Hybrid Recommender System

Yaqi Zhou; Yanan Huo; Li Luo; Kejia Shi

March 2016

1. Abstract

With the development of Internet, the new era of big data is approaching and personalized recommendation services has become a new fashion. We mainly use R to explore and develop the algorithm with package *Recommenderlab*, where the collaborative filtering algorithms have been developed maturely, to some extent. The two main parts in this report are exploring the R package to predict the ratings based on the 100 thousand MovieLense data set and combine the information from both users and items to develop a new algorithm to solve the cold start problem. Furthermore, we also managed to connect to the Center for High Throughput Computing to realize the algorithm on the 20 million MovieLense data set.

2. Plan

The first thing we did is to explore the 100 thousand MovieLense data set from website, comparing it with the built-in data set in the *Recommenderlab* package and to make sure that the codes in R can be generally utilized into both data sets. The data set from website with several files are not completely accurate or concise as the built-in data set, where there are some overlapped documented ratings for the same user and the same item and even some movies have two item ID numbers. To prevent the similar situation happened to the 20 million data set, we developed our own R code to tidy up the data set. Meanwhile, data visualization is a necessary process to express the data structure in a macroscopical respective, which are shown in the next part.

Secondly, we looked through several papers about the recommender system to learn the algorithm inside the package and even a book by S.K. Gorakala and Michele Usuelli, 2015. Following these papers and book, we learned the mechanism of the recommendation models and evaluated the predict result. We have plotted out the ROC curves and compared them based on one-user prediction.

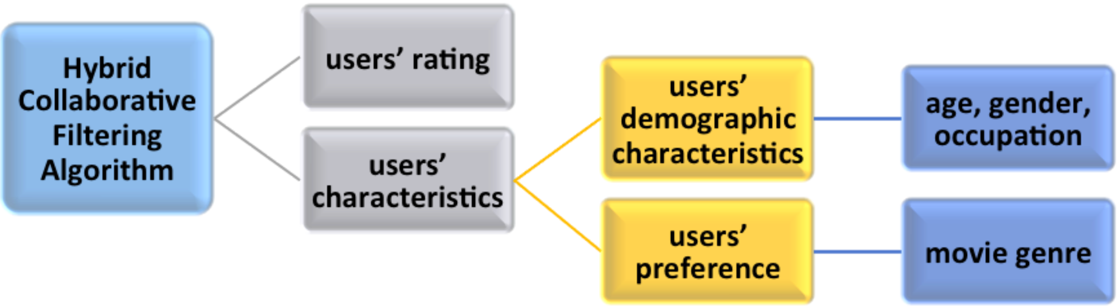
Moreover, we recognized some limitations of collaborative filtering algorithms, such as the data sparsity problem, especially the cold start challenge, which means that recommendations for new items or new users cannot be provided because no historic records are found. The algorithms we used in the Recommenderlab package only take the rating matrix into account; however, not only users' ratings for items, but also users' own features, such as users' demographic characteristics and users' preference for some properties of items, contribute to the similarities between users.

Therefore, we intended to combine other information to improve our previous recommend system and fix the cold start problem.

Here we proposed a hybrid collaborative filtering algorithm based on users' ratings and user's features. Specifically, the users' features could be further divided into users' demographic characteristics, like age, gender and occupation, and user's preferences for movie genres. If we would like to recommend for an old user, we calculated weighted sum of the similarities between this user and any other user based on user's demographic characteristics, user's preferences, as well as user's ratings. Then 60 neighbors were selected and ratings for movies that had not been watched were forecasted. On the other hand, we created an information database of demographic characteristics and preferences for movie genres so that we could tell a new user's movie genre preference given his or her demographic features. Hence, similarities between the new user and every old user could be computed and recommendations for new users are no longer impossible.

3. Hybrid Collaborative Filtering Algorithm

The flow chart for our hybrid collaborative filtering algorithm is given as follows:



There exist two situations when we predict ratings of items for our target user.

1) Predictions for old users

If the target user has already rated some movies, then we can check what kinds of movies he or she rated more frequently and higher. Thus, we have information about his or her preference of movie genres. The overall similarity between two old users a, b are calculated according to the following formula:

$$sim_{overall.old}(a,b) = \alpha sim_{demographic}(a,b) + \beta sim_{preference}(a,b) + (1-\alpha-\beta) sim_{rating}(a,b) \qquad (1)$$

, where $sim_{overall.old}(a,b)$, $sim_{demographic}(a,b)$, $sim_{preference}(a,b)$ and $sim_{rating}(a,b)$ refer to the overall similarity, similarity based on users' demographic characteristics, similarity based on users' preference for movie genres and similarity based on ratings in the past between user a and user b, respectively. It's worth mentioning that similarity based on ratings in the past between user a and user b can be produced using the Recommenderlab package. Besides, $0 < \alpha, \beta < 1$ and $\alpha + \beta < 1$. α, β are weighting parameters whose values will be chosen according to mean absolute error (MAE) during the experiment. The mean absolute error is a quantity used to measure how close forecasts or predictions are to the eventual outcomes, which is defined as

$$MAE_a = \frac{1}{n}\sum_{h=1}^{n}|S_{a,h} - y_{a,h}| \tag{2}$$

, where $S_{a,h}$ is the rating prediction of the h^{th} item for user a, while $y_{a,h}$ is the corresponding true rating. And n is the number of movies that user a has rated. The less the MAE is, the more accurate the prediction is. We use MAE as our standard because it has the same scale as our rating value, which gives us more intuitive feeling about the deviations of predictions from the true values.

After calculating the overall similarities between our target user and any other user in the dataset, it is easy to pick nearest neighbors for the target user. In this study, we chose 60 neighbors. The rating prediction, $S_{a,h}$, of the h^{th} movie for our target user a is thus calculated as belows:

$$S_{a,h} = \overline{P}_a + \frac{\sum_{b \in kNB} sim_{overall.old}(a,b)(P_{b,h} - \overline{P}_b)}{\sum_{b \in kNB} sim_{overall.old}(a,b)} \tag{3}$$

, where $\overline{P}_a$ is the average rating for movies that user a has rated; similarly, $\overline{P}_b$ is the average rating for movies that user b has rated. And user b belongs to the k nearest neighbors, kNB, of user a. Here, k=60.

Obviously, we brought in the average rating for user a instead of using weighted average rating for user a's k nearest neighbors. The reason for this is that different users have different rating habits and rating scales, which can be reflected in the $\overline{P}_a$.

2) Predictions for new users

For new users, we are lack of information about previous ratings. Therefore, only users' demographic characteristics and users' preferences for movie genres are available. What is more, new users' preference for movie genres is forecasted based on his or her demographic features. In other words, we can divide old users into various groups according to some demographic feature and summarise the movie genre preferences for these groups into a lookup table; we can predict what types of movies new users are more likely to watch on the basis of those lookup tables. And then, the similarity between the new user a and any old user b are calculated as belows:

$$sim_{overall.new}(a,b) = |\frac{C_a \cap C_b}{C_a \cup C_b}| \tag{4}$$

, where C_a represents user a's all characteristics, including demographic characteristics and preference for movie genres; C_b represents user b's all characteristics. $C_a \cap C_b$ refers to the characteristics in common for user a and user b. While $C_a \cup C_b$ refers to the all characteristics that either user a or user b has.

The similarity measure above is called Jaccard coefficient, which measures similarity between two finite unordered sample sets. Without users' ratings, it is reasonable to use Jaccard coefficient to calculate similarity in this scenario.

Because new users do not have previous rating records, they do not have average rating for movies that they have rated, $\overline{P}_a$. The rating prediction, $S_{a,h}$, of the h^{th} movie for our target user a is thus calculated using following formula:

$$S_{a,h} = \frac{\sum_{b \in kNB} sim_{overal.new}(a,b) P_{b,h}}{\sum_{b \in kNB} sim_{overal.new}(a,b)} \qquad (5)$$

, where the notations have same meanings as they do in Formula (3) and (4).

4. Result

4.1 Collaborative Filtering Algorithm Using Recommenderlab in R

1) Data Exploring

In the beginning, we developed our own code to tidy up the 100 thousand data set and explore the nature of it. We have 943 users and 1664 movies the data set, with 99392 ratings. Firstly, similarity between users and between items is the main measurements that collaborative filtering algorithms based on, where it is calculated relied on the similarity of rating. To illustrate, we display the first hundred users and first hundred items to lay out the similarity within users and items.

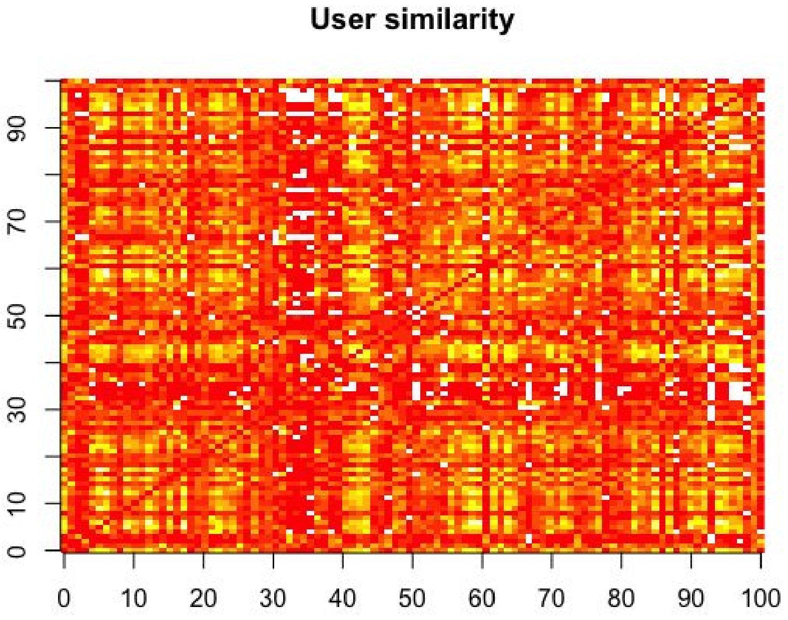


Figure 1.1.1 The similarity among the first-100th users. The deeper the color, the more relevant the two users based on their ratings.

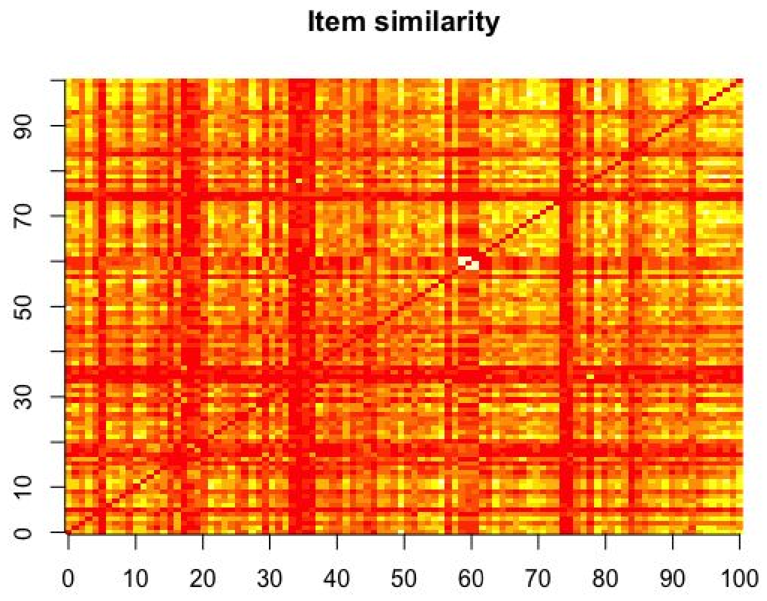


Figure 1.1.2 The similarity among first-100th movies

In the two similarity figures, we can see that both of them are symmetric over the red diagonal line, which is obvious that the similarity between the *i*th user and the *j*th user is the same with that between the *j*th and the *i*th. As expected, there are blank areas in the similarity matrix, which means that there is no overlap on the movies that the two users rated or the two movies have entirely different ratings.

Since there are plenty of unrated items, the rating matrix must be a sparse matrix, with few rating values. We got rid of the large amount of missing values and picked the 99392 ratings out to explore the details. We drew a histogram of the ratings without missing data and it tells that most ratings are larger than 2 and 4 is the most common rating.

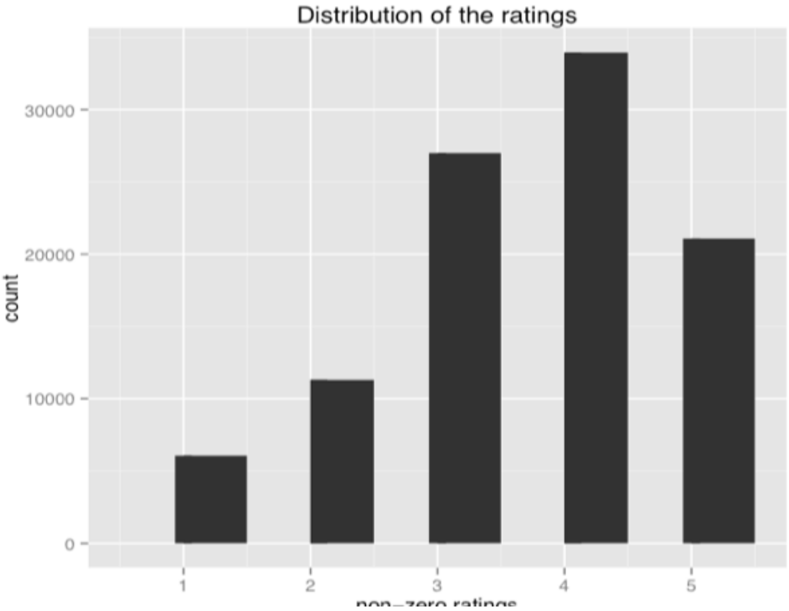


Figure 1.1.3 Histogram of the ratings without missing values

Merely knowing the rough distribution of ratings are far from knowing well the nature of this data set. From the *Recommenderlab* package, we can learn quickly the number of non-missing values for each column and also the average value within columns. Then, we can sort the Movies by the number of rated, which can be interpreted as the popularity ranking, to some extent. The next plot shows that the number of views of the top-ten movies.

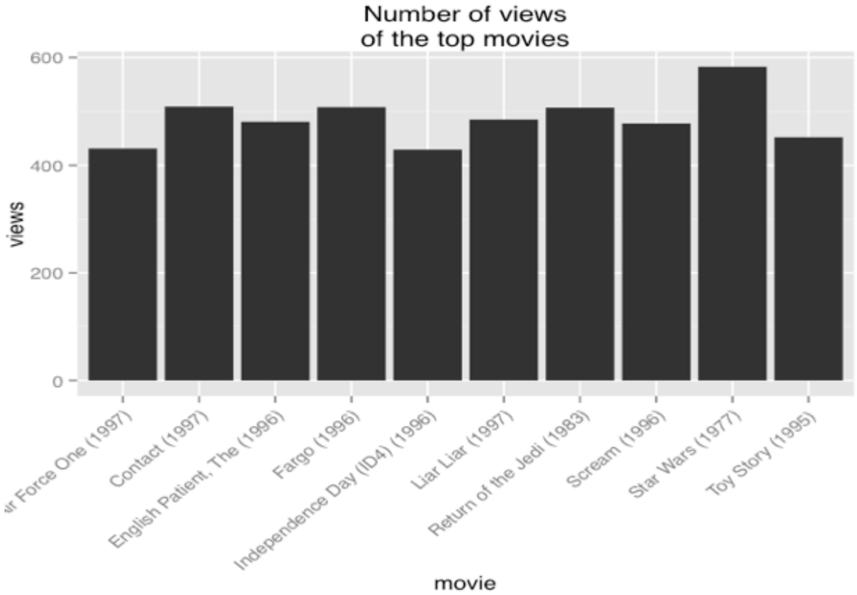
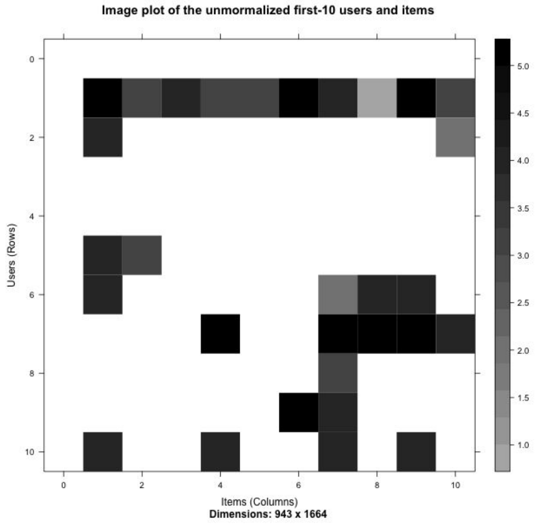


Figure 1.1.4 Number of views of the top-ten movies

Since that different users have their own judgment criteria, there may be bias within a user and sequentially, removing this effect by normalizing the data is the next step. To normalize the data, we expect the average of each user is 0. To see whether there is bias existing within a user, we plot out the heatmap before and after normalization to indicate the necessity of normalization. Of the following plot, we use the "z-score" method to normalize with subtracting the average and then divided by the standard deviation. The other method, "centering", only subtracting the average.



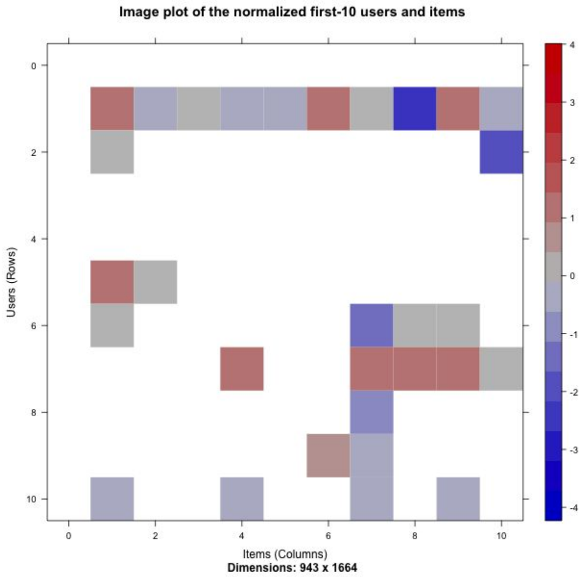


Figure 1.1.5 The heatmap of the first 10 users over the first 10 items before and after normalization.

Comparing the two figures above, the 10th user (row) could be the most typical user to explain the importance of normalization, where he gave all the 4.0 rating of the first four items, while after normalization, the 4.0 rating turns out to be -1.0, which is even below his average rating. The similar situation happens through the whole data set, thus we must normalize the data before applying models on it.

The data set from website also contains the information of demographic characteristics such as gender, age and occupation. To explore the effectiveness of demographic characteristics on ratings, we drew out several histograms to reveal their distribution.

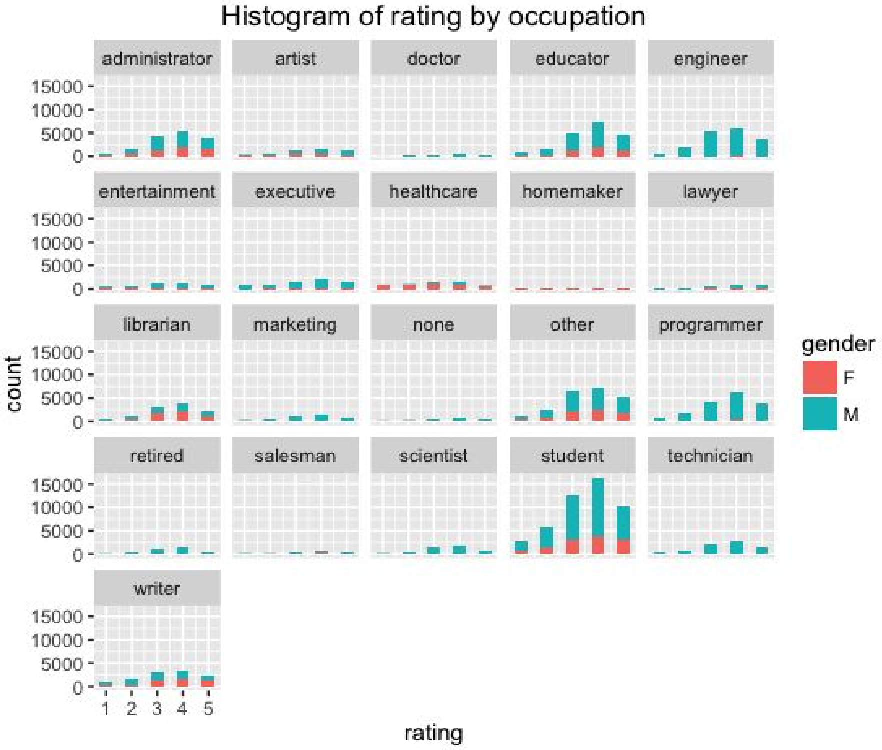


Figure 1.1.6

The figure above indicates the circumstance in the data set that male and female do relate to the type of occupation. For example, most of people doing jobs relevant to healthcare and homemaker are female and most of programmers and male. In general, male rate more than female, while both of male and female fate mostly rate 4 and 1 least. Moreover, we can see that the differences of the amounts of each rating do vary for each occupation. Thus, we may have reason to believe that only utilizing rating matrix losses information about the feature of users.

Also, we plot out other figures about other demographic characteristics over rating to explore the relationship between them, but we are not going to display them here, since figure 1.1.6 is a typical one.

2) Data preparation

To prepare the data, we follow the next two steps:

i. Select the relevant data;
ii. Normalize the data.

Since plenty users rated only a few movies and lots of movies have been rated only a few times, the situation leads to a sparse rating matrix, and, sequentially, much more bias. These missing value could trigger the less accuracy in predicting their rating. Thus, it is necessary to determine the minimum numbers of users per movie and vice versa. We should do several iterations to leave out the correct value of this particular number. We require that users should rated as least 50 movies and movies should be rated at least 100 times. Then the rating matrix we got here contains 560 users and 332 items with 55298 ratings, which decreases the rating matrix of MovieLens by more than half. After lowering the scale of the data set, we normalized it using “z-score” method.

3) Model Evaluating

Six models are built in the package, while we focus on these two, User-based Collaborative Filtering and Item-based Collaborative Filtering, with the comparing group, Popular and Random. We choose the first 905 users and all movies to be the training data and the 906th user to be the user that we predict and evaluate. With the particular training data set, we run Recommenderlab to give the recommender lists of the test user, with top-1, 3, 5, 10, 20, 50 and 200 lists. With *k-fold* method and setting *k* to be 4, we plot out the ROC curves as following. To evaluate the effectiveness of the four methods, we can compute the area under the curves, which shows apparently that the UBCF is the best one.

The Figure 1.3.1 shows that the ranking of the four methods is: UBCF, Popular, IBCF and Random. This ranking is not a coincidence, since it keeps the same after we randomly chose other 5 users to practice the process. It is normal to have the random item to be the last one among them, while we did not expect the IBCF performs worse than Popular item. We believe that Popular item gives the common preference for most people by gathering ratings from the entire 943 users, which can be treated as the most general answer for all users and take advantage of IBCF.

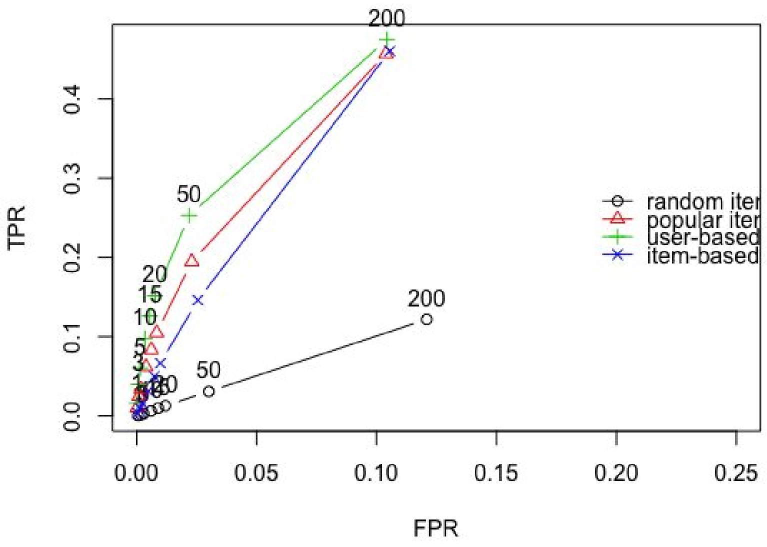


Figure 1.3.1 ROC curves of the four methods.

On the other hand, the running time of the IBCF lasts more than 50 times than the other methods last. With the performance in the ROC figure, we can conclude that the IBCF may not be the best choice within these four methods.

4.2 Hybrid Collaborative Filtering Algorithm

1) For old users

We used an exhaustive search approach to find the best weights, α, β, in Formula (1) so that we can have the minimum MAE for rating predictions. Because calculating MAE for all users can be overwhelming on a single machine, we randomly sampled a user, the one with user ID 917, and produced the following table:

Table 4.2.1 MAE Values with Different α, β

MAE	α=0.1	α=0.2	α=0.3	α=0.4	α=0.5	α=0.6	α=0.7	α=0.8
β=0.1	0.942	0.972	0.987	0.995	0.990	0.997	0.997	1.008
β=0.2	0.942	0.976	0.984	0.982	0.981	0.986	0.993	-
β=0.3	0.938	0.962	0.971	0.970	0.975	0.980	-	-
β=0.4	0.933	0.965	0.965	0.974	0.978	-	-	-
β=0.5	0.935	0.971	0.976	0.979	-	-	-	-
β=0.6	0.944	0.970	0.978	-	-	-	-	-
β=0.7	0.965	0.969	-	-	-	-	-	-
β=0.8	0.971	-	-	-	-	-	-	-

In order to find out the minimum MAE in Table 4.2.1, we computed the relative minimal values of MAE for each column and plot Figure 4.2.1 as belows:

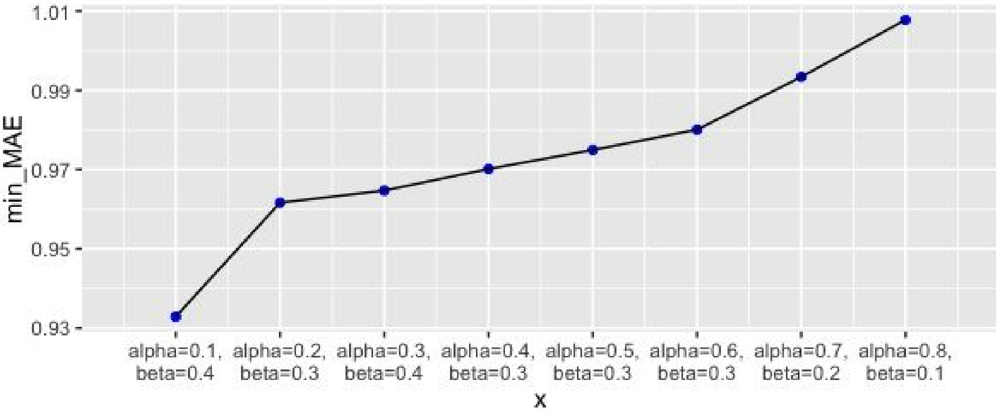


Figure 4.2.1 Line Segment Graph for Age Group Classification

When $\alpha = 0.1$, $\beta = 0.4$, MAE has the minimal value, that is, the algorithm is most accurate.

Thus, we used $\alpha = 0.1$, $\beta = 0.4$ to calculate the overall similarity for every two old users using Formula (1) and predicted ratings using Formula (3). The evaluation of this method is discussed in Section 4.3.

2) For new users

As we mentioned previously, it is important to create an information database, that is, lookup tables for demographic characteristics and preferences for movie genres based on old users. For example, we listed top 5 genres for males and females respectively in Table 4.2.2.

Table 4.2.2 Lookup Table for Gender and Preference for movie genres

Gender	Preference1	Preference2	Preference3	Preference4	Preference5
F	Drama	Comedy	Romance	Thriller	Adventure
M	Drama	Comedy	Action	Thriller	Romance

Thus, according to Table 4.2.2, we see drama and comedy are popular among both females and males. And men prefer action movies while women like romantic ones.

As for the age, we divided the life of people into 11 groups as in the Figure 4.2.2, based on the way United Union classifies age in relation to the minimum ages for enrollment in different education levels.

Lookup tables for age group and preference for movie genres, as well as occupation and preference for movie genres, are given in Appendix Table 1 and Table 2. Some patterns are found as well: 1) children are not likely to insist on watching movies for a long time and following the story clues of drama; instead they love drama more; 2) teenagers are fond of thriller and adventure movies because young people can bear exciting or shocking things; 3) romantic movies are popular among grown-ups; 4) compared to other occupations, the movie genres that lawyers prefer are thriller, action, war.

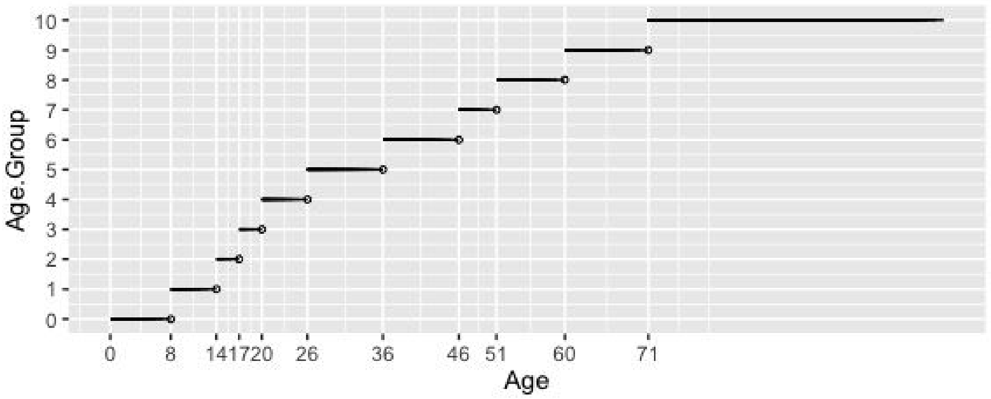


Figure 4.2.2 Line Segment Graph for Age Group Classification

For we did not have new users in reality, we randomly sampled 3 users to be the new users and removed their ratings but kept their demographic features as in Table 4.2.3.

Table 4.2.3 Table for New Users' Characteristics

user_id	age	gender	occupation	Preference 1	Preference 2	Preference 3	Preference 4	Preference 5
75	24	M	entertainment					
519	22	M	other					
611	46	M	librarian					

For every new user, we found out the intersection set of preference for different demographic characteristics from the lookup tables. We filled in the blanks in Table 4.2.3 and produced the Table 4.2.4.

Table 4.2.4 Table for New Users' Characteristics

user_id	age	gender	occupation	Preference 1	Preference 2	Preference 3	Preference 4	Preference 5
75	24	M	entertainment	Drama	Comedy	Romance	Action	Adventure
519	22	M	other	Drama	Comedy	Romance	Action	Adventure
611	46	M	librarian	Drama	Comedy	Romance	Thriller	Sci-Fi

Then, using Formula (4) and (5), we calculated similarities between new users and any old users and recommended according to the rating predictions.

4.3 Comparison between Algorithms

We used MAE as a criterion to evaluate our algorithm.

1) For old users

We randomly sampled 3 users to do predictions considering large amount of time if predicting for all users. We compared our results with the results we got using UBCF algorithm from Recommenderlab package.

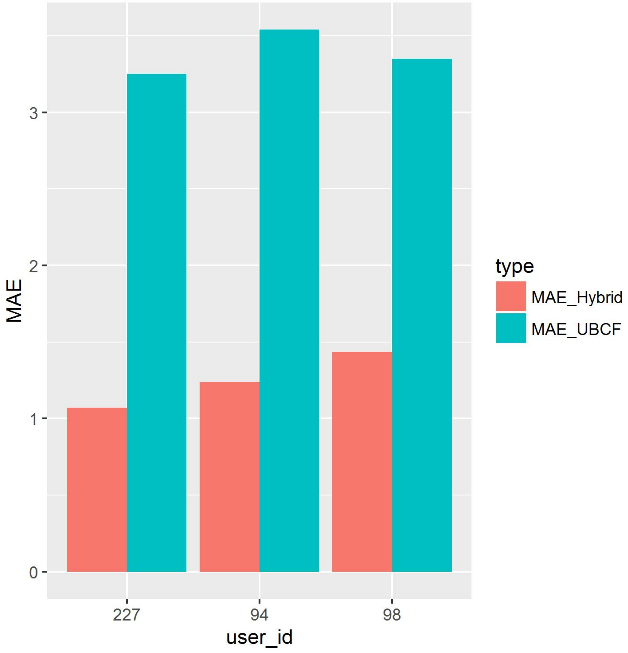


Figure 4.3.1 Predicting MAE for Old Users

From the Figure 4.3.1, we can see that for all of the 3 users, MAEs using the hybrid collaborative filtering algorithm are much smaller than those using traditional user-based collaborative filtering algorithm. Therefore, we can say that adding users' demographic characteristics and preferences helps us improve the prediction accuracy.

2) For new users

We continued to utilize the 3 random samples in Section 4.2.2) 3 to be new users. As it traditional collaborative filtering algorithms can not handle the cold start problem, here we just show MAEs for the ratings of those 3 users.

4.4 Conclusion

In this study, we have compared algorithms in R package Recommenderlab which only consider users' ratings with another hybrid recommender system that also combines the information of users' demographic characteristics and preferences for movie genres. According to the previous work, hybrid collaborative filtering algorithm improves the prediction accuracy and thus the rationality of recommendations based on the MAE criterion. What is more, hybrid collaborative filtering algorithm can solve the cold start challenge which is impossible using traditional collaborative filtering algorithm. Therefore, the hybrid recommender system is better than recommender system using collaborative filtering algorithms from Recommenderlab package.

However, there exists something that can be further explored or improved if we had a lot more time:
1) As we can see in Section 4.3, hybrid collaborative filtering algorithm outperformed user-based collaborative filtering algorithm. However, when we tried to find the best weights, α, β (see Section 4.2), it took us about an hour to do the calculation. This is on the basis of the fact that we just set 8 different values for either α or β, which yielded only a very small amounts of choices, 36 combinations, considering the possible values for α, β. It's better to find a more efficient way to calculate.

2) In our hybrid algorithm, we manually picked 60 nearest neighbors to do predictions. However, the number of nearest neighbors may also affect the prediction accuracy. We can further explore the effect of choosing different numbers of nearest neighbors and determine the most appropriate number of nearest neighbors to predict ratings.

3) In terms of CHTC, we did not have enough time to learn and apply our algorithms on it. Because computing power in a single machine is so limited that we randomly sampled and tuned our parameters or evaluated the predictions between two algorithms based on samples. Taking advantage of high throughput computing, we can make our result more robust.

Reference

[1] Mean absolute error. (n.d.). In *Wikipedia*. Retrieved from https://en.wikipedia.org/wiki/Mean_absolute_error.
[2] Jeff M. Phillips. (2013). *Jaccard Similarity and Shingling*. Retrieved from https://www.cs.utah.edu/~jeffp/teaching/cs5955/L4-Jaccard+Shingle.pdf.
[3] Department of International Economic and Social Affairs. (1982). Provisional Guidelines on Standard International Age Classifications. *Statistics Papers, M*(74). Retrieved from http://unstats.un.org/unsd/publication/SeriesM/SeriesM_74e.pdf.

Appendix

Table 1 Lookup Table for Age Group and Preference for movie genres

Age_Group	Preference1	Preference2	Preference3	Preference4	Preference5
0	Comedy	Romance	Drama	Action	Thriller
1	Drama	Thriller	Comedy	Sci-Fi	Action
2	Drama	Thriller	Romance	Adventure	War
3	Drama	Comedy	Action	Thriller	Romance
4	Drama	Comedy	Romance	Action	Adventure
5	Drama	Comedy	Action	Thriller	Romance
6	Drama	Comedy	Action	Thriller	Romance
7	Drama	Comedy	Romance	Thriller	Sci-Fi
8	Drama	Comedy	Thriller	Romance	Sci-Fi
9	Drama	Comedy	Action	Thriller	Romance
10	Drama	Thriller	Romance	Action	Crime

Table 2 Lookup Table for Occupation and Preference for movie genres

Occupation	Preference1	Preference2	Preference3	Preference4	Preference5
administrator	Drama	Comedy	Action	Thriller	Sci-Fi
artist	Drama	Action	Thriller	Adventure	Romance
doctor	Drama	Thriller	Romance	Comedy	Crime
educator	Drama	Comedy	Romance	Thriller	Sci-Fi
engineer	Drama	Comedy	Action	Thriller	Romance
entertainment	Drama	Comedy	Action	Romance	Sci-Fi
executive	Drama	Comedy	Romance	Thriller	Sci-Fi
healthcare	Drama	Comedy	Action	Thriller	Sci-Fi
homemaker	Drama	Comedy	Action	Thriller	Adventure
lawyer	Drama	Thriller	Action	War	Sci-Fi
librarian	Drama	Comedy	Thriller	Romance	Sci-Fi
marketing	Drama	Comedy	Thriller	Romance	Sci-Fi

none	Drama	Thriller	Romance	Action	Adventure
other	Drama	Comedy	Action	Romance	Sci-Fi
programmer	Drama	Comedy	Romance	Thriller	Adventure
retired	Drama	Comedy	Thriller	Romance	Adventure
salesman	Drama	Comedy	Action	Thriller	Romance
scientist	Drama	Action	Romance	Thriller	Adventure
student	Drama	Comedy	Action	Thriller	Romance
technician	Drama	Comedy	Action	Thriller	Romance
writer	Drama	Comedy	Action	Romance	Adventure