Introduction

Définition CF
Attentes (précision/temps de réponse/ajout rapide d'un rating/cold start...)
Efficacité peut varier selon la forme du jeu de données (sparsity...)

1 Les algorithmes utilisés

1.1 Algorithmes témoins

Algorithmes naïfs servant de point de comparaison.
– Estimer un rating inconnu par la moyenne de tous les ratings connus
– Estimer la note donnée par un utilisateur à un objet par la moyenne des notes données par les autres utilisateurs à cet objet
– Retirer les biais comme dans le cours, estimer tous les ratings inconnus par 0, remettre les biais

1.2 SVD

Variantes sur la question bonus du DM (comment traiter les trous dans la matrice, avec ou sans traitement des biais...).

1.3 Algorithmes Slope One

Idée : régression linéaire, en imposant que la pente vaut 1.
Ajout d'un nouveau rating et traitement de requêtes très rapides, il faut voir à quel point c'est moins précis que d'autres algos plus sophistiqués.
Algorithmes tirés de [2].

1.4 Algorithmes par similarité cosinus

Implémentation de l'algorithme vu en cours

1.5 Analyse en composantes principales : algorithme Eigentaste

Idée : s'appuyer sur un petit sous-ensemble d'objets notés par tous les utilisateurs (*gauge set*) pour projeter un utilisateur sur un espace de petite dimension puis estimer ses notes à partir de celles de ses voisins au sens d'un algorithme de clustering.
Défauts : on demande à un nouvel utilisateur de noter l'intégralité du gauge set pour l'ajouter, et le cold start pose problème.
La précision est-elle vraiment meilleure que celle d'autres algorithmes plus simples ?
Algorithme extrait de [1].

2 Observations expérimentales

2.1 Jeux de données

– Matrice de la question bonus du DM (pleine, on observe seulement une certaine fraction des ratings)
– Jeu de données Jester (ratings d'une centaine de blagues, dix blagues sont notées par toutes les utilisateurs) utilisé pour Eigentaste.

– Jeu de données plus grand (MovieLens) pour mesurer les difficultés liées aux temps d'exécutions dans des conditions plus réalistes ?

2.2 Mesures d'erreur

2.3 Temps d'exécution

Conclusion

Références

[1] Ken Goldberg, Theresa Roeder, Dhruv Gupta, and Chris Perkins. Eigentaste : A constant time collaborative filtering algorithm. *Inf. Retr.*, 4(2) :133–151, July 2001.

[2] Daniel Lemire and Anna Maclachlan. Slope one predictors for online rating-based collaborative filtering. *CoRR*, abs/cs/0702144, 2007.