RecSys Project Report Template

Rui Tu

13206398
COMP 30490

1 Introduction

Collaborative filtering is a common approach in recommend systems. It normally uses the ratings given by users who shares the same interests or experiences to predict a rating or recommend some products. Often a large amount of data of both the predict user and other users are gathered and analysed. Collaborative filtering method then use the similarity between each user and the predict user to give a possible predict rating or recommendation. The result may not be correct, but the larger dataset we use, the more accurate prediction system will make. Collaborative filtering is based on the assumption that people agreed in the past will agree in the future, and that they will like similar kinds of items as they liked in the past.

The main approaches collaborative filtering working can be divided into three parts. The first step is collecting users' behaviours and information. In order to get a better prediction, the data being collected need to be as much as possible and try to be specific to separate users with different attributes. The second step is trying to get a group of users who shares the same interests or attributes with the predict user. There are serval methods to evaluate the similarity. The final step is give the proper prediction based on the closest users because we can assume that the higher similarity get, the closer of ratings or interests will be for two different users.

In my work I use two methods for measuring the similarity, Mean Square Differences (Shardanand & Maes, 1995) and Pearson's Correlation Coefficient. Besides, three different prediction methods is used, which are mean item rating, distance based prediction and Resnick's prediction. The cross validation method selected is leave-one-out testing. The variables of the prediction are neighbourhood size and minimum overlap between user. Then we calculate three values as result, root mean square error, prediction coverage and program running time.

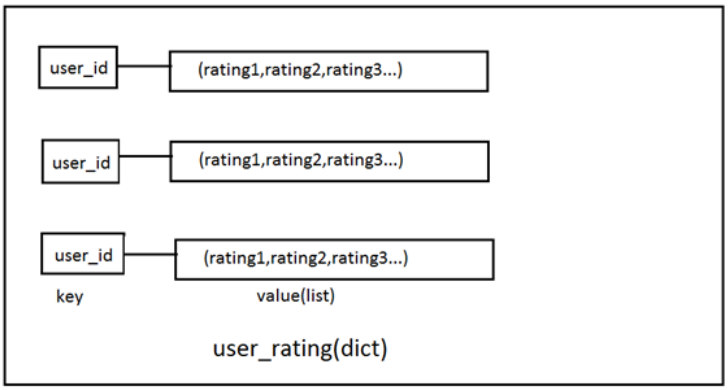
2 Collaborative Filtering

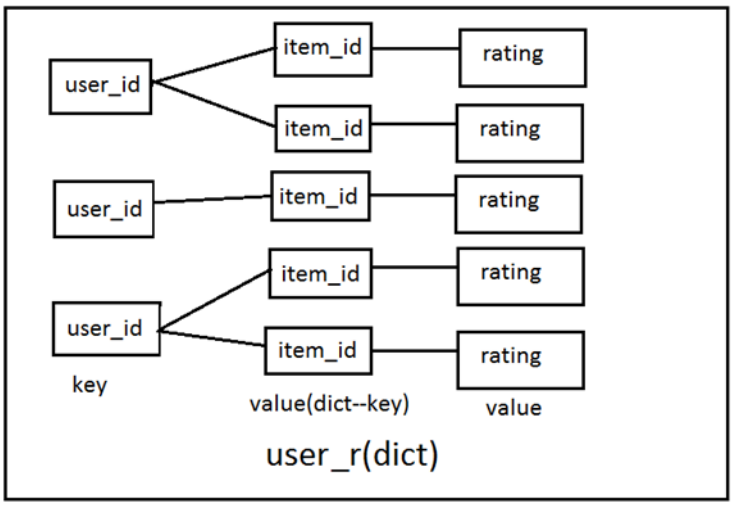
In this section you should provide a brief overview of the collaborative filtering technique, paying particular attention to the role of ratings, neighbourhoods, and prediction.

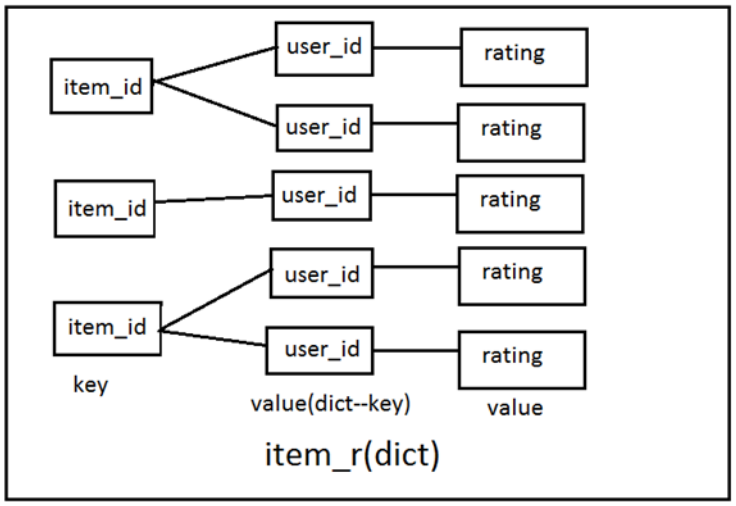
2.1 An Overview of Collaborative Filtering

In this section, I use three kind of approaches of collaborative filtering, mean-rating prediction, distance-based collaborative filtering and Resnick's prediction technique.

The preparation for input data of these three approaches are the same. Firstly, I read the data from 100k.csv and save it into four different dictionaries. 'user_rating' dictionary use each user as key and the value is a list contains all the ratings this user give. 'item_rating' dictionary use each item as key and the value is a list contains all the ratings this item get. These two dictionaries are mainly used for the traversal of data. Another use is counting the average value of user's rating and average rating of a single item. Because the value of these two dictionary are list type, they are much easier than doing traversal on a nested dictionary. The other two dictionaries are 'user_r' and 'item_r'. The key of 'user_r' dictionary is the user id. And the values are a set of dictionaries contains item id that user rated and its rating. 'item_r' dictionary is just the opposite. The item id is the key of the dictionary outside. And the inner dictionary contains the users' id that have rated this item and its ratings.

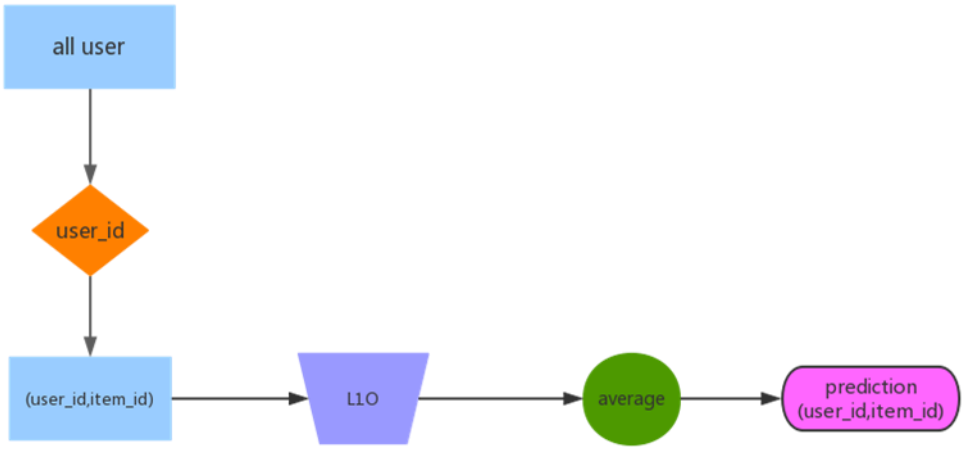






2.2 Approach 1 – Mean-Rating Prediction

This is the most simple and quickest research. I write a method called mean_item_item. It takes a user_id, item_id pair as input and give back a prediction. First, checking if this item has been rated by the user. Or this item can't be predicted for this user. After that check if this item is also rated by other user. If so, we can do leave-one-out to pick out the predict user's rating from the data set. And then get an average value of all the other users' rating as the prediction. Loop all the user, item pair and record the useful information like rmse and coverage.

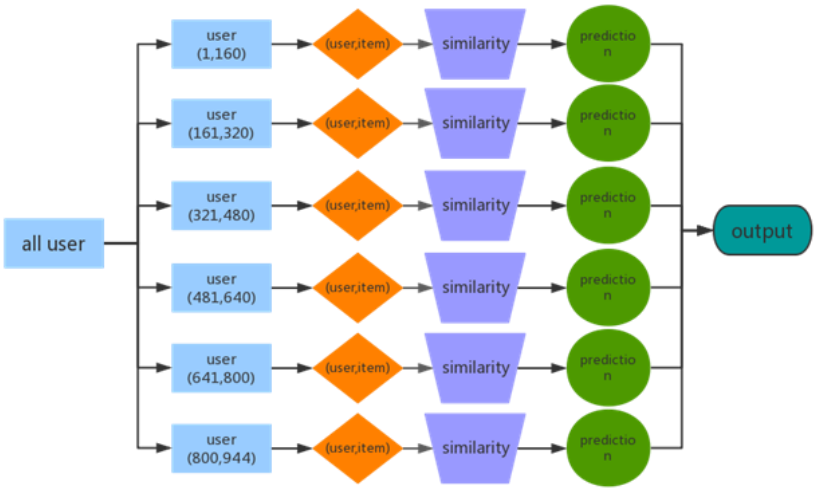


2.3 Approach 2 – Distance-Based CF

This approach can be divided into two parts, the calculation of similarity and prediction. The distance similarity method take two user_id and the predict item_id as the input. For this two users, I get the overlap of item they rated first. This step will reduce calculation of just loop all the items. Then leave the item that going to predict out and get the similarity between these two users. For the calculation of predict rating, we take the user_id, item_id pair that going to predict as the input. First, find all the user that have rated this item as its neighborhood. Leave the predict user out and use all the similarities between predict user and other users given by similarity method to get weight of each user. Finally, calculate the prediction result using weight and rating of this item from its neighborhood. The weight as denominator in the final calculate can be 0 when one user rate the

highest score and the other rate the lowest score. In this situation, we consider it as unpredictable. This will reduce the coverage of final result.

To increase the efficiency of this approach, I use multiprocessing during calculation. Before the final loop to get all users' prediction, I divided all 943 users into 6 group. Then I put each group into the process pool of python, so each group is handled by a processor at the same time as long as the computer has enough number processor. For each group, it calls a multi method to get the range of users in this group and get all the items rated by this user. The item should also be rated by other user to make L1O. Then every user, item pair call the prediction method and get its own prediction and is append into a list. After they all finished, the list of result for each group is joined together and write into the csv file.



Approach 2/3

2.4 Approach 3 – Resnick's Prediction Technique

This approach is similar to approach 2. It can also be divided into similarity part and prediction part. During this approach, the average rating need to be used very frequently. To avoid duplicate calculation of average value, I store the sum and number of ratings given by each user into a list. And save the user_id and list pair into a dictionary.

The reason for not storing the average value directly is that this is more convenient to do L1O later. So, each time I need to use average rating I can get it from the dictionary.

For the similarity between two user, the input is two user_id and a predict item_id. We use the item_id to calculate the average rating after L1O based on the original sum and number in the dictionary. The average user's average rating can be count directly. Then we can easily get the similarity between these two users. In some situations, we cannot get the similarity because denominator is 0. Then we consider this user, item pair as unpredictable.

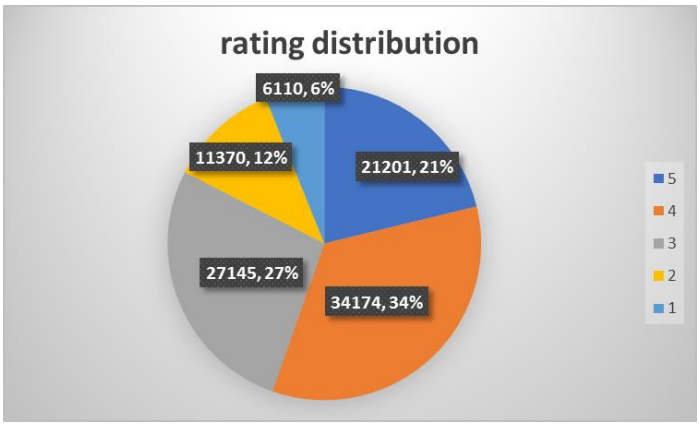
After that, we can use all the similarities and average ratings of one's neighborhood to calculate the prediction rating of a certain user, item pair. For the overall result, I use the same multiprocessing method like approach 2 and greatly improve the efficiency of this approach.

3 Evaluation

3.1 Dataset

The dataset I use is read from a csv file called '100k.csv'. Each row represents a single rating that one user gives to a certain item. Each line contains four values: user_id, item_id, rating and a timestamp. This time we don't use the last value.

- There are 943 users, 1682 items and 10000 ratings in total.
- The rating density of this dataset is 0.06304669364224533.
- The mean, median standard deviation, max, min rating per user is saved in file 'user_detail.csv'.
- The mean, median standard deviation, max, min rating per item is saved in file 'item_detail.csv'.
- There are 21201 5-star ratings, 34174 4-star ratings, 27145 3-star ratings, 11370 2-star ratings and 6110 1-star ratings.



3.2 Methodology & Metrics

The Leave-one-out evaluation methodology leave the test data out every time. Taking the remaining data as training set and evaluate it. Then repeat for every test data. In this recommend system, a complete L1O takes every exist user, item pair as training set and get its predict rating from the training set consists of all other related user, item pairs using three different approaches. The leave-one-out evaluation mythology can exclude the influence of test data itself and give back closer predict result. However, this method may be very time consuming when the approach is complex. Because all the parameters using in the approach is different because of L1O. In our system, we need to calculate the similarity between two same users many times because every time a different item is leaved out.

In approach 2 and 3, we can vary the neighbourhood size and minimum overlap. For each single user, item pair, we sort all the similarities between this predict user and other users so we get a sorted list of similarities of all the neighbourhood for this user. By varying the neighbourhood size, we pick the top n similarity as its n nearest neighbourhood as the training set.

The overlap is the number of items that both two user have rated. We can exclude some user who only rated few same items with the predict user. This kind of user can't make good contribution for the prediction and sometimes can be the noise in the training set. So we set a minimum overlap to get users who rated at least n same items with the predict user into training set.

We use three metrics to evaluate the performance of our recommend system. The first one is accuracy. In this system, the root mean square error is chosen to evaluate the accuracy. For each predict rating, we get the difference between predict and actrual rating. And count the root mean square of every difference during one L1O duration. The advantage of RMSE is that it produces errors that are on the ratings scale.

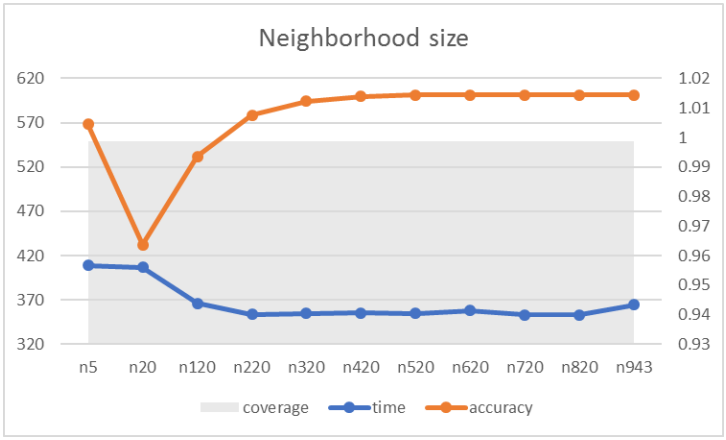
The next metric is coverage. It measures how much ratings can be predicted. Sometimes a method gets a very good accuracy but a low coverage. We still cannot consider it as a good result because many of the test data can't predict. A good system should get the highest accuracy with an acceptable coverage. In our system, coverage can be calculated by the number of predict ratings divided by the total number of rating 100000.

The last metric is efficiency. This is measured by the total time consumed during each L1O procedure. I get the timestamp at the start and end of my code and get the total used when the code running.

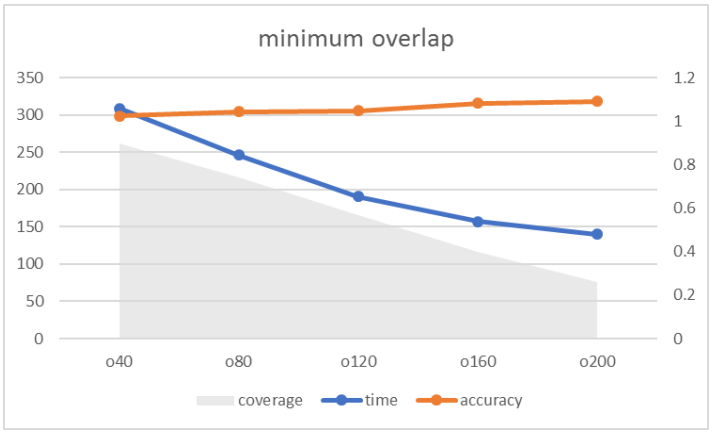
3.3 Approach 1 Results

In mean rating approach, I get 99859 predict ratings. The accuracy is 1.02 and the coverage is 0.99859. The time that this approach consumed is 2.35s. Because this approach is very simple, so it doesn't use so much time to generate the prediction. However, the accuracy is not very good. There are 141 items that only be rated by one user, so they can't use L1O to generate the prediction and influence a little bit to the coverage.

3.4 Approach 2 Results



n size	time	accuracy	coverage
n5	408.9602	1.004473	0.99859
n20	406.7676	0.963711	0.99859
n120	366.1914	0.993542	0.99859
n220	353.4324	1.007529	0.99859
n320	354.4545	1.012286	0.99859
n420	355.2079	1.013923	0.99859
n520	354.4924	1.014355	0.99859
n620	357.7938	1.01442	0.99859
n720	352.8412	1.01442	0.99859
n820	353.0893	1.01442	0.99859
n943	364.5173	1.01442	0.99859



overlap	time	accuracy	coverage
o40	308.5856	1.023649	0.8991
o80	246.0408	1.043922	0.74245
o120	190.7679	1.047894	0.56615
o160	156.9204	1.081942	0.39849
o200	139.7981	1.091071	0.25876

In the distance based approach, there are two parameters that can be varied, the neighborhood size and the minimum overlap. As we can see in the two graphs above, they record the three evaluation matrices. The blue

line is time consumed, the orange line is the accuracy of RMSE values. And the grey area represents the coverage.

By varying the neighborhood size from 5 to 943 which is all the possible user, we can find that the time has a little fluctuation but only in a range of 40 seconds. So a single L1O procedure is about 360 seconds. The efficiency is pretty steady because no matter what neighborhood size we pick, the system still need to calculate all the similarities and sort them later. The most time consuming part in this recommend system is the huge amount of calculation of similarity. So varying neighborhood size doesn't decrease much of the calculation and can't influence much to the time.

The accuracy has some big changes during the variation of neighborhood size. When we set size to 5, we get a high RMSE value actually. But the value drop to bottom and start increase as the size increase. It is because 5 neighbour is too small that they may not cover much items. So it get a high RMSE value. And as the size increase to 20, the advantage of use nearest neighbour as the training set shows. It get a lowest RMSE value. Then the RMSE increase till size come to 620. After that the RMSE value stay the same as neighbourhood increase. it is because no user has more than 600 neighbours, so this change is not effective.

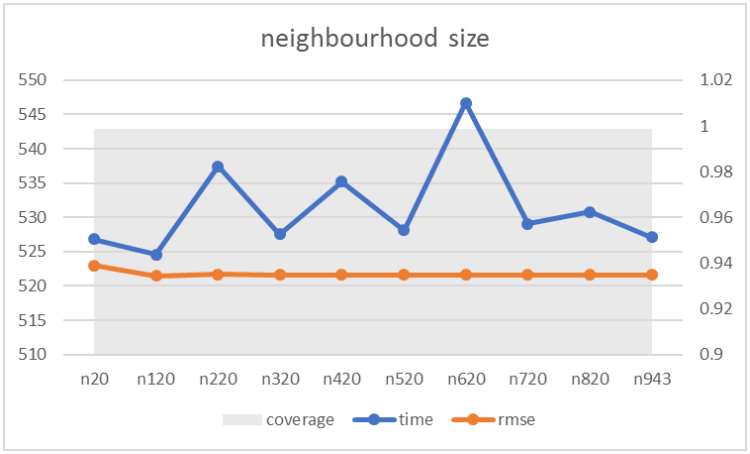
The coverage always stay the same value as 0.99859 during the whole test because varying neighbourhood size doesn't change the number of predict rating. Only the items which has one user rated is not predictable because of L1O.

By varying the minimum overlap from 40 to 200, the time drops as the overlap number increase. It is because as the minimum overlap increase, many users who doesn't match the condition is thrown out of the training set. So the number of users in the training set also decrease as the number goes bigger. And the time use to calculate the similarities of these users is saved. So the time drops greatly from 300 second to 140 second.

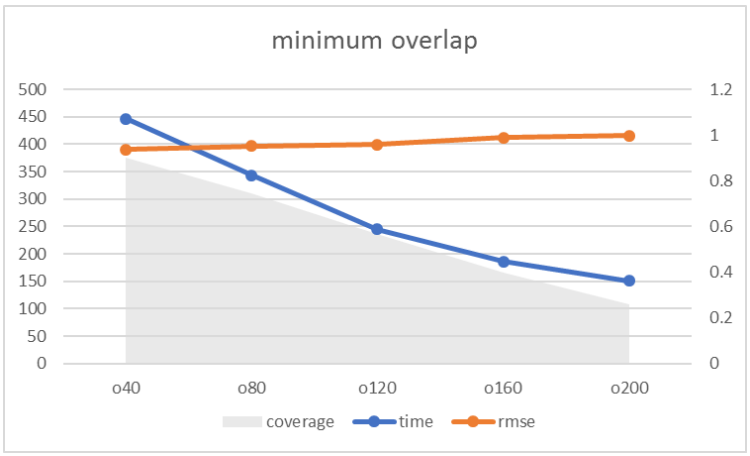
Besides, the shrink of the training set also cause the decrease of coverage. Many user, item pair can't be predicted all the users satisfied with the minimum overlap haven't rated that item. So as the overlap number increase, more and more items haven't be rated even once by the users in the training set. As a result, the area of coverage in the graph get smaller steady.

However, as the overlap number get bigger, the users in the training set become closer to the predict user become they share many item in common. This makes great contribution to the accuracy of prediction. As the minimum overlap set to 40, the RMSE is smaller than the original one. However, as the overlap number increase, the users involve in prediction become less. So for a single item the training set becomes smaller, this betray the idea of collective intelligence. The accuracy start to drop so the RMSE value increase as shown in the graph.

3.5 Approach 3 Results



n size	time	accuracy	coverage
n20	526.8328	0.938867	0.99859
n120	524.5718	0.934423	0.99859
n220	537.443	0.935103	0.99859
n320	527.5292	0.934942	0.99859
n420	535.195	0.934771	0.99859
n520	528.0884	0.93477	0.99859
n620	546.6879	0.934924	0.99859
n720	529.0874	0.934924	0.99859
n820	530.7885	0.934924	0.99859
n943	527.1355	0.934924	0.99859



o40	446.5238	0.93735	0.8991
o80	343.6149	0.95124	0.74245
o120	244.9534	0.958908	0.56615
o160	185.7734	0.988151	0.39849
o200	150.9399	0.997239	0.25876

The approach 3 use Resnick technique. We can still doing the experiment by varying the neighborhood size and the minimum overlap. In the two graphs above, three evaluation matrices are recorded. The blue line is time consumed, the orange line is the accuracy of RMSE values. And the grey area represents the coverage.

By varying the neighborhood size from 5 to 943 users, we find that as the size increase the time doesn't change much. Only small fluctuation appears. A complete L1O procedure of Resnick is about 520 second. The efficiency is pretty steady because no matter what neighborhood size we pick, the system still need to calculate all the similarities and sort them later. The most time consuming part in this recommend system is the huge amount of calculation of similarity. So varying neighborhood size doesn't decrease much of the calculation and can't influence much to the time.

The accuracy in Resnick doesn't change much as the neighborhood size increase. It is because the formula of Resnick itself. The target item is assigned a rating that is an adjusted form of the target user's average rating. When changing the size, we only changes how much users who is not so close to the target user because they are in the tail of sorted similarity list. And these changes don't takes too much effect on the average rating of target user.

The coverage always stay the same value as 0.99859 during the whole test because varying neighbourhood size doesn't change the number of predict rating. Only the items which has one user rated is not predictable because of L1O.

By varying the minimum overlap between 40 to 200, the time drops from 450 second to 150 second which is quite a lot. It is because as the minimum overlap increase, many users who doesn't match the condition is exclude from the training set. So the number of users in the training set also decrease as the number goes bigger. And the time use to calculate the similarities and prediction of these users is saved.

The trend of changes happens on coverage is the same as time. Because the shrink of training set, less items can be covered by the users in the training set. Only item which has been rated at least one time by the users in the set can be predicted. So as the overlap number increase, the coverage area gets smaller and smaller.

The accuracy is decrease as the RMSE value increase smoothly. However, set a minimum overlap number such as 40 will increase the accuracy comparing to the original result. Because of the training set is consist of only close users to the target user because they have lots of rated items in common. But as the overlap number increase, the size of training set decrease greatly because only a small part of user can meet the overlap requirement, which makes the prediction inaccurate than before.

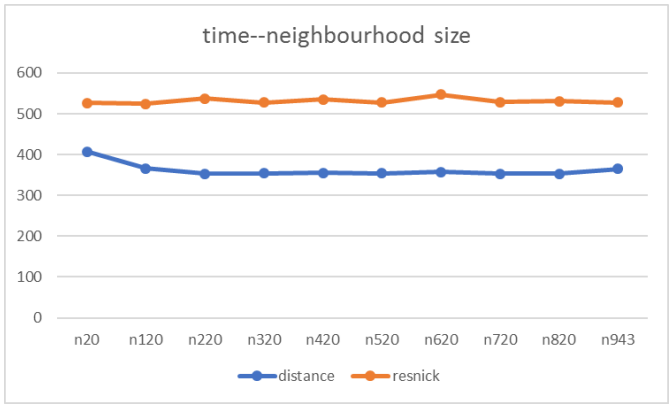
4 Discussion

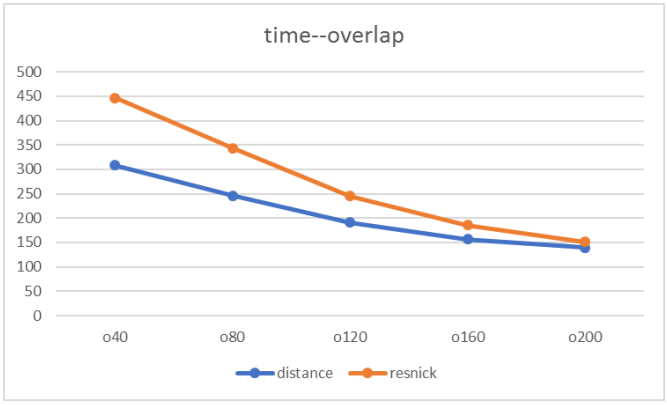
In this part I will compare the three evaluation matrices of three different approaches.

The first is the efficiency matric. All the running time used in this section is the average value of 10 times running time. The table on the right side is the time used for a complete leave one out procedure without changing any neighbourhood policy. As we can see, mean rating approach is much faster than the other two approaches with less than 3 second. The efficiency of distance based approach is better than Resnick approach. It is because both of these method have similarity and prediction

approach	time/s
mean rating	2.35
distance	365
Resnick	529

part, but Resnick approach has a much complex way for calculating the similarity. So the overall time Resnick approach used is about twice as the distance based approach.



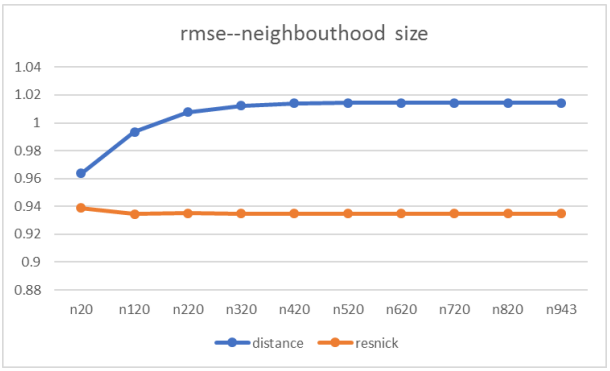


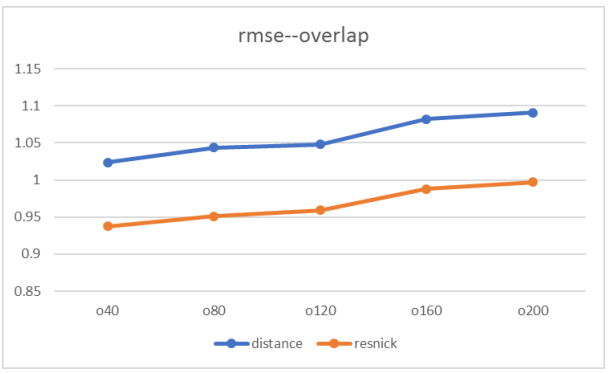
By varying the neighbourhood size, we can find that there is no obvious change on both approach. Distance based approach always use 150 second less than Resnick approach.

As is shown in the right graph, by varying the minimum overlap, we can find that both two approach drops significantly and finally use nearly the same time when overlap is set to 200. Because the set of minimum overlap, the training set shrinks a lot as number increase. And less times of similarity and prediction calculations need to be done so both these approaches use less time than before. When we set 200 minimum overlap, the coverage is about 0.25 which means that only very few user is in the training set, so there is no obvious different between two different approaches.

The second is accuracy matric. We can find from the table on the right side, when we don't change the neighbourhood policy, the mean rating approach gets the lowest accuracy. And the Resnick approach get the best accuracy which is much better than the other two approaches. Because of using different way to generate prediction, the RMSE value is different. Mean rating just calculate the mean rating during L1O, so it gets the lowest accuracy. Distance based approach use neighbours' rating to generate the target user's prediction. Resnick approach use the ratings of target user's neighbour to adjust its own average rating. So Resnick get better accuracy than distance prediction.

approach	rmse
mean rating	1.020593
distance	1.01442
Resnick	0.934924





By varying the neighbourhood size, we can find that the RMSE value of distance based approach first rise up a little bit and then become steady. The value of Resnick approach is steady all time during the experiment. This is because the prediction of distance approach is complete based on the user from training set and its similarity, so if all user in the training set is the closer user, it will get a high accuracy. However, Resnick approach is based on the average rating of target user. The neighbour is only working on adjustment. It will not be influenced greatly by the neighbourhood size.

By varying the overlap, both these two approaches show a same trend, the RMSE value increase as the minimum overlap number increase. The only difference is that Resnick approach is more accurate.

For the coverage matric. The distance based approach and Resnick approach always get the same coverage when varying the neighbourhood policy. It is because this makes the same changes to the training set, which directly influence the coverage. So no matter what approach we select, the coverage will be the same.

Overall, there is not perfect solution. The Resnick approach get the highest accuracy, but its calculation is too complex. Its efficiency is the worst. The mean rating approach runs really fast, but it can't give back an accurate enough prediction. So all these approaches have their own advantages and shortages. We can based on the situation and the first goal to select which approach to use.

For the improvement, I think we can add similarity threshold to the neighbourhood policy. The training set will then only select neighbours exceeding a fixed similarity threshold. Besides, the further experiment can be a combination of these three different neighbourhood policy to strike a balance and get the most suitable training set for an accurate result and acceptable coverage.

5 Conclusions

Review what you have achieved and comment on any aspects that you could not complete satisfactorily.

For this assignment, I build a simple recommend system. First, I analyze the dataset and get some useful information such as user number. Next, the data is saved into a proper structure. What I selected is nested dictionary. Then, three different approach based on leave one out is implemented. Finally, the prediction result is evaluated using three different matrices.

	n-size	overlap	accuracy	efficiency	coverage
mean rating	/	/	1.020593	2.356308	0.99859
distance	20	1	0.963711	406.7676	0.99859
resnick	120	1	0.934423	524.5718	0.99859

The best result I got is in the table. This is the case that the highest accuracy get for the three different approach. Resnick get the best accuracy of 0.934 but it is the slowest one. If your aim is find the most accurate prediction, you can use Resnick approach. The distance approach has a good prediction with an acceptable accuracy. So if you want to get a good result with a limit time, you should select this approach.

The mean rating approach is the quickest, but its accuracy is the worst. If you want a quick prediction, you can select this approach.

6 References