Système de recommandation de livres

Dylan Fagot

25 mai 2023

Table des matières

1 Introduction

Vous avez sûrement déjà rencontré sur des pages internet des recommandations telles que :
— Des films qui pourraient vous plaire au vu de ce que vous avez regardés ;
— Des vêtements que vous pourriez acheter au vu de ce que vous avez commandés ;
— Des articles de presse susceptibles de vous intéresser sur la base de vos lectures.

Ces services reposent sur un système de recommandation, ensemble d'algorithmes qui utilise un historique des interactions entre des utilisateurs et des objets (e.g. films, livres) afin de proposer des recommandations pertinentes aux utilisateurs.

Ce projet est un système de recommandation de livres basé sur des notes émises par des lecteurs.

2 Données d'entrée

Nous utilisons l'ensemble de données () disponible librement sur le site Kaggle. Ces données sont réparties dans trois fichiers :

— BX_Books.csv : liste les livres disponibles. Ces livres sont identifies par leurs identifiants ISBN et par leurs titres.

— BX-Users.csv : liste anonymisée des différents utilisateurs ayant émis des évaluations. Ce fichier n'est pas utilisé dans le cadre de ce projet.

— BX-Book-Ratings.csv : un ensemble d'évaluations données par les utilisateurs sur les différents livres disponibles.

3 Pré-traitements sur les données

3.1 Matrice d'utilité

Les systèmes de recommandation se basent généralement sur un tableau à deux dimensions livre / utilisateur appelé "matrice d'utilité". Cette matrice donne pour chaque livre et chaque utilisateur la note que ce dernier a donné au livre.

Une particularité de cette matrice est qu'elle est creuse ("sparse" en anglais), ce qui signifie qu'elle est majoritairement constituée de zéros : cela vient du fait que les utilisateurs interagissent avec un nombre limité de l'ensemble des livres disponibles.

Cette matrice va être construite sur la base d'une matrice initialement remplie de zéro, dans laquelle on va insérer les notes lues dans le fichier listant les notes. Afin d'améliorer les performances de recommandation en terme de précision et de complexité calculatoire, deux mécanismes de rejet ont été mis en places.

3.2 Réjection de notes

Les données d'évaluations sont comprises entre 0 et 10. 0 indique que l'utilisateur a lu le livre mais ne l'a pas évalué. La matrice d'utilité étant initialisée à 0, la lecture du fichier de notes exclura ces valeurs.

3.3 Représentativité statistique

Certains livres ont peu de notes, tandis que d'autres ont beaucoup de notes. La note globale d'un livre avec beaucoup d'évaluations est représentative de sa popularité : si 10.000 utilisateurs l'ont noté 10/10, il y a fort à parier que c'est un excellent livre. Au contraire, un livre avec une seule évaluation de 10/10 peut être un livre médiocre évalué par un lecteur peu exigeant ou de très bonne humeur. Ces deux livres ne seront donc pas tout à fait comparables.

Pour cette raison, le pré-traitement du système de recommandation va exclure les livres ayant trop peu de notes (moins de 100).

4 Recommandation

Une fois la matrice d'utilité construite, celle-ci va être exploitée par des algorithmes spécifiques afin d'émettre des recommandations de livres proches d'un livre donné.

4.1 Décomposition en valeurs singulières

Plusieurs approches existent pour la tâche de recommandation. Ce projet se base sur la décomposition en valeurs singulières (SVD pour Singular Values Decomposition en anglais) de la matrice d'utilité. Cette approche a été adoptée par l'équipe ayant gagné le premier prix du concours "Netflix Prize" en 2007 [1]. Cette décomposition s'écrit généralement : $\mathbf{M} = \mathbf{USV}^\top$, ou encore $\mathbf{M} = \sum_i s_i u_i v_i^\top$ avec s_i la i-ème valeur singulière et u_i et v_i les i-èmes colonnes de $\mathbf{U}$ et $\mathbf{V}$, respectivement.

4.2 Représentation tronquée

En regroupant les vecteur singuliers de gauche et leurs valeurs singulières (**U** et **S**), cette décomposition permet d'exprimer la matrice d'utilité comme un produit matriciel entre une matrice dite dictionnaire (**US**), et une matrice dite d'activation ($\mathbf{V}^\top$).

En sélectionnant uniquement les valeurs singulières les plus fortes, il est possible de reconstruire une matrice d'utilité très proche de l'initiale.

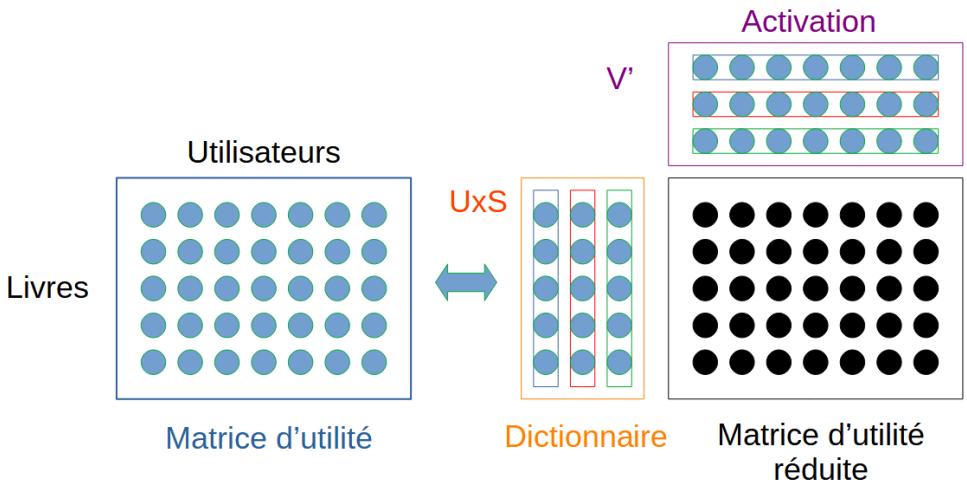

FIGURE 1 – Représentation de la matrice d'utilité décomposée via SVD tronquée ($K = 3$)

Les colonnes de la première matrice représentent des tendances globales en terme de notation de livres. Par exemple, cela peut fait apparaître qu'un livre m est évalué de façon similaire à un autre livre n. Cela peut avoir plusieurs causes, notamment :

— Les livres m et n ont été écrits par le même auteur ;
— Les livres m et n font partie d'une même série ;
— Les livres m et n ont un genre très similaire.

Les lignes de la deuxième matrice (matrice d'activation) quantifie de combien les similarités s'expriment pour chaque utilisateur : un utilisateur peut avoir attribué une même note les livres m et n, tandis que les autres utilisateurs ont évalués les livres suivant un autre schéma (préférences littéraires différentes).

Le deuxième intérêt de cette décomposition est qu'elle quantifie la prévalence de ces schémas de notations. Une stratégie est alors de conserver uniquement les schémas de notations les plus prépondérants (appelons ce nombre K), ce afin d'expliquer les mécanismes de notation, tout en réduisant au maximum la quantité de données produites. Dans la figure 1, seules les trois premières colonnes du dictionnaire et les trois premières lignes de la matrice d'activation sont conservées ($K = 3$) pour donner une matrice d'utilité réduite approchant la matrice d'utilité.

Finalement, chaque livre n'est plus vu comme un ensemble de notes attribuées par chaque utilisateur, mais comme un ensemble de notes obéissant à K tendances de notation. Ce nombre K doit être choisis :

— Suffisamment faible en comparaison au nombre de livres afin d'identifier des tendances de notations globales et limiter la taille des données générées ;
— Suffisamment fort pour ne pas trop réduire la complexité du mécanisme de notation, au risque de perdre la correspondance matrice d'utilité = dictionnaire x activation.

Nous prendrons K comme étant égal à la racine carrée du nombre de livre.

4.3 Mesure de similarités

Ce projet se base sur la mesure de ressemblances entres différents livres. Ces ressemblances sont mesurées ici via la fonction de similarité cosinus. Cette ressemblance est maximale (1) si les vecteurs formés par les notes de chaque livres sont identiques, et minimale (-1) s'ils sont de sens opposé.

Formellement, pour deux vecteurs de notes u et v, cette similarité a pour valeur $\frac{u.v}{||u||.||v||}$ Cette similarité peut être interprété géométriquement : c'est le cosinus de l'angle entre les vecteur u et v, d'où le nom de cette mesure de similarité.

Les similarités entre livres sont mesurées dans leur représentation de faible dimension (lignes du dictionnaire).

4.4 Mise à jour SVD

Dans le cas où un livre ou un utilisateur doit être ajouté à la base de données, ou qu'une nouvelle note apparaît, il est nécessaire de recalculer la décomposition. Cela peut être une opération très coûteuse calculatoirement au vu de la dimension du problème. Une alternative est d'ajuster la décomposition courante avec une mise à jour dite de rang 1. La publication [2] détaille explicitement les calculs de mise à jour à effectuer.

5 Analyse des résultats

5.1 Pré-traitement des données de notes

La fichier de notes contient 11,9% de notes non nulles. Parmi ces notes, seuls 2,7% des livres disposent de suffisamment de notes pour être comparés. Après suppression des notes nulles et des livres avec trop peu d'évaluations, le nombre d'utilisateur initial a été mécaniquement réduit de 91,0%.

Cela illustre la proportion de données réellement exploitable pour effectuer les recommandations. La matrice d'utilité construite a un taux de complétion de 0.07%. Cela confirme la pertinence de définir cette matrice avec le format de matrice creuse offert par la librairie scipy. Sous ce format, la matrice nécessite moins d'espace mémoire pour être stockée, et cela bénéficie également à l'algorithme de décomposition qui va pouvoir profiter du grand nombre de zéros pour accélérer les calculs.

5.2 Recommandations

Une fois la matrice de similarités obtenues, on propose un livre de référence ≪The Hobbit : The Enchanting Prelude to The Lord of the Rings≫ pour lequel on souhaite avoir des recommandations de livres similaires avec les scores associés. Nous obtenons les trois recommandations suivantes :

— The Return of the King (The Lord of the Rings, Part 3), similarité = 0.72
— The Two Towers (The Lord of the Rings, Part 2), similarité = 0.71
— The Fellowship of the Ring (The Lord of the Rings, Part 1), similarité = 0.69

On remarque que les recommandations tournent autour de l'œuvre de Tolkien. Cela est cohérent du comportement des utilisateurs appréciant cet auteur : les lecteurs ayant apprécié le Hobbit auront tendance à poursuivre leur voyage dans les Terres du Milieu.

6 Alternatives

6.1 NMF

La décomposition en valeurs singulières tronquée n'assure pas que la matrice d'utilité estimée (dictionnaire x activation) ait des valeurs positives : cela revient à dire que le traitement admet la possibilité de notes négatives ! Une alternative serait d'utiliser la factorisation en matrice non-négative (NMF, Nonnegative Matrix Factorization en anglais) qui permet d'assurer la décomposition d'une matrice à termes positifs comme le produit de matrices dictionnaire/activation à termes positifs [3].

6.2 Auto-encodeur

La force de la décomposition en valeurs singulières est sa simplicité d'expression. C'est aussi sa faiblesse : cette décomposition détecte uniquement des schémas de notation qui sont des fonctions linéaires des notes de chaque livre. Une alternative capable de détecter des relations plus complexes (non-linéaires) est le réseau de neurones en architecture dite "auto-encodeur" [4].

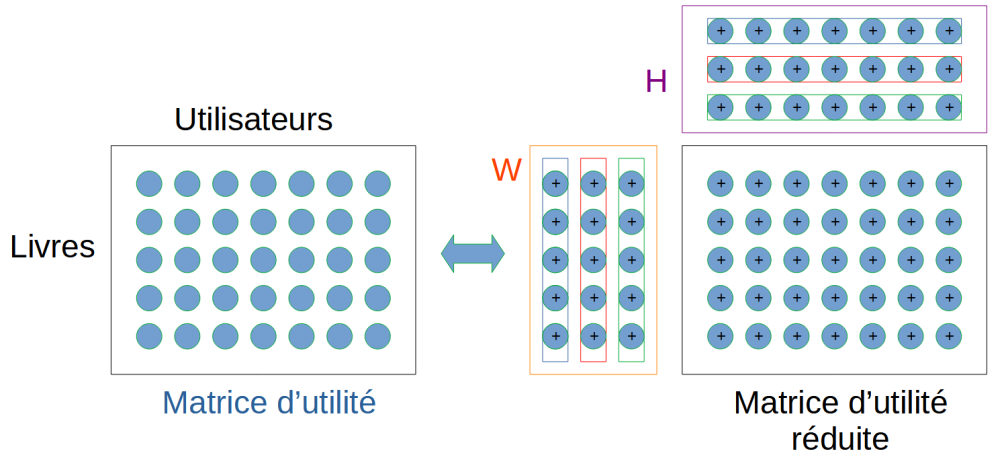

FIGURE 2 – Factorisation en matrice non-négative. Les termes du dictionnaire $\mathbf{W}$, de la matrice d'activation $\mathbf{V}^\top$ et de la matrice d'utilité réduite sont tous positifs.

Ce type de réseau de neurone est conçu pour reproduire en sortie (couche de sortie) ce dont il dispose en entrée (couche d'entrée). Leur particularité est de fonctionner avec des couches intermédiaires de tailles limitées. Ainsi, ce type de réseau de neurone va apprendre à "compresser" l'information (encodage) puis à la "décompresser" (décodage) sans la dénaturer (entrée = sortie). Par simplicité, nous considérons une auto-encodeur à une couche d'entrée, une couche intermédiaire et une couche de sortie, illustré dans la figure 3.

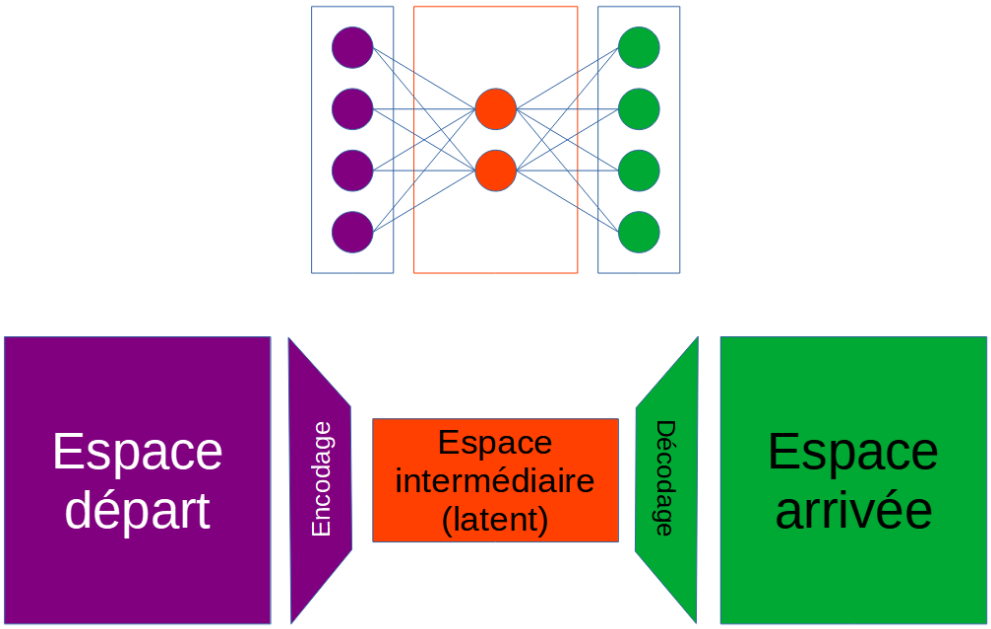

FIGURE 3 – Représentation d'un autoencodeur à une couche intermédiaire (orange). Les valeurs en sortie (vert) coincident avec celles données en entrée (mauve) une fois l'entraînement effectué.

Dans la SVD et la NMF, l'encodage se fait via la matrice de dictionnaire, et le décodage via la matrice d'activation. Dans ces cas, les fonctions d'encodage/décodage sont matricielles et donc linéaires. Il serait tout à fait possible d'augmenter le nombre de couches intermédiaires et le nombre de neurones par couche pour obtenir des fonctions d'encodage/décodage plus évoluées. Cela se fera toute fois au détriment de la capacité du réseau de neurones à extraire une représentation généralisée

(phénomène dit de «overfitting») et complexifie davantage le problème de l'entraînement du réseau, de par l'augmentation du nombre de paramètre et de la non-convexité du problème.

7 Améliorations

7.1 SVD

Le principal axe d'amélioration de la méthode présentée dans ce projet est la question de la prise en compte de l'absence de notes. En effet, notre système de recommandation exploite actuellement les notes absentes, et les traitent telles de vrais zéros. Une autre approche, présentée dans la publication consiste à exploiter uniquement les données réellement présentes pour calculer les liens entre les livres. Avec des liens mieux calculer, il devient alors possible de réaliser de la prédiction de notes : la note qu'un utilisateur attribuera à un livre peut être calculée comme une moyenne des notes qu'il a attribué à des livres similaires : c'est le principe du "K-nearest neighbor SVD" présenté dans la publication [1].

7.2 NMF / auto-encodeur

Sur les deux autres approches, il est également possible d'empêcher l'approximation de notes manquantes en les excluant de la fonction de coût. Une fois l'apprentissage effectué, des estimations de notes apparaîtront alors naturellement aux emplacements des notes manquantes.

8 Références

[1] Bell, R. M., Koren, Y., & Volinsky, C. (2007). The bellkor solution to the netflix prize. KorBell Team's Report to Netflix.

[2] Brand, M. (2006). Fast low-rank modifications of the thin singular value decomposition. Linear algebra and its applications, 415(1), 20-30.

[3] Seung, D., & Lee, L. (2001). Algorithms for non-negative matrix factorization. Advances in neural information processing systems, 13, 556-562.

[4] Hinton, G. E., & Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. science, 313(5786), 504-507.