

协同过滤的基础理论

目录

CONTENT

1. 什么是CF
2. Memory-based CF
3. Model-based CF
4. 小结

CSL
Career
Science
Lab

目录

CONTENT

1. 什么是CF

2. Memory-based CF

3. Model-based CF

4. 小结

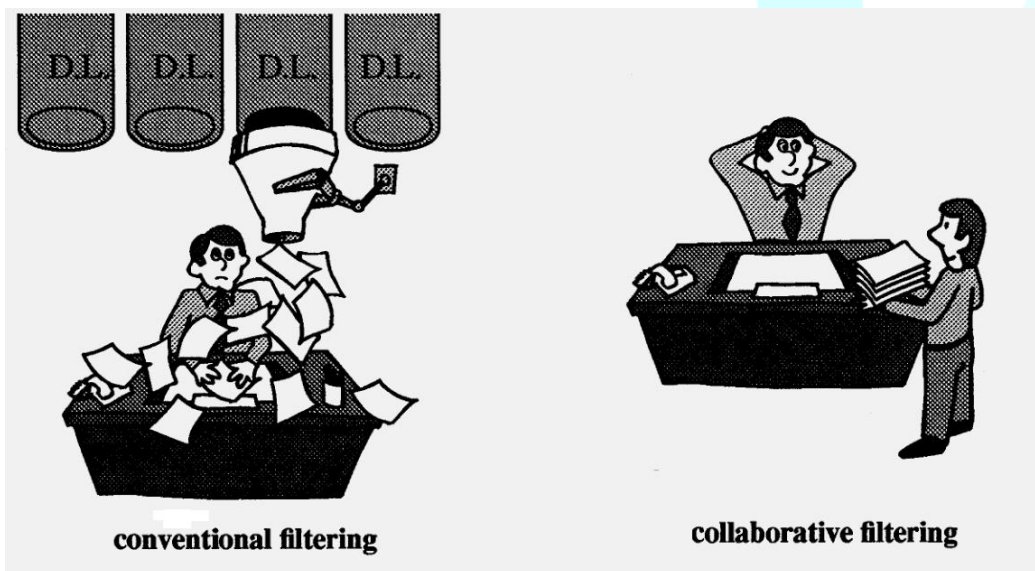
CSL
Career
Science
Lab

数据->信息->知识->智慧 (目标)

信息检索 (搜索和推荐)

信息过载 => 信息**过滤**

- 内容: Content-based filtering (self's X as feature)
- 行为: Collaborative filtering (other's Y as feature)























1. 协同过滤

Collaborative Filtering (CF) A means of filtering items for one user of a system based on [the implicit or explicit rating \(feedback\)](#) of items by other users of that system. For example, filtering emails based on others' responses to the same emails or recommending movies based on others' ratings of those movies.

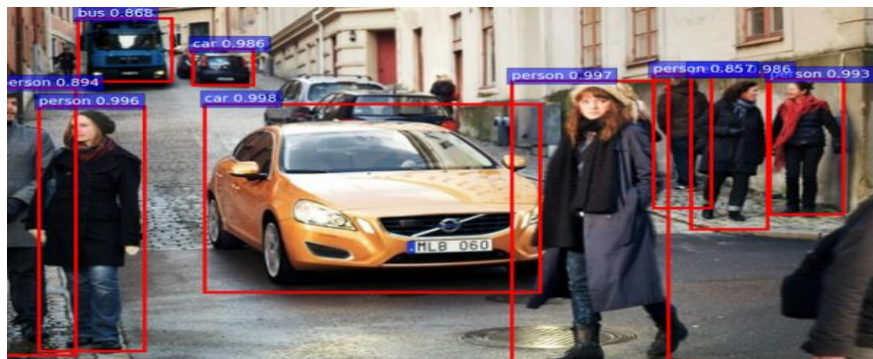
“CF makes predictions (filtering) about a user's interest by collecting preferences information from many users (collaborating)” --- Wikipedia

原理: "物以类聚, 人以群分" (The birds of a feather gather together.)

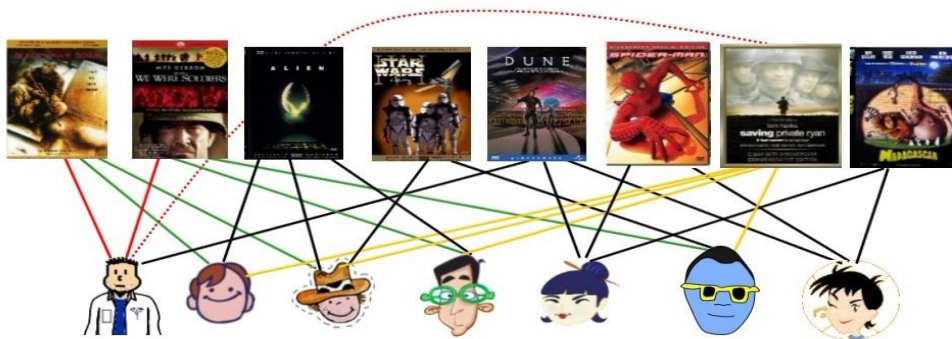
Movie					
The Lion King					
Lethal Weapon					
The Sound of Music					
Amadeus					
When Harry Met Sally					

两类任务

- 机器学习人的认知能力（理性）：CV, NLP



- 机器学习/理解人的行为（有限理性、非理性）：RS

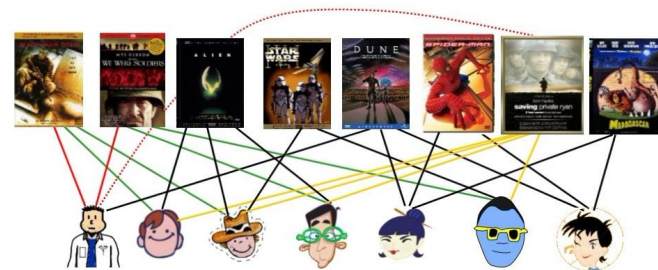


偏好挖掘 (理解人类的行为)

偏好: \geq (1) 完备性: $x \geq y$ 或 $y \geq z$;
(2) 传递性: 若 $x \geq y$ 且 $y \geq z$, 则有 $x \geq z$.

Debreu定理: 连续偏好 \Rightarrow 连续的效用函数.

效用函数: $U(x) \geq U(y) \leftrightarrow x \geq y$



	item 1	item 2	item 3	item 4	item 5	item 6	item 7
user 1	5	3	4	1	?	?	?
user 2	4	2	3	1	5	2	5
user 3	5	?	4	1	5	3	?
user 4	1	3	2	5	1	4	2
user 5	4	?	4	4	4	?	4

用户:

物品

用户 u 对物品 i 的喜好程度 (打分)

$$u, v \in U$$

$$i, j \in I$$

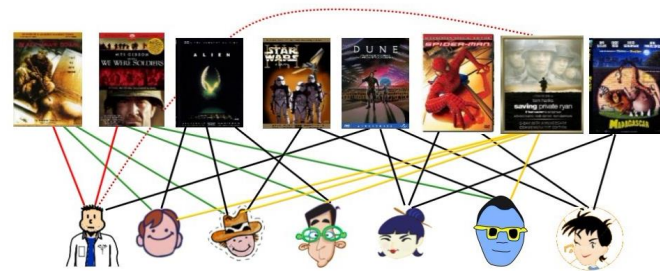
$$r_{u,i}$$

偏好挖掘 (理解人类的行为)

偏好: \geq (1) 完备性: $x \geq y$ 或 $y \geq z$; (2) 传递性: 若 $x \geq y$ 且 $y \geq z$, 则有 $x \geq z$.

Debreu定理: 连续偏好 \Rightarrow 连续的效用函数.

效用函数: $U(x) \geq U(y) \Leftrightarrow x \geq y$



	item 1	item 2	item 3	item 4	item 5	item 6	item 7
user 1	5	3	4	1	?	?	?
user 2	4	2	3	1	5	2	5
user 3	5	?	4	1	5	3	?
user 4	1	3	2	5	1	4	2
user 5	4	?	4	4	4	?	4

- 显示反馈 (explicit feedback) : 用户直接反映其对产品的喜好信息, 如评分等等。
- 隐式反馈 (implicit feedback) : 间接反映出用户对于产品的喜好, 如: 购买历史、浏览记录、搜索记录等

CF分类

- **Memory-based CF** : instance-based learning
 - User-based CF: 利用用户的相似性
 - Item-based CF : 利用物品的相似性
- **Model-based CF**: model-based learning
 - Latent factor model (latent semantic model)
 - ...

目录

CONTENT

1. 什么是CF

2. **Memory-based CF**

3. **Model-based CF**

4. 小结

CSL
Career
Science
Lab

Memory-based CF

kNN (k-Nearest Neighbor)

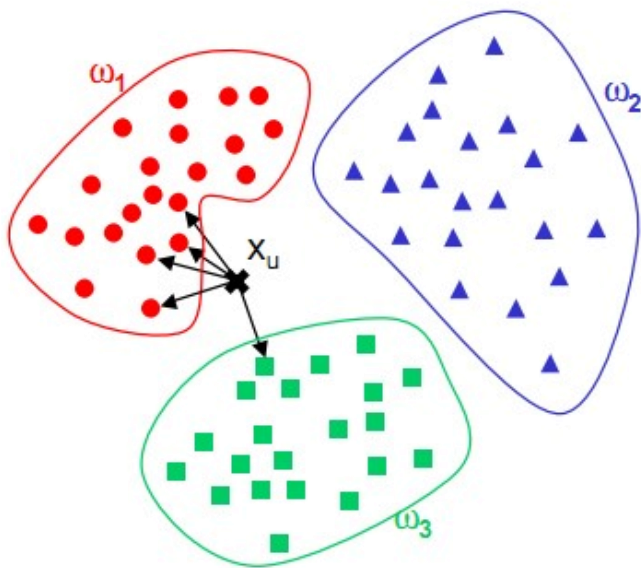
$$r_{u,i} = f_{user}(r_{v_1,i}, \dots, r_{v_k,i})$$

$$r_{u,i} = f_{item}(r_{u,j_1}, \dots, r_{u,j_k})$$

- kNN算法是数据挖掘技术中最简单的方法之一
- kNN是基于实例的学习 (instance-based learning), 没有显式的学习过程, 使用新样本可以直接进行预测.

Memory-based CF

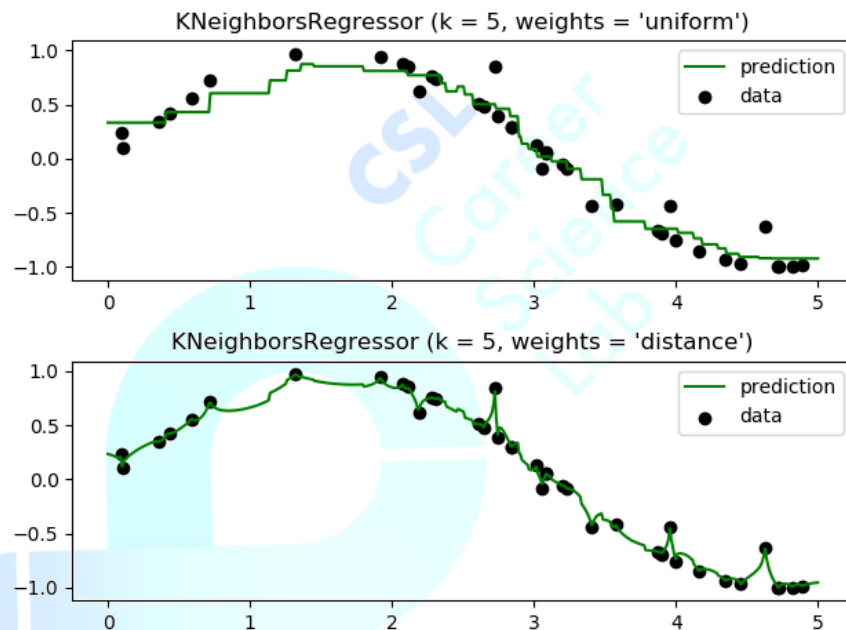
kNN分类



$$p(y = c|x, k) = 1/k \sum_{i=1}^k I_{x_i \in N_x}(y_i = c)$$

```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors = 3)
knn.fit(X_train, y_train)
```

kNN回归



$$y_i = \frac{1}{\sum_{x_j \in N_{x_i}} sim(x_j, x_i)} \sum_{v \in N_{x_i}} sim(x_j, x_i) y_j$$

```
from sklearn.neighbors import KNeighborsRegressor
knn = KNeighborsRegressor(n_neighbors = 5)
knn.fit(X, y)
```

Memory-based CF

相似性（距离）

cosine相似性

$$sim(x, y) = \frac{\sum_i x_i y_i}{\sqrt{\sum_i x_i^2} \sqrt{\sum_i y_i^2}}$$

Pearson相关系数:

$$sim(x, y) = \frac{\sum_i (x_i - \bar{x}_i)(y_i - \bar{y}_i)}{\sqrt{\sum_i (x_i - \bar{x}_i)^2} \sqrt{\sum_i (y_i - \bar{y}_i)^2}}$$

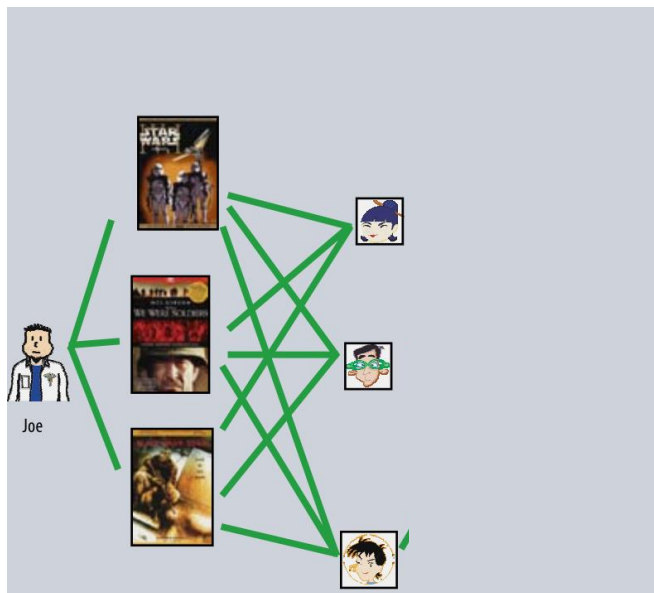
Jaccard相似性

$$sim(x, y) = \frac{|N(x) \cap N(y)|}{|N(x) \cup N(y)|}$$

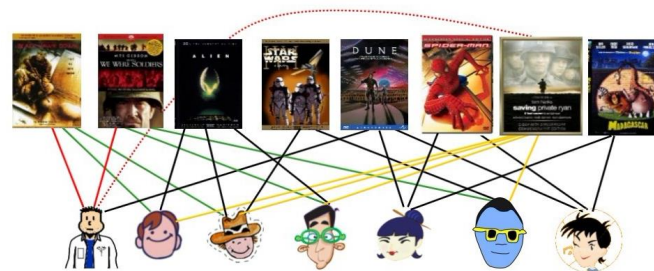
Memory-based CF

user-based CF

user与user的相关性



$$r_{u,i} = f_{user}(r_{v_1,i}, \dots, r_{v_k,i})$$



	item 1	item 2	item 3	item 4	item 5	item 6	item 7
user 1	5	3	4	1	?	?	?
u2	4	2	3	1	5	2	5
u3	5	?	4	1	5	3	?
user 4	1	3	2	5	1	4	2
user 5	4	?	4	4	4	?	4

$$\text{sim}(u2, u3) = \cos(u2, u3)$$

$$\hat{r}_{u,i} = \frac{1}{\sum_{v \in N_u} \text{sim}(v, u)} \sum_{v \in N_u} \text{sim}(v, u) r_{v,i}$$

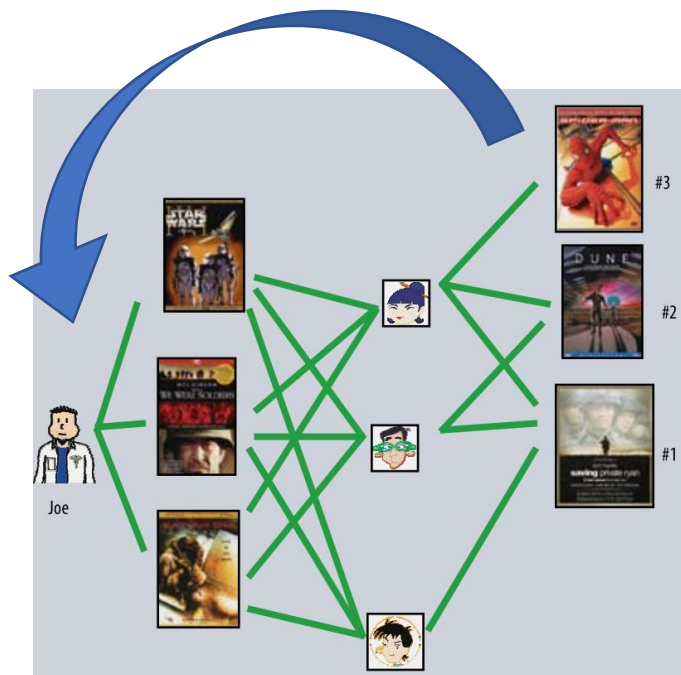
$$\hat{r}_{u,i} = \bar{r}_u + \frac{1}{\sum_{v \in N_u} \text{sim}(v, u)} \sum_{v \in N_u} \text{sim}(v, u) (r_{v,i} - \bar{r}_v)$$

[Goldberg et al., 1992] Goldberg, D., Nichols, D., Oki, B. M., and Terry, D. (1992). Using collaborative filtering to weave an information tapestry. Communications of the ACM, 35:61–70.

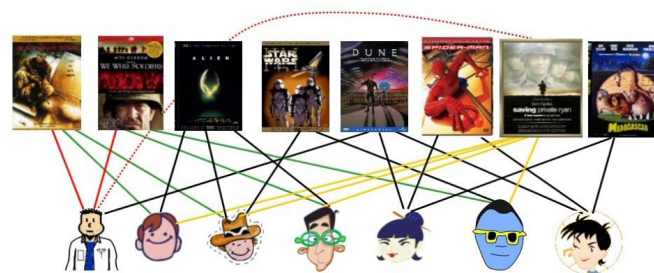
Memory-based CF

user-based CF

user与user的相关性



$$r_{u,i} = f_{user}(r_{v_1,i}, \dots, r_{v_k,i})$$



	item 1	item 2	item 3	item 4	item 5	item 6	item 7
user 1	5	3	4	1	?	?	?
u2	4	2	3	1	5	2	5
u3	5	?	4	1	5	3	?
user 4	1	3	2	5	1	4	2
user 5	4	?	4	4	4	?	4

$$\text{sim}(u2, u3) = \cos(u2, u3)$$

$$\hat{r}_{u,i} = \frac{1}{\sum_{v \in N_u} \text{sim}(v, u)} \sum_{v \in N_u} \text{sim}(v, u) r_{v,i}$$

$$\hat{r}_{u,i} = \bar{r}_u + \frac{1}{\sum_{v \in N_u} \text{sim}(v, u)} \sum_{v \in N_u} \text{sim}(v, u) (r_{v,i} - \bar{r}_v)$$

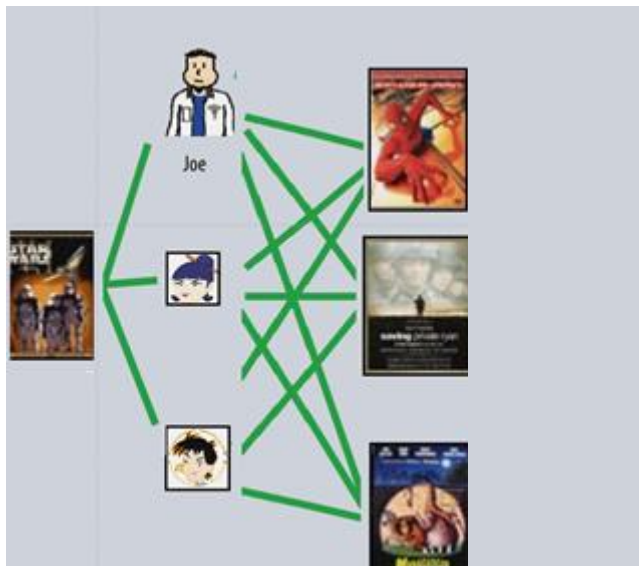
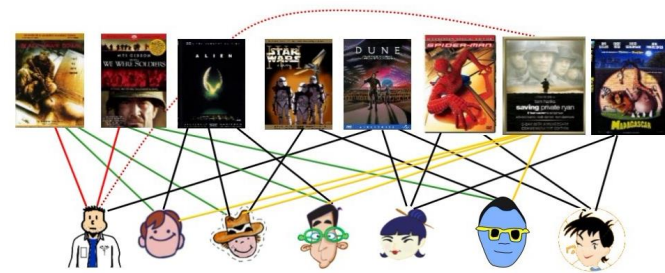
[Goldberg et al., 1992] Goldberg, D., Nichols, D., Oki, B. M., and Terry, D. (1992). Using collaborative filtering to weave an information tapestry. Communications of the ACM, 35:61–70.

Memory-based CF

item-based CF

Item与item的相关性

$$r_{u,i} = f_{item}(r_{u,j_1}, \dots, r_{u,j_k})$$



			i3	i4			
	item 1	item 2	item 3	item 4	item 5	item 6	item 7
user 1	5	3	4	1	?	?	?
user 2	4	2	3	1	5	2	5
user 3	5	?	4	1	5	3	?
user 4	1	3	2	5	1	4	2
user 5	4	?	4	4	4	?	4

$$\text{sim}(u2, u3) = \cos(i3, i4)$$

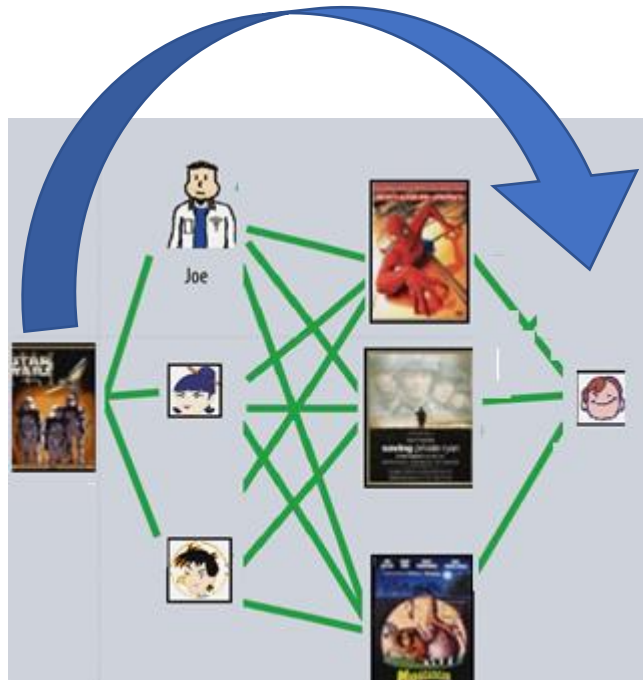
$$\hat{r}_{u,i} = \frac{1}{\sum_{j \in N_i} \text{sim}(j, i)} \sum_{j \in N_i} \text{sim}(j, i) r_{u,j}$$

[Sarpar et al., 2001] Sarpar, B. M., Karypis, G., Konstan, J. A., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. the pnb conference, 285-295.

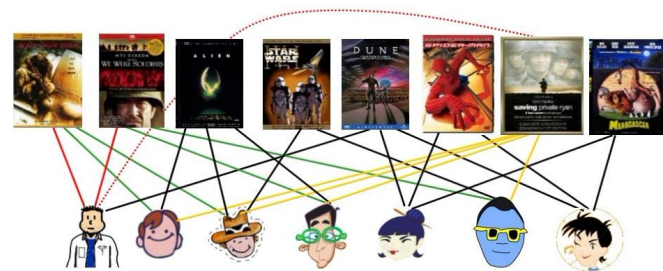
Memory-based CF

item-based CF

Item与item的相关性



$$r_{u,i} = f_{item}(r_{u,j_1}, \dots, r_{u,j_k})$$



			i3	i4			
	item 1	item 2	item 3	item 4	item 5	item 6	item 7
user 1	5	3	4	1	?	?	?
user 2	4	2	3	1	5	2	5
user 3	5	?	4	1	5	3	?
user 4	1	3	2	5	1	4	2
user 5	4	?	4	4	4	?	4

$$\text{sim}(u2, u3) = \cos(i3, i4)$$

$$\hat{r}_{u,i} = \frac{1}{\sum_{j \in N_i} \text{sim}(j, i)} \sum_{j \in N_i} \text{sim}(j, i) r_{u,j}$$

[Sarpar et al., 2001] Sarpar, B. M., Karypis, G., Konstan, J. A., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. the pnb conference, 285-295.

Memory-based CF

user-based CF 还是 item-based CF ?

经验

- 推荐场景 : user-user的相关性, item与item的相关性
- 计算复杂度 : $\hat{r}_{u,i} = \frac{1}{\sum_{j \in N_i} sim(j, i)} \sum_{j \in N_i} sim(j, i) r_{u,j}$ $\hat{r}_{u,i} = \frac{1}{\sum_{v \in N_u} sim(v, u)} \sum_{v \in N_u} sim(v, u) r_{v,i}$
- 推荐结果 : item-based方法推荐结果的多样性不如user-based
- 偏好稳定性 : 如果偏好稳定 : user-based; 若偏好不稳定 : item-based
- 冷启动 : item-based能应对新用户 ; user-based能应对新item

目录

CONTENT

1. 什么是CF
2. Memory-based CF
3. **Model-based CF**
4. 小结

CSL
Career
Science
Lab

Model-based CF

Latent factor model (LFM) (Latent semantic model)

- p 表示latent space中的偏好向量
- q 表示latent space中的属性向量

$$Preference(u, i) = r_{u,i} = p_u q_i^T$$

矩阵分解

	item 1	item 2	item 3	item 4	item 5	item 6	item 7
user 1	5	3	4	1	?	?	?
user 2	4	2	3	1	5	2	5
user 3	5	?	4	1	5	3	?
user 4	1	3	2	5	1	4	2
user 5	4	?	4	4	4	?	4

Model-based CF

Latent factor (known):

	item 1	item 2	item 3	item 4	item 5	item 6	item 7
user 1	5	3	4	1	0	0	0
user 2	4	2	3	1	5	2	5
user 3	5	0	4	1	5	3	0
user 4	1	3	2	5	1	4	2
user 5	4	0	4	4	4	0	4

$$Preference(u, i) = r_{u,i} = p_u q_i^T$$

user 1: $p_1 = (p_{1,1}, p_{1,2}, p_{1,3}, p_{1,4})$

线性回归

item因素 (身高、颜值、收入、学历)

- item 1: 180cm, 0.7, 20k, PhD = 4
- item 2: 160cm, 0.9, 15k, Master = 3
- item 3: 175cm, 0.8, 20k, bachelor = 2
- item 4: 185cm, 0.5, 17k, bachelor = 2

$$p_{1,height} \cdot 180 + p_{1,face} \cdot 0.7 + p_{1,income} \cdot 20 + p_{1,edu} \cdot 4 \approx 5$$

$$p_{1,height} \cdot 160 + p_{1,face} \cdot 0.9 + p_{1,income} \cdot 15 + p_{1,edu} \cdot 3 \approx 3$$

$$p_{1,height} \cdot 175 + p_{1,face} \cdot 0.8 + p_{1,income} \cdot 20 + p_{1,edu} \cdot 2 \approx 4$$

$$p_{1,height} \cdot 185 + p_{1,face} \cdot 0.5 + p_{1,income} \cdot 17 + p_{1,edu} \cdot 2 \approx 1$$

Model-based CF

Latent factor (known):

	item 1	item 2	item 3	item 4	item 5	item 6	item 7
user 1	5	3	4	1	0	0	0
user 2	4	2	3	1	5	2	5
user 3	5	0	4	1	5	3	0
user 4	1	3	2	5	1	4	2
user 5	4	0	4	4	4	0	4

$$Preference(u, i) = r_{u,i} = p_u q_i^T$$

user 2: $p_2 = (p_{2,1}, p_{2,2}, p_{2,3}, p_{2,4})$

线性回归

item因素 (身高、颜值、收入、学历)

- item 1: 180cm, 0.7, 20k, PhD = 4
- item 2: 160cm, 0.9, 15k, Master = 3
- item 3: 175cm, 0.8, 20k, bachelor = 2
- item 4: 185cm, 0.5, 17k, bachelor = 2

$$p_{2,height} \cdot 180 + p_{2,face} \cdot 0.7 + p_{2,income} \cdot 20 + p_{2,edu} \cdot 4 \approx 4$$

$$p_{2,height} \cdot 160 + p_{2,face} \cdot 0.9 + p_{2,income} \cdot 15 + p_{2,edu} \cdot 3 \approx 2$$

$$p_{2,height} \cdot 175 + p_{2,face} \cdot 0.8 + p_{2,income} \cdot 20 + p_{2,edu} \cdot 2 \approx 3$$

$$p_{2,height} \cdot 185 + p_{2,face} \cdot 0.5 + p_{2,income} \cdot 17 + p_{2,edu} \cdot 2 \approx 1$$

Model-based CF

Latent factor (known):

		q1	q2	q3	q4	q5	q6	q7
		item 1	item 2	item 3	item 4	item 5	item 6	item 7
p1	user 1	5	3	4	1	?	?	?
p2	user 2	4	2	3	1	5	2	5
p3	user 3	5	?	4	1	5	3	?
p4	user 4	1	3	2	5	1	4	2
p5	user 5	4	?	4	4	4	?	4

Item特征向量 q

- item 1: 180cm, 0.7, 20k, PhD = 4
- item 2: 160cm, 0.9, 15k, Master = 3
- item 3: 175cm, 0.8, 20k, bachelor = 2
- item 4: 185cm, 0.5, 17k, bachelor = 2

线性回归:

$$PQ^T \approx R$$

$$PQ^T Q \approx RQ$$

$$P = RQ(Q^T Q)^{-1}$$

$$\min \|R - PQ^T\|_2$$

Model-based CF

Latent factor (unknown):

- p 表示latent space中的偏好向量
- q 表示latent space中的属性向量

$$Preference(u, i) = r_{u,i} = p_u q_i^T$$

给定 R , 求解 P, Q

$$\min ||R - PQ^T||_2$$

$$\min_{P, Q} \sum_{(u,i) \in R} (r_{ui} - p_u q_i^T)^2 + \lambda(||q_i||^2 + ||p_u||^2)$$

Model-based CF

矩阵分解

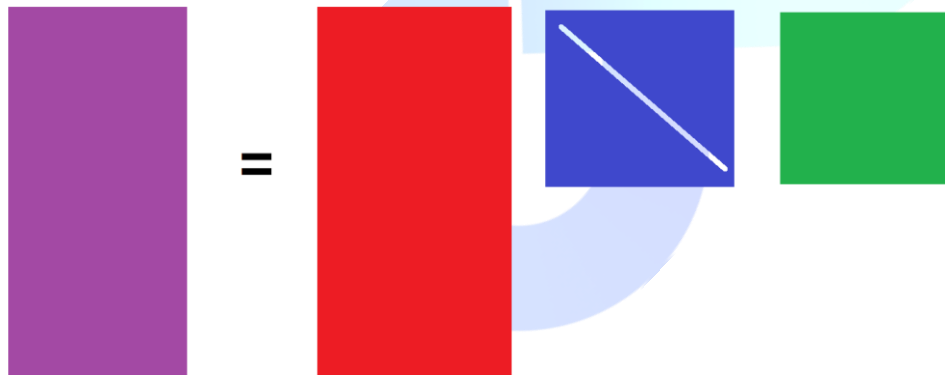
SVD (Singular Value Decomposition)

$$X = U\Sigma V^T$$

Full SVD



reduced SVD



Model-based CF

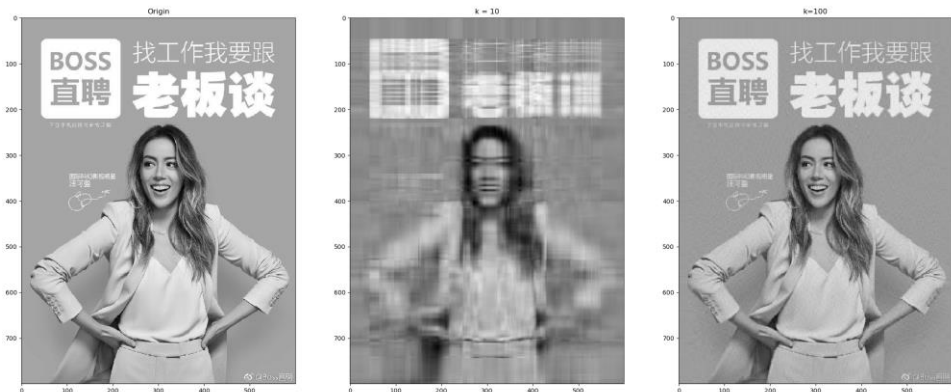
矩阵分解

pure **SVD** (Singular Value Decomposition)

$$X = U\Sigma V^T$$

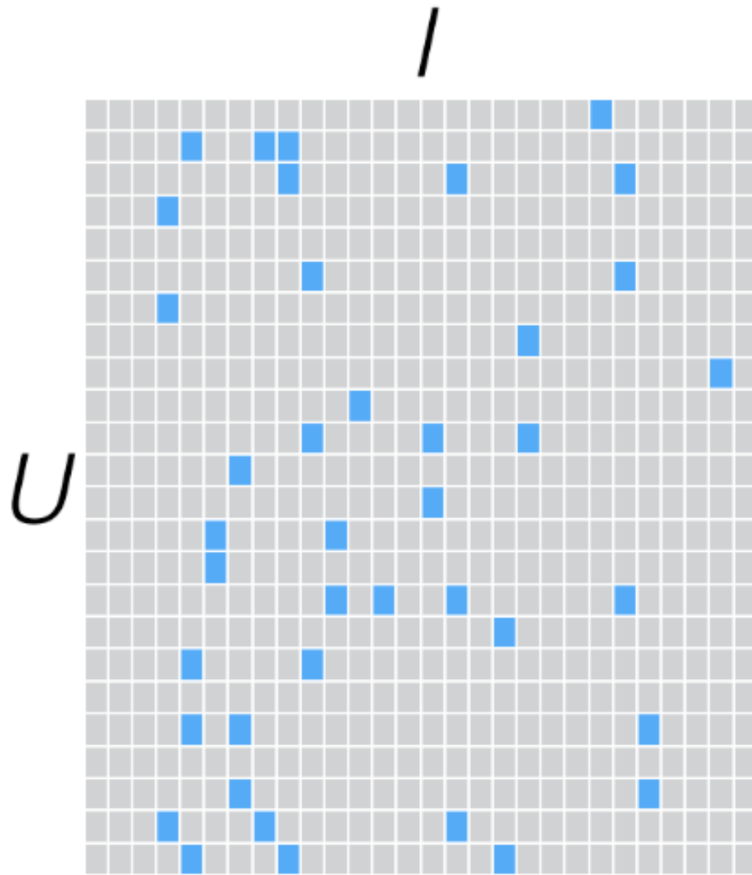


```
from PIL import Image
import numpy as np
import matplotlib.pyplot as plt
from numpy.linalg import svd
img = Image.open('boss.jpg')
x0 = np.array(img)
U, S, VT = svd(x0)
x1 = np.dot(np.dot(U[:, :10], np.diag(S[:10])), VT[:10, :])
x2 = np.dot(np.dot(U[:, :100], np.diag(S[:100])), VT[:100, :])
plt.subplot(1,3,1), plt.imshow(x0, cmap = 'gray'), plt.title('Origin')
plt.subplot(1,3,2), plt.imshow(x1, cmap = 'gray'), plt.title('k = 10')
plt.subplot(1,3,3), plt.imshow(x2, cmap = 'gray'), plt.title('k=100')
plt.show()
```



Model-based CF

Popularized by **Simon Funk**
during the Netflix Prize



Funk SVD

$$\min_{P,Q} \sum_{r_{u,i} \in R} (r_{u,i} - q_i^T p_u)^2 + \lambda(\|q_i\|^2 + \|p_u\|^2)$$

$$\hat{r}_{u,i} = \mu + b_i + b_u + q_i^T p_u$$

Stochastic gradient descent (SGD):

$$e_{u,i} \equiv r_{u,i} - q_i^T p_u$$

$$q_i \leftarrow q_i + \gamma(e_{u,i} - p_u - \lambda q_i)$$

$$p_u \leftarrow p_u + \gamma(e_{u,i} - q_i - \lambda q_i)$$

Alternate least square (ALS)

- 随机生成P和Q
- Repeat until convergence {
 固定Q, 使用最小二乘法更新P.
 固定P, 使用最小二乘法更新Q. }

Model-based CF

Matrix Factorization-based algorithms

```
class surprise.prediction_algorithms.matrix_factorization.SVD
```

Bases: `surprise.prediction_algorithms.algo_base.AlgoBase`

The famous SVD algorithm, as popularized by [Simon Funk](#) during the Netflix Prize. When baselines are not used, this is equivalent to Probabilistic Matrix Factorization [\[SM08\]](#) (see [note](#) below).

The prediction \hat{r}_{ui} is set as:

$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T p_u$$

If user u is unknown, then the bias b_u and the factors p_u are assumed to be zero. The same applies for item i with b_i and q_i .

For details, see equation (5) from [\[KBV09\]](#). See also [\[RRSK10\]](#), section 5.3.1.

To estimate all the unknown, we minimize the following regularized squared error

$$\sum_{r_{ui} \in R_{train}} (r_{ui} - \hat{r}_{ui})^2 + \lambda (b_i^2 + b_u^2 + \|q_i\|^2 + \|p_u\|^2)$$

Getting started, example

Here is a simple example showing how you can (down)load a dataset, split it for 5-fold cross-validation, and compute the MAE and RMSE of the SVD algorithm.

```
from surprise import SVD
from surprise import Dataset
from surprise.model_selection import cross_validate

# Load the movielens-100k dataset (download it if needed).
data = Dataset.load_builtin('ml-100k')

# Use the famous SVD algorithm.
algo = SVD()

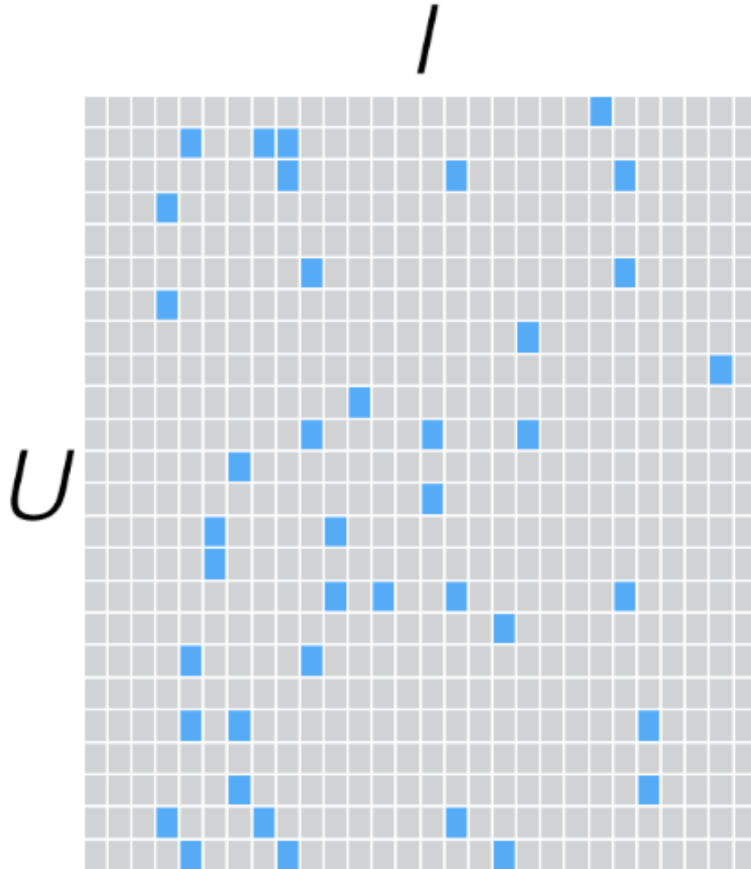
# Run 5-fold cross-validation and print results.
cross_validate(algo, data, measures=['RMSE', 'MAE'], cv=5, verbose=True)
```

`pip install surprise`

https://surprise.readthedocs.io/en/stable/matrix_factorization.html#unbiased-note

Model-based CF

$$\min_{P,Q} \sum_{r_{u,i} \in R} (r_{ui} - q_i^T p_u)^2 + \lambda(\|q_i\|^2 + \|p_u\|^2)$$



bias: $\hat{r}_{u,i} = \mu + b_i + b_u + q_i^T p_u$

svd++:

$$\hat{r}_{u,i} = \mu + b_i + b_u + q_i^T (p_u + |I_u|^{-1/2} \sum_{j \in I_u} x_j)$$

```
class surprise.prediction_algorithms.matrix_factorization.SVDpp
```

Bases: `surprise.prediction_algorithms.algo_base.AlgoBase`

The SVD++ algorithm, an extension of `svd` taking into account implicit ratings.

The prediction \hat{r}_{ui} is set as:

$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T \left(p_u + |I_u|^{-\frac{1}{2}} \sum_{j \in I_u} y_j \right)$$

Where the y_j terms are a new set of item factors that capture implicit ratings. Here, an implicit rating describes the fact that a user u rated an item j , regardless of the rating value.

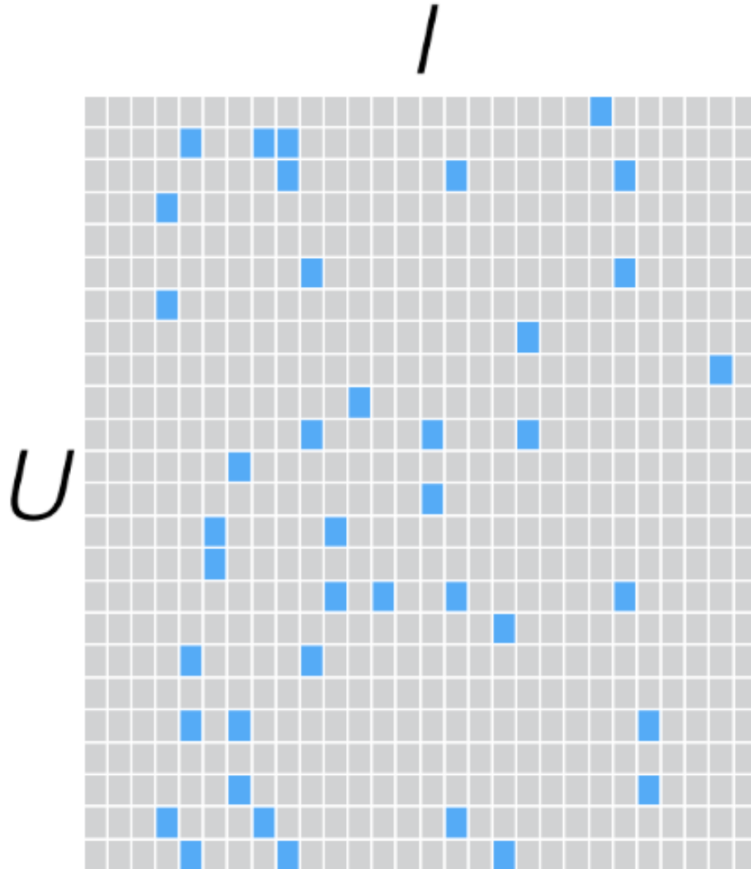
If user u is unknown, then the bias b_u and the factors p_u are assumed to be zero. The same applies for item i with b_i , q_i and y_i .

Model-based CF

$$\min_{P,Q} \sum_{r_{u,i} \in R} (r_{ui} - q_i^T p_u)^2 + \lambda(\|q_i\|^2 + \|p_u\|^2)$$

temporal dynamics:

$$\hat{r}_{u,i}(t) = \mu + b_i(t) + b_u(t) + q_i^T p_u(t)$$



$$\hat{r}_{ui} = \mu + b_i + b_u + q_i^T [p_u + |N(u)|^{-0.5} \sum_{i \in N(u)} x_i + \sum_{a \in A(u)} y_a]$$

$$\min_{p^*, q^*, b^*} \sum_{(u,i) \in K} c_{ui} (r_{ui} - \mu - b_u - b_i - p_u^T q_i)^2 + \lambda (\|p_u\|^2 + \|q_i\|^2 + b_u^2 + b_i^2)$$

[Koren et al., 2009] Koren, Y. , Bell, R. , & Volinsky, C. . (2009). Matrix factorization techniques for recommender systems. *Computer*, 42(8), 30-37.

目录

CONTENT

1. 什么是CF
2. memory-based CF
3. model-based CF
4. 小结

CSL
Career
Science
Lab

小结

1. CF是一种信息过滤的方法：充分利用其他人（显式/隐式）反馈(行为) 信息

collaborative filtering A means of filtering items for one user of a system based on the implicit or explicit rating of items by other users of that system.

2. CF包括memory-based CF和model-based CF

memory-based CF: kNN -> user-based CF, item-based CF

model-based CF: matrix factorization (SVD) -> Latent factor model

3. CF依赖于群体智慧，利用群体的行为信息

4. CF -> MF(matrix factorization) = FM(factorization machine) -> DL (deep learning)

5. “Are We Really Making Much Progress? A Worrying Analysis of Recent Neural Recommendation Approaches” 发表于RecSys 2019, Best Paper

谢谢!

CSL
Career
Science
Lab