香港科技大学(广州)
THE HONG KONG
UNIVERSITY OF SCIENCE AND
TECHNOLOGY (GUANGZHOU)

Recommender System

Xing Daiyan
50015641
dxing004@connect.hkust-gz.edu.cn

3 notebooks | dataset


```python
ratings = pd.read_csv("data/ratings.csv")
ratings.head()
```

	userId	movieId	rating	timestamp
0	1	1	4.0	964982703
1	1	3	4.0	964981247
2	1	6	4.0	964982224
3	1	47	5.0	964983815
4	1	50	5.0	964982931

```python
movies = pd.read_csv("data/movies.csv")
movies.head()
```

	movieId	title	genres
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
1	2	Jumanji (1995)	Adventure|Children|Fantasy
2	3	Grumpier Old Men (1995)	Comedy|Romance
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance
4	5	Father of the Bride Part II (1995)	Comedy

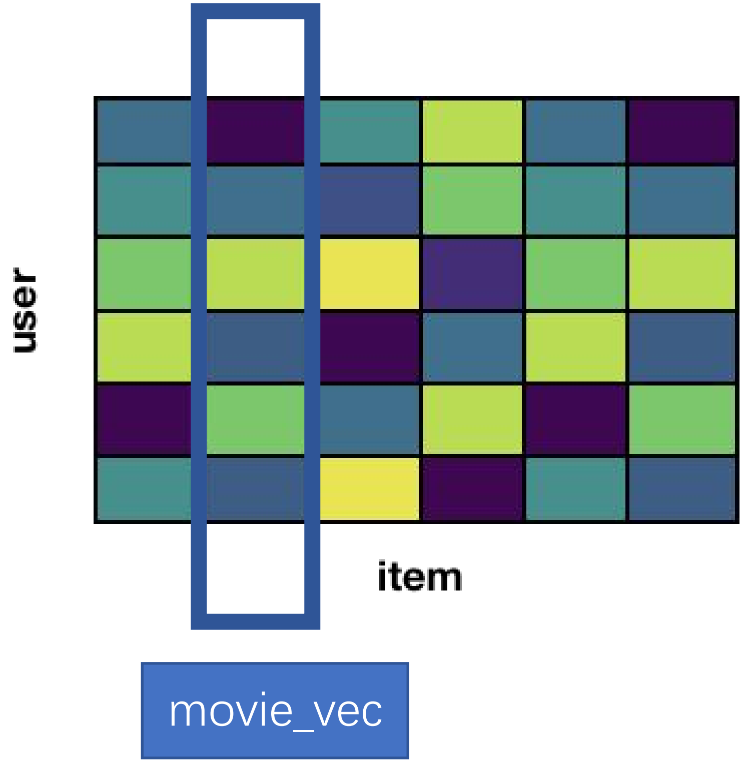
```python
n_ratings = len(ratings)
n_movies = ratings['movieId'].nunique()
n_users = ratings['userId'].nunique()

print(f"Number of ratings: {n_ratings}")
print(f"Number of unique movieId's: {n_movies}")
print(f"Number of unique users: {n_users}")
print(f"Average number of ratings per user: {round(n_ratings/n_users, 2)}")
print(f"Average number of ratings per movie: {round(n_ratings/n_movies, 2)}")
```

```
Number of ratings: 100836
Number of unique movieId's: 9724
Number of unique users: 610
Average number of ratings per user: 165.3
Average number of ratings per movie: 10.37
```

3 notebooks | item-item recommendation using collaborative filtering

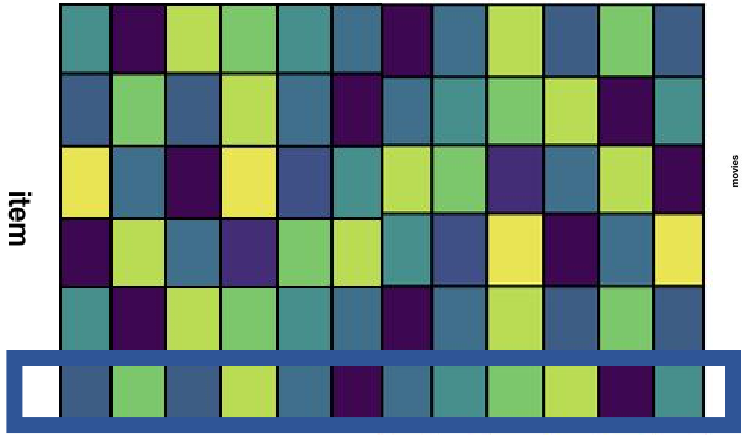
- Data
  - ratings, movies
- Create X
  - user-item matrix
  - (m_{users}, n_{movies})
  - sparsity (1.7%)
- Find similar movies
  - use kNN
  - movie_vec = X[movie_ind]
  - given a movie id, find a list of k movies that are similar to the movieId of interest



Collaborative filtering relies solely on user-item interactions within the utility matrix.
The issue with this approach is that brand new users or items with no interactions get excluded from the recommendation system.

3 notebooks | item-item recommendation using content-based filtering

- Data
  - movies
    - especially movies['genres'], movie_decades
- Create X
  - movie features cosine similarity matrix
  - (n_{movies}, n_{movies})
  - values between 0 and 1
- Find similar movies
  - base on the sim_scores from cosine_sim

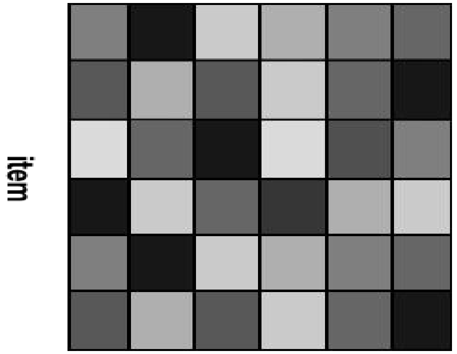


genres

movies	Comedy	Drama	Thriller	Romance	Family	Horror
Movie A	0	1	0	1	0	0
Movie B	1	0	0	1	1	0
Movie C	0	1	0	0	0	1
Movie D	0	0	1	1	0	1
Movie E	0	0	0	0	1	0
Movie F	1	0	0	1	0	0

```
movie_features = pd.concat([movies[genres], movie_decades], axis=1)
```

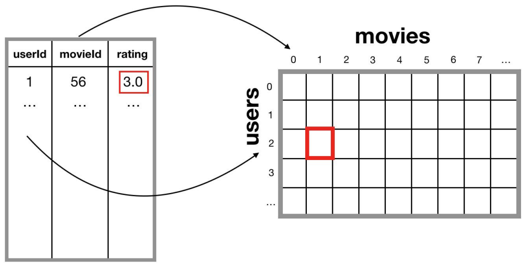
```
cosine_sim = cosine_similarity(movie_features, movie_features)
```



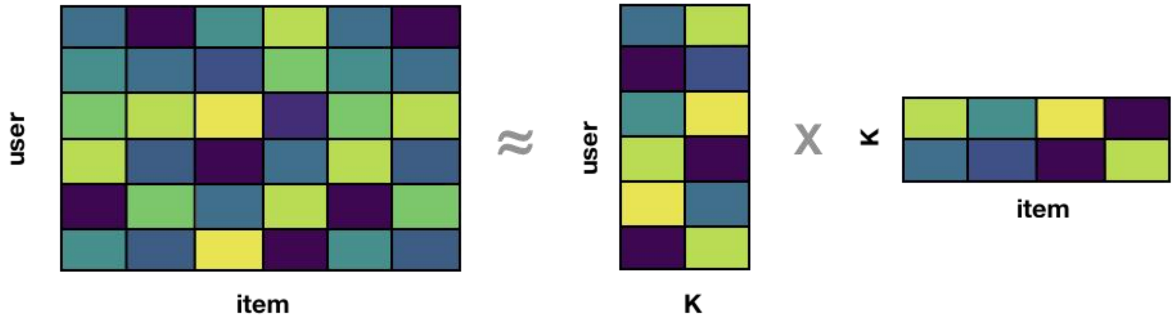
Handling the Cold Start Problem
Generating recommendations based on user and item features

3 notebooks | implicit feedback recommender system using the implicit package

- Data
  - ratings, movies
    - treat movie ratings as the number of times that a user watched a movie



- Create X
  - Alternating Least Squares (ALS)
  - user-factor matrix
    - (n_users, k)
  - item-factor matrix
    - (k, n_items)



- Find similar movies
  - implicit.als.AlternatingLeastSquares
  - given a movie id, find a list of k movies that are similar to the movieId of interest
  - given a use id, find a list of k movies that are similar to the movieId of interest (why id=95 has an error)

Y. Hu, Y. Koren and C. Volinsky, "Collaborative Filtering for Implicit Feedback Datasets," 2008 Eighth IEEE International Conference on Data Mining, Pisa, Italy, 2008, pp. 263-272, doi: 10.1109/ICDM.2008.22.

- Rahul Mazumder et al. proposed a matrix completion algorithm based on Alternating Least Squares (ALS) that efficiently solves low-rank matrix completion and singular value decomposition problems.
- For the movie rating matrix R, assuming there are m users and n items, and we want to discover k latent factors, our task is to find the user matrix P (m x k) and the item matrix Q (k x n). The objective function is:

$$L(P,Q) = \sum_{(u,i)\in R_0} (R_{ui} - P_u^\top \cdot Q_i)^2 + \lambda \sum_u ||P_u||^2 + \lambda \sum_i ||Q_i||^2$$

- The steps of alternating ridge regression are as follows:
  - Initialize Q with an initial value Q_0 , which can be randomly generated or the global average.
  - Fix the current value of Q_0 and solve for P_0
  - Fix the current value of P_0 and solve for Q_1
  - Fix the current value of Q_1 and solve for P_1
  - Repeat steps 3 and 4 until convergence
  - Continue the iterations until the loss function value L converges
- For example, let's consider fixing Q and solving for P. Minimizing the objective function ($\min_{P,Q} L$) is equivalent to: $\min_P \left[\sum_{(u,i)\in R_0} (R_{ui} - P_u^\top \cdot Q_i)^2 + \lambda \sum_u ||P_u||^2 \right]$
- This equation can be further simplified as: $\sum_u \min_P \left[\sum_i (R_{ui} - P_u^\top \cdot Q_i)^2 + \lambda ||P_u||^2 \right]$

香港科技大学(广州)
THE HONG KONG
UNIVERSITY OF SCIENCE AND
TECHNOLOGY (GUANGZHOU)

- Let's define $L_u(P_u) = \left[\sum_i (R_{ui} - P_u^\top \cdot Q_i)^2 + \lambda ||P_u||^2\right]$
- Our goal is to find the user feature vector P_u that minimizes $L_u(P_u)$.
- Taking the derivative of L_u with respect to P_u, we have:

$$\frac{\partial L_u}{\partial P_u} = \frac{\partial\left[\sum_i (R_{ui} - P_u^\top \cdot Q_i)^2 + \lambda ||P_u||^2\right]}{\partial P_u} = \sum_i 2\,(P_u^\top \cdot Q_i - R_{ui})Q_i + 2\lambda P_u = 2\left(\sum_i P_u^\top Q_i Q_i - \sum_i R_{ui} Q_i + \lambda P_u\right)$$

- Setting the above equation to zero, we get: $\sum_i P_u^\top Q_i Q_i - \sum_i R_{ui} Q_i + \lambda P_u = 0$
- We can rewrite it as: $\left(\sum_i Q_i Q_i^\top + \lambda I\right) P_u = \sum_i R_{ui} Q_i$
- Therefore, we can obtain: $P_u = (QQ^\top + \lambda I)^{-1} Q R_u$
- Similarly, when fixing P and solving for Q, we have: $Q_i = (PP^\top + \lambda I)^{-1} P R_i$

code： https://blog.csdn.net/alionsss/article/details/104302010

Mazumder, Rahul, Trevor Hastie, and Robert Tibshirani. "Matrix completion and low-rank SVD via fast alternating least squares." Journal of Machine Learning Re-search 11.Jan (2010): 19-60.

- The objective function is defined as: $L(P,Q)=\sum_{(u,i)\in R_0}(R_{ui}-P_u^\top\cdot Q_i)^2+\lambda\sum_u\|P_u\|^2+\lambda\sum_i\|Q_i\|^2$

- Similarly, let's take the derivative with respect to each P_u:

$$\frac{\partial L_u}{\partial P_u}=\frac{\partial\left[\sum_i(R_{ui}-P_u^\top\cdot Q_i)^2+\lambda\|P_u\|^2\right]}{\partial P_u}=\sum_i 2\,(P_u^\top\cdot Q_i-R_{ui})Q_i+2\lambda P_u$$

- The gradient descent algorithm can be written as:

$$P_{u+1}=P_u-\alpha\cdot\frac{\partial L_u}{\partial P_u}=P_u-\alpha\left[\sum_i 2\,(P_u^\top\cdot Q_i-R_{ui})Q_i+2\lambda P_u\right]$$

- Similarly, for the iteration of the item matrix, the algorithm can be written as:

$$Q_{i+1}=Q_i-\alpha\cdot\frac{\partial L_u}{\partial Q_i}=Q_i-\alpha\left[\sum_u 2\,(P_u^\top\cdot Q_i-R_{ui})P_u+2\lambda Q_i\right]$$

code：https://blog.csdn.net/alionsss/article/details/104302010

- Convex relaxation techniques are employed to offer a sequence of regularized low-rank solutions for large-scale matrix completion problems. The algorithm described in the paper, SOFT-IMPUTE, iteratively replaces missing elements with values obtained from a soft-thresholded singular value decomposition (SVD). By initializing with a warm start, the algorithm can efficiently compute a complete regularization path of solutions across various values of the regularization parameter.

Algorithm 1 SOFT-IMPUTE

1. Initialize $Z^{\text{old}} = 0$.
2. Do for $\lambda_1 > \lambda_2 > \ldots > \lambda_K$:
   (a) Repeat:
      i. Compute $Z^{\text{new}} \leftarrow \mathbf{S}_{\lambda_k}(P_\Omega(X) + P_\Omega^\perp(Z^{\text{old}}))$.
      ii. If $\frac{\|Z^{\text{new}} - Z^{\text{old}}\|_F^2}{\|Z^{\text{old}}\|_F^2} < \varepsilon$ exit.
      iii. Assign $Z^{\text{old}} \leftarrow Z^{\text{new}}$.
   (b) Assign $\hat{Z}_{\lambda_k} \leftarrow Z^{\text{new}}$.
3. Output the sequence of solutions $\hat{Z}_{\lambda_1}, \ldots, \hat{Z}_{\lambda_K}$.

Mazumder R, Hastie T, Tibshirani R. Spectral Regularization Algorithms for Learning Large Incomplete Matrices. J Mach Learn Res. 2010 Mar 1;11:2287-2322. PMID: 21552465; PMCID: PMC3087301.

香港科技大学(广州)
THE HONG KONG
UNIVERSITY OF SCIENCE AND
TECHNOLOGY (GUANGZHOU)

Thank you!

Xing Daiyan
50015641
dxing004@connect.hkust-gz.edu.cn