Le génie pour l'industrie

RAPPORT TECHNIQUE
PRÉSENTÉ À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
DANS LE CADRE DU COURS GTI792 PROJET DE FIN D'ÉTUDES EN GÉNIE DES TI

Système de recommandation de livres

Auteur
GUILLAUME LÉPINE
LEPG14099201
GUSLEP@GMAIL.COM
HTTPS://GITHUB.COM/GUSLEP

DÉPARTEMENT DE GÉNIE LOGICIEL ET DES TI

Professeur-superviseur

April Alain

MONTRÉAL, 30 JUILLET
ÉTÉ 2016

CRÉATION D'UN SYTÈME DE RECOMMANDATION

GUILLAUME LÉPINE
LEPG14099201

RÉSUMÉ

Ce projet consiste à réaliser un système de recommandation de livres utilisables dans les bibliothèques. Les recommandations doivent être en fonction des habitudes de consommation des utilisateurs et leurs appréciations des différents livres préalablement lus. La solution finale retenue consiste en un système utilisant des recommandations par contenus et des recommandations de type item-item. L'approche des recommandations item-item a été validée par une erreur absolue moyenne de 2.6, se traduisant par une précision de 74%. La précision obtenue concorde avec les résultats que d'autres équipes ont obtenus en utilisant d'autres approches sur le même jeu de donnée. Pour l'algorithme de recommandation par contenu la précision n'a pas pu être validée par une métrique, cependant des vérifications manuelles suggèrent que les recommandations sont en générale consistente avec les gouts des utilisateurs. Certaines améliorations pourraient être apportées à l'algorimte de recommandation de contenu lors de la génération des profils pour limiter l'influence des catégories ayant peu de notes associées. Ce projet prouve l'utilité d'utiliser différents algorithmes lors des recommandations, il met aussi en lumière les difficultés liées aux recommandations par contenus, principalement l'enrichissement des données et la validation des résultats et offre certaines stratégies pour contourner ces difficultés.

Mot clef: Système de recommandation, Recommandation item-item, Recommandation par contenu, BookCrossing, Recommandation de livre.

TABLE DES MATIÈRES

1 Pages préliminaires

1.1 REMERCIMENTS

Je tiens à remercier le professeur supervisant ce projet Alain April pour m'avoir dirigé vers plusieurs ressources qui m'ont grandement aidé, pour avoir tenté de sauver la relation avec biblio mundo. Je tiens aussi à remercier Thomas Maketa étudiant au doctorat sous Alain April, pour avoir pris du temps pour me rencontrer, pour ces conseils et les discussions qui m'ont permis de confirmer mon approche.

1.2 LISTE DES FIGURES

1.3 Glossaire

crowdsourcer : Action de laisser des utilisateurs externes effectuer une partie du travail.

API : Application Programming Interface, Interface de programmation qu'un système expose pour recevoir des interactions extérieures.

screen scraping : Extraire de l'information visible à l'écran ex aller sur chaque page web et parcourir la page pour extraire certaines informations.

capctha : Completely Automated Public Turing Test To Tell Computers and Humans Apart, Image ayant des lettres et des chiffres cachés à l'intérieur que l'utilisateur doit entrer pour accéder à une page ou une section du site. Permets de départager les utilisateurs des robots.
ISBN : Numéro d'identification international attribué à chaque ouvrage publié.
ID : Identifiant
MAE : Mean Absolute Error Erreur Absolut Moyenne. Permets de calculer à quel point une prédiction se rapproche de la valeur réelle

1.4 Préface

Ce projet était initialement prévu être réalisé en partenariat avec l'entreprise Bibliomundo cependant plusieurs différents surs la vision du projet et l'utilisation de leurs données les ont amené à rompre tout contact 3 semaines après le début du projet. Bien que cet événement m'a forcé à réorienter le projet, l'idée générale est restée similaire.

2 INTRODUCTION

Les systèmes de recommandation font partie intégrante de nos vies, ils nous aident à mieux acheter,trouver du nouveau contenu, etc. Pour les entreprises, les systèmes de recommandations sont un outil essentiel pour augmenter leurs ventes et la satisfaction des clients. La majorité des bibliothèques n'ont pas de systèmes de recommandation de livre, cette lacune les empêche de faire découvrir aux utilisateurs des livres qui pourraient les intéresser. Par conséquent, si les bibliothèques étaient capables de recommander automatiquement des livres, elles pourraient mieux promouvoir la lecture. Bien que BiblioMundo se soit retiré du projet, la problématique reste toujours présente. De plus, après réflexion, j'ai remarqué que plusieurs secteurs souffrent du manque de système de recommandation ou du manque de précision de ceux-ci. Ainsi, la solution proposée tentera d'être la plus générale possible pour ainsi pouvoir s'appliquer à d'autres industries. Le but

du projet est de créer un système de recommandation de livres en se basant sur l'historique de notation de livre. Le système sera capable de supporter l'introduction de nouveaux utilisateurs et de nouveaux livres. Le système livré permettra de générer, automatiquement ou sur demande, des recommandations de livres et d'adapter les recommandations basées. Les retombés du projet seraient d'avoir un système fonctionnel qui peut être intégré aux bibliothèques, permettant aux utilisateurs de découvrir des livres qui les intéressent dont ils ne soupçonnaient pas l'existence. Comme le problème des systèmes de recommandation est présent dans plusieurs industries le système devra être le plus indépendant possible du contexte des bibliothèques, de cette façon il sera facile à adapter à une autre industrie.

3. Survol de l'univers des système de recommandation

Les systèmes de recommandations sont des systèmes qui servent à suggérer à l'utilisateur des choix ou des produits qu'il est suceptible d'aimer. Pour ce faire, le système peut prendre ce baser sur les choix d'autres utilisateurs, l'historique de l'utilisateur et sur toute autres informations suceptible de permettre une correlation entre différents produits ou services.

3.1 Algorithme existant

Les systèmes de recommandations sont des systèmes qui servent à suggérer à l'utilisateur des choix ou des produits qu'il est susceptible d'aimer. Pour ce faire, le système peut prendre ce baser sur les choix d'autres utilisateurs, l'historique de l'utilisateur et sur toute autre information susceptible de permettre une corrélation entre différents produits ou services.

3.1.1 Recommandation par contenu

L'algorithme de recommandation par contenu fonctionne en tentant d'identifier des produits Y ayant les mêmes caractéristiques que le produit X. Le prédicat à la base de cet algorithme est que les utilisateurs aiment un produit pour ses attributs et donc si un utilisateur X a aimé les attributs du produit Y il devrait aimer le produit Z qui a des attributs similaires. Un

avantage de cette méthode est que dès qu'un produit est introduit dans le catalogue il peut être recommandé. De plus, comme l'algorithme ne se base pas sur les autres utilisateurs il peut recommander des produits moins connus. Cette caractéristique se nomme l'augmentation de la sérendipité. Cependant, cette méthode comporte plusieurs désavantages. Souvent les recommandations effectuées auront peu de valeurs aux yeux de l'utilisateur. Un exemple serait un utilisateur qui a lu et aimé Bilbo le hobbit en version de poche. Le système a de fortes chances de lui recommander Bilbo le hobbit en version couverture rigide. Aux yeux de l'algorithme, ces recommandations ont du sens, car ce sont deux produits distincts qui ont les mêmes attributs.

3.1.2 Recommandation Utilisateur-Utilisateur

L'algorithme de recommandation collaboratif utilisateur-utilisateur se base sur le prédicat suivant: pour un utilisateur X il existe des utilisateurs Y qui ont des goûts similaires donc ce que les Y ont aimé, X devrait aimer. En pratique, il fonctionne bien, cependant il comporte quelques problèmes. En premier lieu, les goûts d'un utilisateur ne sont pas statiques, ils évoluent dans le temps. À moins de recalculer constamment les utilisateurs similaires, il est difficile de prendre en compte ces changements. Ensuite, il souffre du problème du démarrage à froid. En effet, lorsqu'un nouvel utilisateur arrive, on doit attendre qu'il ait noté suffisamment produits avant de lui trouver des utilisateurs similaires. La même situation se produit avec les produits, avant de pouvoir être recommandés ils doivent avoir été notés par plusieurs utilisateurs. Un autre problème vient de la complexité asymptotique de l'algorithme, pour trouver les utilisateurs similaires on doit, pour chaque utilisateur, comparer ce qu'il a noté à tous les autres utilisateurs, cela nous amène à une complexité N^2. Finalement, la majorité des utilisateurs auront noté peu de livres, il est donc difficile de trouver des associations d'utilisateurs qui aient plusieurs produits en communs, cela affecte donc la précision et la qualité des recommandations.

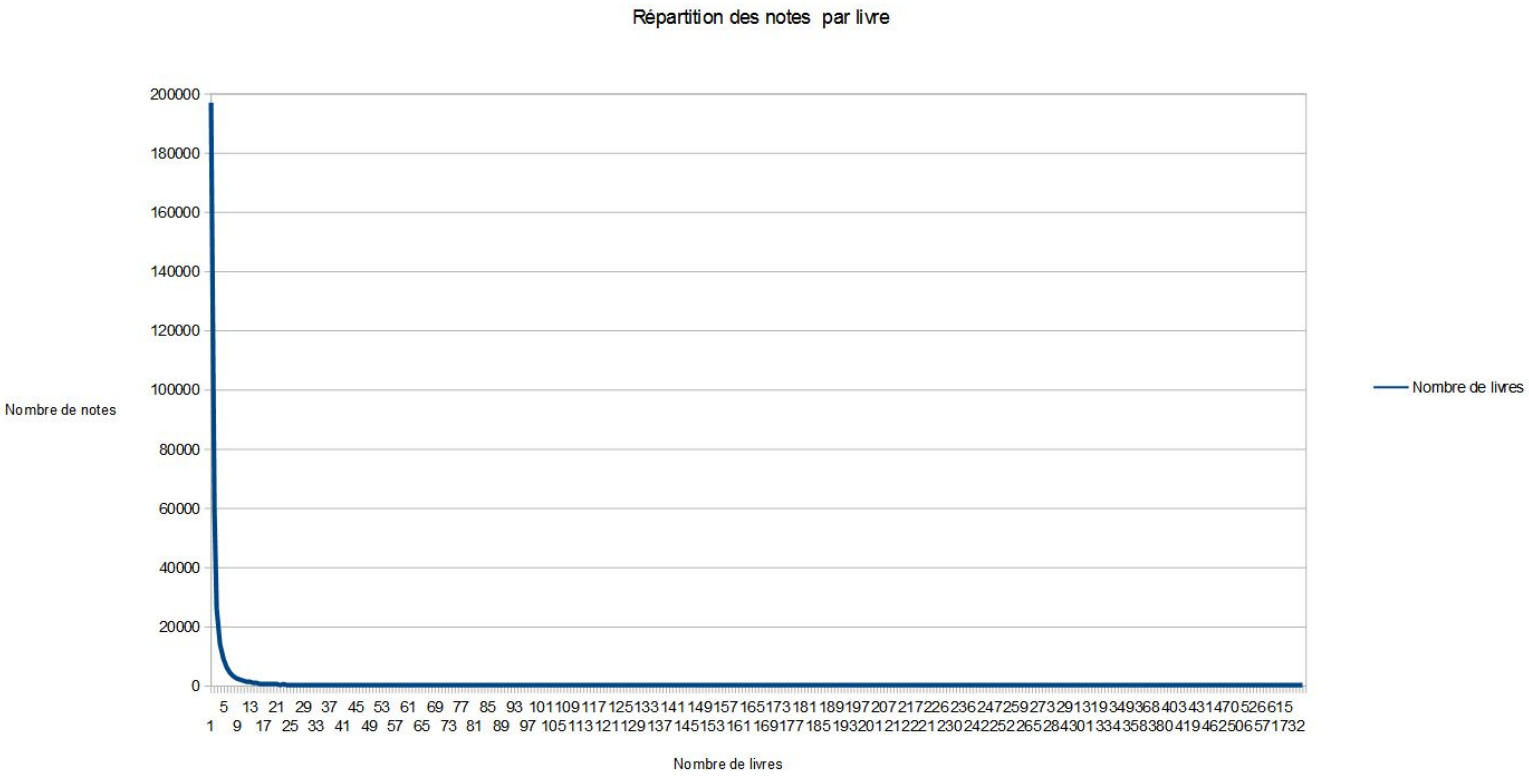
3.2 Choix d'algorithme

La méthodologie adoptée pour choisir l'algorithme est d'implémenter la recommandation par item-item et par contenu puis évaluer la combinaison des deux qui fonctionne le mieux.Si les résultats des deux algorithmes sont acceptables alors leurs recommandations seront combinées. Pour évaluer chaque algorithme différents facteurs seront utilisés: l'erreur absolue moyenne, la similarité intra liste et la sérendipité. La décision de prendre un algorithme item-item plutôt qu'utilisateur utilisateur vient du fait qu'il sera possible de faire des recommandations lorsqu'un nouvel utilisateur ou qu'un utilisateur non enregistré visite le site web. Pour effectuer la recommandation par contenu, une étape d'enrichissement des données a été effectuée pour récupérer les catégories applicables à chaque livre. Pour ce faire, l'API d'Amazon a été utilisé, avec un compte gratuit l'accès était limité à 3600 requêtes par heure et chaque requête permettait d'obtenir l'information de 10 livres maximum.

4 Analyse des données

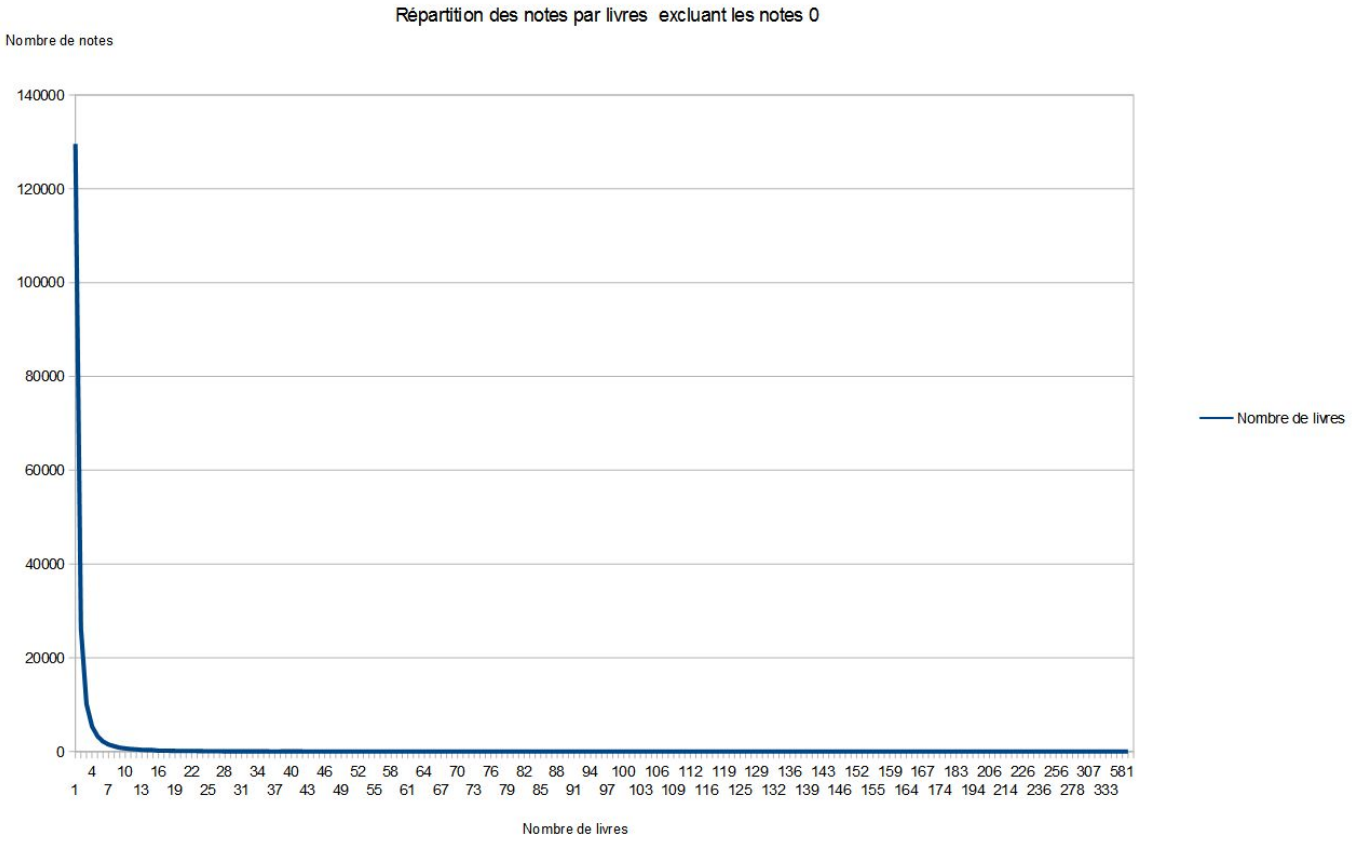
Le jeu de donné utilisé a été trouvé sur internet , il a été collecté par Cai-Nicolas Ziegler pendant 4 semaines en d'aout à septembre 2004. Les données ont été extraites du site Book Crossing en accord avec le CTO Ron Hornbaker. Les données des utilisateurs ont été anonymisées pour préserver leurs vies privées. Le jeu de donnée brut contient 278,858 utilisateurs, 1,149,780 notes explicites et implicites et 271,379 livres. Une note explicite est une note entre 1 et 10 qui a été volontairement attribuée par un utilisateur à un livre. Une note implicite est une note de 0 qui a été attribué d'une manière qui est inconnue par le système de Book Crossing. En consultant la littérature sur les systèmes de recommandation le consensus est que généralement une note implicite est basée sur les pages vues, les cliques, les recherches, etc. On peut soumettre l'hypothèse que l'utilisateur a un intérêt pour ce livre, il est probable que ces notes soient les livres qu'un utilisateur a consultées. Une analyse du format des données a révélé que certains livres avaient des problèmes d'encodage lors de l'extraction, ces livres ont donc été retirés du jeu de donné, car ils ne pouvaient pas être importé dans la base de données postgresql et n'auraient pas pût être relié à des notes. Le nombre de livres problématique était d'environ 500, cela n'aura donc pas d'impact sur le système de recommandation.
Une fois l'importation des données l'analyse a permis de visualiser la forme générale de ces dernières. Cette étape permet s'assurer que le jeu de donné est de bonne qualité. La principale information que j'ai regardée est la répartition des données. Es-ce qu'il ya un petit groupe d'utilisateur qui ont la majorité des notes, est-ce que qu'un petit nombre de livres ont la majorité des notes. La courbe idéale pour les utilisateurs serait une l'oie normale avec son sommet autour de 20 à 30 notes. Pour les livres la courbe idéale serait un loir normal ayant son sommet autour de 40 à 50 notes. Ces distributions permettraient d'être certain que lorsque l'algorithme de recommandation par item va tenter de trouver des livres similaires il va trouver assez d'utilisateurs ayant noté les deux livres. L'analyse des données a plutôt révélé une courbe de style logarithme inverse. Ceci est probablement dû à la courte durée pendant laquelle les données ont été extraites. Néanmoins, vu les contraintes de temps et le fait que ce soit un des seuls jeux de données sur des livres disponibles librement ce jeu de donnée a été utilisé. L'analyse des données a permis de confirmer que les notes implicites devraient être utilisées. En effet, j'estime qu'il faut à un livre au minimum de 15 à 20 notes pour être capable de générer un profile de similarité, en déca de ce seuil les chances que plusieurs utilisateurs aient noté un livre a et b devient très bas. En en incluant les notes de 0 il y a entre 10774 et 68973 livres qui ont au moins 15 ou 20 notes. En en excluant les notes de 0 il y a entre 3244 et 2179 livres qui ont au moins 15 ou 20 notes.

Figure 1 - Répartition des notes par livre



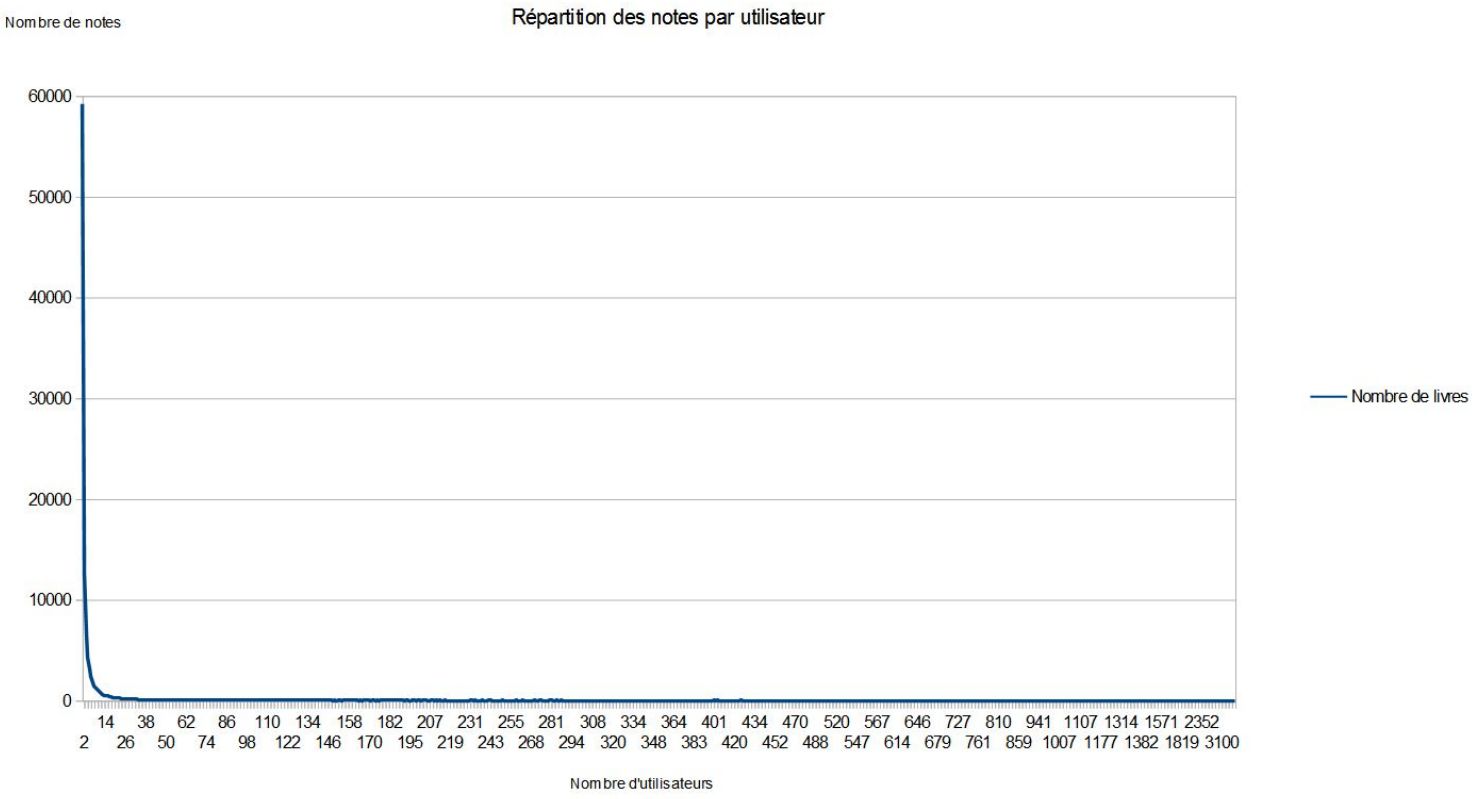
Répartition des notes par livre. Voir Annexe 1 pour les données brutes

Figure 2-Répartition des notes par livre excluant les notes 0



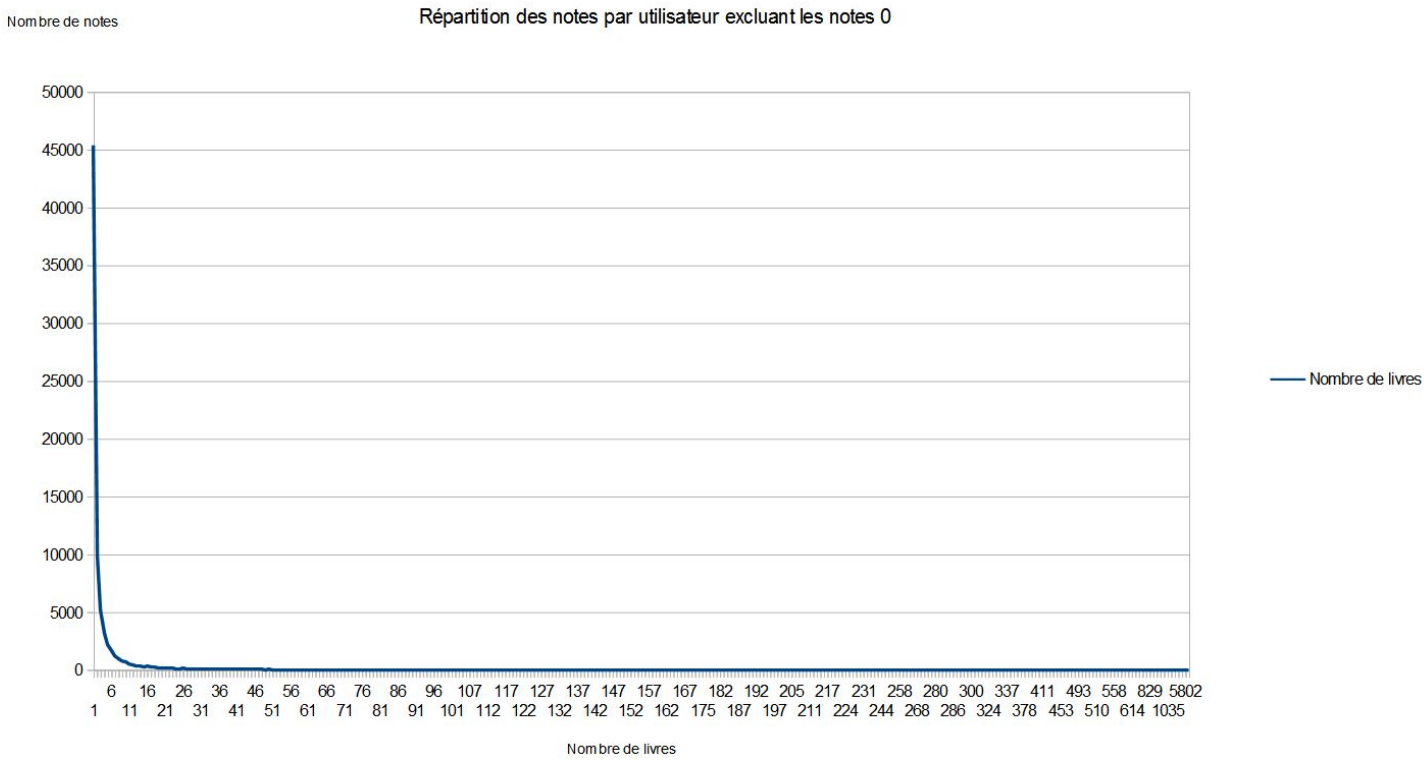
Répartition des notes par livre en excluant les notes 0. Voir Annexe 1 pour les données brutes

Figure 3-Répartition des notes par utilisateur



Répartition des notes par utilisateur. Voir Annexe 1 pour les données brutes

Figure 4-Répartition des notes par utilisateur excluant les notes 0



Répartition des notes par utilisateur. Voir Annexe 1 pour les données brutes

5 Enrichissement des donnés

L'enrichissement des données est un concept clef dans les systèmes de recommandations par contenu. En effet, les données peuvent êtres enrichies de descripteurs servant à préciser ce qu'ils représentent. L'enrichissement peut être fait par de multiples sources différentes, plus les descripteurs sont précis meilleur sera la précision des recommandations effectués. Ainsi, une des meilleures méthodes est d'avoir des experts pour aposer les descripteurs. Une autre technique qui pourrait être prometteuse serait de crowdsourcer l'aposition de descripteurs. Si c'est possible on peut utiliser l'API d'Amazon pour extraire les

catégories dans laquelle fait partie un produit et se servir des catégories comme descripteurs.

Pour ce projet j'ai commencé par tenté de faire du "screen scraping" pour extraire les informations des livres. Malheureusement après environ 50 livres analysés de la sorte Amazon m'ont bloqué en demandant que je remplisse un capcha avant d'obtenir les informations du livres. Suite à cet échec j'ai utilisé l'API d'Amazon, je fournissais le code ISBN puis Amazon me renvoyais une liste d'arbre contenant les catégories auxquelles le livre appartient. La première catégorie est la plus général je devais donc parcourir tous les arbres et prendre seulement les catégories uniques, de plus j'ai volontairement retiré certaines catégories comme "Books" ou bien "Subjects" car elles sont trop générales et partagés par tous les livres. L'objet d'Amazon se nomme un Browse Node, il représente une catégorie d'un livre et contient le nom de la catégorie, le ID de la catégorie et un object Ancestors. Cet objet contient un BrowseNode qui est le parent du node précédent. On peut donc parcourir de manière récursive l'arbre jusqu'à ce qu'on trouve un BrowseNode contenant la propriété isRootcategory ou qu'il n'aie pas de propriété Ancestors. Chaque catégorie rencontré est ajouté dans un HashMap qui à la fin est converti en liste et inséré dans la base de donné.

Plusieurs problèmes sont survenus, premièrement l'API d'Amazon ne permet que 1 requête par seconde. Ayant 268981 livres il m'aurait fallu environ 75h pour traiter tout les livres. Pour contourner ce problème j'ai dû grouper mes livres en groupe de 10, le maximum d"item qu'il est permis d'envoyer par requête. Ensuite, L'API d'Amazon n'offre aucune garantie, dans certains cas j'obtenais aucune catégorie et dans d'autre cas j'obtenais NULL. J'ai dû travailler avec l'apporche qu'aucun paramètre n'était garantie d'être retenu, par exemple Amazon a renvoyé une réponse contenant aucun ISBN.

6 Architecture

L'architecture utilisée pour le projet est une architecture comportant 5 modules indépendants. Cette architecture permet de déployer les modules sur des serveurs différents en utilisant des technologies différentes pour chacun d'eux. Les cinq modules sont "DataStore","Feature Enrichment Processing","Profile Generator", "Recommandation Generator" et "WEB API". L'approche modulaire permet de remplacer les composants indépendamment les uns des autres, d'utiliser des technologies optimales pour chaque composant. Ainsi, il serait facile de migrer les modules "Profile Generator" et "Recommandation

Generator” pour les migrer vers la plateforme spark tout en gardant les autres modules intacts..De plus, cela permet de simplifier le projet et permet une meilleure commercialisation future en permettant à chaque client de prendre les composants qui lui conviennent et même d’utiliser des composants déjà présents.

Figure 5- Schéma d’intéraction des modules

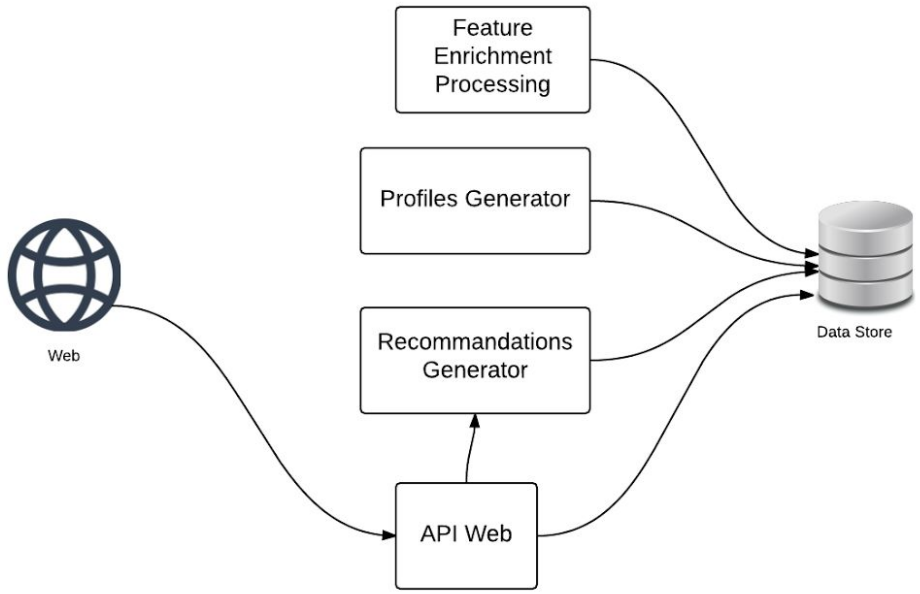


Schéma démontrant l’intéraction entre les différents modules du système. L’orientation des flèches démontre la direction de la requête ex: l’API Web appel le Data Store.

6.1 Data Store

Le Data Store sert à stocker les données, il peut être constitué d’un ou des plusieurs systèmes de base de donnés. Pour ce projet, une seule base de données de type PostgreSQL a été utilisée. Cependant, dans un système qui irait en production il serait mieux d’utiliser un une base de données Postgresql pour les données qui seront en petit nombre comme les livres ou les utilisateurs. Pour les données qui auront un plus gros volume, il serait mieux d’utiliser une base de donnée Impala, car elle est partitionnée et peut gérer plusieurs millions d’enregistrements. Cette base de données serait adaptée pour les tables de recommandations et de votes, car ces tables vont être 10 à 100 fois plus grandes que les tables d’utilisateurs ou de livres.

6.2 Feature Enrichment Processing

Ce module sert à enrichir les livres de différents descripteurs. Pour ce projet, le module a été implémenté en Node.JS pour sa facilité et la rapidité d'exécution. Ce module reçoit la liste des livres à enrichir, contact ensuite Amazon pour obtenir les catégories associées, parcours l'arbre de catégorie reçu et insère les résultats dans le "Data Store".

6.3 Profiles Generator

Ce module sert à générer les profils des gouts des utilisateurs basés sur leurs notes passés. Il obtient les données du Data Store, principalement l'historique de note et la description des livres, génère les profils des utilisateurs et ensuite écrit les profiles générés dans le Data Source. Le module est implémenté en Java, car il partage des classes avec "Profile Generator", "Recommandation Generator". Une autre fonctionnalité est la génération des similitudes intra livre qui sera utilisé dans les recommandations par item.

6.4 Recommandations Generator

Ce module sert à générer les recommandations, il obtient les données du Data Store et écrit ensuite les recommandations dans le Data Store. La génération des recommandations se fait en deux temps. Dans un premier lieu, un algorithme de recommandation par contenu est employé. Ensuite, un algorithme de recommandation de type Item-Item est employé. Les recommandations sont insérées en groupe de 1000 pour ne pas surcharger le serveur.

6.5 Web API

Ce module sert d'interface pour interagir avec les différents modules. Il permet d'ajouter des des livres, des utilisateurs, d'enregistrer les votes, d'obtiendras les recommandations générées pour un utilisateur et de générer des recommandations sur le vif. Pour ce projet, cet API a été implémenté en Node.JS, il obtient les recommandations générées en contactant le DataStore. L'obtention des recommandations sur le vif se fait en contactant directement le module de recommandation generator.

7 Implémentation des alghoritmes

7.1 Recommandation par contenu

7.1.1 Génération de profiles

La génération de profilé consiste à faire une moyenne pondérée pour chacun des descripteurs associés aux livres que l'utilisateur a notés. Pour ce faire, la première étape consiste à calculer la moyenne des votes de l'utilisateur. Ensuite, pour chaque livre noté la liste des descripteurs est extraite puis la note qui avait été attribuée au livre est ajoutée à la moyenne des notes pour ce descripteur. Dans le cas où la note est 0 la moyenne des notes de l'utilisateur est utilisé. L'hypothèse est que l'utilisateur a montré un intérêt envers le livre, mais ne l'a pas noté donc son intérêt pour le livre devrait se situer dans la moyenne de ces votes. Une fois, toutes les notes ont été analysées le résultat obtenu est une liste de descripteurs associés à un poids entre 1 à 10. Ce poids représente l'intérêt que porte l'utilisateur pour une catégorie en particulier.[1].

7.1.2 Génération des recommandations

La recommandation par contenu pour un utilisateur consiste à attribuer un score d'appréciation potentiel à tous les livres. Pour chaque livre la liste des descripteurs associés est extraire, ensuite une sommation sur les intérêts de l'utilisateur est effectuée pour chaque descripteur qui est présent dans le profile de l'utilisateur et dans les descripteurs associés au livre. Lorsque tous les livres ont été traités les 100 livres ayant le meilleur score sont gardées et insérées dans la base de données, ils feront office de recommandation de livre [2]

[1] Voir Équation 1
[2] Voir Équation 2

7.2 Recommandation par item

7.2.1 Génération de la similarité intra-livre

La première étape de la recommandation par item est de précalculer la corrélation de Pearson entre tous les livres. Cela consiste à comparer les notes qu'un utilisateur a attribuées à deux livres pour trouver leurs niveaux de similitudes. L'hypothèse est que si plusieurs personnes ont apprécié un livre A et un livre B, il y a de fortes chances que si tu as apprécié un livre A tu vas apprécier le livre B. Ainsi, pour comparer deux livres Bi et Bj la liste des utilisateurs ayant noté ces deux livres est extraite. Ensuite, une sommation est effectuée sur le résultat de la multiplication des deux notes soustraites à la moyenne des notes de leurs livres associés. Dans le cas où un utilisateur a attribué la note 0 la moyenne des notes plus grande que 0 de l'utilisateur est utilisée. Pour chaque livre, une liste d'au maximum 100 livres qui sont les plus similaires sont ensuite insérés dans la base de données.[3]

7.2.2 Génération de recommandation

Finalement, lorsqu'on veut générer des recommandations pour un utilisateur il faut commencer par extraire la liste des livres notés, substituer les notes 0 par la moyenne des notes de l'utilisateur. Ensuite pour prédire si un utilisateur va apprécier un livre il faut calculer la somme des notes données par l'utilisateur aux livres similaires. Une fois cette somme calculée il faut calculera diviser par la somme des similarités. Cette somme pondérée permet d'arriver avec une note entre 1 et 10, cette note représente le score que l'utilisateur devrait attribuer au livre une fois l'avoir lu.[4]

7.3 Hybride

La combinaison de deux ou plus algorithme est appelé un algorithme hybride. En constatant les avantages des deux algorithmes, principalement l'augmentation de la sérendipité grâce la recommandation par contenu et la précision grâce à la recommandation item-item, les deux algorithmes ont donc été combinés pour créer un algorithme hybride. Lorsque les recommandations sont retournées, ils sont alors

[3] Voir Équation 3
[4] Voir Équation 4

classés en ordre de celui ayant la plus haute note en premier. Différentes approchent pourraient être testées pour trouver l'approche optimale. Une autre approche intéressante serait de s'assurer qu'il y ait au moins 4 recommandations par contenu et ait moins 4 recommandations par item - item dans les 10 premiers résultats. L'hypothèse est que lorsque les recommandations seront montrées dans un courriel ou sur une page web l'utilisateur va regarder les 5 à10 premières il faut donc attirer son attention rapidement. Bien que les recommandations par contenus ne seront pas toujours très intéressantes, car ils recommandent des choix communs, certaines peuvent provoquer la curiosité. La même règle serait appliquée aux recommandations par item, car on ne veut pas que l'utilisateur perde confiance envers les recommandations. Si les 10 premières recommandations sont des livres inconnus qui n'inspirent pas confiance alors l'utilisateur risque de ne plus faire confiance aux systèmes de recommandation. Cependant, l'introduction de recommandation par item vient contrebalancer cela de par ça précision plus élevée. L'approche hybride est celle qui a été retenue, combinant les résultats des approches par item et par contenu. La liste est ensuite triée par ordre des notes les plus élevées au plus basses. Bien qu'aucune métrique de précision ne permette de conclure que les recommandations par contenus sont bonnes, elles permettent à l'utilisateur de découvrir de nouveaux livres.

8 Analyse des résultats

Pour analyser les résultats, deux techniques différentes ont été employées. Pour la recommandation par contenu l'approche suivante a été utilisée. En premier lieu tous les utilisateurs ayant au moins 20 notes ont été sélectionnés. Ensuite, toutes les notes sont séparées aléatoirement en 2 groupes, une liste contenant 75% notes qui servira à entrainer le système et un contenant 25% des notes qui servira de liste de référence. La plus grande liste sert à entrainer l'algorithme de recommandation par contenu. Pour les utilisateurs concernés, la liste des scores d'appréciations est régénérée puis à partir de cette liste de nouvelles recommandations sont créées. La liste de recommandations est calculée et comparée à la liste de référence pour savoir quelle proportion de la liste de référence fait partie des recommandations. En effectuant ce test, la précision moyenne de l'algorithme était moins de 1%. La raison de ce faible taux de précision est la nature de l'algorithme de recommandation par contenu, cet algorithme va, comparer tous les livres ensemble pour trouver certains qui sont similaire. Il va ainsi recommander des livres plus obscures qui sont moins connues qui ont moins de chance d'avoir été déjà notées et ainsi de se retrouver dans la liste de référence . Une meilleure manière de tester l'approche serait de faire valider les résultats par des utilisateurs. Pour valider les résultats j'ai moi-même effectué certaines validations pour voir si les résultats étain censé. Un exemple flagrant est une personne qui a

noté un livre x en version de poche c'est fait recommander le même livre, mais en version bibliothèque. Bien que cette recommandation soit inutile, elle montre que le système est capable de trouver des livres qui correspondent au gout des utilisateurs.

Pour analyser la recommandation du type item-item le but est de comparer les prédictions effectuées avec les notes attribuées. Pour ce faire, la même technique de séparation des données est réutilisée. Cependant, la métrique de l'erreur moyenne au carré (Mean Square Error) est utilisée, elle est particulièrement appréciée, car elle ne différencie pas si la note attribuée était plus basse ou plus haute que celle réellement donnée par le client. En utilisant encore une fois 75% des votes pour entrainer les modèles de similarité et en utilisant 25% des données comme liste maitre. En utilisant cette technique la précision de cet algorithme varie entre 74% et 76% ce qui est excellent, cela veut dire que lorsque le système attribue une note elle sera à plus ou moins 26% de ce que l'utilisateur aurait attribué.

Pour calculer la précision de l'algorithme hybride, le même problème qu'avec l'algorithme à recommandation par contenu s'impose. L'utilisation de la MAE ne donnera pas le résultat escompté et ne représentera pas la réelle précision du système de recommandation. La meilleure indication serait de faire tester le système par des utilisateurs réels.

Une inspection manuelle des résultats a été effectuée pour tenter de voir si les résultats des recommandations avaient du sens selon mon expérience et si les recommandations étaient en lien avec le profilé de l'utilisateur. Les figures 6 et 7 démontrent des résultats de recommandations qui sont concluantes. Cependant, la figure 6 démontre les désavantages d'un système de recommandation par contenu. Il est difficile de savoir si le fait que l'utilisateur n'aime pas les technothriller va influencer son appréciation du livre. De plus, il est difficile de savoir combien de livres de type technothriller il a lus. La figure 7 démontre la puissance de la recommandation par item, cette recommandation est pertinente et l'utilisateur va assurément aimer ces deux livres. Cependant, il y a peu de chance que l'utilisateur n'ait jamais entendu parler des deux livres recommandés. Néanmoins, ces exemples démontrent la puissance et le bon fonctionnement du système développé.

Figure 6 - Résultat d'une recommandation par attribut

Cette figure démontre une recommandation par item. Les deux images de gauche démontrent deux notes qu'un utilisateur a entrées. Il a donné 10 aux deux premiers livres de la trilogie du seigneur des anneaux. Le système lui a donc recommandé (image de droite) le troisième livre de la trilogie et Bilbo le hobbit qui est considéré comme un prélude à la trilogie.

Figure 7- Résultat d'une recommandation par item

Cette figure démontre une recommandation par item. Les deux images de gauche démontrent deux notes qu'un utilisateur a entrées. Il a donné 10 aux deux premiers livres de la trilogie du seigneur des anneaux. Le système lui a donc recommandé (image de droite) le troisième livre de la trilogie et Bilbo le hobbit qui est considéré comme un prélude à la trilogie.

9 CONCLUSION

La réalisation d'un système de recommandation permet d'augmenter l'avantage compétitif d'une compagnie. Ils permettent de faire découvrir des produits aux utilisateurs en leurs offrants des choix qu'ils n'auraient pas naturellement considérés. Certains sites implémentent des recommandations manuelles. Si un utilisateur voit le produit A voici les 5 produits qui seront montrés dans un style les gens qui ont acheté A ont aussi acheté ces articles. Ce projet a permis de prouver qu'il est possible d'implémenter un système de recommandation qui soit satisfaisant en terme de précision et en terme de sérendipité. Bien que ce projet ne sera pas utilisé, il pourrait facilement être intégré dans une bibliothèque pour permettre aux utilisateurs de recevoir des recommandations de lectures. De manière plus générale, ce projet pourrait être utilisé dans n'importe quel magasin en ligne. Un produit comme un chandail est très similaire à un livre, il contient des attributs comme la couleur, le style, la marque. Ce projet a permis d'établir une démarche à suivre pour la réalisation de système de recommandation. Le taux de précision de 74 % est plus haut que le taux initial fixé de 70%, on peut donc conclure que le projet est un succès, car il satisfait les critères de succès fixé initialement. Ce projet a permis de produire un système modulaire permettant de recommander des livres aux utilisateurs basés sur leurs préférences. Il permet d'être constamment réentrainer pour s'assurer de la précision des recommandations et de prendre en compte les dernières notes, par conséquent de prendre en compte les changements de gout des utilisateurs et des modes qui sont passagères.

10 RECOMMANDATIONS

D’autre approche pourrait être exploré, en effet l’idée du professeur Galit Shumeli d’utiliser un algorithme Bayesien naïf pour prévoir si un utilisateur va aimer un livre ou pas et ainsi développer des profils de gouts par âge est location pourrait donner de bons résultats. Cette approche, combiné à l’algorithme de génération de profile permettrait de générer des profiles plus précis. Une autre amélioration serait de migrer les modules de ”Profile Generator” et “Recommandation Generator” vers la platforme de big data spark. Cela permettrait de réduire drastiquement le temps de génération des profilés en permettant d’utiliser de manière coordonnée plusieurs machines pour effectuer les calculs. D’autres améliorations cosmétiques pourraient être apportées comme l’ajout d’un site web pour présenter les données. Au niveau des algorithmes, une modification majeure devrait être apportée à l’algorithme de génération des profilés d’utilisateurs . Cette version améliorée permettrait de prendre en compte l’incertitude lorsqu’un utilisateur a noté peu de livres d’une catégorie. La version actuelle a une faible dans ces cas, si un utilisateur note un livre 10 dans une certaine catégorie alors sont profile favorisera cette catégorie lors des recommandations, cependant le système n’est pas sur si l’appréciation de cette catégorie est un hasard ou la réalité.L’ajout d’un logarithme permettrait d’ajouter une pénalité aux catégories dont le système n’est pas certain. Un logarithme en base 5 est peut être trop agressif et pourrait être remplacé par un logarithme en base 3 si nécessaire, par exemple si l’utilisateur a noté que 4 ou 5 livres.

11 Équations

11.1 Profile d'un utilisateur

Équation 1

$$R_{uta} = R_u \bigcap \varepsilon(\{Ta, \mathbb{N}\})$$

$$P_{uta} = \frac{\sum_{rui\ \varepsilon\ R_{uta}}^{rui} rui | rui != 0 \rightarrow rui \wedge \neg rui = 0 \rightarrow \overline{RU}}{len(R_{uta})}$$

Le profile d'un utilisateur pour une catégorie i est calculé de la manière suivante. Une sommation est effectuée sur toutes les notes données par un utilisateur pour une catégorie particulière (Ruta), si une note est de 0 alors la moyenne des notes de l'utilisateur est utilisée. Le résultat est ensuite divisé par le nombre de notes étant dans cette catégorie (Ruta).

11.2 Génération de recommandation par conteu

Équation 2

$$R_{uI} = \frac{\sum_{dui\ \varepsilon\ Ti \bigcap Pi} dui * 10}{len(Ti \bigcap Pi)}$$

La note prédite pour un livre est calculée de la manière suivante. Une sommation est effectuée sur la multiplication par 10 de tous les coefficients de gouts associés aux catégories présentes dans le profil de l'utilisateur et dans le livre (Ti Pi).

11.3 Similarité entre deux livres

Équation 3

$$sim(Bi,Bj) = \frac{\sum_{u \in U} (Re_{u,bi} - \overline{R}_i)(Re_{u,bj} - \overline{R}_j)}{\sqrt{\sum_{u \in U} (Re_{u,bi} - \overline{R}_u)^2} \sqrt{\sum_{u \in U} (Re_{u,bj} - \overline{R}_u)^2}}$$

Pour tous les utilisateurs ayant noté les livres i et j. Une sommation est effectuée sur la multiplication de la différence entre la note attribuée par un utilisateur u à un livre i et la moyenne des notes attribuées au livre i et la note attribuée par un le même utilisateur u à un livre j et la moyenne des notes attribuées au livre j. Le résultat est ensuite divisé par la multiplication de la racine carrée des notes attribuées par un utilisateur u à un livre i et la moyenne des notes attribuées au livre i au carré et la racine carrée des notes attribuées par un utilisateur u à un livre j et la moyenne des notes attribuées au livre j'ai carré. La valeur obtenue est un coefficient de similarité entre les livres i et j

11.4 Génération de recommandation par item

Équation 4

$$R_{UI} = \frac{\sum_{j \in Si \bigcap R_U} S_{ij} * R_{Uj}}{\sum_{j \in Si \bigcap R_U} S_{ij}}$$

La recommandation pour un livre i à un utilisateur u se fait en effectuant une moyenne pondérée sur la multiplication des notes attribuées avec le coefficient similarité entre tous les livres que l'utilisateur u a noté et qui sont similaires au livre i.

11.5 MAE

Équation 5

$$mae = \frac{1}{N}\sum\left|r_i - r_p\right|$$

L'erreur absolue moyenne est la somme de la valeur absolue de la différence d'une note attribuées par le système et la valeur réelle attribué par un utilisateur

11.6 Profile d'un utilisateur amélioré

Équation 6

$$P_{UTa} = \varepsilon(\{Ta, \mathbb{N}\}) = log_5(len(R_{uTa} \,!= 0)) * \frac{\sum_{rui\ \varepsilon\ RU}^{rui} rui | rui!=0 \rightarrow rui \wedge \neg rui=0 \rightarrow \overline{RU}}{len(R_{uTa})}$$

Le profile d'un utilisateur pour une catégorie i est calculé de la manière suivante. Une sommation est effectuée sur toutes les notes données par un utilisateur pour une catégorie particulière (Ruta), si une note est de 0 alors la moyenne des notes de l'utilisateur est utilisée. Le résultat est ensuite divisé par le nombre de notes étant dans cette catégorie (Ruta) puis multiplié par un log en base 5 du nombre de notes différentes de 0 présente dans cette catégorie. Le logarithme permet d'éliminer les catégories avec un nombre insuffisant de votes. Si le logarithme en base 5 est trop agressif alors il pourrait être remplacé par un logarithme en base 3.

12 LISTE DE RÉFÉRENCES

12.1 BIBLIOGRAPHIE

Joseph Konstan et Michael D Ekstrand, N/A : Introduction to Recommender Systems, Internet, Disponible en ligne :https://www.coursera.org/learn/recommender-systems/ , consulté le 01/01/ 2016.

Institut für Informatik, N/A : Book-Crossing Dataset , Internet, Disponible en ligne :http://www2.informatik.uni-freiburg.de/~cziegler/BX/ , consulté le 27/04/ 2016.

GroupLens Research Group,NA,Item-Based Collaborative Filtering Recommendation Algorithms, Internet, Disponible en ligne, http://files.grouplens.org/papers/www10_sarwar.pdf, 4/05/ 2016.

Anurag Sharma Shashvat Rai Siddhartha Chatterji Siddharth Raman Singh Nitesh Batra Sandip Chaudhuri , N/A,A recommendation system for BookCrossing.com, N/A, Internet, Disponible en ligne, consulté le 01/08/2016 http://www.galitshmueli.com/sites/galitshmueli.com/files/BookCrossingREPORT.pdf

Asela Gunawardana, Guy Shani, publié Décembre 2009, A Survey of Accuracy Evaluation Metrics of Recommendation Tasks, revue scientifique, Disponible en ligne, http://jmlr.csail.mit.edu/papers/volume10/gunawardana09a/gunawardana09a.pdf Consulté le 01/08/2016

13 ANNEXE

13.1 Analyse des données

Les données ayants permis de générer les graphiques sont situés dans le fichié données. xslx

13.2 Diagramme de base de données