Blabla Netflix
Eléments logiciels pour le traitement de données massives

Benjamin DONNOT
Thibault LAUGEL

Janvier 2015

1 Introduction

De nos jours, de nombreux sites commerciaux font appel à des systèmes de recommandation. Ces modèles sont utilisés afin d'aider leurs clients à trouver des informations ; par exemple, le système de recommendation d'Amazon aide les clients à trouver les produits idéaux. Dans un autre contexte, le système de recommendation de Netflix propose aux utilisateurs des films que ceux-ci seraient susceptibles d'apprécier.

Améliorer la qualité de ces recommendations a été l'un des enjeux majeurs du marketing en ligne au cours de ces dernières années. En 2009, dans le but d'améliorer son système de recommandation, Netflix organisa une compétition promettant un million de dollars à l'équipe parvenant à battre son algorithme.

Notre projet vise à répondre au problème posé par ce concours en construisant un système de recommandations personnalisées pour les films. Nous avions déjà travaillé sur ce sujet par le passé, mais la trop grande quantité de données disponibles dans la base d'origine nous avait forcé à nous concentrer sur une petite partie de ces observations. Dans ce projet, nous avons donc cherché à prendre en compte l'ensemble des données disponibles en utilisant la technologie Map Reduce. À l'origine nous pensions utiliser d'autres outils pour réaliser ce système de recommandation, et notamment Mahout [1]. Ce logiciel n'étant pas installé, nous avons assez vite abandonné cette possibilité.

Dans ce rapport, nous commençons par présenter les données puis expliquons la méthode utilisée pour les recommandations ainsi que son implémentation avec *PIG*.

Finalement, nous présenterons les résultats obtenus et discuterons des limites de notre travail.

1. Mahout est un regroupement d'outils pour faire, entre autre, des calculs distribués et est notamment populaire pour des systèmes de recommandations. Un introduction est disponible à cette adresse : http://www.ibm.com/developerworks/java/library/j-mahout/

2 Création de la base de données

Nous utilisons les données originellement mises à disposition pour le concours Netflix afin de construire notre système de recommendations.

Les données mises à disposition regroupent 100.480.507 revues faites par 480.189 utilisateurs uniques à propos de 17.770 films différents.

Nous disposons, pour chaque film, d'un fichier texte contenant l'ensemble des revues faites par les utilisateurs pour ce film. Chaque revue est définie par l'*id* de l'utilisateur ayant fait la revue, la date à laquelle celle-ci a été écrite et la note donnée au film (entier compris entre 1 et 5).

La première étape de notre travail a donc été de regrouper l'ensemble de ces fichiers textes en une grosse base de données, en ajoutant aux variables précédentes l'*id* du film. La base finale fait 2,6 GB environ.

Notre but dans ce projet n'est pas à proprement parler la création d'un système de recommandation mais plutôt la prédiction de la note qu'un utilisateur spécifique donnerait à un film spécifique. La création d'un vrai système de recommandation nécessiterait l'implémentation d'autres étapes, telles que la sélection des films à recommander à partir de ces prédictions.

3 Méthodologie

La base a tout d'abord été découpée en deux parties : l'une pour entraîner nos modèles, l'autre pour tester les résultats. Nous avons choisi de découper ces bases par ordre chronologique : la base d'entraînement comporte environ 70% des revues, à savoir toutes celles postées avant le 31 mai 2005.

Compte tenu de la quantité de données que nous utilisons, nous avons ensuité décidé de tester tous nos programmes *PIG* sur un petit sous-échantillon des données d'entraînement. Cette base a été sélectionnée de façon aléatoire : environ $1/1000^{\text{e}}$ du jeu de données original a été sélectionné, soit quelques 70 000 lignes. L'ensemble de nos scripts ont tout d'abord été testés sur cette petite base que nous espérons représentative de la base complète. Nous n'avons lancé les jobs *PIG* sur la base totale qu'après avoir jugés les résultats obtenus corrects sur la plus petite base.

Nous avons utilisé le langage *PIG* pour construire un système de prédiction des notes données par les utilisateurs aux films évalués avant le 31 mai 2005. Le détail de ces implémentations sera présenté dans la partie suivante.

Nous avons enfin réalisé les prévisions a proprement parler sur la base de test, puis testé nos résultats grâce aux vraies notes données par les utilisateurs avec le critère du RMSE. Etant donnée la vraie valeur d'une note $(t_i)_i$ et une prédiction $(p_i)_i$, l'erreur est alors définie par :

$$\text{RMSE}(t,p) \stackrel{\text{def}}{=} \frac{1}{n}.\sum_{i=1}^{n}(t_i - p_i)^2$$

Avec n le nombre d'observations dans notre échantillon de test.

4 Algorithme utilisé

L'idée générale de notre algorithme est regrouper les films par similarité, et d'utiliser les revues faites par les utilisateurs pour en déduire celles qu'ils donneraient à un film similaire.

Les modèles proposés dans la littérature sur le sujet proposent souvent de "corriger" les notes des utilisateurs afin d'améliorer les résultats en rendant plus cohérentes les observations les unes par rapport aux autres. Deux effets peuvent en effet être identifiés dans nos données, à savoir :

l'effet "film" : il y a des bons films (notes très hautes) et de films plus mauvais. Il parait donc nécessaire de centrer les notes par rapport aux films. On compute donc, pour chaque film i :

$$\text{MAvg}_i \overset{\text{def}}{=} \frac{\sum_u r_u}{\sum_u e_{u,i}}$$

où $r_{u,i}$ est la note donnée par l'utilisateur u au film i, et $e_{u,i}$ vaut 1 si l'utilisateur u a noté le film i et 0 sinon.

l'effet "utilisateur" : De la même manière, des utilisateurs sont plus ou moins enclin à mettre des mauvaises notes. On va donc calculer une moyenne de note par utilisateur à partir de des notes corrigées de l'effet film, afin de centrer une seconde fois nos observations.

On calcule donc cette fois la moyenne par utilisateur u des notes corrigées de l'effet film :

$$\text{UAvg}_u \overset{\text{def}}{=} \frac{\sum_i r_{u,i} - \text{MAvg}_i}{c_u}$$

De là, on calcule la note centrée finale donnée par un utilisateur u à un film i en posant simplement :

$$\text{centered\_rating}_{u,i} \overset{\text{def}}{=} r_{u,i} - \text{UAvg}_u$$

Les observations une fois centrées, nous avons ensuite pu procéder au calcul des similarités entre les films. L'idée repose sur la construction pour chaque paire de films, de deux vecteurs de notes, un pour chaque film, de longueur le nombre d'utilisateurs ayant regardé ces films. Une fois ces vecteurs construits, la similarité entre les deux films est calculée à l'aide du coefficient de corrélation de Pearson, dont la formule pour deux films i et j est rappelée ci-dessous :

$$\text{pearson\_correlation}(i,j) \overset{\text{def}}{=} \frac{Cov(i,j)}{\sqrt{Var(i)Var(j)}}$$

Cette similarité est calculée pour chaque paire de films et a été stockée dans une table de taille $(n^2, 2)$. Elle a ensuite ainsi pu être utilisée dans la prédiction finale des notes. Pour ce faire, nous avons élaboré un algorithme k-NN permettant d'identifier, pour chaque utilisateur de la base de test, les films qu'il a regardés par le passé les plus similaires au film dont nous cherchons à prédire la note. Ladite note est alors calculée en prenant le vote majoritaire parmi ces plus proches voisins.

C'est précisément ce script qui n'a malheureusement pas abouti. Ce script ("compute_reco.pig") a été lancé le samedi 9 février 2015 en matinée ("job_1422615021176_2290"), malheureusement à l'heure actuelle, soit plus de 2 jours plus tard, celui-ci n'est toujours par terminé (20%). Nous ne serons donc pas en mesure de présenter des résultats portant sur l'intégralité de la base d'apprentissage, malheureusement.

5 Résultats et conclusion

Comme expliqué précédemment, nous n'avons pas pu lancer notre modèle sur la base de test, et n'avons pas donc pas pu computer de valeur. Le RMSE final trouvé est de XXX. De plus, Y% des revues ont été correctement prédites. C'est sans surprise que l'on découvre que ces résultats, obtenus grâce à un apprentissage sur une base très réduite, ne sont pas satisfaisants. En effet, le faible nombre de revues disponible par utilisateur dans l'apprentissage rend délicat toute prédiction.

En plus du temps de calcul important rendant impossible l'apprentissage sur la base de données totale, la principale difficulté rencontrée est venue de la complexité de prise en main du langage *PIG*, qui bien que similaire au langage *SQL* comporte de nombreuses particularités. De plus, la faible quantité de documentation disponible rend obscure l'écriture du code et la correction des erreurs.

Si les résultats obtenus sur l'échantillon ne sont guère probants, nous sommes néanmoins confiants sur le fait que l'utilisation de la technologie Map Reduce permette d'améliorer la qualité globale des recommandations lorsque l'ensemble des données sont prises en compte. Toutefois, le temps de calcul actuel rend impossible l'utilisation de notre modèle en production. On pourrait penser à l'utilisation de OVJMBmohoi pour rendre les calculs plus rapides.