TP Systèmes de recommandation
SD-TSIA 211

Encadrement : Pascal Bianchi, Olivier Fercoq, Alex Lambert, Eugène Ndiaye et Umut Simsekli
27 novembre 2017

Le TP se fait en binôme. Vous ferez un rapport accompagné des fonctions associées aux questions à envoyer au plus tard le 5 décembre sur le site pédagogique. Vous pouvez rendre le rapport sous forme d'un notebook python ou d'un fichier pdf.

Nous allouerons ensuite à chaque binôme un rapport à évaluer. Vous devrez donner une note en utilisant la grille d'évaluation fournie sur le site pédagogique.

1 Présentation du modèle

U est l'ensemble des utilisateurs, I est l'ensemble des items (ici les films). Pour chaque couple (u, i), soit l'utilisateur u n'a pas regardé le film i et nous n'avons pas de donnée, soit nous connaissons une note $R_{u,i}$ du film i par l'utilisateur u.

Suivant [KBV09], on fait l'hypothèse qu'il existe un espace latent de caractéristiques jointes C tel que les interactions utilisateur-item sont des produits scalaires dans cet espace. Suivant ce modèle, on devrait avoir $R_{u,i} \approx \sum_{c \in C} Q_{u,c} P_{c,i}$ où $Q_{u,:}$ est une représentation de l'utilisateur u dans l'espace C et $P_{:,i}$ est une représentation de l'item i dans ce même espace C. La force du modèle est de prédire une note probable que donnerait l'utilisateur u s'il regardait le film i, et donc de lui proposer les films qu'il n'a pas vus mais qu'il est susceptible d'aimer.

Il suffit ensuite d'entraîner ce modèle en utilisant des moindres carrés régularisés.

$$\begin{aligned}(\hat{P}, \hat{Q}) &= \arg\min_{P,Q} \frac{1}{2} \sum_{(u,i) \in K} \left(R_{u,i} - \sum_{c \in C} Q_{u,c} P_{c,i} \right)^2 + \frac{\rho}{2} \left(\sum_{u \in U, c \in C} Q_{u,c}^2 + \sum_{i \in I, c \in C} P_{c,i}^2 \right) \\ &= \arg\min_{P,Q} \ \frac{1}{2} \|1_K \circ (R - QP)\|_F^2 + \frac{\rho}{2} \|Q\|_F^2 + \frac{\rho}{2} \|P\|_F^2\end{aligned} \tag{1}$$

où K est l'ensemble des couples (u, i) pour lesquels $R_{u,i}$ est connu, $\|\cdot\|_F$ est la norme de Frobenius, $(1_K)_{u,i} = 0$ si $(u, i) \notin K$ et $(1_K)_{u,i} = 1$ si $(u, i) \in K$, $(A \circ B)_{u,i} = A_{u,i} B_{u,i}$ et $\rho > 0$ est un paramètre de régularisation. Dans la suite, nous prendrons $\rho = 0.3$ et $C = \{0, \ldots, 3\}$.

Quand $\rho = 0$ et $K = U \times I$, la solution de ce problème correspond à une SVD tronquée. Dans le cas général, il faut utiliser un algorithme d'optimisation.

Question 1.1
Récupérer la base de données Movielens [HKBR99] sur le site suivant :

http://files.grouplens.org/datasets/movielens/ml-100k.zip.

Lancer la fonction `load_movielens` de `movielens_utils.py` avec le bon nom de fichier et vérifier que la matrice R retournée est bien de taille $943 \times 1\,682$. Que fait l'option `minidata` ?

Question 1.2
Combien y a-t-il d'utilisateurs, de films référencés dans la base de données ? Quel est le nombre total de notes ?

Question 1.3
La fonction objectif est-elle convexe ? Quel est son gradient ? Est-il lipschitzien ? Donner la constante de Lipschitz le cas échéant. On pourra remarquer que la fonction que l'on cherche à minimiser est un polynôme.

2 Trouver P quand Q_0 est fixé

On initialise l'algorithme de résolution avec Q^0 (resp. P^0) défini comme les F vecteurs singuliers à gauche (resp. à droite) de R associés à ses $|C|$ plus grandes valeurs singulières. Les valeurs manquantes de R seront fixées à 0 pour cette étape d'initialisation. Vous pourrez utiliser la fonction `scipy.sparse.linalg.svds`.

On cherche pour l'instant à résoudre le problème plus simple suivant :

$$\begin{aligned} g(P) &= \frac{1}{2}\|1_K \circ (R - Q^0 P)\|_F^2 + \frac{\rho}{2}\|Q^0\|_F^2 + \frac{\rho}{2}\|P\|_F^2 \\ P^1 &= \arg\min_P \; g(P) \end{aligned}$$

Question 2.1
La fonction objectif g est-elle convexe ? Quel est son gradient ? On admettra que le gradient est lipschitzien de constante $L_0 = \rho + \|(Q^0)^\top Q^0\|_F$.

Question 2.2
La fonction fournie `objective` calcule $g(P)$. Compléter cette fonction pour qu'elle calcule aussi $\nabla g(P)$. Vous pourrez vérifier votre calcul avec la fonction `scipy.optimize.check_grad` (vous aurez peut-être besoin des fonctions `numpy.reshape` et `numpy.ravel` car `check_grad` ne gère pas les variables matricielles).

Question 2.3
Coder une fonction `gradient(g, P0, gamma, epsilon)` qui minimise une fonction g par la méthode du gradient à pas constant γ en partant du point initial P^0 et avec le critère d'arrêt $\|\nabla g(P_k)\|_F \leq \epsilon$.

Question 2.4
Utiliser la fonction codée à la question précédente pour minimiser la fonction g jusqu'à la précision $\epsilon = 1$.

3 Raffinements algorithmiques pour le problème à Q_0 fixé

Question 3.1
Rajouter une méthode de recherche linéaire à la méthode du gradient, de manière à s'affranchir du calcul de la constante de Lipschitz.

Question 3.2
Justifiez que l'on peut utiliser la méthode du gradient conjugué pour ce problème et codez-la.

Question 3.3
Comparer les performances des trois algorithmes.

4 Résolution du problème complet

Question 4.1
Résoudre le problème (1) par la méthode du gradient avec recherche linéaire jusqu'à la précision $\epsilon = 100$. À quoi correspond ce qui est retourné par l'algorithme ?

Question 4.2
Quand Q (resp. P) est fixé, le problème est facile à résoudre. La méthode des moindres carrés alternés utilise ce fait et consiste en l'algorithme suivant :

---

for $k \geq 1$ **do**
  $P_k \leftarrow \arg\min_P \frac{1}{2}\|1_K \circ (R - Q_{k-1}P)\|_F^2 + \frac{\rho}{2}\|Q_{k-1}\|_F^2 + \frac{\rho}{2}\|P\|_F^2$
  $Q_k \leftarrow \arg\min_Q \frac{1}{2}\|1_K \circ (R - QP_k)\|_F^2 + \frac{\rho}{2}\|Q\|_F^2 + \frac{\rho}{2}\|P_k\|_F^2$
end for

Montrer que la valeur de l'objectif décroît à chaque itération. En déduire qu'elle converge.

Question 4.3
Coder la méthode des moindres carrés alternés.

Question 4.4
Comparer la méthode du gradient avec recherche linéaire et la méthode des moindres carrés alternés. Est-ce que les solutions sont les mêmes ? Est-ce que la prédiction $\hat{R} = QP$ est la même ? Est-ce que la valeur de l'objectif est la même ? Comment se comparent les temps de calcul ?

Question 4.5
Quel film recommanderiez-vous à l'utilisateur 300 ?

Références

[HKBR99] Jonathan L Herlocker, Joseph A Konstan, Al Borchers, and John Riedl. An algorithmic framework for performing collaborative filtering. In *Proc. of 22nd ACM SIGIR conference*, pages 230–237. ACM, 1999.

[KBV09] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, (8) :30–37, 2009.