Recommender System (Spring 2022)

Homework #1 (100 Pts, March 9)

Student ID ______________________

Name ______________________

(1) [20 pts] We are given the following user-item rating matrix for five users and six items. Assume that a peer group of size at most 2 is used in each case, and negative correlations are filtered out.

	I1	I2	I3	I4	I5	I6
U1	5	6	7	4	3	?
U2	4	?	3	?	5	4
U3	?	3	4	1	1	?
U4	7	4	3	6	?	4
U5	1	?	3	2	2	5

(a) [10 pts] Predict the values of unspecified ratings of user 2 using the user-based collaborative filtering algorithm. Use adjusted cosine similarity with mean-centering.

Answer:

(b) [10 pts] Predict the values of unspecified ratings of user 2 using the item-based collaborative filtering algorithm. Use adjusted cosine similarity with mean-centering.

Answer:

(2) [20 pts] We are given an $m \times n$ rating matrix, where m is the number of users and n is the number of items.

(a) [5 pts] When building the similarity matrix, calculate the worst-case time complexity for item-based and user-based collaborative filtering algorithms.

Answer:

(b) [5 pts] When making a rating prediction for a target user, calculate the worst-case time complexity for item-based and user-based collaborative filtering algorithms.

Answer:

(c) [10 pts] Assuming m=10,000 and n=1,000,000, which of the two algorithms, user-based and item-based collaborative filtering, is better? Explain why.

Answer:

(3) We provide template code and dataset in Python. Refer to 'models/userKNN_explicit.py,' write your code to implement the item-based collaborative filtering algorithm on 'models/itemKNN_explicit.py.' Run '1_main.py' to run the source code.

(a) [20 pts] Implement the function "fit" in 'models/itemKNN_explicit.py' using the Adjusted cosine similarity. It is defined as follows:

$$sim(i,j) = \frac{\sum_{u \in \mathcal{S}_{ij}} (r_{ui} - \bar{r}_{u*})(r_{uj} - \bar{r}_{u*})}{\sqrt{\sum_{u \in \mathcal{S}_{ij}} (r_{ui} - \bar{r}_{u*})^2} \sqrt{\sum_{u \in \mathcal{S}_{ij}} (r_{uj} - \bar{r}_{u*})^2}}$$

Note: Fill in your code here. You also have to submit your code to i-campus.

Answer:

(b) [20 pts] Implement the function "predict" in 'models/itemKNN_explicit.py. (1) Find the items user rated, (2) sort by similarity, (3) get Top-K Neighbors using self.top_k, and (4) predict using neighbors. Note that user_mean is subtracted at __init__(), so you should consider it at predict().

Note: Fill in your code here. You also have to submit your code to i-campus.

Answer:

(c) [20 pts] Given the data ('naver_movie_dataset_small.csv' and 'movielens_100k.csv'), draw the plots of RMSE by adjusting k (i.e., the number of nearest neighbors) for UserKNN and ItemKNN, respectively. (i) For varying k, explain the results and how much k affects RMSE. (ii) evaluate whether UserKNN is better/worse than ItemKNN. Run '2_search.py' to run the source code.

Note: Please show the results for two datasets in the code.

Note: Show your plots and explanations in short (3-5) lines.

Answer: