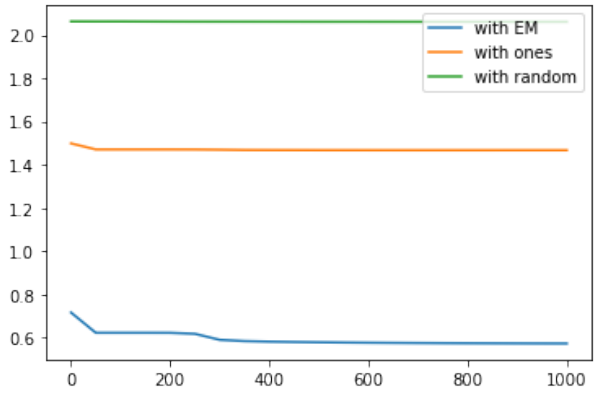

Figure 14: score MAE pour différentes initialisations avec WMNF

Nous avons ensuite essayé de retrouver des groupes caractéristiques pour le k optimal identifié précédemment ($k = 3$). Simplement en triant les valeurs de chaque colonne de I, on obtient une caractérisation de chaque communauté en terme de préférences de films. Nous nous intéressons à deux caractéristiques : le genre et la date du film. On récupère ainsi les 50 films les mieux notés de chaque communauté et on regarde la répartition pour le genre et la date. Afin de ne pas tenir compte de la répartition des genres et des dates dans le dataset, nous nous intéressons aux pourcentages des films d'un genre ou d'une date sélectionnée.

On obtient ainsi trois groupes caractéristiques : Le premier groupe aime les "film-noir" et les films "Mystère". Le second, les films "Musical","Enfant" et "Romance". Le dernier aime bien les films de type documentaire. Les dates des films ne semblent pas donner d'information pertinente pour $k = 3$.

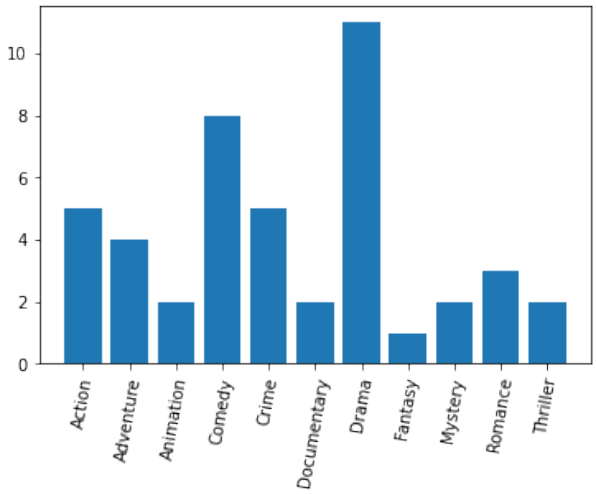

Figure 15: Groupe 1

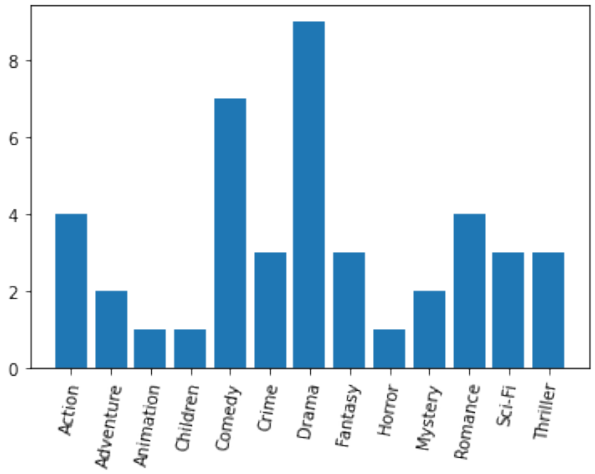

Figure 16: Groupe 2

Nous avons accepté de perdre un petit peu en performance en augmentant le k ($k = 7$), pour essayer d'obtenir d'autres groupes caractéristiques. On obtient ainsi d'autres groupes caractéristiques comme ceux aimant les films noirs et les documentaires et ceux aimant les Western et les films de "Guerre".

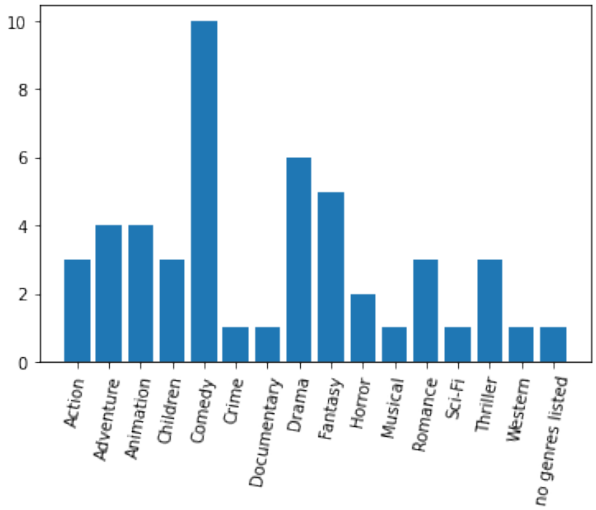

Figure 17: Groupe 3

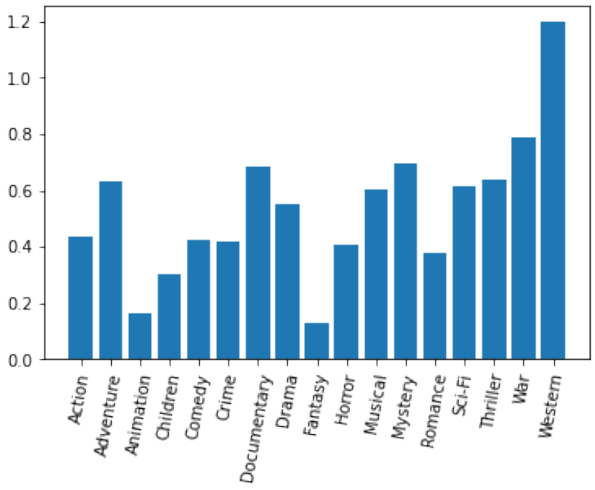

Figure 18: Un groupe pour k = 7

4.3 Analyse du compromis entre l'inter-probabilité et la performance

Dans cette partie, nous nous intéressons à la comparaison entre la méthode de Factorisation de matrice par définition (Baseline) et l'approche probabiliste.
Les contraintes imposées par la méthode probabiliste vont créer une différence qui sera représentable par le comportement de la loss (*RMSE="root mean squared error"*) pour laquelle on observera l'attitude suivante:

> Au fur et a mesure que la loss augmente,l'inter-probabilité décroît. En revanche, nous obtenons une meilleure performance (tradeoff).

5 Conclusions

Nous avons donc vu trois méthodes de factorisation de matrices différentes. La méthode classique (Baseline), la NFM et une approche bayésienne. En terme d'interprétabilité, il est clair que les facteurs latents issus d'une factorisation de matrice classique sont inintelligibles. Cependant nous avons vu qu'il était possible de rendre ces données compréhensibles via les deux autres méthodes de factorisation de matrices étudiées ici. Dans les deux cas, le but est de supprimer les facteurs négatifs auxquels on ne peut donner aucune interprétation. Ensuite on cherche à créer des groupes d'utilisateurs ayant les mêmes goûts que l'on peut facilement identifier dans les matrices U en rajoutant des contraintes lors de l'entraînement du modèle quitte à perdre en performance.

References

[1] Antonio Hernando, Jesus Bobadilla, and Fernando Ortega. A non negative matrix factorization for collaborative filtering recommender systems based on a bayesian probabilistic model. *Knowledge-Based Systems*, 97, 01 2016.

[2] Jean-François Pessiot, Tuong-Vinh Truong, Nicolas Usunier, Massih-Reza Amini, and Patrick Gallinari. Factorisation en matrices non négatives pour le filtrage collaboratif. pages 315–326, 01 2006.

[3] Sheng Zhang, Weihong Wang, James Ford, and Fillia Makedon. Learning from incomplete ratings using non-negative matrix factorization, 04 2006.