Course Number: COEN 272

Name: Dengwei Wang

SCUID: 00001599188

Assignment Number: Project#2

1. MAE-Table for different algorithm

	Test5	Test10	Test20	Overall
Cosine-Similarity	0.8277	0.7793	0.7633	0.7884
Pearson-Correlation	0.8382	0.7872	0.7840	0.8026
Pearson-Correlation with IUF	0.8664	0.8217	0.8250	0.8378
Pearson-Correlation with Case Amplification	0.8844	0.8717	0.8879	0.8828
Item-based Collaborative Filtering	0.9792	0.8903	0.8307	0.8941
Own Algorithm	0.8071	0.7515	0.7374	0.7637

2. Discussion

a) Overall, the accuracy increased as the number of ratings given for a single user, that is because given more information about one user, we can get more accurate similarity and find more similar users to predict the rating.

b) The algorithm that has the best performance(excluding my own algorithm) is the basic Cosine-Similarity algorithm, the second best one is the basic Pearson-Correlation algorithm.

c) I think the result is reasonable because the number of training data is relatively small with only 200 users and 1000 movies. A simpler algorithm can thus provide a better result, on the other hand, the database for recommendation systems in the real world is much bigger and simpler algorithms would not satisfy the accuracy.

d) The accuracy of Pearson-Correlation with IUF is slightly worse than the basic Pearson-Correlation algorithm. I think the reason is that IUF might cause the modified rating to exceed 5 or less than 1 when there are too few or too many ratings for one particular movie, thus the modified weight could be worse than the original one.

e) The performance of the item-based Collaborative Filtering algorithm is worse than the basic Cosine-Similarity algorithm and Pearson-Correlation algorithm. The reason should be that the number of ratings is only 1 to 5 so that it is harder to find similar items than finding similar users, so the performance of item-based algorithms is worse.

3. Own algorithm

Since the result of the Cosine-Similarity algorithm and Pearson-Correlation algorithm, my own algorithm would be focusing on optimizing the two algorithms and using the weighted results of the two algorithms to get the final result.

The following are some details about my own algorithm.

(a) When calculating the cosine similarity between two users, there might be only one common movie they both rated. In this case, the cos_sim result based on the formula would always be one which is obviously wrong when one rate 1 while the other rate 5, I choose to place their value with the difference between the rating. For example, one rate 1 and the other rate 5 yield the result 0 for no similarity, 1 and 3 get the cos_sim with 0.5.

(b) As for calculating Pearson-Correlation weight between two users, when there is only one common movie, I would put the weight among -1, -0.5, 0, 0.5, 1 based on the difference of the two ratings. When there are no similar users for one prediction, the default rating for this user is set to the mean value of all other movies the user rated.

(c) When choosing the number of k neighbors for calculating the final rating, I choose k = 10, 30, 50 for testing and k = 30 has the best result. So in my own algorithm, I use most 30 neighbors for calculating ratings based on cosine similarity and Pearson-Correlation.

(d) When choosing most similar neighbors for calculating ratings in Cosine-Similarity algorithm and Pearson-Correlation algorithm, only the users with cos_similarity larger than 0.3 or abs(pearson_weight) larger than 0.3 would be used for calculation. This handling avoids poor similar users' ratings being used when there are not enough similar neighbors.

(e) The result of Cosine-Similarity is slightly better than Pearson-Correlation, in my own algorithm, I tried to calculate the final result with the weight of (0.5cos+0.5pearson), (0.6cos+0.4pearson) and (0.7cos+0.3pearson). The combination with (0.6cos+0.4pearson) has the best result.