Systèmes de recommandation

Julien Romero

Limitations des moteurs de recherche

- Il faut savoir ce que l'on cherche
  - Comment découvrir de nouveau films, livres, produits ?
- Les résultats ne sont pas personnalisés
  - Le résultat "album de musique" renvoie la même chose pour quelqu'un qui écoute du métal, du classique, ou du rap
- Les moteurs de recherches sont statiques
  - Ne prend pas en compte ce qui est actuellement populaire

Systèmes de recommandation

- Un système de recommandation est un algorithme capable de suggérer un objet (page web, film, livre, musique, personne) pour un utilisateur et un context donné
- Plus formellement, soit U l'ensemble des utilisateurs et I l'ensemble des objets (items), un système de recommandation prédit un score R d'utilité pour chaque couple d'utilisateur et d'objets. Ce score peut être un entier entre 0 et 1, ou une note sur 5.

$$u : U \times I \rightarrow R$$

Exemples

Google News

La personnalisation de la recommandation

- Recommandations non personnalisées
  - Souvent choisies pour plaire à une catégorie de personnes
  - Articles dans un journal
  - Vidéo Youtube “10 livres à lire pour devenir riche”
  - Top 10
  - Plus récents
- Recommandations personnalisées
  - Choisies pour un individu donné
  - Amazon, Netflix, Youtube, ...
  - Ce qui nous intéresse aujourd’hui !

D'où viennent les données ?

- Pour entraîner et tester un système de recommandation, il nous faut des données
  - Ce sont des valeurs connues de notre fonction d'utilité
- Données explicites
  - On peut demander à l'utilisateur de noter (de 1 à 5, like/dislike)
  - Ne marche pas trop en pratique
- Données implicites
  - On va capturer des signaux venant des utilisateurs
  - Signaux forts : Achat d'un objet, visionnage d'un film, page web visitée
  - Signaux faibles : click sur un produit, temps passé sur une vidéo, nombre d'interactions par minute, ...

Matrice d'utilité

- Matrice contenant les valeurs connues de la fonction d'utilité
  - Matrice très creuse en général

	Harry Potter	Le seigneur des anneaux	Good bye Lenin	La chute	OSS 117	Les visiteurs
Marie	5	?	3	2	3	?
Louise	4	5	4	1	2	2
Jean	1	?	?	?	?	?
Maurice	?	3	5	4	2	?

Recommandation basée sur le contenu

Création d'un profil d'objet et d'utilisateur

- On peut créer un vecteur de caractéristiques pour chaque objet
  - Film : Genre, réalisateur, durée, date de production, langue, titre
  - Livre : auteur, genre, nombre de pages, contenu (avec TF-IDF)
  - Offre d'emploi : type de contrat, compétences, expérience requise, formation
- Idem pour l'utilisateur
  - Moyenne (pondérée) des objets avec lesquels il a interagit
  - Sexe, age, formation, compétences, genre préféré, ...

Recommander un nouvel objet

- Recommander l'objet le plus proche des objets connues de l'utilisateur
  - K plus proches voisins
  - Heuristiques/approximations quand trop d'objets
  - On peut utiliser la similarité cosinus
- Entraîner un modèle de régression/classification
  - Approche classique de machine learning

Avantages de l'approche par contenu

- Pas besoin d'avoir accès aux données des autres utilisateurs
- Marche même si l'utilisateur a des goûts particuliers
  - (Avec les bonnes caractéristiques)
- Marche avec les objets nouveaux et non populaires
- Facile de produire des explications sur la recommandation

Inconvénients de l'approche par contenu

- Il faut créer soi-même le vecteur de caractéristique
  - Souvent difficile
- Surspécialisation
  - Pas de recommandation en dehors de ce qui est connu
  - Difficile de modéliser des intérêts multiples
  - N'exploite pas le jugement des autres utilisateurs
- Ne fonctionne pas bien avec les nouveaux utilisateurs sans interactions
  - Le problème du "Cold-start"

Le problème du “cold-start”

- Comment traiter le cas particulier du “démarrage” du système ?
  - À la création d’un objet ou utilisateur, nous n’avons pas d’interaction le concernant
- La recommandation suivra souvent une stratégie simple et générique
  - Top 10, Tendance en ce moment, …
- Problème récurrent

Le filtrage collaboratif

Le filtrage collaboratif

- **Idée principale** : On base la recommandation d'un objet pour un utilisateur sur l'ensemble des interactions entre les objets et les utilisateurs.
- Le vecteur de caractéristique va être obtenu à partir de la matrice d'utilité
  - Les colonnes nous donnent les vecteurs des objets
  - Les lignes nous donnent les vecteurs des utilisateurs
- Deux variantes :
  - Filtrage collaboratif basé sur les utilisateurs
  - Filtrage collaboratif basé sur les objets

Filtrage collaboratif basé sur les utilisateurs

- Pour un *u* utilisateur donné :
  - Trouver les N plus proches utilisateurs (selon la matrice d'utilité)
  - Estimer le vecteur de *u* à partir des N autres utilisateurs
- Mesures de similarité
  - Similarité de Jaccard
    - Mais en ignorant les valeurs des notes
  - Similarité cosinus
    - Mais en traitant les valeurs manquantes comme négatives
  - Coefficient de corrélation de Pearson (S(xy) est l'ensemble des interactions communes)

$$sim(x,y) = \frac{\sum_{s \in S_{xy}} (r_{xs} - \bar{r_x})(r_{ys} - \bar{r_y})}{\sqrt{\sum_{s \in S_{xy}} (r_{xs} - \bar{r_x})^2} \sqrt{\sum_{s \in S_{xy}} (r_{ys} - \bar{r_y})^2}}$$

Prédire le vecteur de caractéristique

- Si *r(x)* est le vecteur de caractéristique d'un utilisateur *x*, et *N* est le set des plus proches utilisateurs ayant prédit un objet *i*, alors on pourrait définir plusieurs métriques pour construire le score final

$$r_{xi} = \frac{1}{k} \sum_{y \in N} r_{yi}$$

$$r_{xi} = \frac{\sum_{y \in N} sim(x,y) * r_{yi}}{\sum_{y \in N} sim(x,y)}$$

Filtrage collaboratif basé sur les objets

On peut appliquer les mêmes principes sur les objets.

$$r_{xi} = \frac{1}{k} \sum_{j \in N} r_{xj} \qquad r_{xi} = \frac{\sum_{j \in N} sim(i,j) * r_{xj}}{\sum_{j \in N} sim(i,j)}$$

Exemple - Basé sur l'utilisateur

	Harry Potter	Le seigneur des anneaux	Good bye Lenin	La chute	OSS 117	Les visiteurs	Similarité
Marie	5	?	3	2	3	?	
Louise	4	4	4	1	3	2	
Jean	1	?	?	5	?	?	
Maurice	?	3	5	4	2	?	

Exemple - Basé sur l'utilisateur - Calcul de la similarité

	Harry Potter	Le seigneur des anneaux	Good bye Lenin	La chute	OSS 117	Les visiteurs	Similarité
Marie	5	?	3	2	3	?	1
Louise	4	4	4	1	3	2	0.75
Jean	1	?	?	5	?	?	-0.99
Maurice	?	3	5	4	2	?	-0.22

Exemple - Basé sur l'utilisateur - Sélection des K (=2) plus proches

	Harry Potter	Le seigneur des anneaux	Good bye Lenin	La chute	OSS 117	Les visiteurs	Similarité
Marie	5	?	3	2	3	?	1
Louise	4	4	4	1	3	2	0.75
Jean	1	?	?	5	?	?	-0.99
Maurice	?	3	5	4	2	?	-0.22

Exemple - Basé sur l'utilisateur - Moyenne pondérée

	Harry Potter	Le seigneur des anneaux	Good bye Lenin	La chute	OSS 117	Les visiteurs	Similarité
Marie	5	4.41	3	2	3	?	1
Louise	4	4	4	1	3	2	0.75
Jean	1	?	?	5	?	?	-0.99
Maurice	?	3	5	4	2	?	-0.22

Exemple - Basé sur l'objet

	Harry Potter	Le seigneur des anneaux	Good bye Lenin	La chute	OSS 117	Les visiteurs
Marie	5	?	3	2	3	?
Louise	4	4	4	1	3	2
Jean	1	?	?	5	?	?
Maurice	?	3	5	4	2	?
Similarité						

Exemple - Basé sur l'objet - Calcul de la similarité

	Harry Potter	Le seigneur des anneaux	Good bye Lenin	La chute	OSS 117	Les visiteurs
Marie	5	?	3	2	3	?
Louise	4	4	4	1	2	2
Jean	1	?	?	?	?	?
Maurice	?	3	5	4	2	?
Similarité	1.0	1.0	-0.71	-0.95	0.95	-1

Exemple - Basé sur l'objet - Sélection des K (=2) plus proches

	Harry Potter	Le seigneur des anneaux	Good bye Lenin	La chute	OSS 117	Les visiteurs
Marie	5	?	3	2	3	?
Louise	4	4	4	1	2	2
Jean	1	?	?	?	?	?
Maurice	?	3	5	4	2	?
Similarité	1.0	1.0	-0.71	-0.95	0.95	0

Exemple - Basé sur l'objet - Moyenne pondérée

	Harry Potter	Le seigneur des anneaux	Good bye Lenin	La chute	OSS 117	Les visiteurs
Marie	5	4.03	3	2	3	?
Louise	4	4	4	1	2	2
Jean	1	?	?	?	?	?
Maurice	?	3	5	4	2	?
Similarité	1.0	1.0	-0.71	-0.95	0.95	0

Filtrage collaboratif basé sur les objets ou les utilisateurs ?

- En général, le filtrage collaboratif basé sur les objets donne de meilleurs résultats
  - Pourquoi ? Le objets sont généralement plus simples
  - Les utilisateurs ont de nombreux intérêts, qui changent avec le temps

Avantages et inconvénients du filtrage collaboratif

+ Marche avec tous les types d'objets
  + Pas besoin de passer beaucoup de temps à créer des caractéristiques
- Cold start
  - Il faut beaucoup d'utilisateurs et un minimum d'interaction pour chaque utilisateur et objet
  - Difficile pour les nouveaux objets ou les objets qui sortent de l'ordinaire
- Matrice d'utilité sparse
  - Difficile de trouver des utilisateurs ayant noté les mêmes objets
- Biais de popularité
  - Tendance à recommander les objets que tout le monde aime

Méthodes hybrides

- Souvent, il faut implémenter plusieurs modèles différent et combiner les résultats
- On prendra l'approche basée sur le contenu pour les utilisateurs et objets avec peu d'interaction
- On bascule vers du filtrage collaboratif par la suite

Évaluation

Évaluer un système de recommandation

- Comme pour toutes les tâches de machine learning, on divise notre dataset en un jeu d'entraînement et un jeu de test (et un jeu de validation si possible)
- On a ensuite le choix entre plusieurs métriques
  - Racine de l'erreur quadratique moyenne (Root Mean Square Error)
    - Utile quand on a des scores

$$\sqrt{\frac{1}{N}\sum_{xi}\left(r_{xi} - r^*_{xi}\right)^2}$$

Évaluer un système de recommandation

- Comme pour toutes les tâches de machine learning, on divise notre dataset en un jeu d'entraînement et un jeu de test (et un jeu de validation si possible)
- On a ensuite le choix entre plusieurs métriques
  - Racine de l'erreur quadratique moyenne (Root Mean Square Error)
    - Utile quand on a des scores
  - Métriques basées sur le rang
    - Pour un utilisateur donné, on prédit le score pour tous les objets et on les classe
    - On associe à chaque objet s'il est pertinent ou pas
    - On calcule des métriques basées sur le rang des objets pertinents

Métrique sur les rangs

	Pertinent ?
Objet 1	Oui
Objet 2	Non
Objet 3	Non
Objet 4	Oui
Objet 5	Non
Objet 6	Non

- Precision@K : Pourcentage d'objets pertinents dans les K premiers
  - Precision@3 = 1 / 3
- Recall@K : Nombre d'objets pertinents dans les K premiers, divisé par le nombre d'objets pertinents
  - Recall@3 = 1 / 2
- Mean Reciprocal Rank (MRR) : Moyenne des inverses des rangs du premier objet pertinent
  - MRR = (1/1 + RR autres utilisateurs) / (# utilisateurs)

Métrique sur les rangs

	Pertinent ?
Objet 1	Oui
Objet 2	Non
Objet 3	Non
Objet 4	Oui
Objet 5	Non
Objet 6	Non

- Average Precision (AP) @ K :
  - AP@3 = 1/2 * (1 * 1 + 1/2*0 + 1/3*0) = 1/2
  - N = Nombre d'objets pertinents pour u
  - rel(k) = pertinence de l'objet au rang k

$$AP@K(u) = \frac{1}{N} \sum_{k=1}^{K} Precision@K * rel(k)$$

- Mean Average Precision (MAP) @ K : Moyenne des AP@K sur tous les utilisateurs

Métrique sur les rangs

	Pertinent ?
Objet 1	Oui
Objet 2	Non
Objet 3	Non
Objet 4	Oui
Objet 5	Non
Objet 6	Non

- Discounted Cumulative Gain (DCG) @ K :
  - DCG@3 = 1 / log(2) = 1

$$DCG@K = \sum_{k=1}^{K} \frac{rel(k)}{log_2(i+1)}$$

- Normalized DCG (NDCG) @ K :
  - NDCG@K = 1 / (1/log2(2) + 1/log2(3)) = 0.61
  - IDCG@K = Meilleur DCG@K possible

$$NDCG@K = \frac{DCG@K}{IDCG@K}$$

Filtrage collaboratif basé sur un modèle

Filtrage collaboratif basé sur un modèle

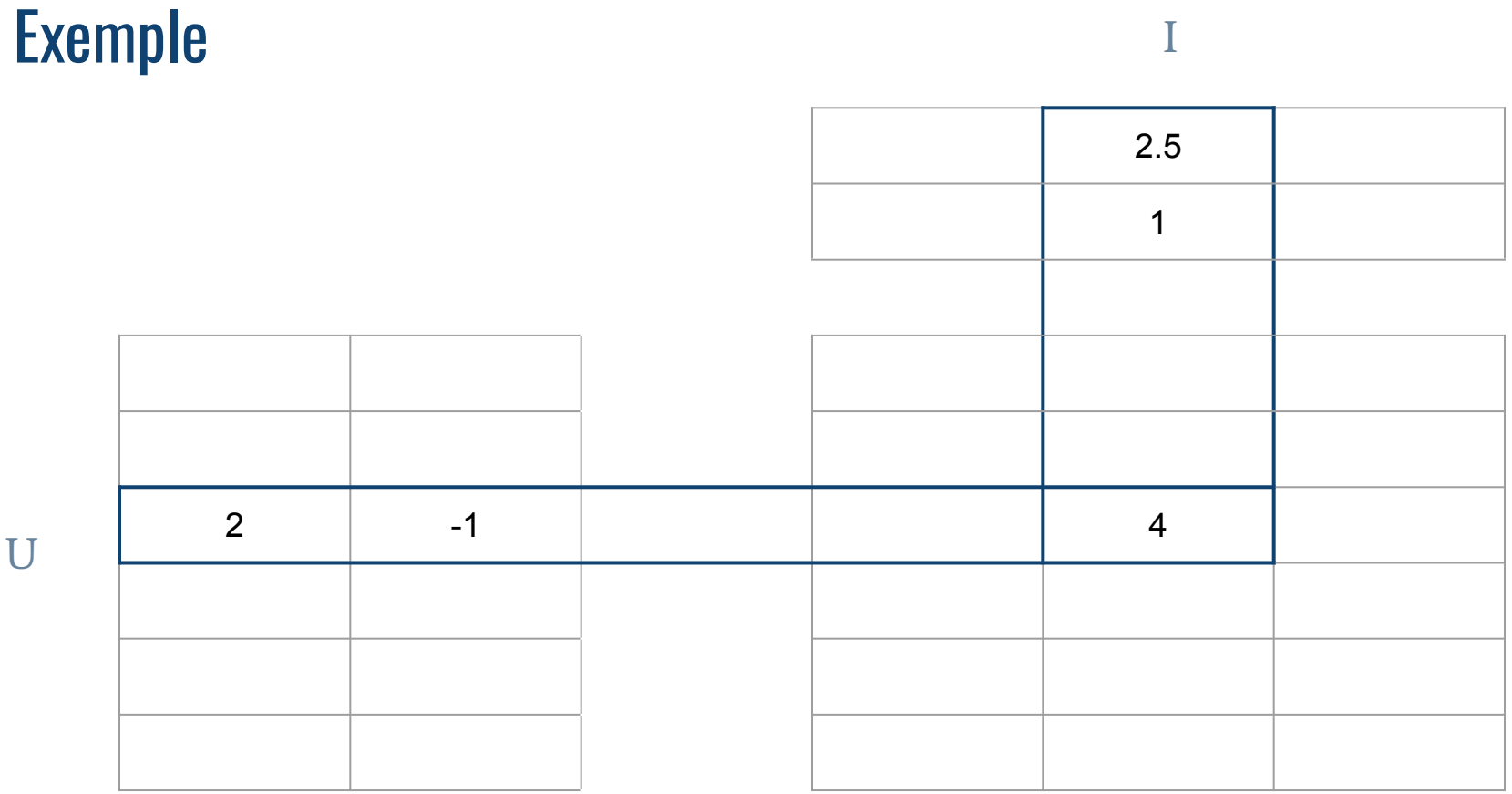
- Jusqu'à présent, nous avons utilisé la matrice d'utilité brute et une approche statistique
  - Communément appelé "basé sur la mémoire" (memory-based)
- Cependant, nous avons une matrice très creuse à manipuler
- L'autre approche consiste à compresser la matrice d'utilité pour la rendre plus dense
  - C'est l'approche basée sur un modèle (model-based)

Factorisation de matrice

- La factorisation de matrice consiste à décomposer une matrice en une multiplication de plusieurs matrices (plus petites)
  - Dans notre cas, nous allons avoir au moins une matrice correspondant aux utilisateurs, et une matrice correspondant aux objets
  - Si nous avons une matrice de taille U (nombre d'utilisateurs) par I (nombre d'objet), nous voulons factoriser la matrice d'utilité de taille U*I en deux matrices de taille U*k et I*k (avec k petit)
  - Les vecteurs associés aux utilisateurs et objets sont appelés des vecteurs latents
  - Pour notre matrice d'utilité M, nous cherchons deux matrices telles que

$$M \approx PQ^T$$

Exemple



Prédire la valeur d'une interaction

- Si p(x) est le vecteur latent d'un utilisateur et q(i) le vecteur latent d'un objet, la valeur de l'interaction sera donnée par :

$$\hat{r_{xi}} = q_i \cdot p_x = \sum q_{i,k} * p_{x,k}$$

Singular Value Decomposition (SVD)

- SVD est un algorithme qui factorise une matrice M de taille m*n en trois matrices U matrice unitaire de taille m*k, Σ matrice diagonale de taille k*l et V matrice unitaire de taille n*l telles que :

$$M \approx U\Sigma V^T$$

- Pour obtenir les vecteurs latents des utilisateurs et objets, on posera

$$P = U \qquad Q = \Sigma V^T$$

Propriété de SVD

- SVD nous donne l'erreur de reconstruction minimale

$$min_{U,V,\Sigma} \sum (M_{i,j} - [U\Sigma V^T]_{i,j})^2$$

- Cependant, notre matrice est creuse ! Il faut donc résoudre le problème sur les interactions connues (jeu d'entrainement):

$$min_{P,Q} \sum_{(x,i)} (M_{i,j} - p_x \cdot q_i)^2$$

- On peut rajouter de la régularisation (conseillé)

$$+\lambda_1 \sum_x ||p_x||^2 + \lambda_2 \sum_i ||q_i||^2$$

Résolution

- Problème résolu avec une descente de gradient (stochastique)

$$min_{P,Q} \sum_{(x,i)} (M_{i,j} - p_x \cdot q_i)^2 + \lambda_1 \sum_x ||p_x||^2 + \lambda_2 \sum_i ||q_i||^2$$

Autres algorithmes

- Analyse en composantes principales, ACP (Principal component analysis, PCA)
- Alternating Least Square (ALS)

Pour aller plus loin…

- Il est possible d'inclure dans la modélisation le score moyen et les déviations de l'utilisateur ou de l'objet
  - Certains utilisateurs sont plus gentils, d'autres plus sévères
  - Certains objets sont globalement meilleurs ou pires

$$r_{xi} = \mu + b_x + b_i + p_x \cdot q_i$$

Score moyen

Biais de l'utilisateur

Biais de l'objet

Interactions

- Ajouter un aspect temporel (sur les biais par exemple)

En route pour le TP