Technical test - Data Science

Fayçal HAFID
Elève ingénieur à IMT Atlantique, Brest, France

Documentation de l'implémentation d'un système de recommandation de films pour un test technique, MoodWork

Sommaire

1. Proposer un regroupement d'utilisateurs basé sur la notation de films :

L'idée est de discriminer les utilisateurs selon le nombre de films qu'ils ont regardés (et qu'ils ont donc notés) pour avoir 4 groupes :

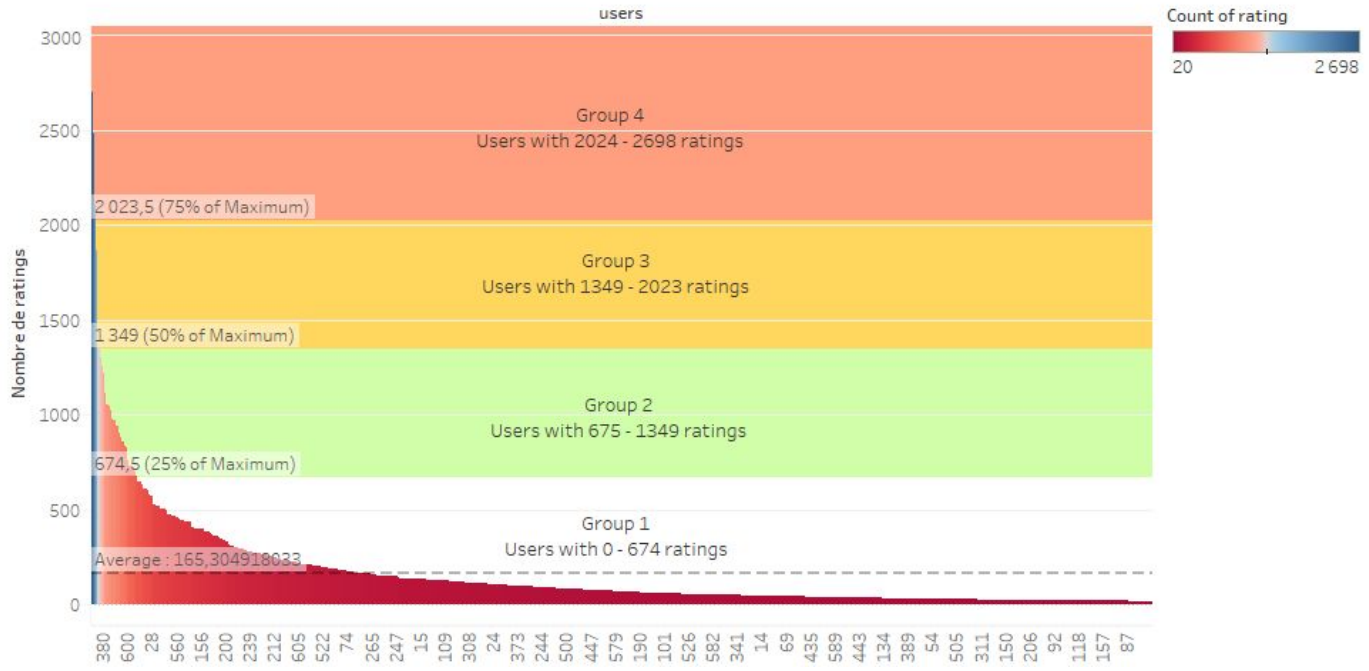
- Groupe 1 : utilisateurs qui ont noté un très petit nombre **(relativement)** de films
- Groupe 2 : utilisateurs qui ont noté un petit nombre **(relativement)** de films
- Groupe 3 : utilisateurs qui ont noté un grand nombre **(relativement)** de films
- Groupe 4 : utilisateurs qui ont noté un très grand nombre **(relativement)** de films

La question qu'on se pose est la suivante : comment déterminer cette relativité ?
Instinctivement, on peut penser de la manière suivante : trouver le nombre minimum de notes *nb_min* (l'utilisateur qui a noté le moins de films) et le nombre maximum de notes *nb_max* (l'utilisateur qui a noté le plus de films). On obtient alors l'intervalle [*nb_min*, *nb_max*], qu'on peut diviser en quatre parties égales.
Les calculs sur Python donnent : nb_min=20 et nb_max=2698 (oui, un utilisateur a apparemment noté près de 2700 films…)
Et donc :

Groupe	1	2	3	4
Intervalle de nombres de notes par utilisateur	[20 ; 674]	[675 ; 1349]	[1349 ; 2023]	[2024 ; 2698]
Nombre d'utilisateurs	583	23	1	3

Le problème qui apparaît est le suivant : **la moyenne du nombre de notes qu'un utilisateur donne est de 165.30 notes/utilisateur**, une valeur très lointaine de la médiane de l'intervalle (1339) ou même du premier quartile (669.5), résultat -> le nombre d'utilisateurs par groupe est très déséquilibré.
La visualisation sur le logiciel Tableau donne :

Nombre de ratings par utilisateur



Presque tous les utilisateurs sont dans le groupe 1, ce qui n'a pas de sens dans notre contexte. Il faut donc redéfinir la relativité non pas par rapport au maximum de nombre de notes dans nos données mais plutôt à **la moyenne du nombre de notes M**.

Groupe	1	2	3	4
Nombre de votes	< M/2	[M/2 ; M]	[M ; 2*M]	> 2*M
Sémantique	Les utilisateurs qui ont noté **bien moins que la moyenne** ont noté un **très petit nombre** de films	Les utilisateurs qui ont noté **moins que la moyenne** ont noté **un petit nombre** de films	Les utilisateurs qui ont noté **plus que la moyenne** ont noté **un grand nombre** de films	Les utilisateurs qui ont noté **bien plus que la moyenne** ont noté un **très grand nombre** de films
Nombres d'utilisateurs dans le groupe	329	124	78	79

Visualisation :

Les résultats sont moins déséquilibrés cette fois ci et beaucoup plus en accord avec la sémantique. On voit que le groupe 1 est toujours beaucoup plus grand que les groupes 3 et 4, mais cela est normal car la plupart des utilisateurs notent peu de films (moins d'une centaine) tandis que certains utilisateurs, une minorité, notent un nombre de films tellement grand (plus de 2000) qu'il augmente la moyenne de manière considérable. Ceci indique que les données sont déséquilibrées (ces minorités peuvent avoir un impact négatif sur l'efficacité de notre système de recommandation).
A la base, il aurait peut être semblé plus pertinent de regrouper les utilisateurs par genres de films qu'ils regardent le plus, mais la question spécifie que le regroupement doit être basé sur la notation des films.

2. Créer un algorithme de recommandation proposant à un nouvel utilisateur N films à regarder :

Comme il s'agit d'implémenter un système de recommandation pour un nouvel arrivant, on ne dispose d'aucune information sur les goûts de ce dernier, on ne peut donc pas privilégier certains films par rapport à d'autres en se basant sur le contenu des films (genre du film par exemple).
La stratégie est alors de simuler une liste de *" Trending Movies "* : les films qui sont à la fois très populaires et qui sont très bien notés par la majorité de ceux qui les ont regardés.
Pour ce faire, on doit attribuer un score aux films, après les avoir filtrés pour ne garder que les films populaires.
Pourquoi ne garder que les films populaires : si on prend en compte un film qui n'a reçu qu'une seule (ou en général très peu de) note valant 5/5, sa note moyenne sera de 5 et son

score sera forcément parmi les meilleurs (si ce n'est le meilleur), or nous n'avons aucune raison de recommander un film que seule une personne a vu et adoré.

Comment sélectionner les films populaires :

- On calcule la moyenne des notes sur tous les films, on obtient qu'un film est en moyenne noté/visionné par 10 utilisateurs. Cette moyenne est calculée pour avoir une idée sur le nombre d'utilisateurs qui doivent visionner un film pour que ce dernier puisse être considéré comme populaire.
- On cherche la valeur d'un quantile qui permette de définir un seuil minimum d'utilisateurs ayant noté le film.
  La valeur (arbitraire) de **0.9** pour le quantile signifie qu'on ne garde que les films qui ont été notés plus que 90% des autres films. Cette valeur nous mène au résultat suivant : le **nombre minimum de notes** pour qu'un film soit considéré comme populaire est de **27** (presque 3 fois plus que la moyenne) et nous retourne 1952 films "populaires" (sur les 9724 films qu'on avait à la base). Ces résultats semblent plausibles.

Comment attribuer un score aux films populaires :

On utilise la formule suivante qui est connue pour donner de bons résultats dans les systèmes de recommandations :

$$\text{Weighted Rating (WR)} = \frac{R * v + C*m}{v+m}$$

Où on a : v le nombre de notes qu'a reçu le film, R la note moyenne du film, m=27 le nombre minimum de votes requis pour scorer le film, et C la note moyenne de tous les films (y compris les films non populaires). C et m sont donc fixes, tandis que v et R dépendent du film -> plus le nombre de notes ET la moyenne des notes d'un film sont grandes, plus son score sera grand, c'est bien ce que l'on veut.

La dernière étape consiste simplement à retourner les N films ayant le plus grand score.

3. Créer un algorithme de recommandations prenant l'historique (rating.csv) d'un utilisateur et proposant N films à regarder :

Pour cette partie, nous voulons recommander des films à un utilisateur sur lequel on dispose des informations : la liste des films qu'il a noté. On peut comparer ses goûts avec ceux des autres utilisateurs pour lui conseiller les films qu'ont aimés ceux qui ont le même goût que lui.

Pour ce faire, on quantifie la similarité de chaque film avec chaque autre grâce à la corrélation : plus des utilisateurs différents donnent des notes similaires à deux films, plus la corrélation entre ces deux films tend vers 1 (valeur maximale pour une corrélation).

Après avoir dressé la matrice de corrélation (mesure de similarité) des films :

- On parcourt les films qu'a noté l'utilisateur à qui on veut faire des recommandations
- Pour chacun de ces films:
  - On vérifie si notre utilisateur lui a donné une note > 2.5 (sinon il n'a pas aimé le film et ça ne servirait à rien de lui recommander des films similaires)

    - On prend les films les plus similaires à ce film (grâce aux mesures de corrélation) et on les pondère avec la note qu'a donné l'utilisateur à ce film, on obtient alors un score (qui va jusqu'à 5, valeur atteinte si l'utilisateur a noté le film courant 5/5 et la corrélation vaut 1)
- Si on voit qu'un film a été recommandé plusieurs fois, cela veut dire qu'il est similaire à plusieurs films que l'utilisateur a aimé à la fois, on somme donc ses différents scores de similarité (son score pourra donc dépasser la valeur 5) pour lui accorder plus d'importance
- On ordonne les recommandations par ordre de score décroissant et on lui retourne les N premiers.

4. Proposer des solutions alternatives à la solution implémentée (à ne pas implémenter) :

- **1ère proposition :**
  Au lieu de comparer la similarité des films en se basant sur les votes des utilisateurs, on peut plutôt utiliser à la fois le timestamp (date de sortie du film) et le genre du film. On tente alors de faire un profiling des goûts de l'utilisateur pour quantifier ses préférences en genres (ex : 70% des films qu'il a aimé sont des Thrillers, 20% des comédies, etc…), pour lui recommander les films les plus populaires (filtrage de la méthode de la question 3) qu'il n'a pas vu ET qui correspondent à ses goûts en genres ET pondérer le score par rapport au timestamp histoire de prioriser les films les plus récents.
- **2nde proposition :**
  Idem que la solution implémentée en question 3, mais au lieu de créer une table de pivot avec les notations des utilisateurs, la créer avec les genres de films pour avoir une mesure de similarité basée sur les genres et non pas sur les notations des utilisateurs. Les scores seront différents.
- **3ème proposition :**
  Hybrider la 2nde proposition avec la solution implémentée en question 3. En effet, les deux méthodes calculent deux scores différents. Une idée serait de normaliser chacun de ces deux scores, et d'en faire la moyenne pour en obtenir un nouveau qui prendrait en compte à la fois la similarité des films basée sur le genre et la similarité basée sur les notations des utilisateurs.