Systèmes de recommandation

Shengrui Wang

November 18, 2024

Université de Sherbrooke

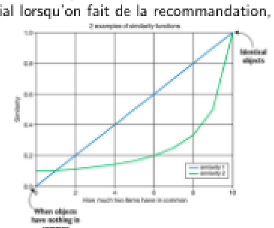
Plan

Introduction

- De manière générale nous avons,
  - Introduit la notion de système de recommandation,
  - Vu quelques techniques de recommandation en particulier celles basées sur le filtrage par contenu et sur le filtrage collaboratif,
  - Constaté qu'on ne pourrait faire de la recommandation sans parler de similarité soit entre utilisateurs soit entre items.
- Pour cette partie du cours, nous allons,
  - Davantages parler des métriques qui nous aide à faire de la recommandation.
  - Voir comment évaluer un système de recommandation.

Mesures de similarité

- Le choix de la métrique ou de la mesure de similarité a un enjeu crucial lorsqu'on fait de la recommandation,



- Il n'y a vraiment pas un choix ultime, tout dépend du jeu de données.

Coefficient de communauté: indice de Jaccard

- Applicable avec des données binaires

	Comedy	Action	Comedy	Action	Drama	Drama
Sara	1	1	0	0	0	0
Jesper	1	1	1	0	0	0
Therese	1	0	0	0	0	0
Helle	0	1	0	1	0	0
Pietro	0	0	0	0	1	1
Ekaterina	0	0	0	0	1	1

$$Sim(U_u, U_v) = \frac{|\boldsymbol{I}_u \cap \boldsymbol{I}_v|}{|\boldsymbol{I}_u \cup \boldsymbol{I}_v|} \text{ avec } \boldsymbol{I}_u, \text{ l'ensemble d'items l'uager } U_u \text{ a achetés.}$$

$$Sim(I_i, I_j) = \frac{|\boldsymbol{U}_i \cap \boldsymbol{U}_j|}{|\boldsymbol{U}_i \cup \boldsymbol{U}_j|} \text{ avec } \boldsymbol{U}_i, \text{ l'ensemble d'usagers qui ont acheté } I_i.$$

L_p Norme

- S'applique sur les *ratings*,

$$dist(U_u, U_v) = \left(\sum_{I_i \in \boldsymbol{I}_u \cap \boldsymbol{I}_v} |r_{u,i} - r_{v,i}|^p \right)^{1/p}$$

$$dist(I_i, I_j) = \left(\sum_{U_u \in \boldsymbol{U}_i \cap \boldsymbol{U}_j} |r_{u,i} - r_{u,j}|^p \right)^{1/p}$$

- Et ainsi obtenir la similarité comme suit

$$sim(\square, \star) = \frac{1}{1 + dist(\square, \star)}$$

Similarité cosinus et corrélation de Pearson

S'applique sur les *ratings*

- Cosinus,

$$sim(U_u, U_v) = \frac{\sum_{I_i \in I_u \cap I_v} r_{u,i} \times r_{v,i}}{\sqrt{\sum_{I_i \in I_u \cap I_v} r_{u,i}^2}\sqrt{\sum_{I_i \in I_u \cap I_v} r_{v,i}^2}}$$

$$sim(I_i, I_j) = \frac{\sum_{U_u \in U_i \cap U_j} r_{u,i} \times r_{u,j}}{\sqrt{\sum_{U_u \in U_i \cap U_j} r_{u,i}^2}\sqrt{\sum_{U_u \in U_i \cap U_j} r_{u,j}^2}}$$

qui pourrait être normalisé comme suit,

$$sim(U_u, U_v) = \frac{\sum_{I_i \in I_u \cap I_v} (r_{u,i} - \bar{r}_u) \times (r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{I_i \in I_u \cap I_v} (r_{u,i} - \bar{r}_u)^2}\sqrt{\sum_{I_i \in I_u \cap I_v} (r_{v,i} - \bar{r}_v)^2}}$$

$$sim(I_i, I_j) = \frac{\sum_{U_u \in U_i \cap U_j} (r_{u,i} - \bar{r}_i) \times (r_{u,j} - \bar{r}_j)}{\sqrt{\sum_{U_u \in U_i \cap U_j} (r_{u,i} - \bar{r}_i)^2}\sqrt{\sum_{U_u \in U_i \cap U_j} (r_{u,j} - \bar{r}_j)^2}}$$

avec $\bar{r}_u$ la moyenne des *ratings* de l'utilisateur U_u sur les items pris dans $I_u \cap I_v$
et $\bar{r}_i$ la moyenne des *ratings* qu'a eu l'item I_i parmis les utilisateurs pris dans $U_i \cap U_j$

Similarité cosinus et corrélation de Pearson

- Corrélation de Pearson,

$$sim(U_u, U_v) = \frac{\sum_{I_i \in I_u \cap I_v} (r_{u,i} - \bar{r}_u) \times (r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{I_i \in I_u \cap I_v} (r_{u,i} - \bar{r}_u)^2}\sqrt{\sum_{I_i \in I_u \cap I_v} (r_{v,i} - \bar{r}_v)^2}}$$

$$sim(I_i, I_j) = \frac{\sum_{U_u \in U_i \cap U_j} (r_{u,i} - \bar{r}_i) \times (r_{u,j} - \bar{r}_j)}{\sqrt{\sum_{U_u \in U_i \cap U_j} (r_{u,i} - \bar{r}_i)^2}\sqrt{\sum_{U_u \in U_i \cap U_j} (r_{u,j} - \bar{r}_j)^2}}$$

avec $\bar{r}_u$ la moyenne des *ratings* de l'utilisateur U_u sur tous les items qu'il a noté.
et $\bar{r}_i$ la moyenne des notes qu'a eu l'item I_i.

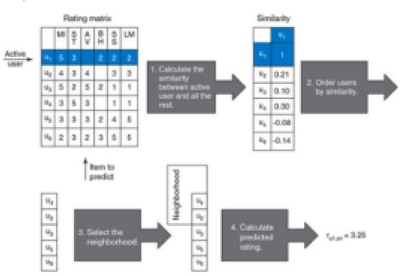
Similarité cosinus et corrélation de Pearson

Faire le calcul des similarités cosinus, cosinus normalisé et Pearson entre utilisateurs et entre items dans le cas suivant:

	Comedy	Action	Comedy	Action	Drama	Drama
Sara	5	3		2	2	2
Jesper	4	5	4		3	3
Therese	5	3	5	2	1	1
Helle	3		3	5	1	1
Pietro	3	3	3	2	4	5
Ekaterina	2	3	2	3	5	5

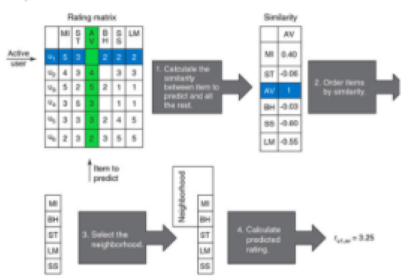
Filtrage collaboratif basé sur le voisinage

Voisinage utilisateurs



Filtrage collaboratif basé sur le voisinage

Voisinage items



Évaluation d'un système de recommandation

- Une fois que votre algorithme de recommandation a été défini, il est important de savoir si ce dernier effectue de bonnes recommandations.
- Pour ce faire nous avons besoin de l'évaluer.
- L'évaluation d'un modèle de recommandation peut se faire de deux manières distinctes:
  1. Évaluation hors ligne
  2. Évaluation en ligne

Évaluation hors ligne

- L'évaluation dans ce cas de figure requiert une bonne quantité de données pour s'assurer que notre modèle de recommandation puisse faire une bonne généralisation.
- À partir des données existantes, on va créer un scénario où l'on prendra une partie des données comme information connue (données d'entraînement) et la partie restante comme information inconnue (données de test)

Évaluation hors ligne

- Dans l'échantillon d'entraînement, on va apprendre notre modèle de recommandation à reconnaître les *ratings* des utilisateurs.
- Pour cela on va cacher certains *ratings* et voir comment notre modèle de recommandation s'adapte dans sa tâche de prédiction: On parle d'entraînement du modèle.
- L'entraînement du modèle recommandation requiert une métrique d'évaluation.
- L'objectif de l'entraînement est de trouver le(s) paramètre(s) de notre modèle de recommandation qui minimiserai(en)t la métrique d'évaluation.

Évaluation hors ligne

- Formellement,
- Étant donné un modèle de recommandation défini par la fonction $f()$ (qui prédit le rating d'un utilisateur), ayant pour paramètre ω et une mesure d'évaluation $Eval()$,
- L'entraînement sur un ensemble *Train* consiste à trouver le meilleur paramètre $\hat{\omega}$ qui minimiserait notre mesure d'évaluation.

$$\hat{\omega} = \underset{\omega}{argmin}\{Eval(f(U_u|\omega), r_{u,i}) \mid \forall U_u,\ I_i \in Train\}$$

Évaluation hors ligne

Exemple,

- Étant donné un système de recommandation où les *ratings* r peuvent prendre les valeurs: 1, 2, 3, 4 ou 5.
- Supposons que notre function de recommandation soit une moyenne pondérée donnée par,

$$f(I_i|U_u,\Omega) = \frac{1}{|\boldsymbol{U}_{v-u}|}\Omega \cdot \left(\sum_{U_v \in \boldsymbol{U}_{v-u}} \sum_{I_j \in I_v} \sum_{r=1}^{5} r_{v,j}\Delta_r(r_{v,j})\right)^T$$

Avec $\boldsymbol{U}_{v-u} = \{U_v, U_v \neq U_u \ / \ \boldsymbol{I}_u \cap \boldsymbol{I}_v \neq \emptyset\}$
$\Omega = (\omega_1, \omega_2, \omega_3, \omega_4, \omega_5)$
$\Delta_r(r_{v,j}) = (1,0,0,0,0)$ si $r = r_{v,j} = 1$ (resp. $\Delta_r(r_{v,j}) = (0,1,0,0,0)$ si $r = r_{v,j} = 2$, ..., $\Delta_r(r_{v,j}) = (0,0,0,0,1)$ si $r = r_{v,j} = 5$)

Évaluation hors ligne

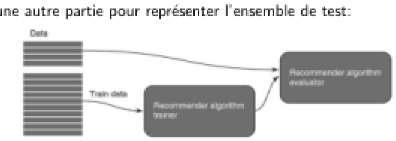
- $Eval(a,b) = |a-b|$ notre function de perte.
- Entraîner notre function de recommandation, reviendrait à chercher les meilleurs paramètres $\hat{\Omega}$ qui minimisent la fonction

$$\hat{\Omega} = \underset{\Omega}{min} \sum_{U_u, I_i \in Train} \left|f(I_i|U_u,\Omega) - r_{u,i}\right|$$

- Ils existent plusieurs moyens d'estimation des paramètres tels que: la descente des gradients, le maximum de vraissemblance, Newton-Raphson etc. ...
- Nous n'aborderons pas les méthodes d'estimation des paramètres dans ce cours ... ☹

Évaluation hors ligne – Validation croisée

- Souvenons nous avons besoin de sélectionner une partie de nos données pour représenter l'ensemble d'entraînement et une autre partie pour représenter l'ensemble de test:

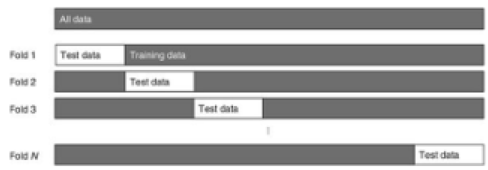


- **Il est important de noter que les paramètres estimés valent particulièrement pour ce cas de figure. On ne saurait encore conclure si notre fonction de recommandation fait une bonne recommandation.**

Évaluation hors ligne – Validation croisée

- La subdivision des données en deux sous-ensembles pourraient générer des cas de figure favorables/défavorables.
- Pour s'assurer qu'on ne favorise pas un cas en particulier, on pourrait générer à partir de nos données, plusieurs sous-ensemble entraînements et tests.
- Ensuite tester notre fonction de recommandation pour chacun des cas et prendre la moyenne.
- Ce type de procédé est généralement connu sous le nom de *validation croisée*.

Évaluation hors ligne – Validation croisée



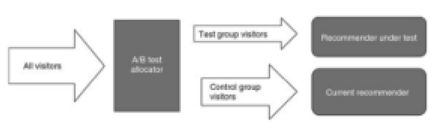
Évaluation en ligne

- Contrairement à la méthode hors ligne qui ne base que sur la connaissance des données existantes,
- l'évaluation en ligne tient compte des nouveaux cas de figures pour juger de la performance du modèle de recommandation.
- Pour une évaluation en ligne, on a très souvent besoin de deux systèmes de recommandation:
  1. Un système de recommandation existant/courant.
  2. Un nouveau système de recommandation.

Évaluation en ligne

- Sur la base des utilisateurs existants, on définit deux sous-ensembles: Un ensemble de *test* et un ensemble de *contrôle*.
- Les utilisateurs pris dans l'ensemble de *contrôle* continueront à être soumis au système de recommandation courant;
- tandis que, les utilisateurs pris dans l'ensemble de *test* seront soumis au nouveau système de recommandation.
- Rouler les deux systèmes de recommandation pendant une certaine durée (quelques jours, semaines ou des mois ...)
- Tirer une conclusion sur celui qui performe le mieux.

Évaluation en ligne



Techniques avancées

- Problèmes avec des approches simples
  1. Couverture limitée
  2. Sensibilité aux données éparses
- Méthodes basée sur la réduction de dimensionnalité
  1. Décomposition de la matrice de "ratings"
  2. Décomposition de la matrice de similarité
- Méthodes basées sur le parcours des graphes (Cette partie de méthodes ne sera pas abordée)
  1. Similarité basée sur chemins de parcours
  2. Similarité basée sur la marche aléatoire

Exemple de LSI

- La matrice mots clés - documents : $\boldsymbol{A}$ =

	c1	c2	c3	c4	c5	m1	m2	m3	m4
human	1	0	0	1	0	0	0	0	0
interface	1	0	1	0	0	0	0	0	0
computer	1	1	0	0	0	0	0	0	0
user	0	1	1	0	1	0	0	0	0
system	0	1	1	2	0	0	0	0	0
response	0	1	0	0	1	0	0	0	0
time	0	1	0	0	1	0	0	0	0
EPS	0	0	1	1	0	0	0	0	0
survey	0	1	0	0	0	0	0	0	1
trees	0	0	0	0	0	1	1	1	0
graph	0	0	0	0	0	0	1	1	1
minors	0	0	0	0	0	0	0	1	1

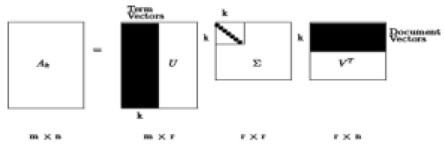
Exemple de LSI

- Décomposition des valeurs singulières (SVD) :

$$\boldsymbol{A} = U \Sigma V^t$$

- SVD réduit :

$$\boldsymbol{A}_k = U_k \Sigma_k V_k^t$$



Exemple de LSI

- U =

0.22	-0.11	0.29	-0.41	-0.11	-0.34	0.52	-0.06	-0.41
0.20	-0.07	0.14	-0.55	0.28	0.50	-0.07	-0.01	-0.11
0.24	0.04	-0.16	-0.59	-0.11	-0.25	-0.30	0.06	0.49
0.40	0.06	-0.34	0.10	0.33	0.38	0.00	0.00	0.01
0.64	-0.17	0.36	0.33	-0.16	-0.21	-0.17	0.03	0.27
0.27	0.11	-0.43	0.07	0.08	-0.17	0.28	-0.02	-0.05
0.27	0.11	-0.43	0.07	0.08	-0.17	0.28	-0.02	-0.05
0.30	-0.14	0.33	0.19	0.11	0.27	0.03	-0.02	-0.17
0.21	0.27	-0.18	-0.03	-0.54	0.08	-0.47	-0.04	-0.58
0.01	0.49	0.23	0.03	0.59	-0.39	-0.29	0.25	-0.23
0.04	0.62	0.22	0.00	-0.07	0.11	0.16	-0.68	0.23
0.03	0.45	0.14	-0.01	-0.30	0.28	0.34	0.68	0.18

Exemple de LSI

- Σ =

3.34								
	2.54							
		2.35						
			1.64					
				1.50				
					1.31			
						0.85		
							0.56	
								0.36

Exemple de LSI

- V =

0.20	0.61	0.46	0.54	0.28	0.00	0.01	0.02	0.08
-0.06	0.17	-0.13	-0.23	0.11	0.19	0.44	0.62	0.53
0.11	-0.50	0.21	0.57	-0.51	0.10	0.19	0.25	0.08
-0.95	-0.03	0.04	0.27	0.15	0.02	0.02	0.01	-0.03
0.05	-0.21	0.38	-0.21	0.33	0.39	0.35	0.15	-0.60
-0.08	-0.26	0.72	-0.37	0.03	-0.30	-0.21	0.00	0.36
0.18	-0.43	-0.24	0.26	0.67	-0.34	-0.15	0.25	0.04
-0.01	0.05	0.01	-0.02	-0.06	0.45	-0.76	0.45	-0.07
-0.06	0.24	0.02	-0.08	-0.26	-0.62	0.02	0.52	-0.45

Méthodes de décomposition de la matrice de "ratings" (cont)

- Problème avec la méthode basée sur LSI :
  - Pour appliquer SVD, $\boldsymbol{R} = \bar{U}\Sigma\bar{V}^t$, $\boldsymbol{R}$ doit être définie partout, ce qui n'est souvent pas le cas.
  - Une solution pratique : affectant les valeurs par défaut à r_{ui}. Cela crée un biais
- Une solution plus juste : créer l'espace latent par optimisation

$$err\left(P, Q\right) = \sum_{r_{u,i} \in \boldsymbol{R}} \left(\left(r_{u,i} - p_u q_i^t\right)^2 + \lambda \left(\|p_u\|^2 + \|q_i\|^2 \right) \right)$$

La minimisation de $err(P, Q)$ se fait pas une approche itérative (ne sera pas abordée dans ce cours. Voir Bell, R. et al (KDD 2007), Koren (KDD 2008), Takacs (JMLR)).

Méthodes de décomposition de la matrice de "ratings" (cont)

- Le sous-produit de la projection est que les coordonnées de projection de u et i peuvent ensuite être utilisées pour estimer la similarité entre les utilisateurs et entre les éléments.

$$w_{u,v} = sim(U_u, U_v) = p_u p_v^t$$
$$w_{i,j} = sim(I_i, I_j) = q_i q_j^t$$

Voir Billsus (ICML 98).

Méthodes de décomposition de la matrice de similitude

- Méthode basée sur la diagonalisation de la matrice de similitude $\boldsymbol{W}$.

$$err(P) = \left\| \boldsymbol{W} - \boldsymbol{P}\boldsymbol{P}^t \right\|^2 = \sum_{u,v} \left(w_{u,v} - p_u p_v^t \right)^2$$

L'optimisation de cette fonction d'erreur donne lieu à la solution suivante :

$$\boldsymbol{W} = V \Lambda V^t$$

où Λ est la matrice diagonale composée des valeurs propres de W, et V est une matrice orthogonale composée des vecteurs propres de W.

Méthodes de décomposition de la matrice de similitude (cont)

- Similaire à l'approche SVD, $\boldsymbol{W}$ peut être approximé par :

$$\hat{\boldsymbol{W}} = \boldsymbol{P}_k \boldsymbol{P}_k^t$$

où $\boldsymbol{P}_k$ est obtenue en utilisant les k plus grandes valeurs propres et les vecteurs propres correspondants, c'est-à-dire :

$$\boldsymbol{P}_k = V_k \Lambda_k^{1/2}$$

Voir Goldberg, K. et al (2001) qui ont construit le système Eigentaste qui recommande des blagues.