Recommender Systems --- Assignment #1

A)

this assignment is about implementing the user base and item base collaborative filtering. In this project we use Pandas library to work with data file.

At the beginning of the code we read the data and printing the first row. The def user_based_sim(rates_by_a, rates_by_b) function is for finding the similar users as the first step .

*Worth mentioning that we pass the rates_by_a ,and rates_by_b as the parameters instead of userId a and b to improving the performance.In this function, first we find the joint movies which rate by both user A and B with this line:

- similar_rated_movies = [x for x in list(rates_by_a.movieId) if x in list(rates_by_b.movieId)]

*we assume that if there is no joint movies we return 0.

then by filtering this list we can find the rate_of_a and rate_of_b

and we calculate the formula in the lecture: sim(a, b)

in the next step we calculate the prediction pred(a,p) with def user_based_pred(rates_by_a, p) function .After these step we randomly select a user and we find the similar users for it through this line:

' •   similarity': user_based_sim(selected_user_rates, rates_by_userId)

then we sort the sort the result of this function in order to find 10_top similar user, following this by 'rate': user_based_pred(selected_user_rates, movieId) we obtain recommended movies for random user and same as the pervious step by using sort method 10_top recommended movies can be find.

B)

in this section we want to find similarity and then prediction according to item_based_approach. Most of the steps are like the user_based_approach. In fact we apply users_rated_a and users_rated_b for finding all rating in which users rate to movie (a) and users for movie (b) and user_rated_both is for shared choice to calculate similarity. Worth mentioning, this code: if len(user_rated_both) == 0: is for the times that there is no same choice.

Finally, we use similarity and prediction function according to the formula in the lecture. With these codes: def item_based_sim(a, b) and def item_based_pred(rated_by_u, p).

in the final step we fine 10-top recommended movies for random user via the same method as pervious section by these codes :

for movieId in data.movieId.unique:()

recommended.append})

'   movieId': movieId,

'   rate': item_based_pred(selected_user_rates, movieId)

({

recommended = sorted(recommended, key=lambda i: i['rate'], reverse=True) # sort