EE219 - UCLA

WINTER 2019

PROJECT 3: REPORT

Collaborative Filtering

GROUP MEMBERS

Anchal Goyanka

Nandan Parikh

Pratik Mangalore

Srishti Majumdar

INTRODUCTION

In this project, we aim to build a recommendation system using collaborative filtering methods. During the course of this project, we implemented and analysed neighbourhood-based collaborative filtering and model based collaborative filtering. We first visualised and understood our dataset, then implementing k-nearest neighbourhood filtering, non-negative matrix factorization and matrix factorization with bias. We evaluate each process and finally, compare them to each other.

MovieLens dataset

Q 1. Sparsity of the movie rating dataset

Sparsity is given by the ratio between the total number of available ratings and the total number of possible ratings. In the table, we can see the values required to calculate the required sparsity.

Values for Calculating Sparsity	Count
Users	610
Movies	9724
Available Ratings	100836

We use $Sparsity = \frac{Available\ Ratings}{Users \times Movies}$ and the sparsity value obtained is 0.01699.

Q 2. Histogram- Frequency of Rating Values

Using the ratings.csv file, we compute the count the number of entries containing each rating as shown in the table.

Frequency of Ratings		Count
Ratings	0.0	0
	0.5	1370
	1.0	2811
	1.5	1791
	2.0	7551
	2.5	5550
	3.0	20047
	3.5	13136
	4.0	26818
	4.5	8551
	5.0	13211

On obtaining these counts, we plot a histogram as shown in Figure 2.1.

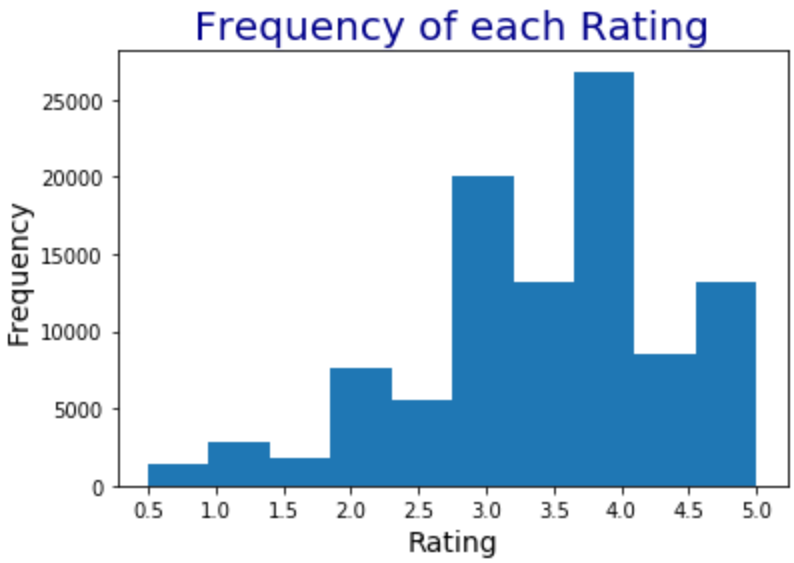


Figure 2.1 : Histogram plot showing the frequency of each rating

On viewing the graph we see, that users usually tend to give ratings of 3.0 or more. It is less common to very low or very high ratings as compared to middle range values. Most movies tend to receive a 3.0 or 4.0. No movie has received a rating 0.9

Q 3. Distribution of Ratings among Movies

For this plot (Figure 3.1), we first counted the number of ratings each movie has and stored this information in a dictionary with key as movie id and value as the obtained count. We then sorted the dictionary by value. The indexing of movies is done as follows: the movie id with the largest number of ratings is assigned index 0 and the rest of the indices are assigned on decreasing value of rating count/frequency. Hence higher the index value, lesser is the number of ratings received by that movie.

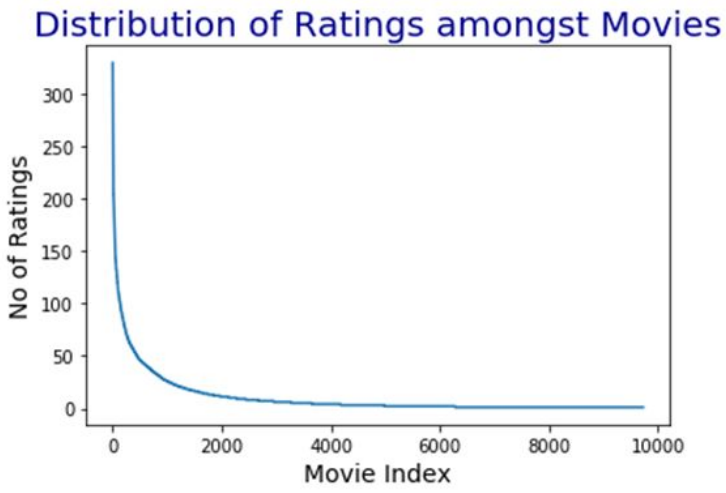


Figure 3.1 : Curve showing the number of ratings received by each movie

Q 4. Distribution of Ratings among Users

The approach for this question is similar to Q3. Like in Q3, the indexing of users begins at 0 and higher the user index value, lesser is the number of ratings given by the user.

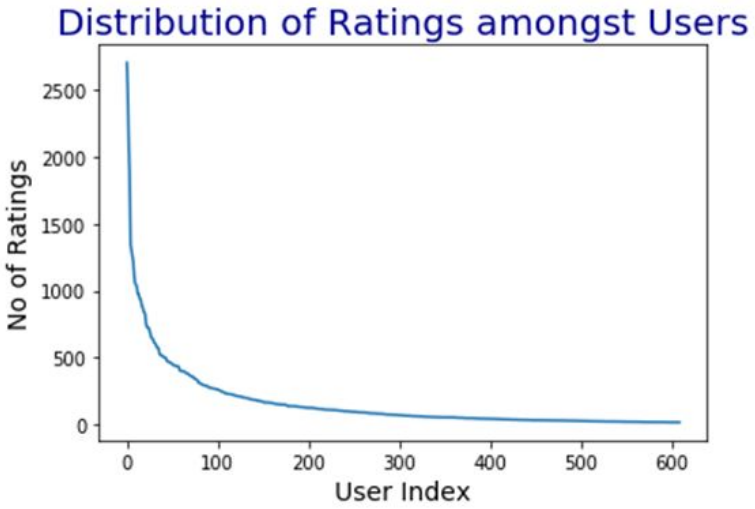


Figure 4.1 : Curve showing the number of ratings given by each user

Q 5. Explanation of Q3 Plot

From figure 3.1, we see that few movies have a lot of ratings. These can prove more useful while predicting which movies to recommend. A movie having a lot of ratings can also be indicator of the popularity of the movie. We see that quite a lot of movies have 20-100 ratings and the expected value for number of ratings can lie within this range. Around four- fifth of the dataset has close to 0 ratings, so we can use sparse-matrix techniques in our predictions.

Q 6. Variance of Rating Values Received by each Movie

For this question, we use a dictionary to store the ratings received by each movie and then calculate the variance using statistics library. For those movies with only one or no rating, we receive a StatisticsError and use variance as 0 for those movies. We then plot the received variances as shown in figure 6.1.

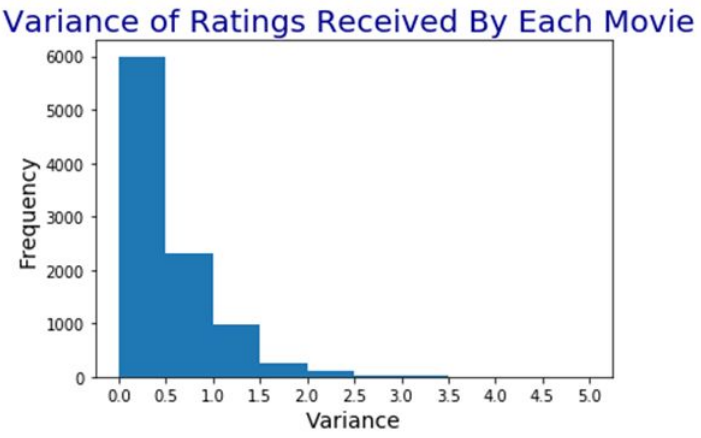


Figure 6.1 : Histogram showing movies and variance in their ratings

Neighborhood-based collaborative filtering

Q 7. Computing Mean Rating for User Using Her Specified Ratings

We know that I_u denotes the set of item indices/movies for which ratings have been specified by user u. Hence $K = |I_u|$ denotes the number of movies rated by user u.

Hence, we can mean rating μ_u of that user using:

$$\mu_u = \frac{1}{K} \sum_{k=0}^{K} r_{uk}$$

Note: r_{uk} denotes the rating of item/movie k by user u.

Q 8. Explaining $I_u \cap I_v$

$I_u \cap I_v$ indicates the number of movies that user u and v both rated. There are over 9000 movies and there are several users who have rated less than 500 movies. Hence, the intersection of I_u and I_v can be an empty set because there can exist two users who have rate completely different movies.

k-Nearest Neighborhood (k-NN)

Q 9. Explaining Mean Centering of Raw Ratings

We conduct mean centering of raw ratings for handling the diversity of rating preferences amongst user. Mean centering helps avoid bias created by inherent user preferences which will affect recommendation systems. If, for example, we have a user who usually tends to give lower ratings but has allotted a high rating to a movie, we can get more information about how good the movie is than when a movie receives a high rating from a user who tends to give high ratings.

Q10. 10-fold Cross Validation results for k neighbors ranging from 2 to 100

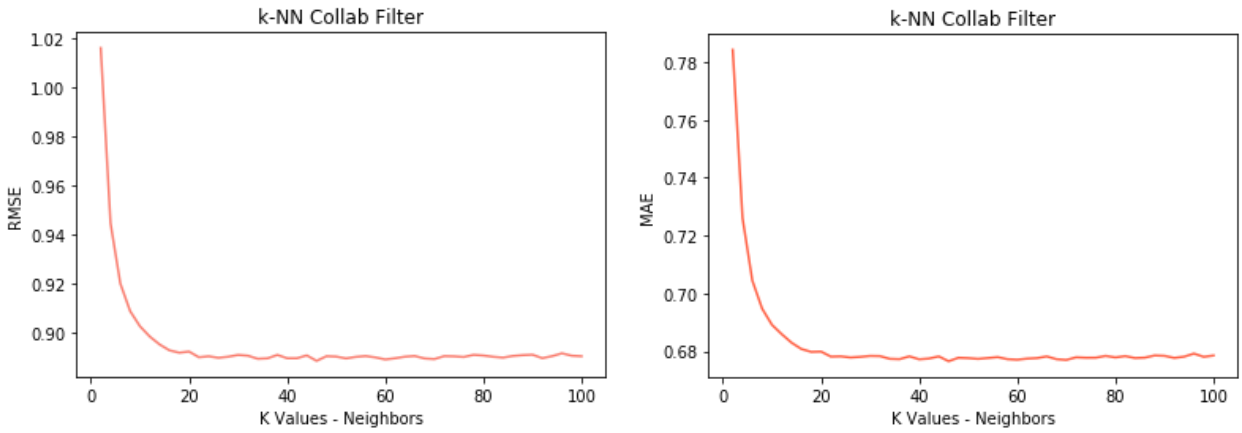


Figure 10.1 : Average RMSE vs k and Average MAE vs k

We designed the k-NN collaborative filter using the surprise module to predict the ratings of the movies. The k neighbors was varied from 2 to 100 to check the mean RMSE and mean MAE for 10-fold cross validation. For each of the 10-fold cross validation, we use the average RMSE and MAE across the 10 runs and plot it as shown in Figure 10.1 above.

```
algo = KNNWithMeans(k=kneighbors, sim_options=sim_options)
result = cross_validate(algo, data, measures=['rmse', 'mae'], cv=10, verbose=True)
RMSE.append(np.mean(result['test_rmse']))
MAE.append(np.mean(result['test_mae']))
```

The minimum RMSE and MAE are found for k = 44 and have the values 0.889 and 0.676 respectively, but the steady state is achieved with k ~ 20 as described below.

Q 11. Minimum ‘k’ value - Steady State

It can be easily seen from the plot above that the after specific value(s) of k, the RMSE and MAE do not drop significantly and reach a steady state. The curve shows that around the region of k=20, the RMSE and MAE do not drop much after increasing the k value of neighbors. We conclude that ‘k’ value is steady just after 20, and can be considered as **22.**

After 22, even if we increase the neighbors we will not a significant improvement in the results.

k-NN on trimmed dataset :

As mentioned we have trimmed the test data using the three filters - Popular Movie trimming, Unpopular Movie trimming and High Variance Movie trimming.

As we can see intuitively, the parts 2 and 3 will result in somewhat bad recommendations as we are using unpopular movies and high variance movies. This can be cross-checked with all of our filters which will be used for the questions.

Q 12. k-NN filter for Popular Movie trimmed test set

We perform the same k-NN filter for the new test set and as expected the curve for RMSE vs k reaches a steady state. Minimum average RMSE is **0.872** for k value 56.

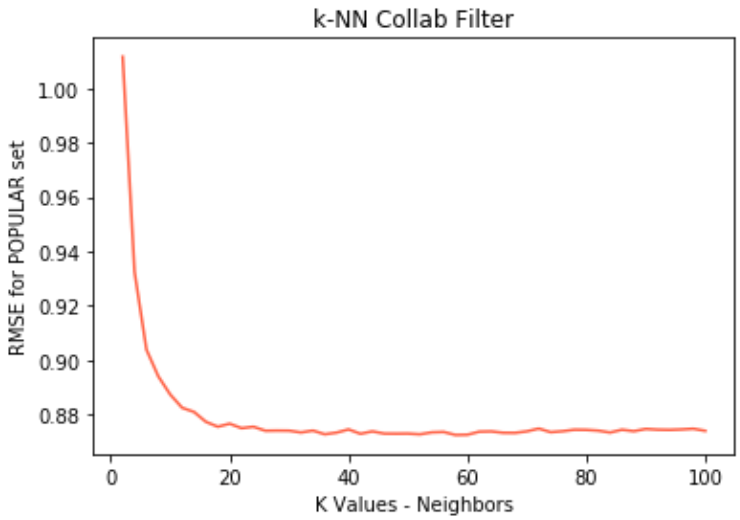


Figure 12.1 : Average RMSE vs K values for popular movie trimmed set

Q 13. k-NN filter for Unpopular Movie trimmed test set

We perform the same k-NN filter for the unpopular movie test set and as described with the intuition above, the curve for RMSE does not reach a steady state and fluctuates.
Minimum average RMSE is **1.110** for k value 78.

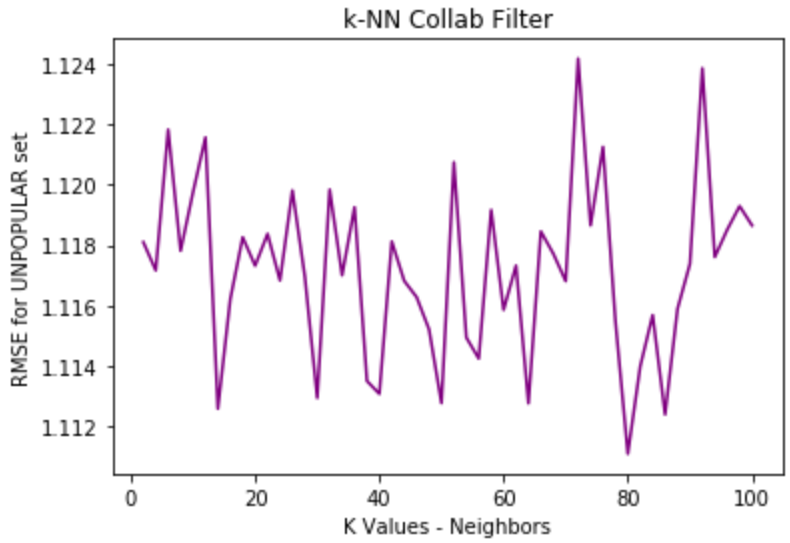


Figure 13.1 : Average RMSE vs K values for unpopular movie trimmed set

Q 14. k-NN filter for High Variance Movie trimmed test set

We perform the same k-NN filter for the high variance movie test set and as described with the intuition above, the curve for RMSE does not reach a steady state and fluctuates.
Minimum average RMSE is **1.482** for k value 58.

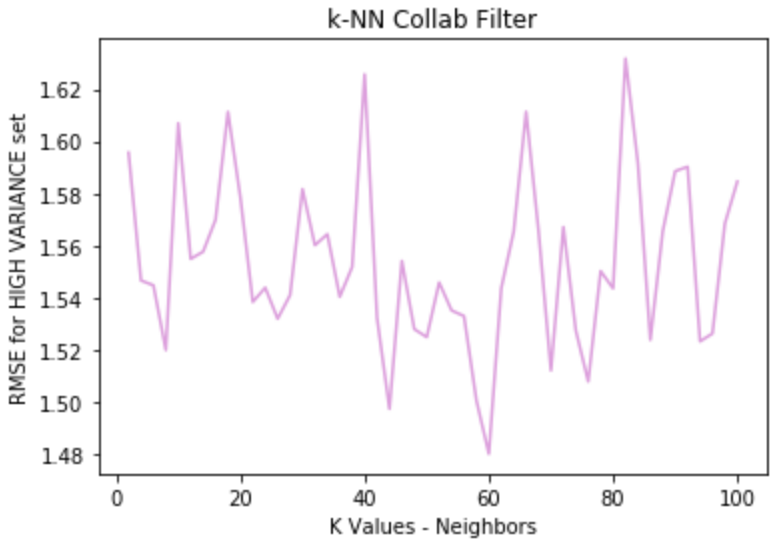


Figure 14.1 : Average RMSE vs K values for high variance movie trimmed set

Q 15. ROC curve for k-NN filter for k = 22

As the question suggests, we have plotted for ROC curve for k = 22. The area under the curves is mentioned below :

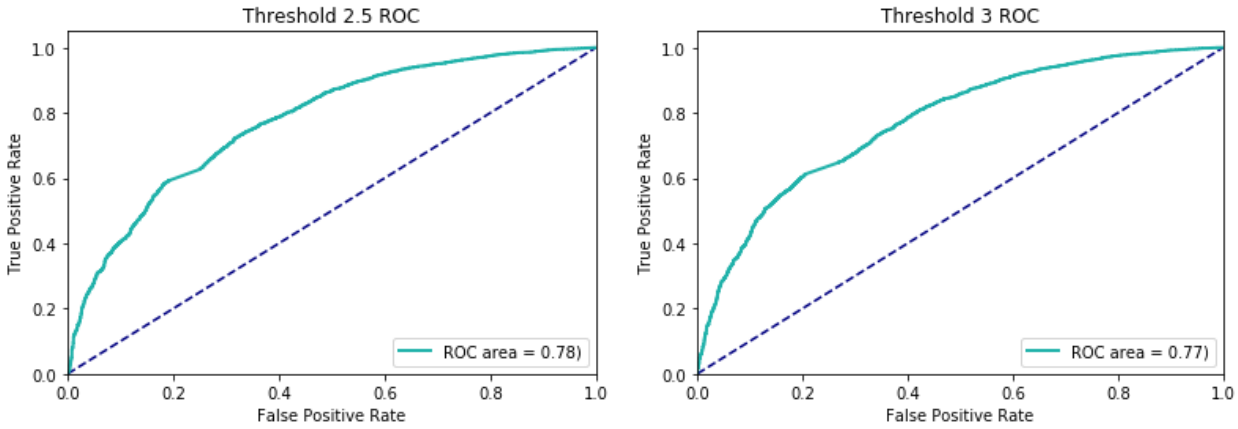


Figure 15.1 : ROC curve using threshold as 2.5 and 3

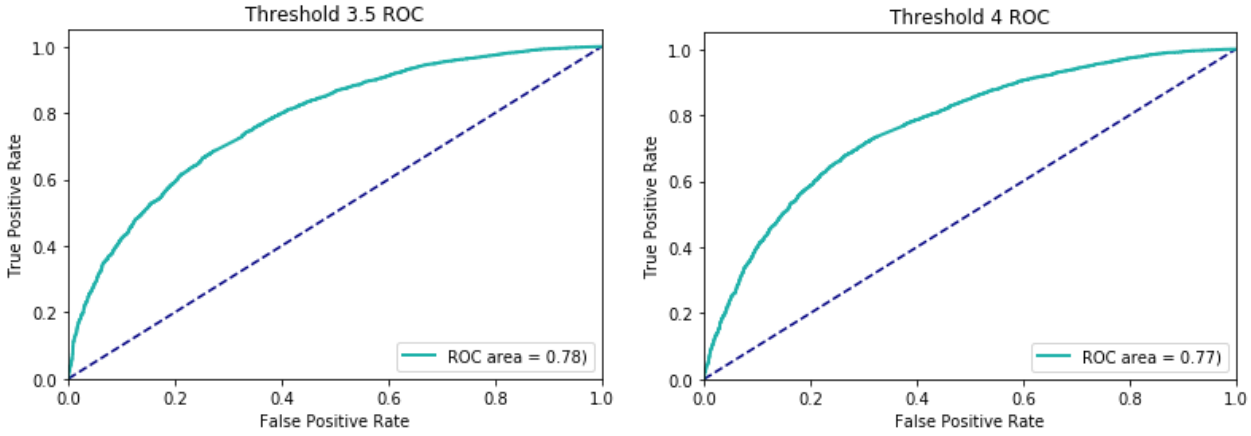


Figure 15.2 : ROC curve using threshold as 3.5 and 4

Threshold	Area
2.5	0.78
3.0	0.77
3.5	0.78
4.0	0.77

Table 15.1 : Area under ROC curve

Note : We have represented all ratings >=threshold as 1 and rest as 0.

Model-based collaborative filtering

Non-negative matrix factorization (NNMF)

For this part we do NNMF using an already written function that uses SGD from the Surprise package.

Q 16. Convexity and least square formulation

The optimization problem given by equation 5 is non convex. The hessian of this problem is not Positive Semi Definite and if the hessian is not PSD then the problem is non convex.
If we solve the optimization problem using ALS, we first keep U fixed and solve for V and the keep V fixed and solve for U and repeat the process until convergence. The question asks to formulate the problem as least squares when U is fixed. So we can look at r_{ij} and say that is the true label for our data i.e. $y_{true.}$ Also we can say that the corresponding predicted value y_{pred} is UV_{ij}^{T}. Thus the given equation changes to minimizing a least square function as belows:

$$\text{minimize}_{U,V} \sum_{i=1}^{n} \sum_{j=1}^{m} W_{ij} (y_{true} - y_{pred})^2$$

Q 17. 10-fold Cross Validation results for NMF ranging from 2 to 50

Here we design a NNMF-based collaborative filter to predict the ratings of the movies in the MovieLens dataset and evaluate its performance.

The following curves show average MAE and RMSE for different k from 2 to 50:

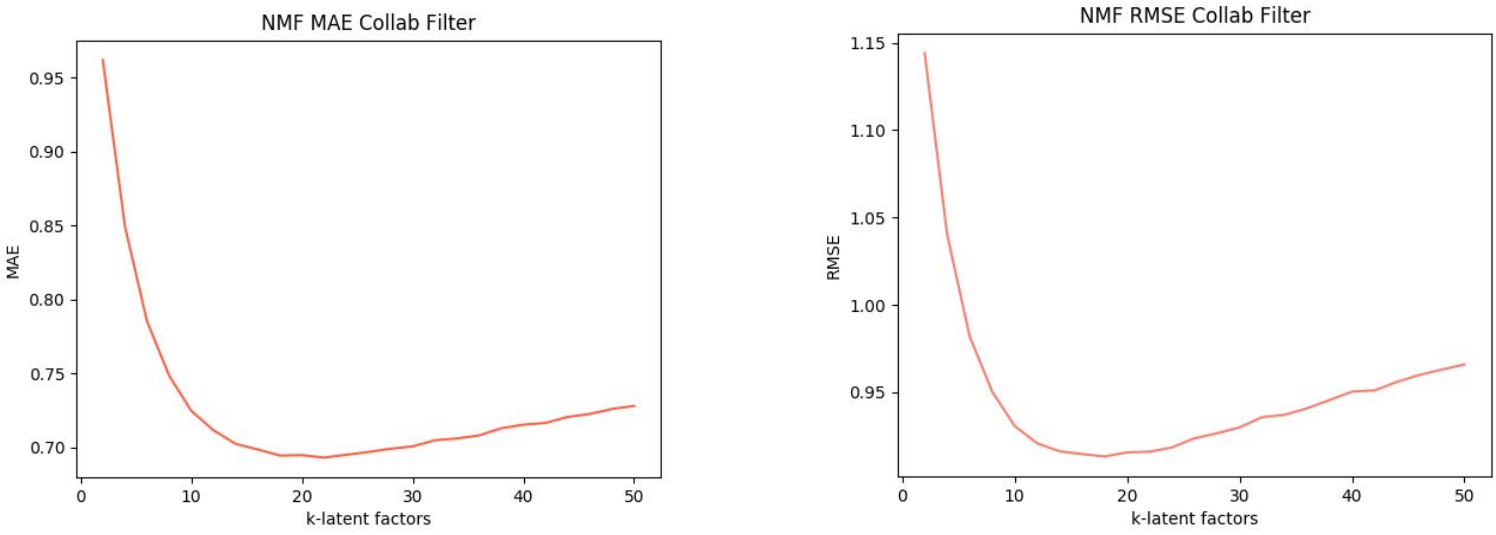


Figure 17.1 : Average RMSE vs k and Average MAE vs k

Q 18. Optimal K

RMSE has a smallest of 0.9131 value at k = 18, while MAE has minimum of 0.6935 at k=20. The number of movie genres is 18. If we take k to be minimum of RMSE we get 18 equal to number of movie genres and if we take it to be minimum of MAE we get k = 20 not equal to number of movie genres but not very different from it.

Q 19. NMF for Popular Movie trimmed test set

The average minimum RMSE is 0.8927340045898283.

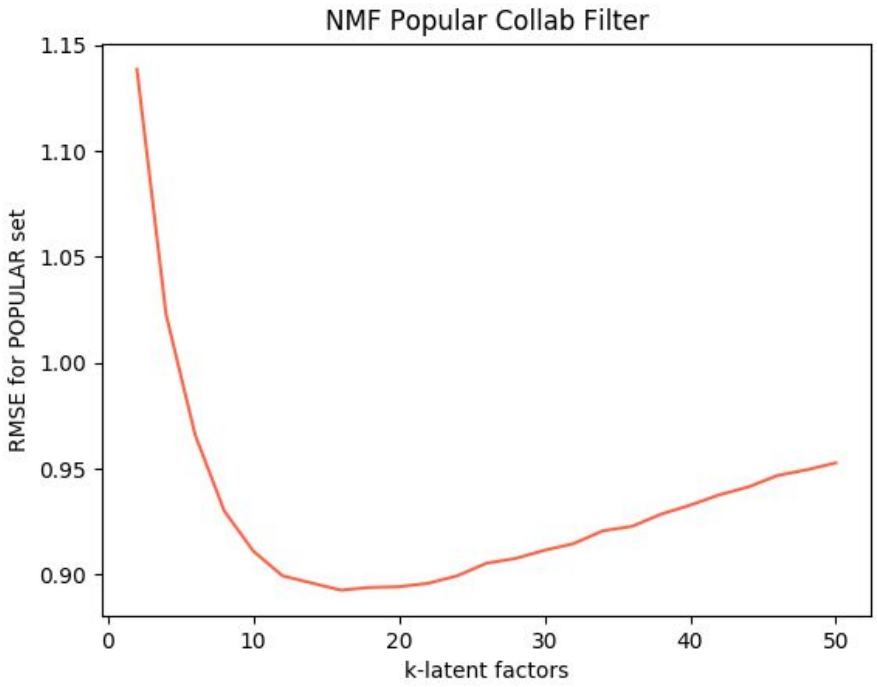


Figure 19.1 : Average RMSE vs K values for popular movie trimmed set

Q 20. NMF for Unpopular Movie trimmed test set

The average minimum RMSE is 1.1711530968098125.

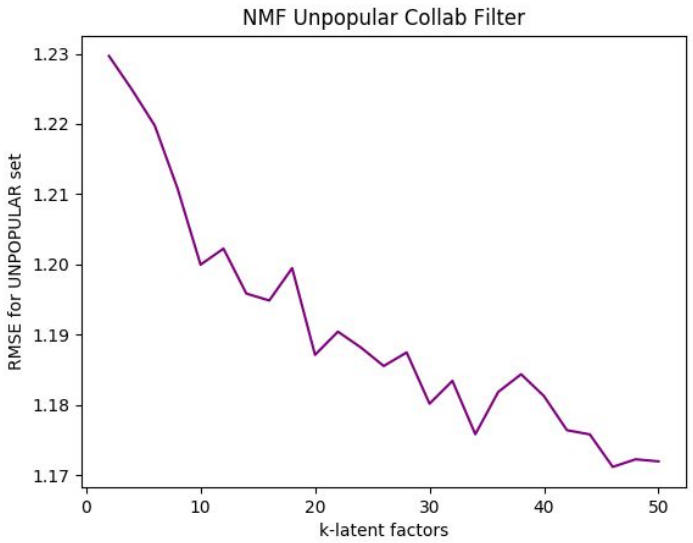


Figure 20.1 : Average RMSE vs K values for popular movie trimmed set

Q 21. NMF for High Variance Movie trimmed test set

The average minimum RMSE is 1.5735100198188499.

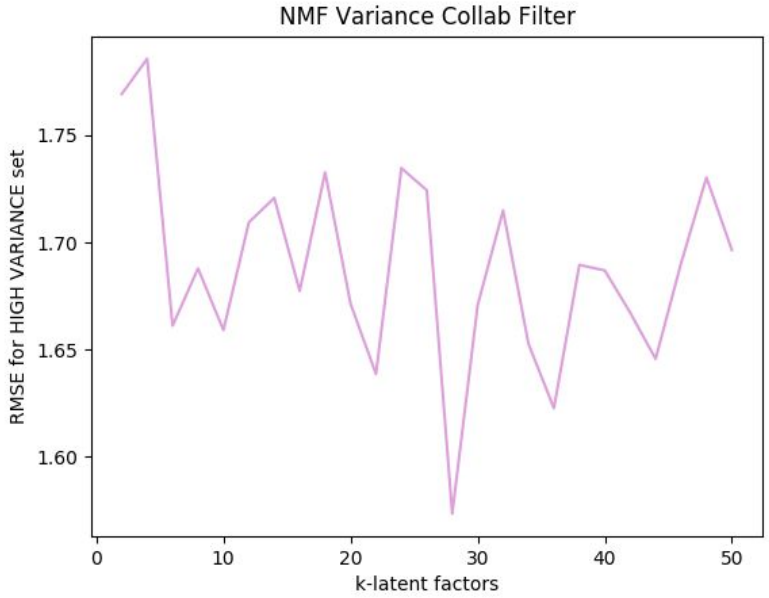


Figure 21.1 : Average RMSE vs K values for popular movie trimmed set

Q 22. ROC curves and AUC

We will use k = 18 (the optimal value using RMSE) to plot ROC and get AUC as follows:

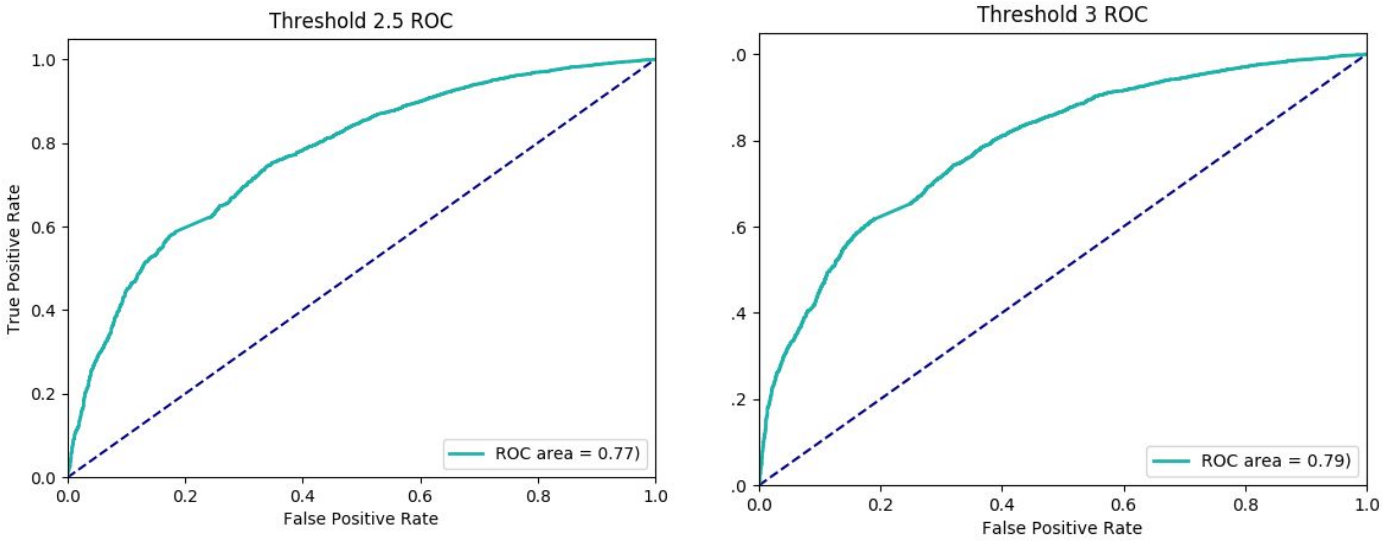


Figure 22.1 : ROC curve using threshold as 2.5 and 3

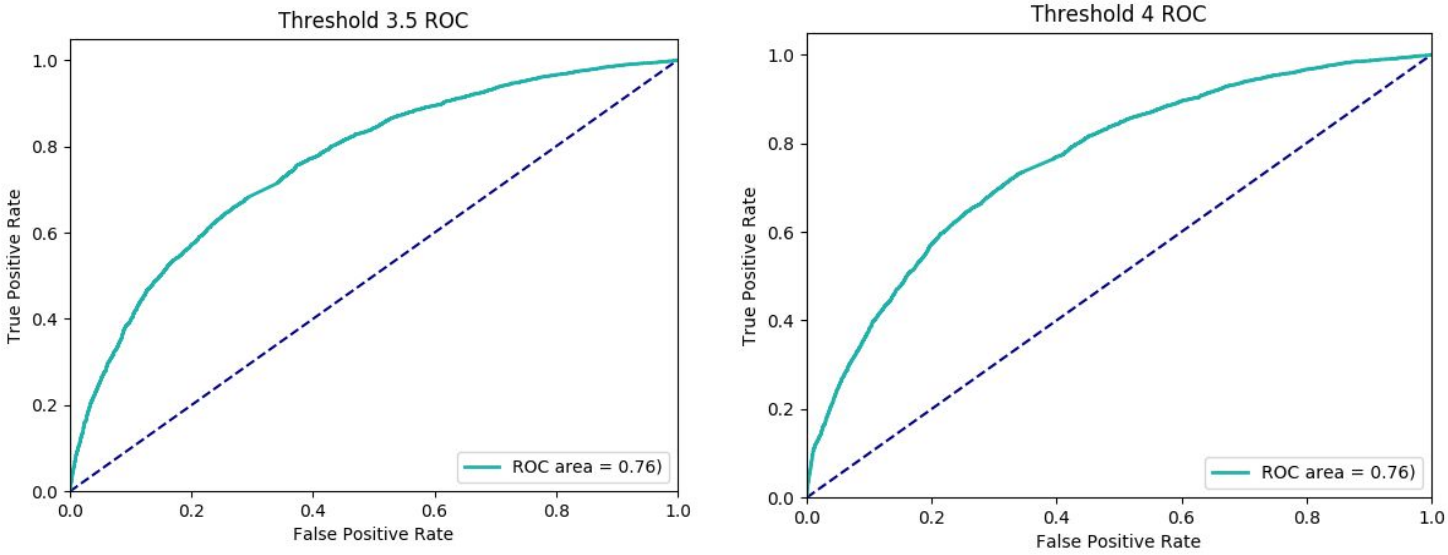


Figure 22.2 : ROC curve using threshold as 3.5 and 4

Threshold	Area
2.5	0.77
3.0	0.79
3.5	0.76
4.0	0.76

Table 22.1 : Area under ROC curve

Q 23. Interpretability of NNMF

Here we perform Non-negative matrix factorization on the ratings matrix R to obtain the factor matrices U and V , where U represents the user-latent factors interaction and V represents the movie-latent factors interaction (use k = 20). The genres of the top 10 movies are as follows:

The top 10 movies for column zero mostly has Comedy and Drama genres. Similarly sixth column has Comedy, Romance and Drama. 10th column has Comedy, Mystery and Thriller 16th column has Horror, Romance and Thriller. So we see that some set of genres keep repeating within one column. We can also say that if we see each set of genres as a feature then each latent factor represent each feature. Also, we see that for higher column numbers, the total number of genres is less than the initial columns.

0	1	2	3
Romance|Western Musical Comedy Comedy|Drama Comedy|Drama Action|Crime|Drama Comedy|Sci-Fi Action|Crime Thriller Comedy|Horror|Thriller	Drama Crime|Drama|Sci-Fi|Thriller Horror Adventure|Drama|Romance|War Comedy|Crime Comedy|Drama Action|Adventure|Sci-Fi Drama Action|Adventure|Comedy|Thriller Drama|Horror|Thriller	Animation|Children|Fantasy|Musical Comedy Action|Sci-Fi|Thriller Action|Comedy|Musical Romance|Thriller Drama Musical Comedy|Sci-Fi|Thriller Horror|Thriller Comedy|Drama|Romance	Drama|Romance Adventure|Animation|Children|Comedy|Fantasy Action Comedy|Horror Drama Drama Documentary Drama Comedy Action|Sci-Fi|Thriller

6	8	10	12
Comedy|Drama|Sci-Fi Comedy Comedy|Romance Comedy|Romance Comedy|Drama Drama|Thriller Comedy Comedy|Drama Drama Comedy	Comedy Comedy|Drama|Fantasy Adventure|Children|Comedy Documentary Documentary Drama Comedy|Drama Horror Animation|Drama Drama|Mystery|Romance	Drama|Mystery|Thriller Comedy Comedy|Drama Comedy Documentary Drama|Mystery|Thriller Comedy Action|Adventure|Drama|Romance|Thriller|Western Crime|Drama|Film-Noir|Thriller Drama	Comedy|Drama|Fantasy Comedy|Horror|Thriller Action|Sci-Fi Horror|Thriller Comedy|Crime|Drama|Romance|Thriller Crime|Mystery|Thriller Action|Mystery|Thriller Comedy|Romance Thriller Horror|Thriller
14	16	18	19
Action|Crime|Drama Drama Drama Action|Comedy|Romance|War Documentary Action|Romance|Thriller Children|Comedy Comedy|Romance Action|Horror|Sci-Fi Drama|Romance	Comedy|Crime|Drama|Romance|Thriller Documentary Action|Adventure|Thriller Horror Comedy Horror Adventure|Animation|Children|Fantasy|Musical Horror|Thriller Comedy|Drama|Romance Drama|Romance	Drama|Romance Action Comedy Comedy Drama|Mystery|Thriller Drama Comedy|War Drama Action|Animation|Sci-Fi Action|Comedy|Horror|Sci-Fi	Comedy|Romance Comedy|Crime Horror Drama|War Adventure|Romance Comedy|Drama|Romance Drama Drama|War Comedy Comedy|Fantasy|Romance

Matrix Factorization with bias (MF with bias)

Q. 24 Designing and testing MF with bias using cross-validation

To perform matrix factorization with bias, we add a bias term for each user and item. The concept here, is to integrate out the effect of the cumulative bias which could be added by users (for example, say that a person A tends to rate all movies a bit harshly, while another person B tends to rate all movies leniently). Since the bias get added to our optimization formula, the algorithm will try to make up for that bias. For users who rate harshly, a positive value will be added to their ratings, and for users who rate too leniently, a negative bias will be added to their ratings. Note - these biases are parameters of the model which shall be learnt.

We have designed a MF with bias collaborative filter using k-fold cross validation. K acts a hyperparameter here, and we shall obtain the value of K for which the average RMSE (root mean squared error) on the training set is minimum. We have also obtained the average MAE. (mean absolute error).

Here is the plot for average RMSE vs K -

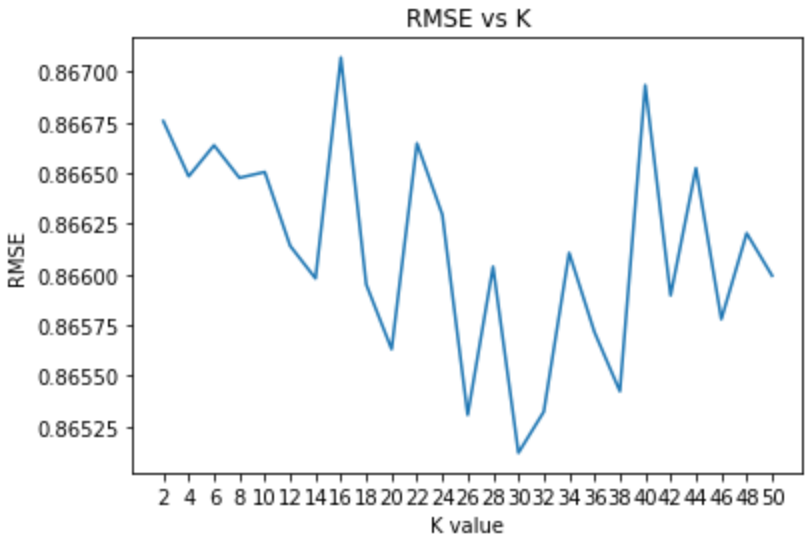


Figure 24.1 : RMSE vs K (Number of latent factors)

Here is the plot for average MAE vs K -

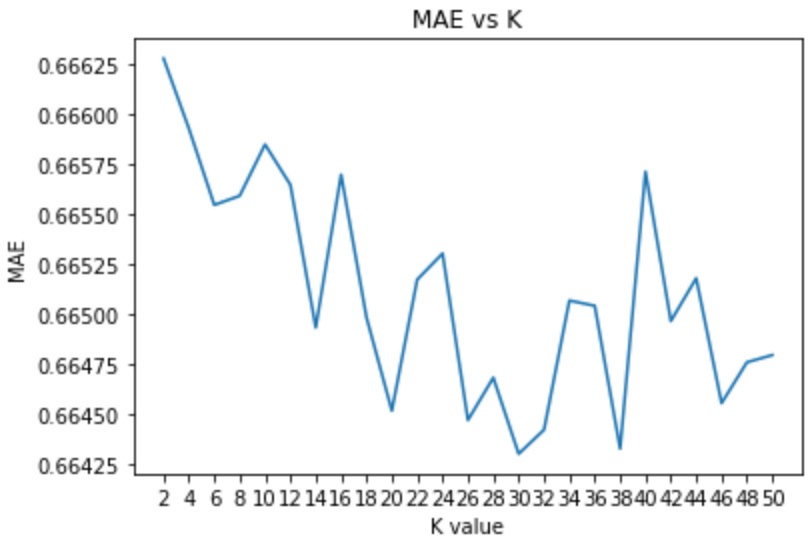


Figure 24.2 : MAE vs K (Number of latent factors)

Q. 25 Obtaining optimal number of latent factors

Using the plots above, we obtain that the optimal number of latent factors K that gives the minimum average RMSE is - **30**.
We also obtain that the optimal K which gives us the minimum average MAE is - **30**.

The minimum average RMSE obtained is - **0.865119905741753**
The minimum average MAE obtained is - **0.6643008352687779**

Q. 26 MF with bias performance on popular movies test set

On using the MF with bias collaborative filter via K-fold cross validation (where K - number of latent factors, ranged from 2 to 50 in steps of 2) on the popular movie test set, the following plot for RMSE vs K was obtained.

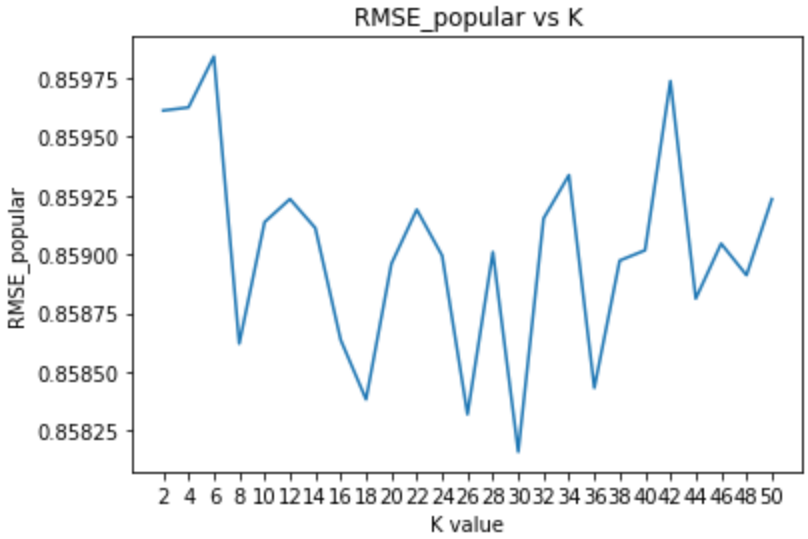


Figure 26.1 : RMSE vs K (Number of latent factors) for popular movies test set

We can see that the optimal K is **30.**
Also, the minimum average RMSE obtained is - **0.8581599503628057**

Q. 27 MF with bias performance on unpopular movies test set

On using the MF with bias collaborative filter via K-fold cross validation (where K ranged from 2 to 50 in steps of 2) on the unpopular movie test set, the following plot for RMSE vs K was obtained.

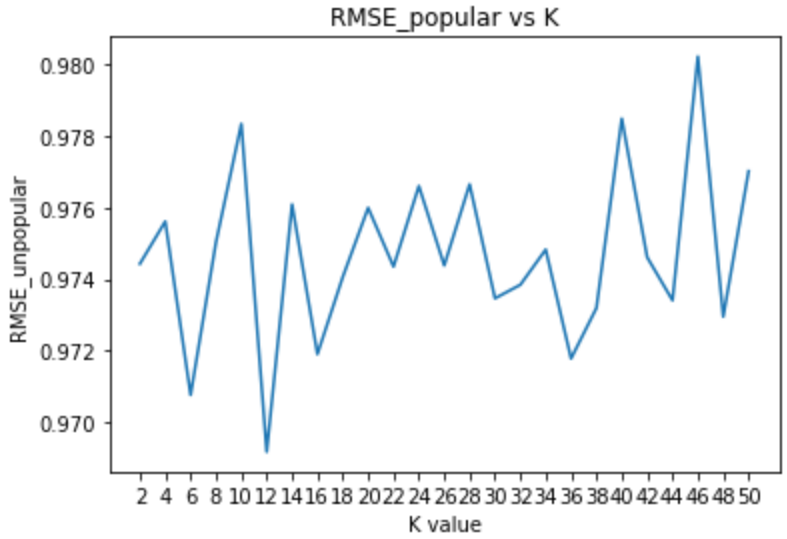


Figure 27.1 : RMSE vs K (Number of latent factors) for unpopular movies test set

We can see that the optimal K is **12.**
Also, the minimum average RMSE obtained is - **0.9691853711668379**

Q. 28 MF with bias performance on high variance movies test set

On using the MF with bias collaborative filter via K-fold cross validation (where K ranged from 2 to 50 in steps of 2) on the high variance movie test set, the following plot for RMSE vs K was obtained.

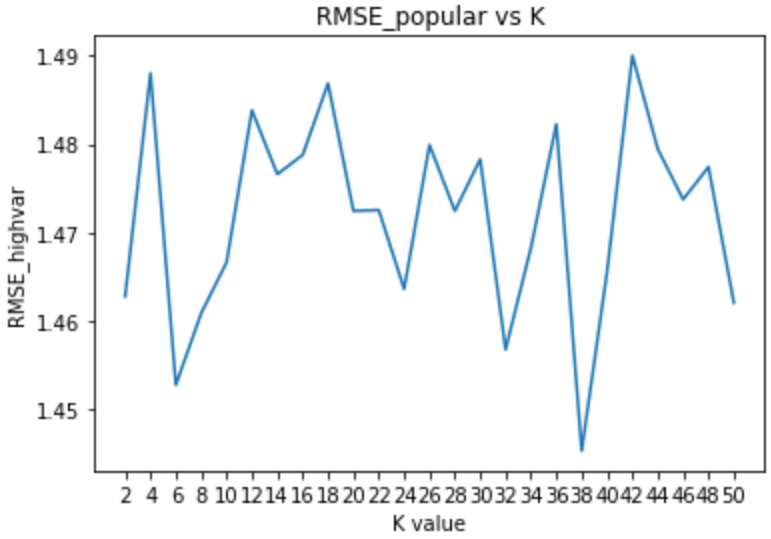


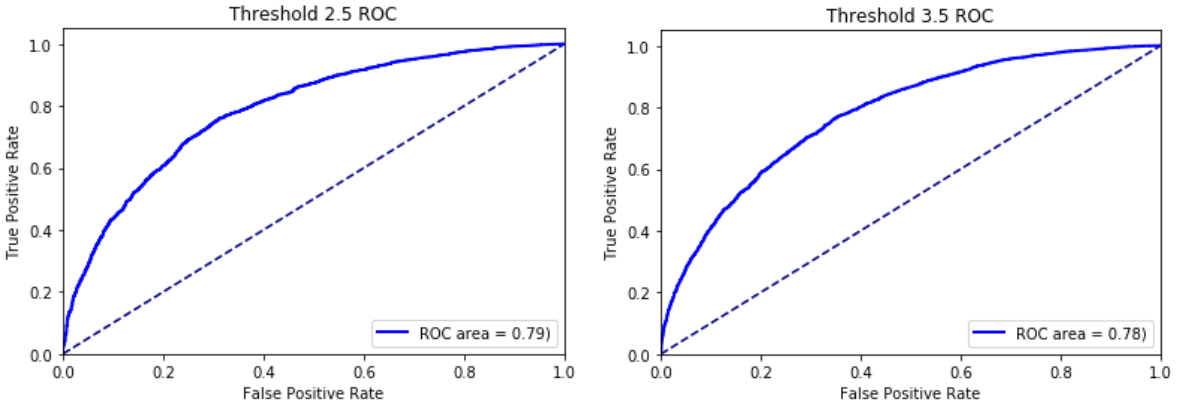
Figure 28.1 : RMSE vs K (Number of latent factors) for high variance movies test set

We can see that the optimal K is **38.**
Also, the minimum average RMSE obtained is - **1.4453300056512584**

We notice that the RMSE is least for the popular movie set, since we can expect that most users would rate these movies similarly, whereas in case of unpopular or movies with high variance of ratings, we can expect that many dissimilarities between user ratings exist.

Q. 29 ROC curves and AUC for different threshold values

We will use k = 30 (the optimal value using RMSE) to plot ROC and get AUC as follows:



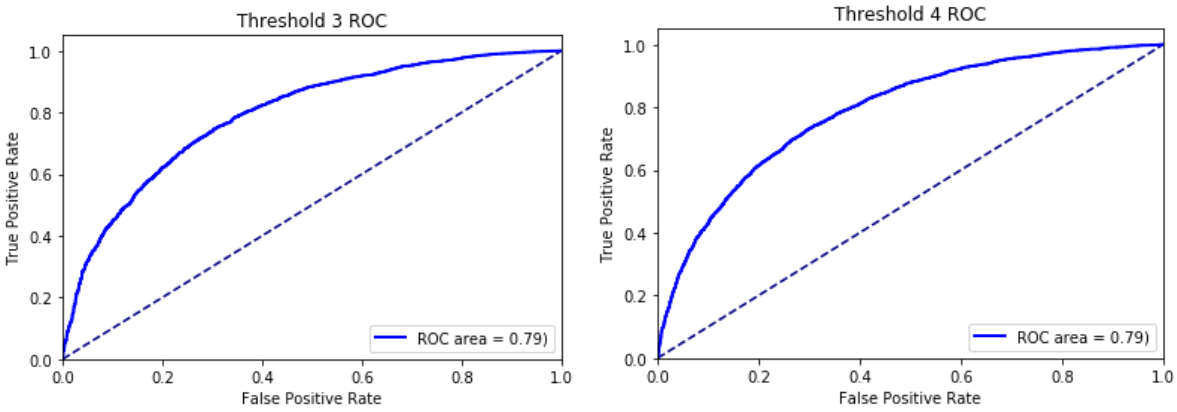


Figure 29.1 : ROC curves for threshold values (2.5,3,3.5,4)

Threshold	Area
2.5	0.79
3.0	0.79
3.5	0.78
4.0	0.79

Table 29.1 : Area under ROC Curve

Naive collaborative filtering

Q. 30 Designing and testing naive collaborative filter

This filter returns the mean rating of the user as the predicted rating for an item, in this case movie. We have designed a naive collaborative filter using k-fold cross validation. Here we have used 10-fold cross validation. We have obtained the average RMSE.
The average RMSE obtained is - **0.9412176872887089**

Q. 31 Naive collaborative filter performance on popular movie test set

We have designed a naive collaborative filter using 10-fold cross validation and tested it on the popular movie test set.
The average RMSE obtained is - **0.9387869947079297**

Q. 32 Naive collaborative filter performance on unpopular movies test set

We have designed a naive collaborative filter using 10-fold cross validation and tested it on the unpopular movie test set.
The average RMSE obtained is - **0.9749625048537409**

Q. 33 Naive collaborative filter performance on high variance movies test set

We have designed a naive collaborative filter using 10-fold cross validation and tested it on the high variance movie test set.
The average RMSE obtained is - **1.454694701198991**

We observe that the naive collaborative filter has performed more poorly than our other filters. This stems from the fact that we do not use a lot of information to perform a prediction. We only use a user's average rating and hence the filter is unable to include ratings from other users.

Performance Comparison

Q 34. Comparing k-NN, NNMF, and MF with bias based collaborative filters

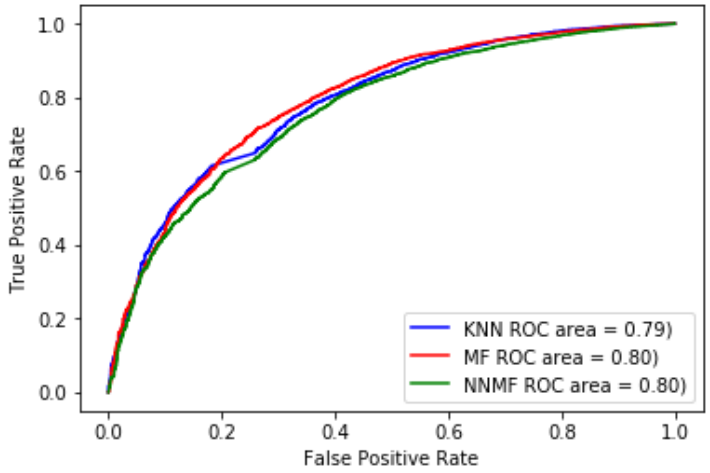


Figure 34.1 : ROC Curve Comparing kNN, NNMF and MF with Bias

We see that NNMF and MF with bias have same ROC area under the curve, and hence, similar accuracy at predicting ratings. However, since the MF with bias curve looks smoother and seems to have more area under the curve, it will be the best method. KNN shows the least accuracy.

Ranking Predictions

Q 35. Precision and Recall

Given the equations 12 and 13, we see:

Precision is the proportion of the items recommended that are relevant. In other words, proportion of the recommended items liked by user.

Recall is the ratio of items recommended and liked by user to the number of items liked by user. In other words, recall checks how well we can recommend items that are liked by a user.

Q 36. k-NN collaborative filter predictions

As can be seen in figure 36.1 below, we get precision consistently decreasing for increasing values of t best rated movies. The precision consistently decreases from 0.92 to 0.88 as recall increases as can be seen in the zoomed- in version as well.

Similarly for recall against t best movies continuously increases and ranges from approximately 0.20 to 0.60.

Recall vs precision is as expected as it is decreasing slightly with different thresholds. Area under curve is quite good as it is giving good output results.

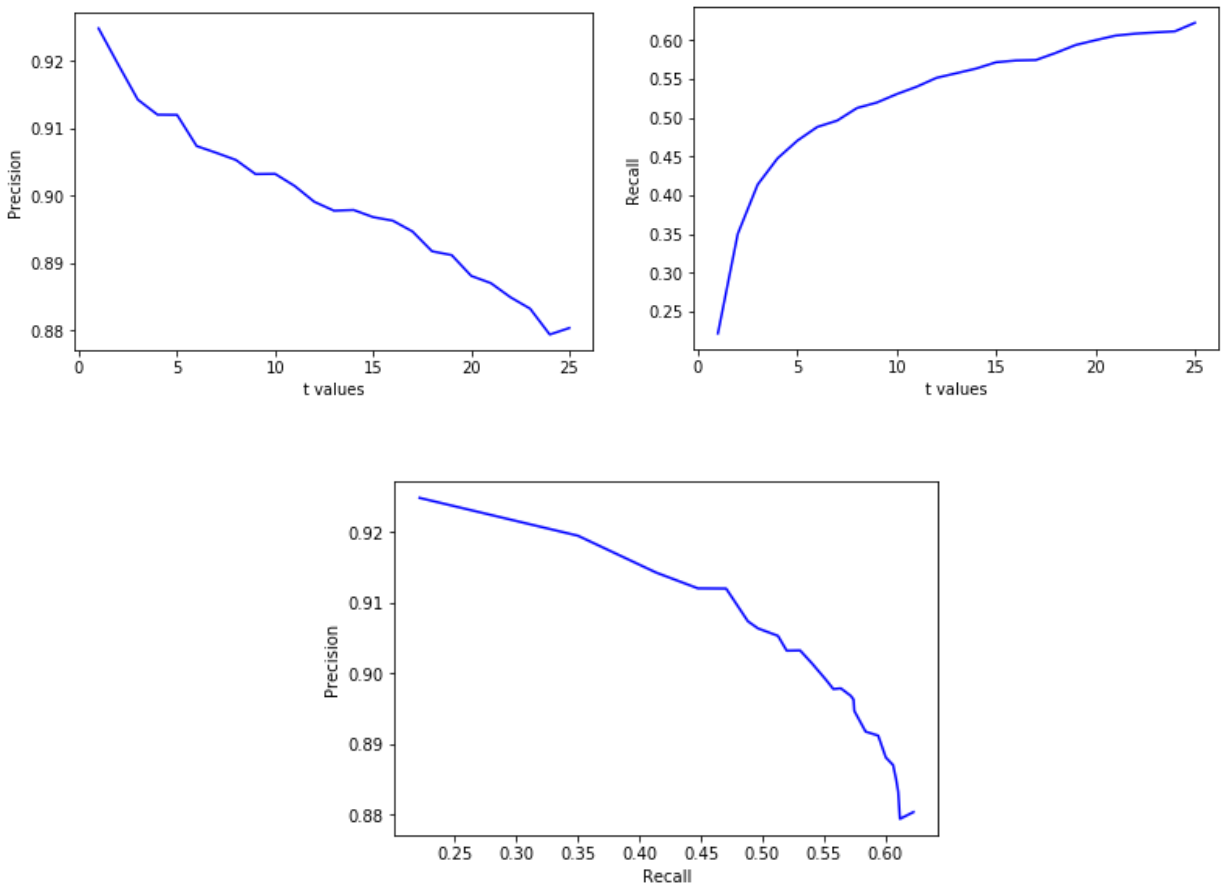


Figure 36.1 : The plots correspond to precision vs t, recall vs t and recall vs precision respectively for kNN

Q 37. NNMF predictions

As can be seen in figure 36.1 below, we get precision consistently decreasing for increasing values of t best rated movies. The precision consistently decreases from 0.93 to 0.88 as recall increases as can be seen in the zoomed- in version as well.

Similarly for recall against t best movies continuously increases and ranges from approximately 0.21 to 0.60.

Recall vs precision is as expected as it is decreasing slightly with different thresholds. Area under curve is quite good as it is giving good output results.

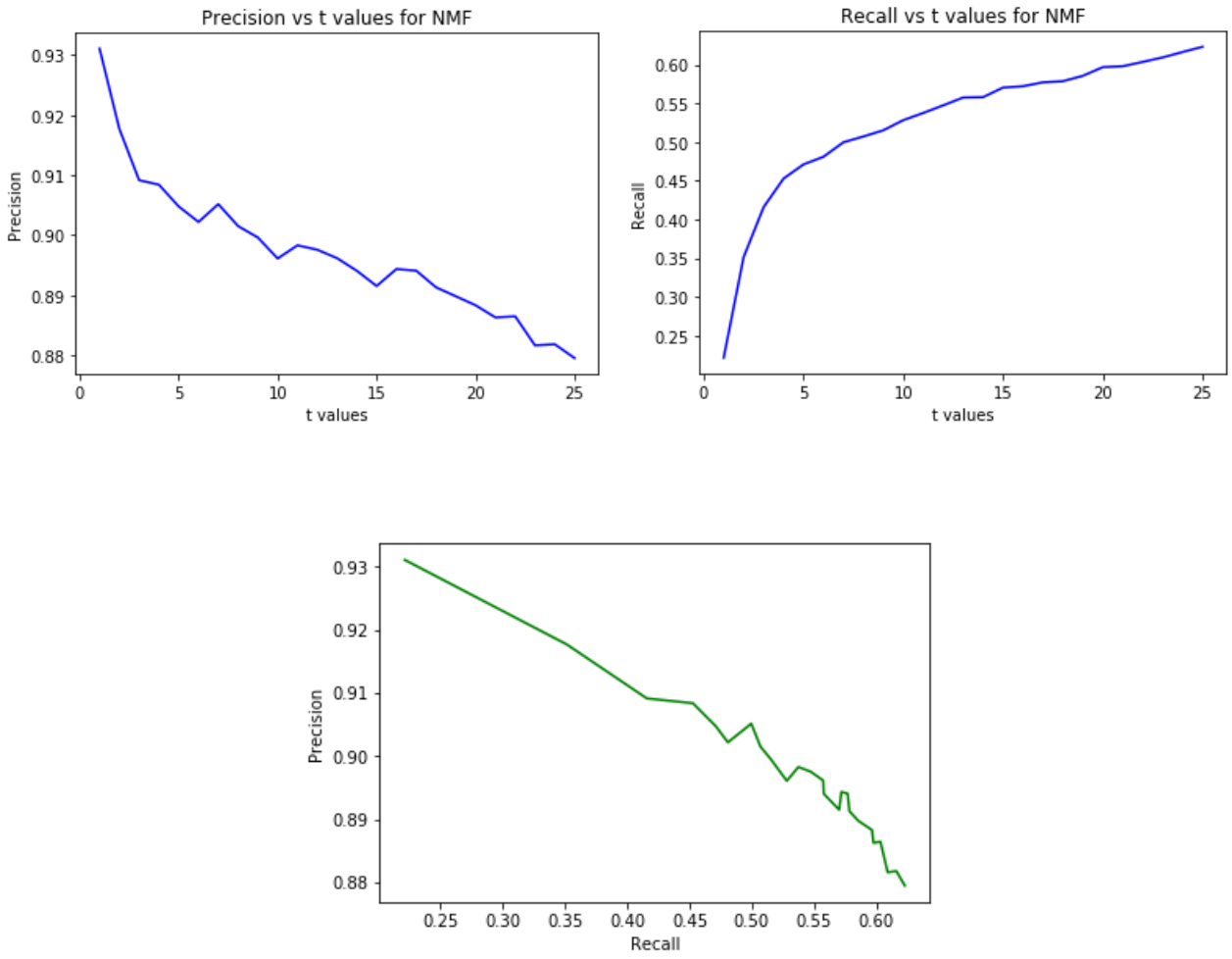


Figure 37.1 : The plots correspond to precision vs t, recall vs t and recall vs precision respectively for NNMF

Q 38. MF with bias predictions

As can be seen in figure 36.1 below, we get precision consistently decreasing for increasing values of t best rated movies. The precision consistently decreases from 0.94 to 0.89 as recall increases as can be seen in the zoomed- in version as well.

Similarly for recall against t best movies continuously increases and ranges from approximately 0.23 to 0.60.

Recall vs precision is as expected as it is decreasing slightly with different thresholds. Area under curve is quite good as it is giving good output results.

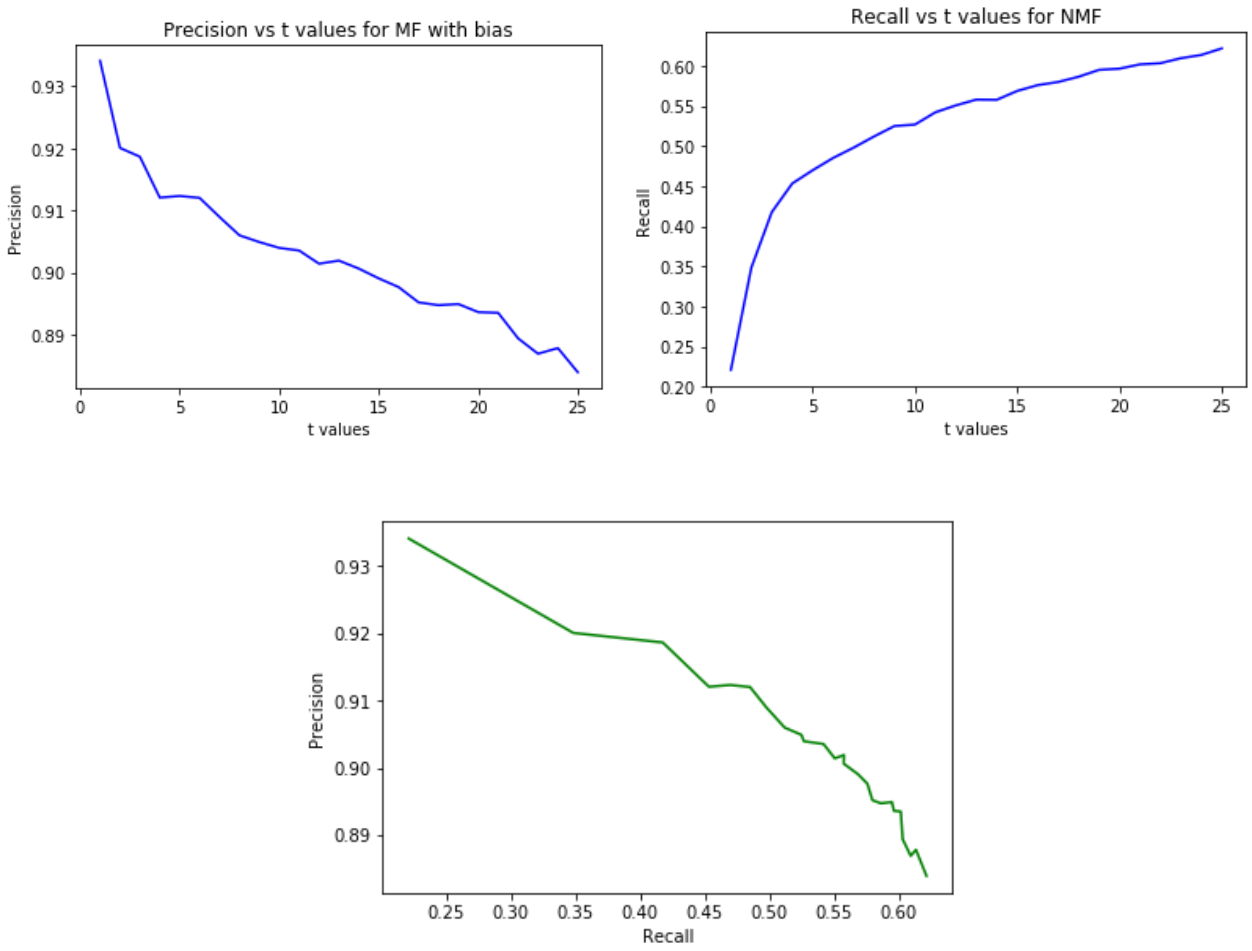


Figure 38.1 : The plots correspond to precision vs t, recall vs t and recall vs precision respectively for MF with bias

Q 39. Precision Recall curve for all combined

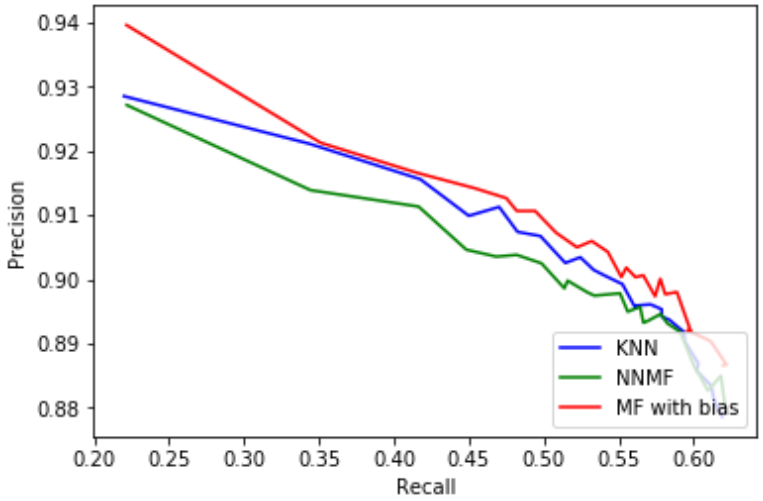


Figure 39.1 : ROC Curve Comparing kNN, NNMF and MF with Bias

We see the best results are for MF with bias (red curve in Figure 39.1). The precision recall curve shows the tradeoff for different thresholds. We get high areas under the curve for all filtering methods, but MF with bias gives best results. As explained in previous sections, the result is consistent with previous graphs with MF being the best method.