**EE219 PROJECT\_3 REPORT**

**COLLABORATIVE FILTERING**

1. **Introduction**

The increasing importance of the web as a medium for electronic and business transactions has served as a driving force for the development of recommender systems technology. An important catalyst in this regard is the ease with which the web enables users to provide feedback about their likes or dislikes. The basic idea of recommender systems is to utilize these user data to infer customer interests. The entity to which the recommendation is provided is referred to as the user, and the product being recommended is referred to as an item.­

The basic models for recommender systems works with two kinds of data:

1. User-Item interactions such as ratings

2. Attribute information about the users and items such as textual profiles or relevant keywords

Models that use type 1 data are referred to as collaborative filtering methods, whereas models that use type 2 data are referred to as content-based methods. In this project, we will build recommendation system using collaborative filtering methods.

1. **Collaborative filtering models**

Collaborative filtering models use the collaborative power of the ratings provided by multiple users to make recommendations. The main challenge in designing collaborative filtering methods is that the underlying ratings matrices are sparse. Consider an example of a movie application in which users specify ratings indicating their like or dislike of specific movies. Most users would have viewed only a small fraction of the large universe of available movies and as a result most of the ratings are unspecified.

The basic idea of collaborative filtering methods is that these unspecified ratings can be imputed because the observed ratings are often highly correlated across various users and items. For example, consider two users named John and Molly, who have very similar tastes. If the ratings, which both have specified, are very similar, then their similarity can be identified by the filtering algorithm. In such cases, it is very likely that the ratings in which only one of them has specified a value, are also similar. This similarity can be used to make inferences about incompletely specified values. Most of the collaborative filtering methods focus on leveraging either inter-item correlations or inter-user correlations for the prediction process.

In this project, we will implement and analyze the performance of two types of collaborative filtering methods:

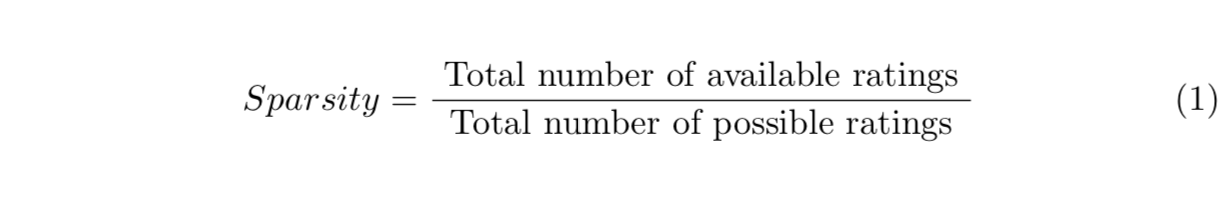
1. Neighborhood-based collaborative filtering

2. Model-based collaborative filtering.

1. **MovieLens dataset**

In this project, we will build a recommendation system to predict the ratings of the movies in the MovieLens dataset. Although the dataset contains movie genre information, we will only use the movie rating information in this project. For the subsequent discussion, we assume that the ratings matrix is denoted by R, and it is an m × n matrix containing m users (rows) and n movies (columns). The (i,j) entry of the matrix is the rating of user ‘i’ for movie ‘j’ and is denoted by ‘rij’. Before moving on to the collaborative filter implementation, we will analyze and visualize some properties of this dataset.

Question 1: Compute the sparsity of the movie rating dataset, where sparsity is defined by equation 1



The sparsity was calculated from the code as:

user\_count = len(ratings\_data['userId'].unique())

movie\_count = len(ratings\_data['movieId'].unique())

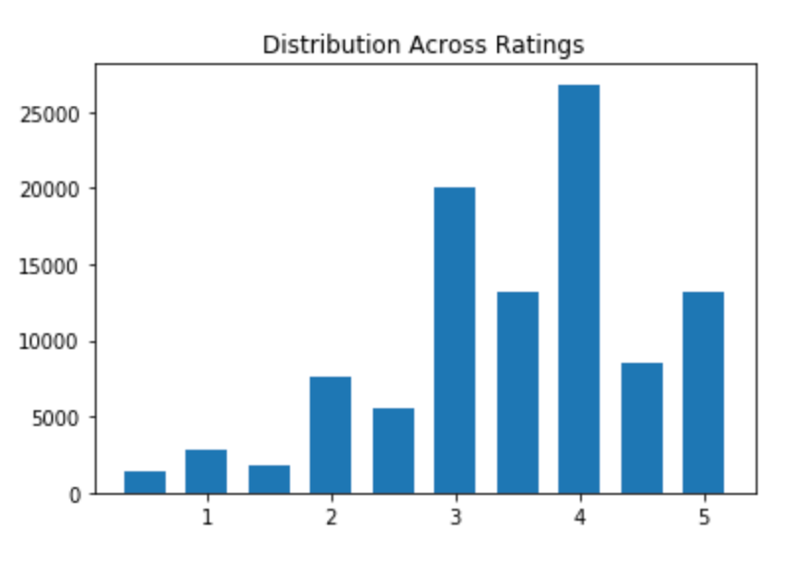
ratings\_count = len(ratings\_data)

sparsity = ratings\_count/(user\_count\*movie\_count)

**sparsity: 0.016999683055613623**

Question 2: Plot a histogram showing the frequency of the rating values. To be specific, bin the rating values into intervals of width 0.5 and use the binned rating values as the horizontal axis. Count the number of entries in the ratings matrix R with rating values in the binned intervals and use this count as the vertical axis. Briefly comment on the shape of the histogram

The graph of the number of ratings received for each rating looks like a negatively skewed gaussian curve. The mean of the curve seems to be between 3.0 and 4.0.



Empirically this seems to be related to the Central Limit Theorem. In probability theory the Central Limit Theorem (CLT) establishes that, in some situations, when independent random variables are added, their properly normalized sum tends toward a normal distribution “informally a bell curve” even if the original variables themselves are not normally distributed. (From Wikipedia)

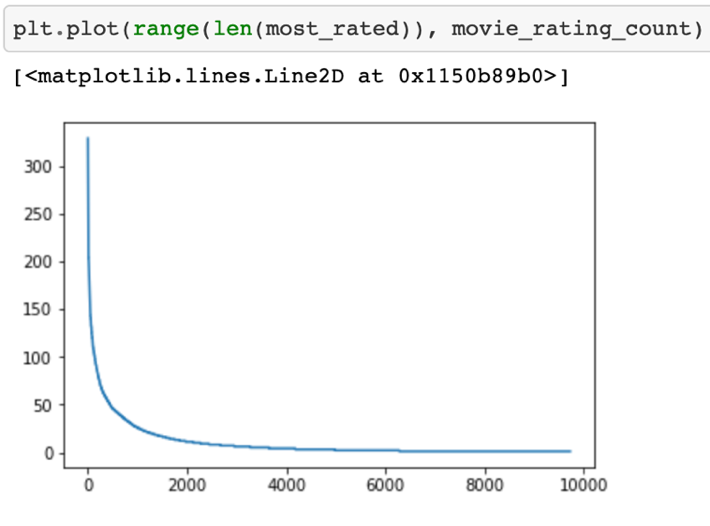
We calculated the mean of the data and again found it to be consistent with what has been empirically determined from prior distributions.

mean = np.sum(ratings\_list\*ratings\_list\_count)/np.sum(ratings\_list\_count)

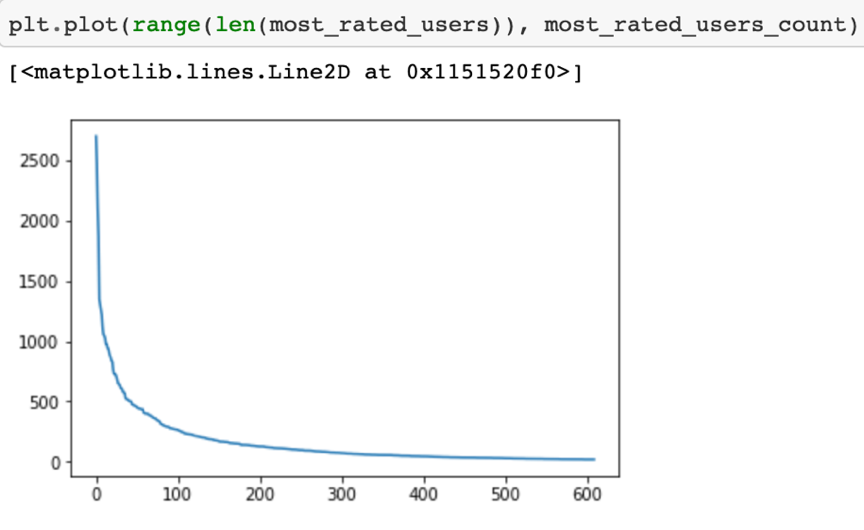
print(mean)

**mean = 3.501556983616962**

Question 3: Plot the distribution of the number of ratings received among movies. To be specific, the X-axis should be the movie index ordered by decreasing frequency and the Y-axis should be the number of ratings the movie has received. For example, the movie that has the largest number of ratings has index 1; ties can be broken in any way. A monotonically decreasing curve instead of a histogram is expected.



Question 4: Plot the distribution of ratings among users. To be specific, the X-axis should be the user index ordered by decreasing frequency and the Y-axis should be the number of movies the user have rated. The requirement of the plot is similar to that in Question 3.

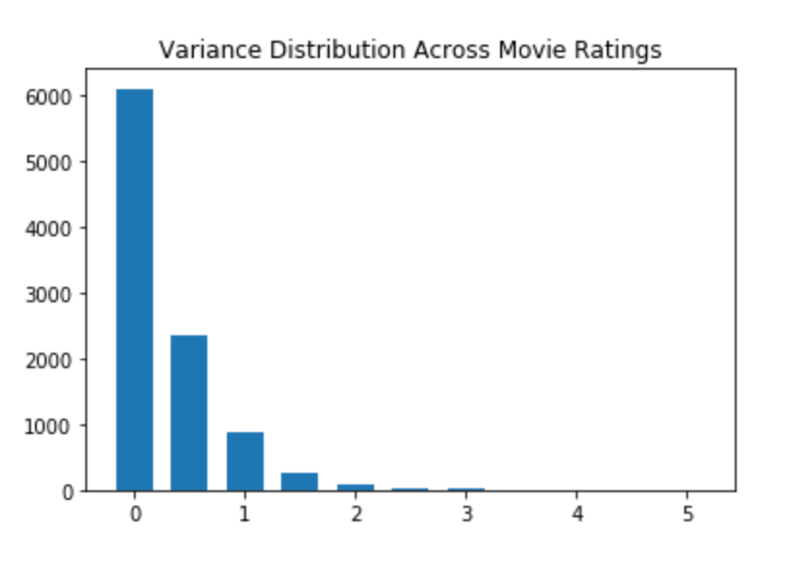


Question 5: Explain the salient features of the distribution found in question 3 and their implications for the recommendation process.

From the graph in Question 3 we can see that of the total number of ratings received by 9724 movies decreases exponentially. The highest number of ratings received by a single movie is 329 (there are 610 users). There are only 3 movies out of 9724 movies that have received 300+ #ratings. 18 have #ratings above 200, 134 have #ratings above 100 and 2121 have #ratings above 10. We can clearly observe the exponentially decreasing trend here. As many as 7603 movies have 10 #ratings or lesser. This is a real concern for the ratings prediction using collaborative filtering techniques.

We are using collaborative filtering models which use the collaborative power of the ratings provided by multiple users to make recommendations. As mentioned previously the main challenge in designing collaborative filtering methods is that the underlying ratings matrices are sparse. Consider an example of a movie application in which users specify ratings indicating their like or dislike of specific movies. Most users would have viewed only a small fraction of the large universe of available movies and as a result most of the ratings are unspecified. (Which is what we observe in this graph too). The basic idea of collaborative filtering methods is that these unspecified ratings can be imputed because the observed ratings are often highly correlated across various users and items.

Question 6: Compute the variance of the rating values received by each movie. Then, bin the variance values into intervals of width 0.5 and use the binned variance values as the horizontal axis. Count the number of movies with variance values in the binned intervals and use this count as the vertical axis. Briefly comment on the shape of the histogram



As can be seen from the graph, most movies have very low variance in the ratings value they received. This means, that if a particular movie was really good, then most viewers gave high ratings; or if a movie was really bad then most viewers gave low ratings. This is a really good dataset for working on since we are trying to predict rating values for a new user based on previous users’ ratings for the same movie. If the variance in ratings values for movies are low, then we have a higher chance of predicting them correctly.

1. **Neighborhood-based collaborative filtering**

The basic idea in neighborhood-based methods is to use either user-user similarity or item-item similarity to make predictions from a ratings matrix. There are two basic principles used in neighborhood-based models:

1. User-based models: Similar users have similar ratings on the same item. Therefore, if John and Molly have rated movies in a similar way in the past, then one can use John’s observed ratings on the movie Terminator to predict Molly’s rating on this movie.

2. Item-based models: Similar items are rated in a similar way by the same user. Therefore, John’s ratings on similar science fiction movies like Alien and Predator can be used to predict his rating on Terminator.

In this project, we will only implement user-based collaborative filtering (implementation of item-based collaborative filtering is very similar).

* 1. User-based neighborhood models

In this approach, user-based neighborhoods are defined in order to identify similar users to the target user for whom the rating predictions are being computed. In order to determine the neighborhood of the target user u, her similarity to all the other users is computed. Therefore, a similarity function needs to be defined between the ratings specified by users. In this project, we will use Pearson-correlation coefficient to compute the similarity between users.

* 1. Pearson-correlation coefficient

Pearson-correlation coefficient between users u and v, denoted by **Pearson(u,v)**, captures the similarity between the rating vectors of users u and v. Before stating the formula for computing Pearson(u,v), let’s first introduce some notation:

Iu : Set of item indices for which ratings have been specified by user u.

Iv : Set of item indices for which ratings have been specified by user v.

μu: Mean rating for user u computed using her specified ratings.

ruk: Rating of user u for item k

Question 7: Write down the formula for μu in terms of Iu and ruk.

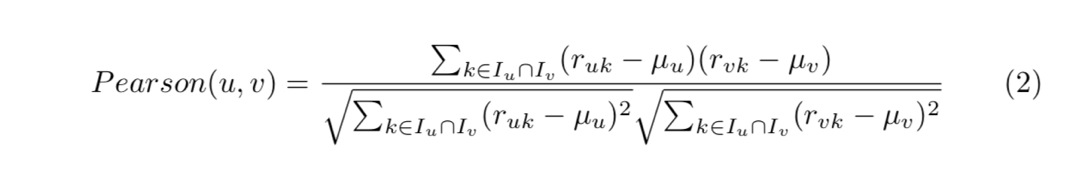
The formula is as follows:

Question 8: In plain words, explain the meaning of Iu ∩ Iv. Can Iu ∩ Iv = ∅

(Hint: Rating matrix R is sparse).

Iu ∩ Iv is the intersection between the two sets of indices of user (u and v) ratings. This can indeed by null set since the users may not have rated a single movie in common. As we can see from the previous plots there are some users who have rated a single movie out of 9724 movies (the exponentially decreasing curve for movie rating count in question 3); and there are like 610 users. Hence there will be a lot of users who have not rated any movies in common.

Then, with the above notation, the Pearson-correlation coefficient between users u and v is defined by equation 2:

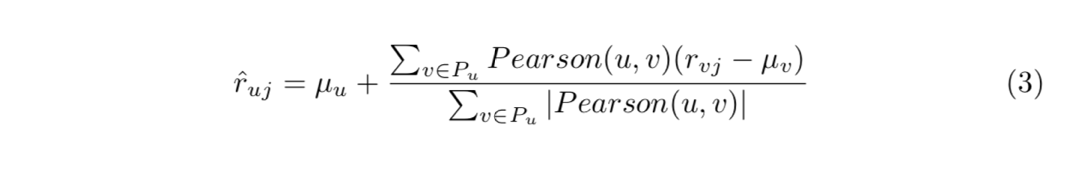


* 1. k-Nearest neighborhood (k-NN)

Having defined similarity metric between users, now we are ready to define neighborhood of users. K - Nearest neighbor of user u, denoted by Pu, is the set of k users with the highest Pearson-correlation coefficient with user u.

* 1. Prediction function

We can now define the prediction function for user-based neighborhood model. The predicted rating of user u for item j, denoted by rˆuj, is given by equation 3



Question 9: Can you explain the reason behind mean-centering the raw ratings (rvj − μv) in the prediction function? (Hint: Consider users who either rate all items highly or rate all items poorly and the impact of these users on the prediction function)

In order to handle the problem of user bias we use mean centering of data. For example, consider user ‘u’ who always gives a rating of 4 for movies that he/she find of mediocre quality. If such a user gives a rating of 5 or 1 this should be quantified properly by taking the difference of the mean rating given by the user. By quantified properly I mean that the difference in the general rating given by the user and the current rating given by the user is important than the absolute value of the rating. Similarly, we could have a user who always gives a rating of 2 for mediocre movies but if such a user gave a rating of 5 for a movie then that would have been an exceptional movie for the user. So, we try to quantify user preferences in terms of this difference and then find similar users and do movie rating predictions.

* 1. k-NN collaborative filter

We shall now implement a k-NN collaborative filter for predicting ratings of the movies.

* + 1. Design and test via cross-validation

In this part of the project, we will design a k-NN collaborative filter and test its performance via 10-fold cross validation. In a 10-fold cross-validation, the dataset is partitioned into 10 equal sized subsets. Of the 10 subsets, a single subset is retained as the validation data for testing the filter, and the remaining 9 subsets are used to train the filter. The cross-validation process is then repeated 10 times, with each of the 10-subsets used exactly once as the validation data.

Question 10: Design a k-NN collaborative filter to predict the ratings of the movies in the MovieLens dataset and evaluate its performance using 10-fold cross validation. Sweep k (number of neighbors) from 2 to 100 in step sizes of 2, and for each k compute the average RMSE and average MAE obtained by averaging the RMSE and MAE across all 10 folds. Plot average RMSE (Y-axis) against k (X-axis) and average MAE (Y-axis) against k (X-axis).

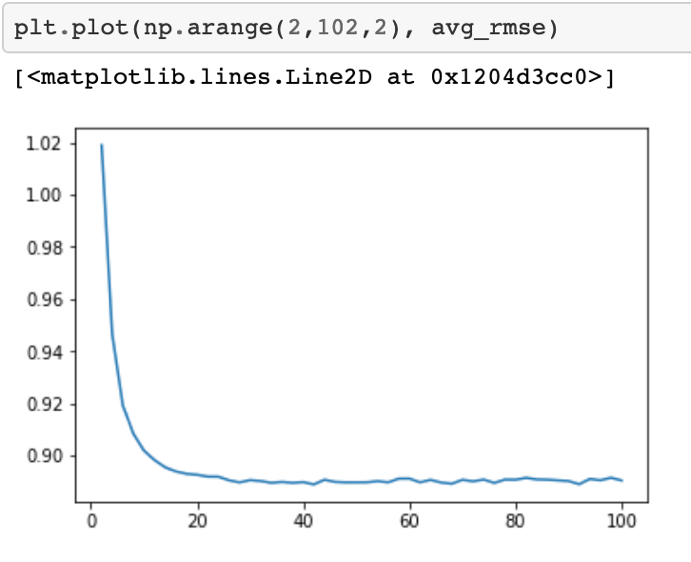


Figure 1: Average RMSE

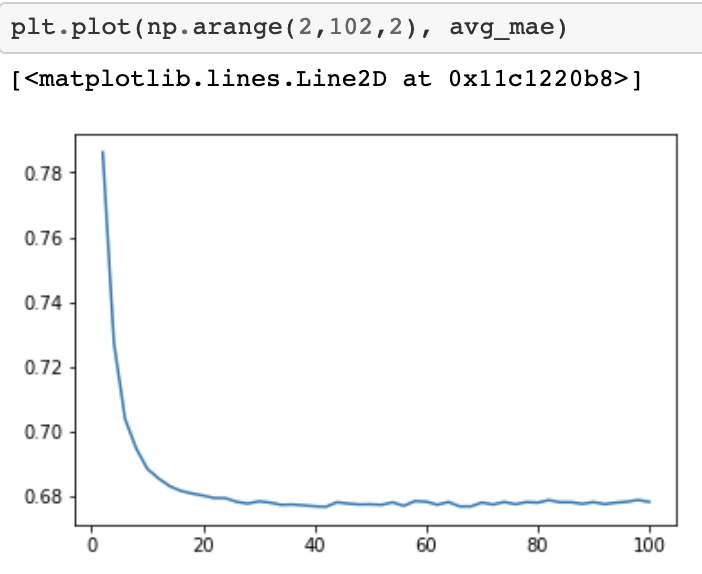


Figure 2: Average MAE

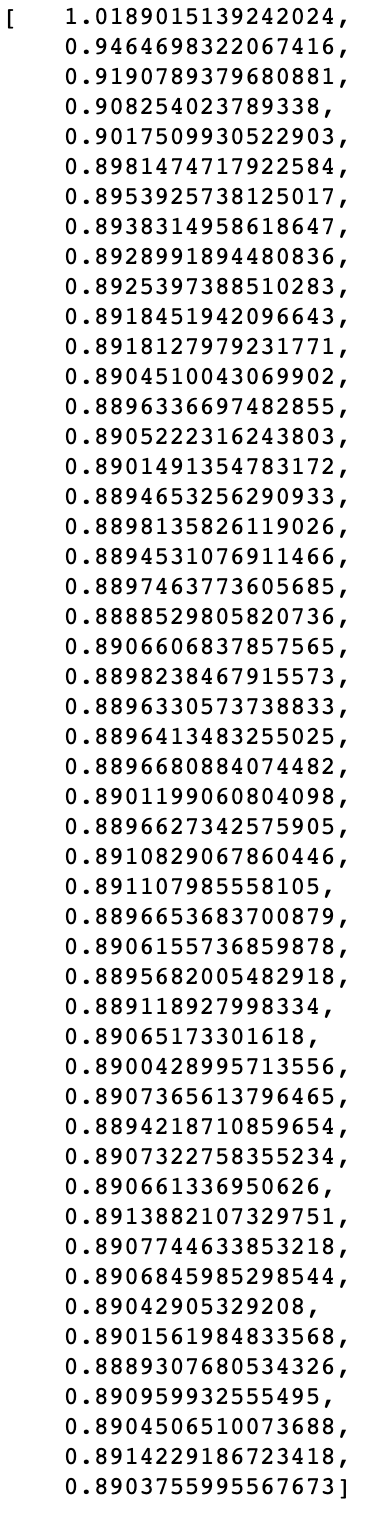
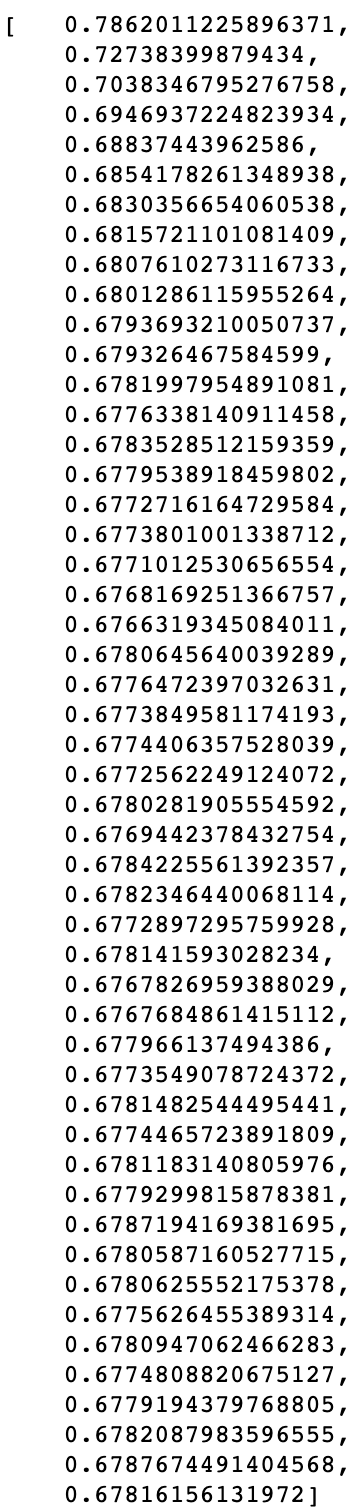


Figure 4: Average MAE

Figure 3: Average RMSE

Question 11: Use the plot from question 10, to find a ’minimum k’. Note: The term ’minimum k’ in this context means that increasing k above the minimum value would not result in a significant decrease in average RMSE or average MAE. If you get the plot correct, then ’minimum k’ would correspond to the k value for which average RMSE and average MAE converges to a steady-state value. Please report the steady state values of average RMSE and average MAE

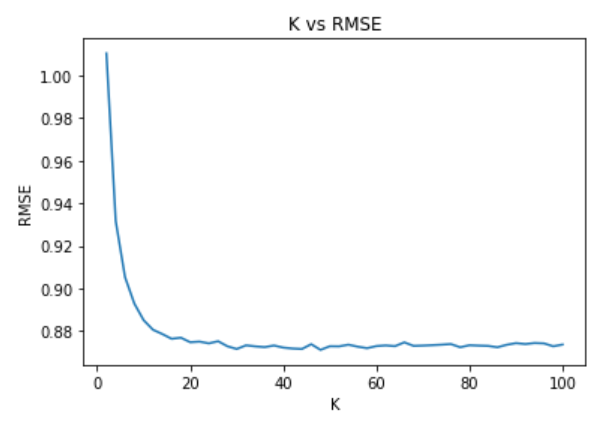
Looking at the graph and also when printing out the values, we see that most of the values are almost the same after a certain point. We chose the k value which had the minimum value of RMSE in the beginning where it just stabilizes. So, we chose **28** as the k value where the RMSE stabilizes. We believe that any values we choose won’t really matter since the RMSE values are almost the same.

In this part of the question, we perform 3 kinds of trimming operations to the test rating dataset we got by reading the data from using the surprise library.

1. Popular movie trimming: Trim the test set to contain movies that received more than 2 ratings. If a movie in the test set has received less than or equal to 2 ratings in the entire dataset, then we delete the move from the test set.
2. Unpopular movie trimming: Trim the test set to contain movies that less than or equal to 2 ratings. If a movie in the test set has received more than 2 ratings in the entire dataset, then we delete the movie from the test set.
3. High variance movie trimming: Trim the test set to contain movies that have variance of at least 2 and have received at least 5 ratings. If a movie in the test set has variance less than 2 have received less than 5 ratings, then we delete the movie from the test set.

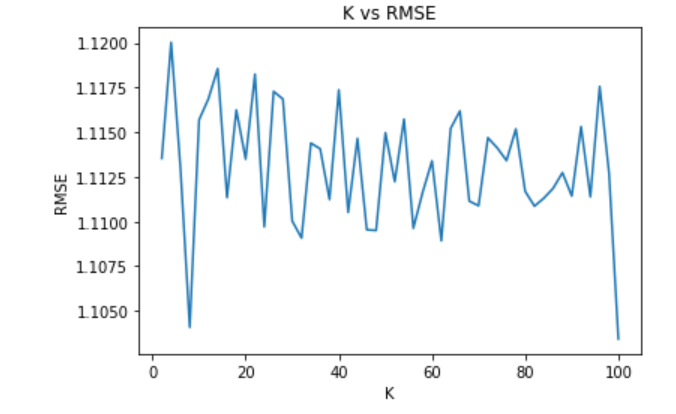
After defining the trimming function, we will apply k-NN collaborative filters that is trained on the training set and testing it on these trimmed test set. We sweep over the values of k from 2 to 102 with step size of 2.

Question 12: Design a k-NN collaborative filter to predict the ratings of the movies in the popular movie trimmed test set and evaluate its performance using 10-fold cross validation. Sweep k (number of neighbors) from 2 to 100 in step sizes of 2, and for each k compute the average RMSE obtained by averaging the RMSE across all 10 folds. Plot average RMSE (Y-axis) against k (X-axis). Also, report the minimum average RMSE.



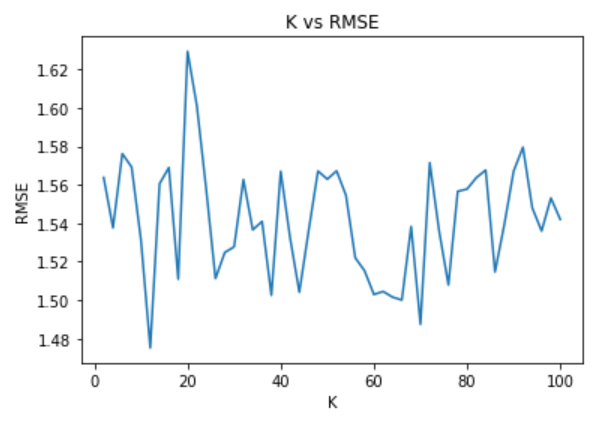
Lowest RMSE = 0.87121593879334

Question 13: Design a k-NN collaborative filter to predict the ratings of the movies in the unpopular movie trimmed test set and evaluate its performance using 10-fold cross validation. Sweep k (number of neighbors) from 2 to 100 in step sizes of 2, and for each k compute the average RMSE obtained by averaging the RMSE across all 10 folds. Plot average RMSE (Y-axis) against k (X-axis). Also, report the minimum average RMSE.



Lowest RMSE = 1.1034293478688482

Question 14: Design a k-NN collaborative filter to predict the ratings of the movies in the high variance movie trimmed test set and evaluate its performance using 10-fold cross validation. Sweep k (number of neighbors) from 2 to 100 in step sizes of 2, and for each k compute the average RMSE obtained by averaging the RMSE across all 10 folds. Plot average RMSE (Y-axis) against k (X-axis). Also, report the minimum average RMSE.

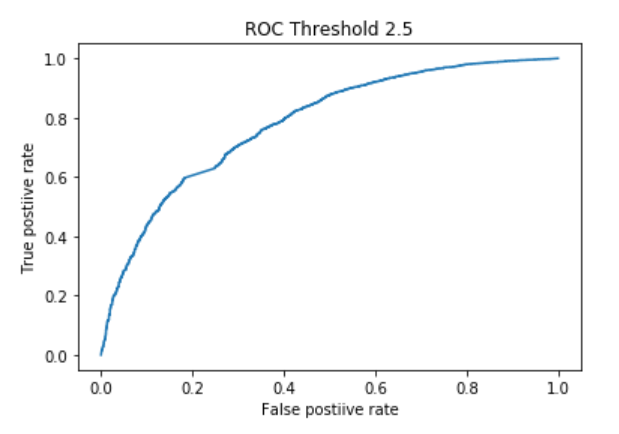


Lowest RMSE = 1.4751680197617105

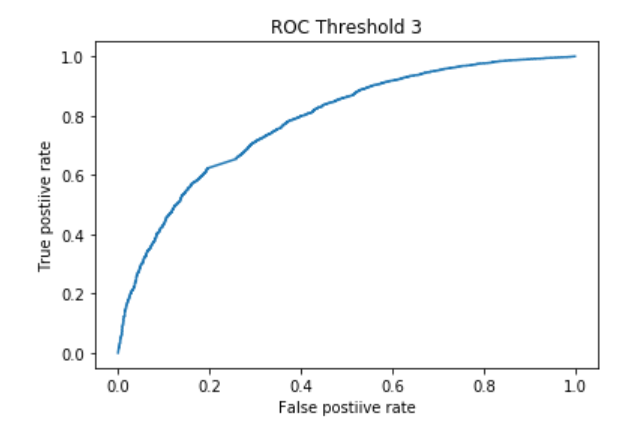
Question 15: Plot the ROC curves for the k-NN collaborative filter designed in question 10 for threshold values [2.5; 3; 3.5; 4]. For the ROC plotting use the k found in question 11. For each of the plots, also report the area under the

curve (AUC) value.

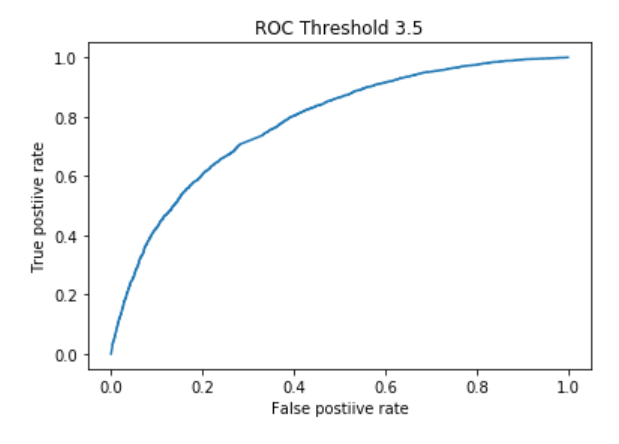
Next, we use the k values we got from question 11, which is k = 28 in order to plot the ROC curve. Before we call the plotting functions, we threshold the ground truth which are [2.5, 3, 3.5, 4]. Anything that is greater than the threshold value, we set that to 1, and lesser than we set it to 0. This can be interpreted as 1 = liking the movie and 0 = not liking the movie. We then plot the ROC curve giving us the graphs below for different threshold:



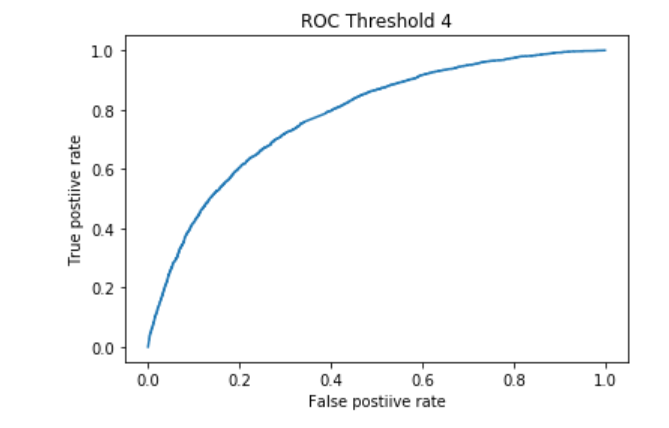
AUC = 0.7815000404306595



AUC = 0.7833463333634785



AUC = 0.7805204768910815



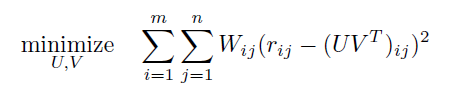
AUC = 0.7792879250252334

1. Model-based collaborative filtering

In model-based collaborative filtering, models are developed using machine learning algorithms to predict users’ rating of unrated items. Some examples of model-based methods include decision trees, rule-based models, bayesian meth- ods, and latent factor models. In this project, we will explore latent factor based models for collaborative filtering.

Question 16: Is the optimization problem given by equation 5 convex? Consider the optimization problem given by equation 5. For U fixed, formulate it as a least-squares problem.

The optimization problem is given by the formula:



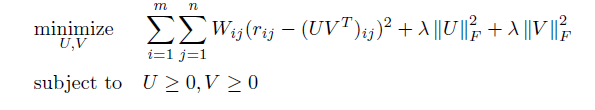
We can see that this problem is nonconvex because its Hessian Matrix is not positive semidefinite. If we fixed U, we’re only solving for V since U is fixed. This is intuitive that the equation above becomes a least squared problem.



We only need to minimize with respect to V because we have already fixed our U. We can actually convert this into a least square problem because we can look at *r* as our y\_true and *UV* as our y\_pred. This is then just reduced to a simple least square problem.

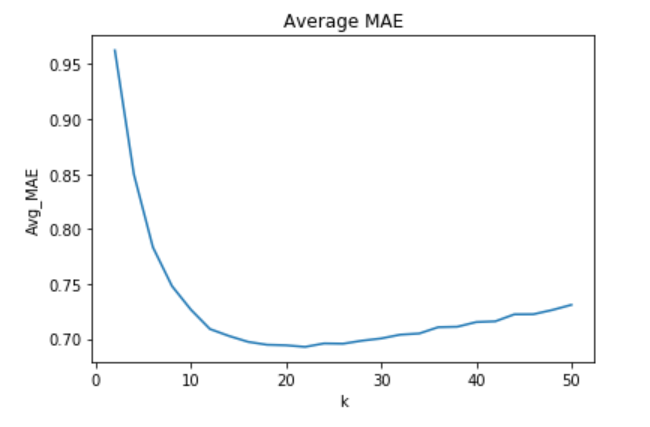
Question 17: Design a NNMF-based collaborative filter to predict the ratings of the movies in the MovieLens dataset and evaluate its performance using 10-foldcross-validation. Sweep k (number of latent factors) from 2 to 50 in step sizes of 2, and for each k compute the average RMSE and average MAE obtained by averaging the RMSE and MAE across all 10 folds. Plot the average RMSE (Y-axis) against k (X-axis) and the average MAE (Y-axis) against k (X-axis). For solving this question, use the default value for the regularization parameter.

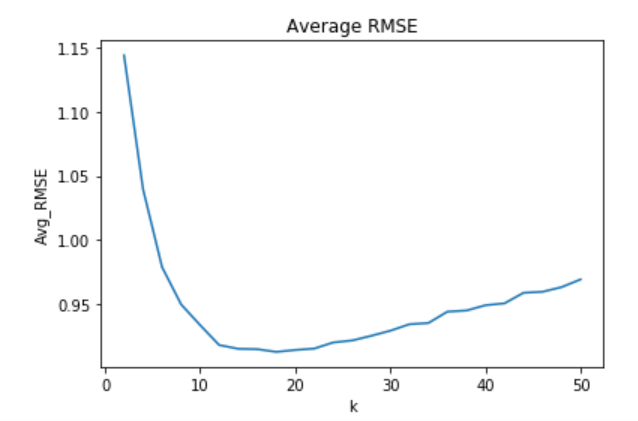
We used NNMF because of its interpretability in understanding user and item interaction. The difference from other method is that U and V have to which are the latent factors have to be non-negative. In order to find these vectors, we solve the optimization problem below:



There are many ways to solve this optimization problem, but in our project, we used SGD because the package we’re using only allows for SGD.

In this question, we perform NNMF with different number of latent vectors. We sweep over 2 to 50 latent vectors in step size of 2 giving us the result below:





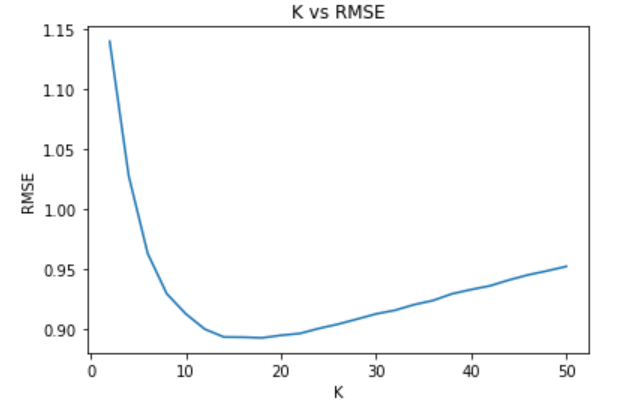
Question 18: Use the plot from question 17, to find the optimal number of latent factors. Optimal number of latent factors is the value of k that gives the minimum average RMSE or the minimum average MAE. Please report the minimum average RMSE and MAE. Is the optimal number of latent factors same as the number of movie genres?

From the results, we got the best number of latent vectors based on RMSE to be 16 or18 depending on the time we ran it and MAE to be 22. From the dataset, we have 19 genres of movies as below:

* Action
* Adventure
* Animation
* Children's
* Comedy
* Crime
* Documentary
* Drama
* Fantasy
* Film-Noir
* Horror
* Musical
* Mystery
* Romance
* Sci-Fi
* Thriller
* War
* Western
* (no genres listed)

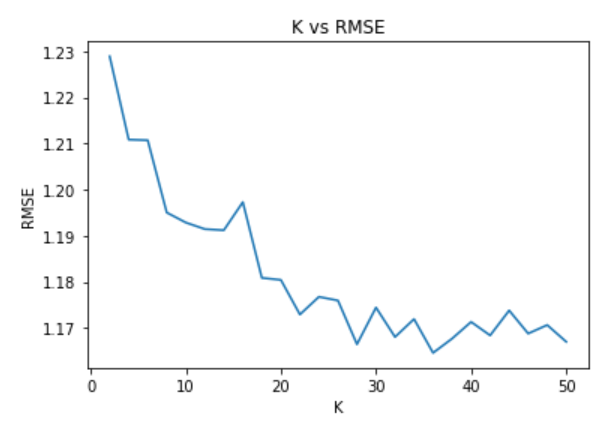
We can see that although the numbers are not equal, it is very close. Actually, if we do not include the no genres listed, the number of latent vectors based on average RMSE actually is the same as the number of genres. This is very intuitive since each genre should have some specific characteristic that distinguishes it from other genres. This is why if we reduce it to that number, the NNMF will give us the least error.

Question 19: Design a NNMF collaborative filter to predict the ratings of the movies in the popular movie trimmed test set and evaluate its performance using 10-fold cross validation. Sweep k (number of latent factors) from 2 to 50 in step sizes of 2, and for each k compute the average RMSE obtained by averaging the RMSE across all 10 folds. Plot average RMSE (Y-axis) against k (X-axis). Also, report the minimum average RMSE.



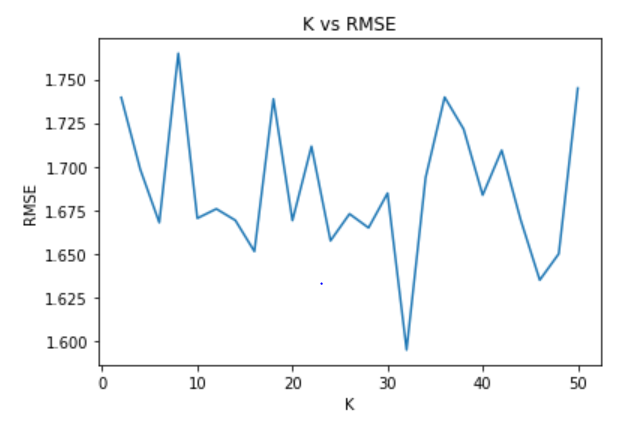
RMSE = 0.8925079927544513

Question 20: Design a NNMF collaborative filter to predict the ratings of the movies in the unpopular movie trimmed test set and evaluate its performance using 10-fold cross validation. Sweep k (number of latent factors) from 2 to 50 in step sizes of 2, and for each k compute the average RMSE obtained by averaging the RMSE across all 10 folds. Plot average RMSE (Y-axis) against k (X-axis). Also, report the minimum average RMSE.



RMSE = 1.1645629486931273

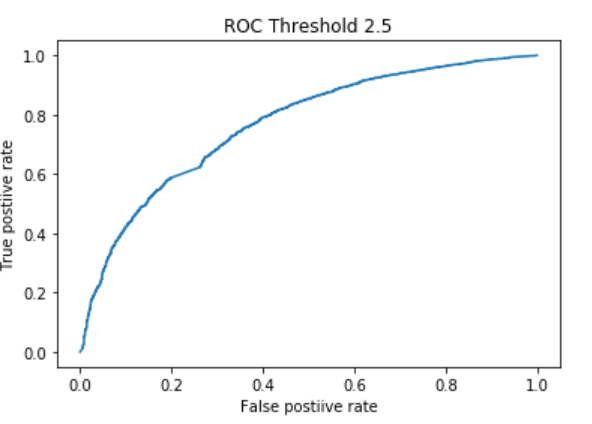
Question 21: Design a NNMF collaborative filter to predict the ratings of the movies in the high variance movie trimmed test set and evaluate its performance using 10-fold cross validation. Sweep k (number of latent factors) from 2 to 50 in step sizes of 2, and for each k compute the average RMSE obtained by averaging the RMSE across all 10 folds. Plot average RMSE (Y-axis) against k (X-axis). Also, report the minimum average RMSE.



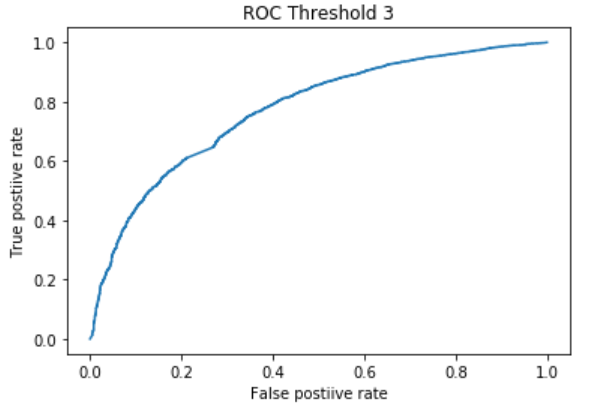
RMSE = 1.59487738976261

Question 22: Plot the ROC curves for the NNMF-based collaborative filter designed in question 17 for threshold values [2:5; 3; 3:5; 4]. For the ROC plotting use the optimal number of latent factors found in question 18. For each of the plots, also report the area under the curve (AUC) value.

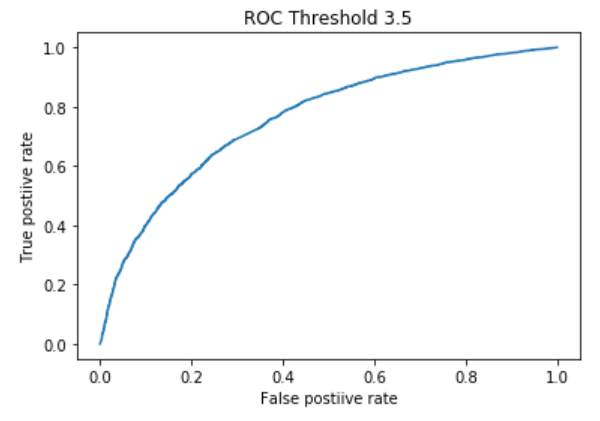
Next, we use the number of latent factors we got from question 18, which is k = 18 in order to plot the ROC curve. Before we call the plotting functions, we threshold the ground truth which are [2.5, 3, 3.5, 4]. Anything that is greater than the threshold value, we set that to 1, and lesser than we set it to 0. This can be interpreted as 1 = liking the movie and 0 = not liking the movie. We then plot the ROC curve giving us the graphs below for different threshold:



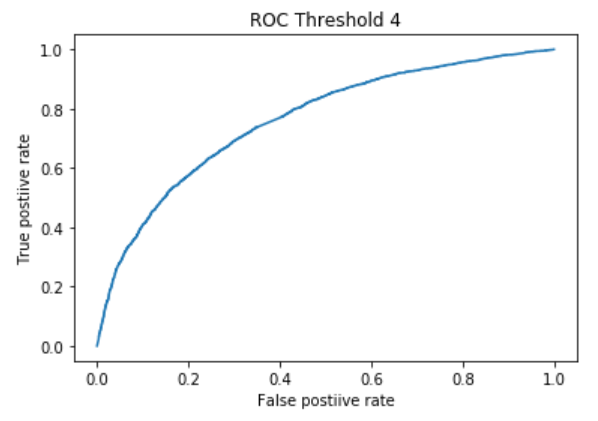
AUC = 0.7672953441655048



AUC = 0.7722796750733076



AUC = 0.7613905594702324



AUC = 0.7616785273386322

Question 23: 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). For each column of V, sort the movies in descending order and report the genres of the top 10 movies. Do the top 10 movies belong to a particular or a small collection of genre? Is there a connection between the latent factors and the movie genres?

We printed out some columns. Theoretically, each column should correspond to some genre. We can also see this in some of the columns, but some of them are very noisy. For instance, the first and the fourth column are shown below:

Column 1:

['Documentary', 'Action|Horror|Sci-Fi|Thriller', 'Action|Sci-Fi|Thriller', 'Horror|Thriller', 'Adventure|Children|Comedy|Mystery', 'Drama', 'Animation|Children|Fantasy', 'Drama', 'Drama', 'Drama|Romance']

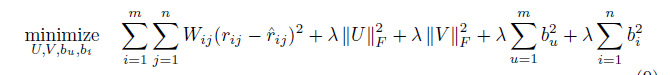
Column 4:

['Comedy|Romance', 'Comedy|Drama|Romance', 'Comedy|Romance', 'Action|Comedy|Crime|Thriller', 'Adventure|Children|Comedy|Mystery', 'Film-Noir|Thriller', 'Drama|Horror|Thriller', 'Comedy|Crime', 'Drama', 'Drama']

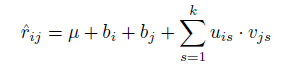
In the first column, most of the genres belong to a superset of Action / Sci-Fi / Thriller / Drama. In the fourth column, most of the movies contain drama. Other column also exhibit the same characteristics, but they are very noisy and might not be as obvious as these columns.

In Section 5.3 we try predicting movie ratings for users using Matrix Factorization with the bias included.

The cost function is then calculated using the following equation:



And the prediction is now calculated using:



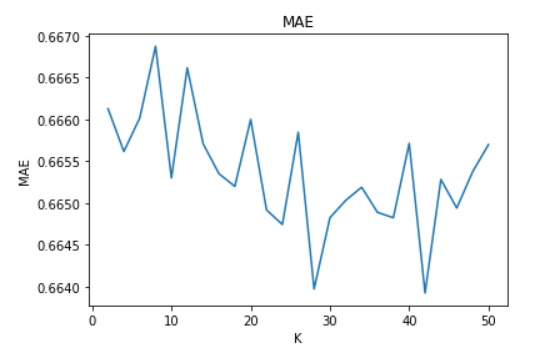
The difference between NNMF and MF solution is the inclusion of the bias term, which are also learned and updated as per the following equation:

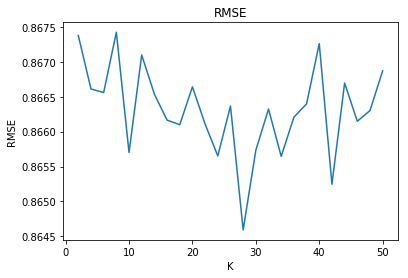


Question 24: Design a MF with bias collaborative filter to predict the ratings of the movies in the MovieLens dataset and evaluate it's performance using 10-fold cross-validation. Sweep k (number of latent factors) from 2 to 50 in step sizes of 2, and for each k compute the average RMSE and average MAE obtained by averaging the RMSE and MAE across all 10 folds. Plot the average RMSE (Y-axis) against k (X-axis) and the average MAE (Y-axis) against k (X-axis). For solving this question, use the default value for the regularization parameter.

We now perform collaborative filtering to predict rating of movies which would have been given by the user, to make the recommendations suited for him. Using the surprise built-in library, we create a model like the ones we created from k-NN and NNMF.

The filter is evaluated over for different number of factors between 2 to 50, and the average RMSE and MAE is calculated for all values of K.





There is a trend visible in the RMSE and MAE calculated over the spread, the errors have a downward trend till around k=28, and then they increase again.

Question 25: Use the plot from question 24, to find the optimal number of latent factors. Optimal number of latent factors is the value of k that gives the minimum average RMSE or the minimum average MAE. Please report the

minimum average RMSE and MAE.

To calculate the optimal value of latent factor, we analyze the graphs generated in the Question 24, and use the K values where the average RMSE was minimum.

Selected latent factors k by MAE: 42

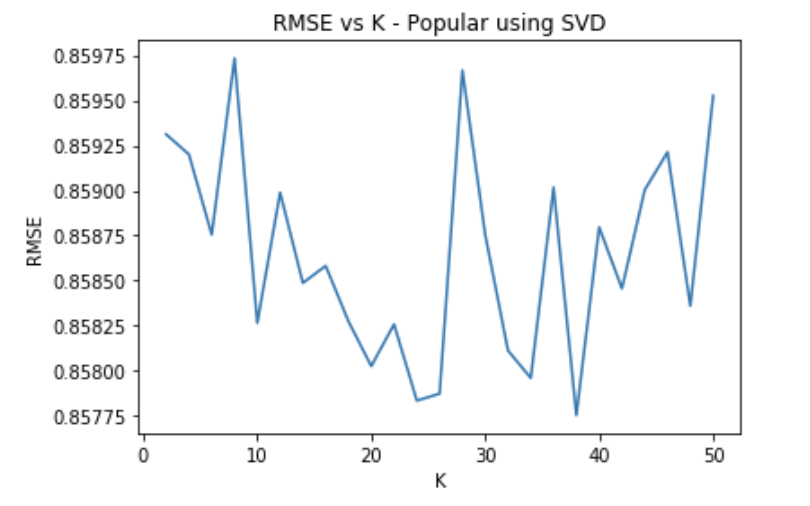
Minimum average MAE 0.663925636234687

Selected latent factors k by RMSE: 28

Minimum average RMSE: 0.8645905588627626

Question 26: Design a MF with bias collaborative filter to predict the ratings of the movies in the popular movie trimmed test set and evaluate its performance using 10-fold cross validation. Sweep k (number of latent factors) from 2 to 50 in step sizes of 2, and for each k compute the average RMSE obtained by averaging the RMSE across all 10 folds. Plot average RMSE (Y-axis) against k (X-axis). Also, report the minimum average RMSE

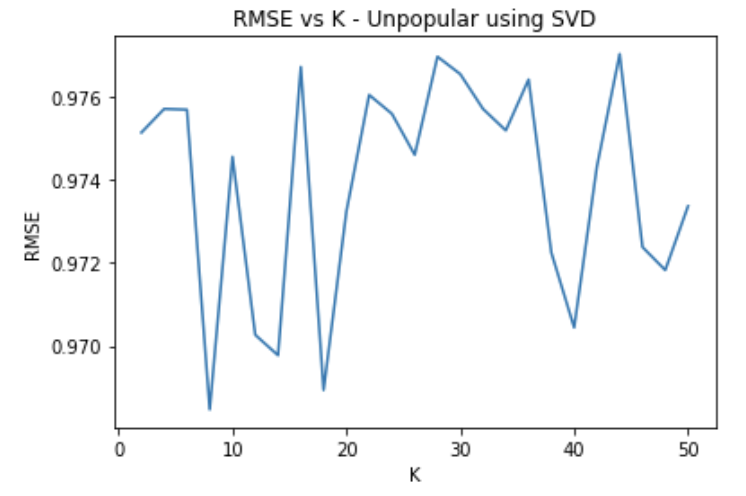
We repeat the same process above with the MF with bias (using built-in SVD) collaborative filter using the popular movie trimmed test set which removed movies that have more than 2 ratings.



We can infer from the figure above, the difference in average RMSE is quite low, with very small fluctuating variance, as k increases.

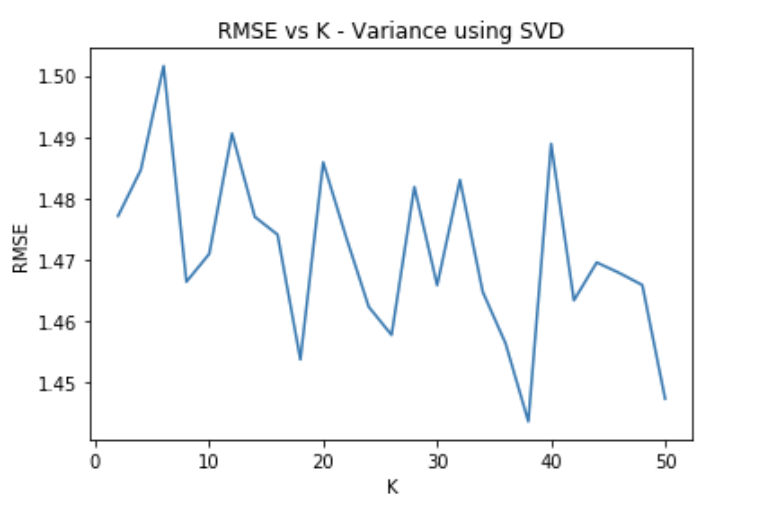
Question 27: Design a MF with bias collaborative filter to predict the ratings of the movies in the unpopular movie trimmed test set and evaluate it's performance using 10-fold cross validation. Sweep k (number of latent factors) from 2 to 50 in step sizes of 2, and for each k compute the average RMSE obtained by averaging the RMSE across all 10 folds. Plot average RMSE (Y-axis) against k (X-axis). Also, report the minimum average RMSE

We repeat the same process above with the MF (using built-in SVD) with bias collaborative filter using the unpopular movie trimmed test set.



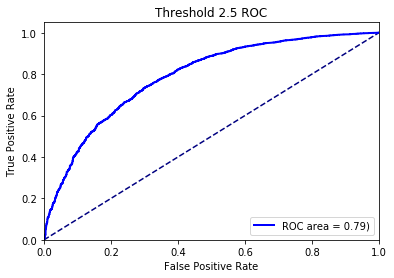
Question 28: Design a MF with bias collaborative filter to predict the ratings of the movies in the high variance movie trimmed test set and evaluate its performance using 10-fold cross validation. Sweep k (number of latent factors) from 2 to 50 in step sizes of 2, and for each k compute the average RMSE obtained by averaging the RMSE across all 10 folds. Plot average RMSE (Y-axis) against k (X-axis). Also, report the minimum average RMSE

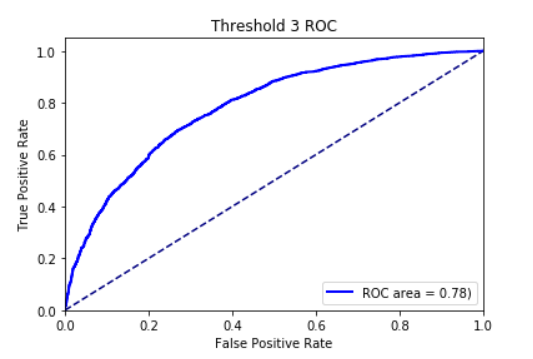
We repeat the same process above with the MF with bias collaborative filter using the high variance movie trimmed test set. In this dataset, the RMSE error shows high variation over different number of filters used, and it performs worse than the other two datasets.

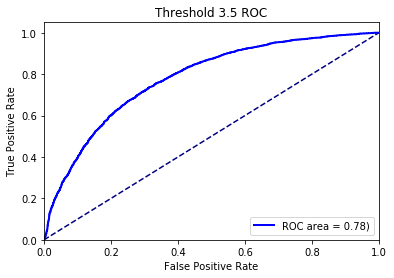


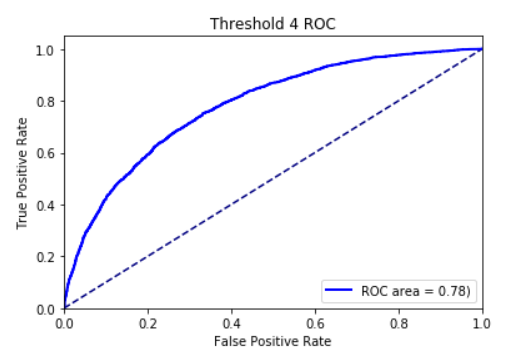
Question 29: Plot the ROC curves for the MF with bias collaborative filter designed in question 24 for threshold values [2:5; 3; 3:5; 4]. For the ROC plotting use the optimal number of latent factors found in question 25. For each of the plots, also report the area under the curve (AUC) value.

We evaluated the performance of this filter using the ROC curve. The ROC curves to asses MF with bias collaborative filter are made using the same methodology as the ROC curve with the k-NN and NNMF filters.









|  |  |
| --- | --- |
| Thresholds | AUC |
| 2.5 | 0.79 |
| 3 | 0.78 |
| 3.5 | 0.78 |
| 4 | 0.78 |

1. **Naive collaborative filtering**

We also used a naive collaborative filter to predict ratings in the dataset, which uses the mean rating of each user to predict movies that have not been rated.

The prediction function shall be calculated as rij = µi , where i is the user and j is the item, and µi is the mean rating out of the ratings for the user.

Question 30: Design a naive collaborative filter to predict the ratings of the movies in the MovieLens dataset and evaluate it's performance using 10-fold cross validation. Compute the average RMSE by averaging the RMSE across all 10 folds. Report the average RMSE.

To implement the Naïve collaborative filtering we implemented our own prediction algorithm with a estimate prediction similar to the estimate prediction of the AlgoBase class of the surprise package.

We create an object of the surprise algorithm that we built and use cross validation to train and test the data in the original ratings dataset in MovieLens, calculating the RMSE in each iteration. The mean RMSE is calculated at the end.

For the Naïve Filter when calculated over the complete dataset, we calculate the RMSE as **0.941.**

Question 31: Design a naive collaborative filter to predict the ratings of the movies in the popular movie trimmed test set and evaluate its performance using 10-fold cross validation. Compute the average RMSE by averaging the RMSE across all 10 folds. Report the average RMSE.

We repeat the Naïve Filtering over the popular dataset and calculate the mean RMSE same as we implemented in the last question. The average RMSE we calculated over the trimmed popular dataset is **0.9311**.

Question 32: Design a naive collaborative filter to predict the ratings of the movies in the unpopular movie trimmed test set and evaluate it's performance using 10-fold cross validation. Compute the average RMSE by averaging the RMSE across all 10 folds. Report the average RMSE.

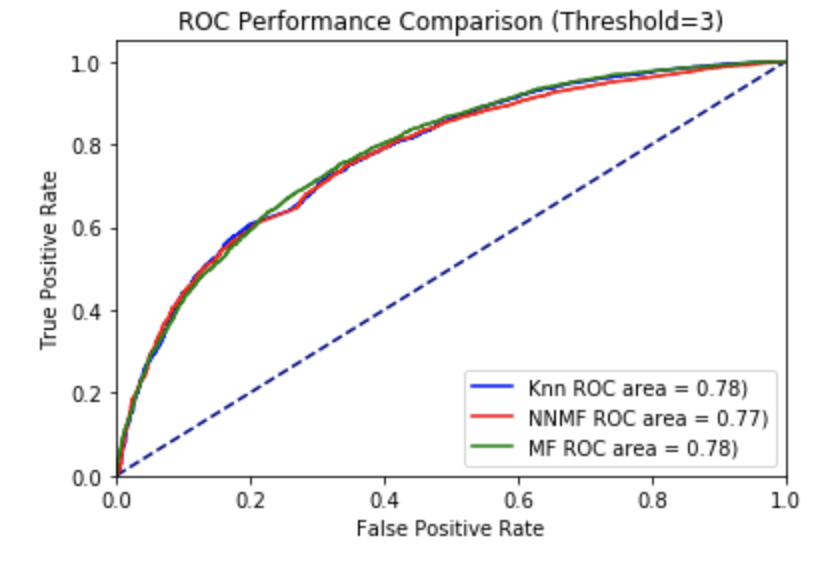
We repeat the Naïve Filtering over the unpopular dataset and calculate the mean RMSE same as we implemented in the last question. The average RMSE we calculated over the trimmed unpopular dataset is **0.962**.

Question 33: Design a naive collaborative filter to predict the ratings of the movies in the high variance movie trimmed test set and evaluate it's performance using 10-fold cross validation. Compute the average RMSE by averaging the RMSE across all 10 folds. Report the average RMSE.

We lastly implement Naïve Filtering over the high variance dataset and calculate its mean RMSE value. The average RMSE value found over this dataset is **1.433**.

We can infer from the above values that Naïve Filtering much like the other filtering methodologies we experimented on works poorly on the high variance dataset and gives better results for the popular and unpopular dataset.

Question 34: Plot the ROC curves (threshold = 3) for the k-NN, NNMF, and MF with bias based collaborative filters in the same figure. Use the figure to compare the performance of the filters in predicting the ratings of the movies.



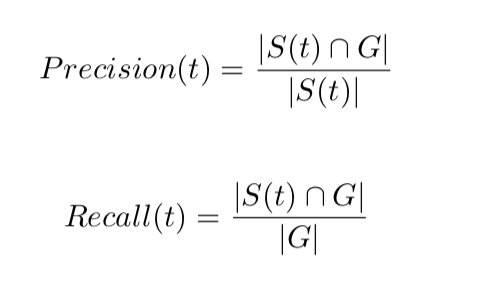
Plotting all the three curves corresponding to the bias based collaborative filters, we are able to do a performance comparison between them. We see that the Matrix Factorization and Knn have a higher area under the curve and outperforms NMF. Plotting the curves together, we see that they almost coincide with each other and the difference between their performances are very little.

Question 35: Precision and Recall are defined by the mathematical expressions given by equations 12 and 13 respectively. Please explain the meaning of precision and recall in your own words.

The two sets we are interested in wrt calculating precision and recall are,

S(t): The set of items of size t recommended to the user. In this recommended set, ignoring the items for which we don’t have a ground truth rating.

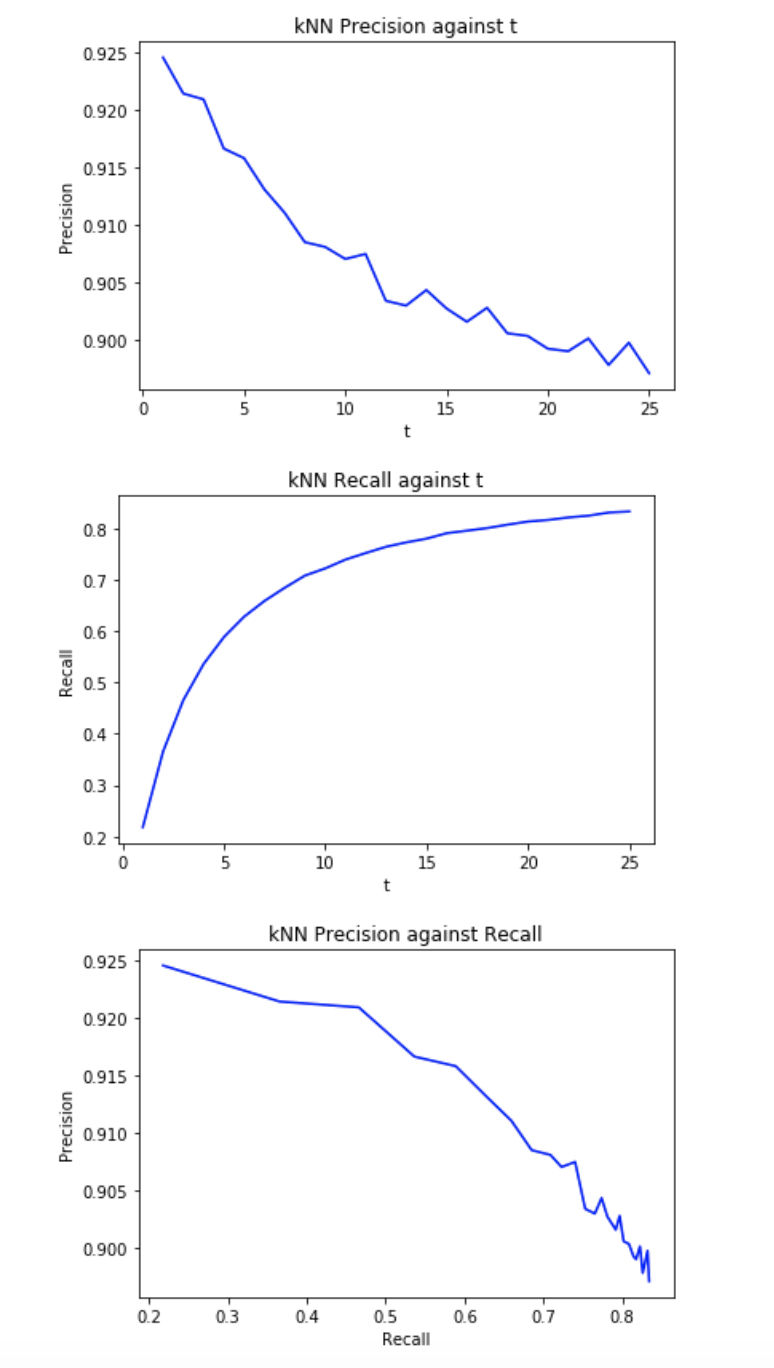
G: The set of items liked by the user (ground-truth positives)



These equations imply, that precision is the correctness of our prediction, i.e. It tells us how many positively predicted items are actually positive. It is a measure of accuracy. Whereas recall defines the completeness of our prediction, i.e. it tells us what percentage of the positive items were predicted correctly.

Question 36: Plot average precision (Y-axis) against t (X-axis) for the rank- ing obtained using k-NN collaborative filter predictions. Also, plot the average recall (Y-axis) against t (X-axis) and average precision (Y-axis) against average recall (X-axis). Use the k found in question 11 and sweep t from 1 to 25 in step sizes of 1. For each plot, briefly comment on the shape of the plot.

We plot the following curves for the Knn algorithm using the best k=28.

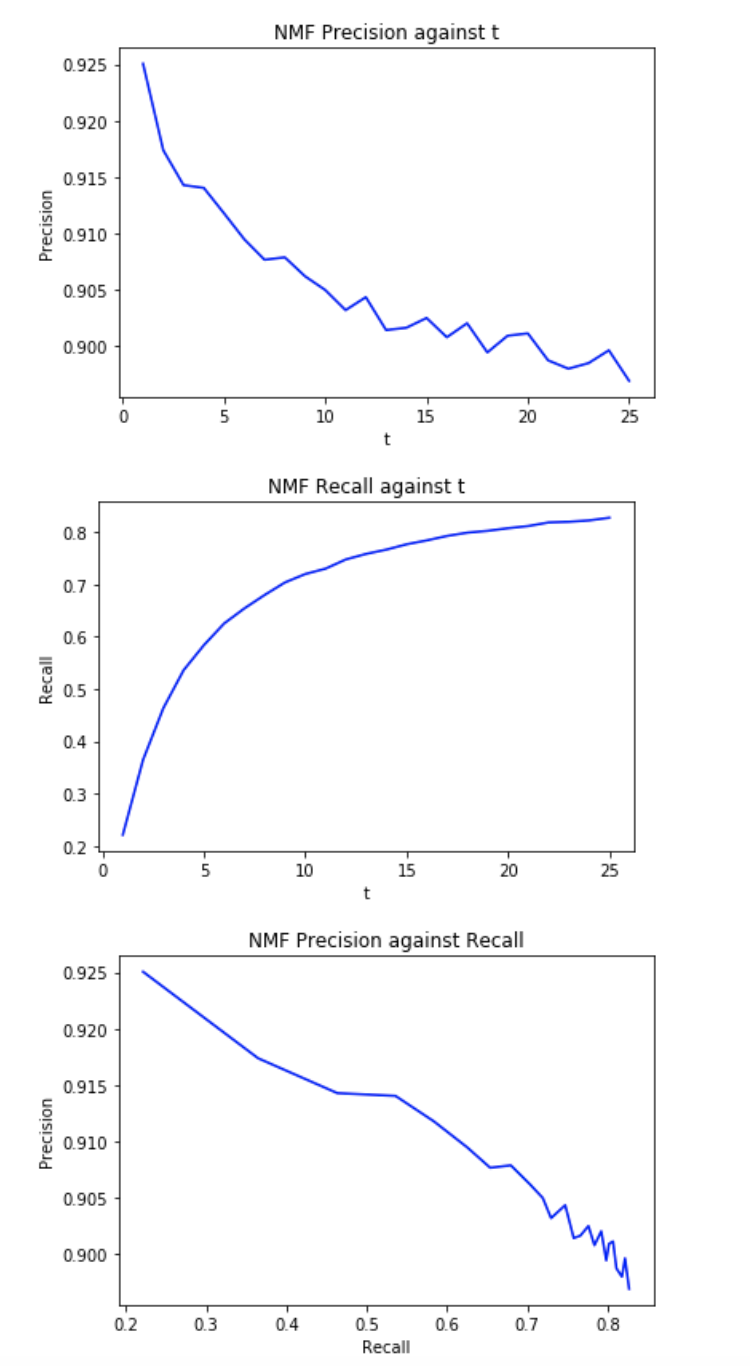


We see that the plot for Precision vs t decreases as we keep increasing t. This is because it is easier to predict the top 1 movie that a particular user will like and get that right, as opposed to getting top 25 movies that he will like. Though we should notice that the graph still shows accuracy of more than 90% even on increasing t, which means it doesn’t go down drastically.

For the Recall vs t graph, we see an exponential rise in recall as we increase the number of recommended items. This is not surprising, since Precision and recall are all ways in a tug of war. Increasing one, reduces the other.

The precision vs recall plot also shows a decreasing behavior. As recall value increases, the precision value decreases which is a known characteristic trait and the graphs is in accordance with the theoretical knowledge of the same.

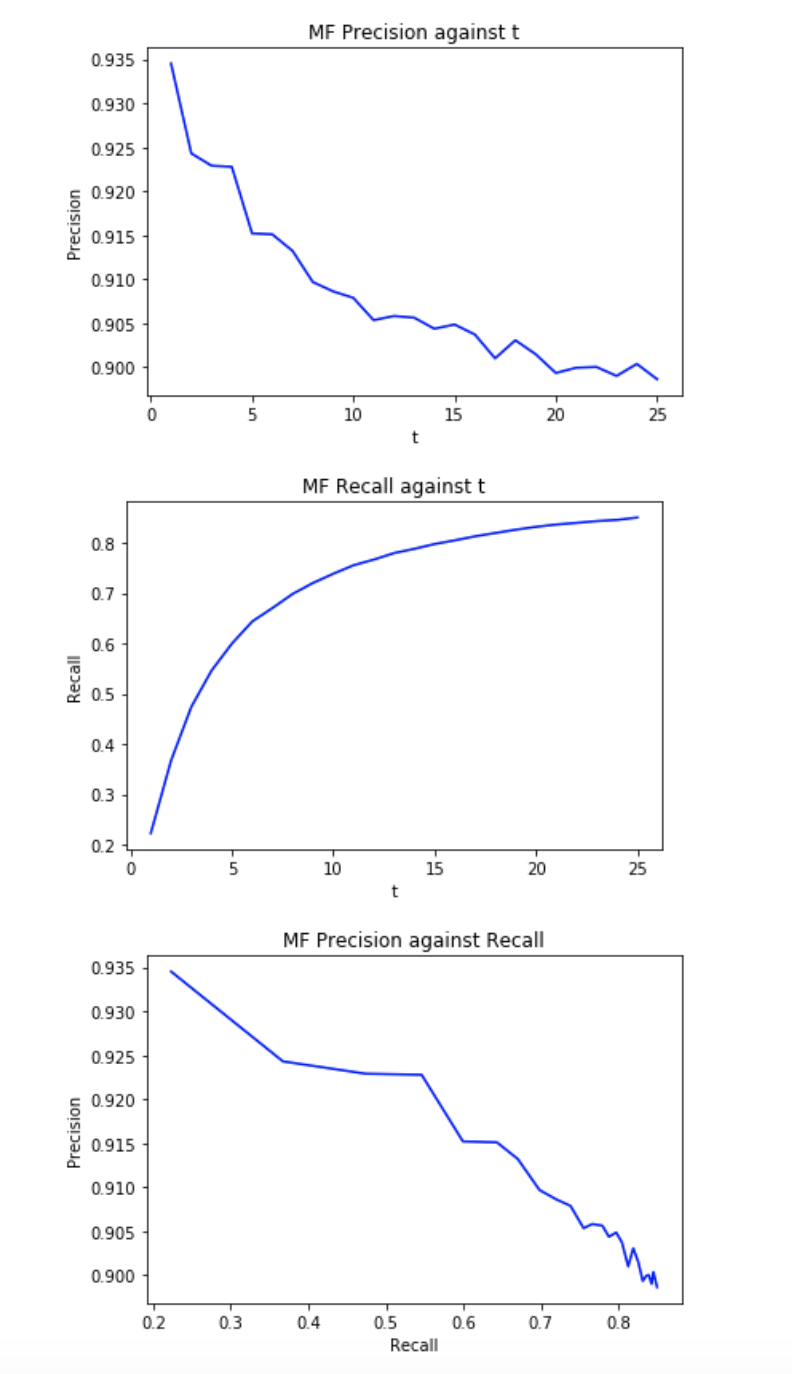
Question 37: Plot average precision (Y-axis) against t (X-axis) for the rank- ing obtained using NNMF-based collaborative filter predictions. Also, plot the average recall (Y-axis) against t (X-axis) and average precision (Y-axis) against average recall (X-axis). Use optimal number of latent factors found in question 18 and sweep t from 1 to 25 in step sizes of 1. For each plot, briefly comment on the shape of the plot.



We plot the following curves for the NNMF algorithm using the best k=18.

The curves follow the same trend as they did for knn.

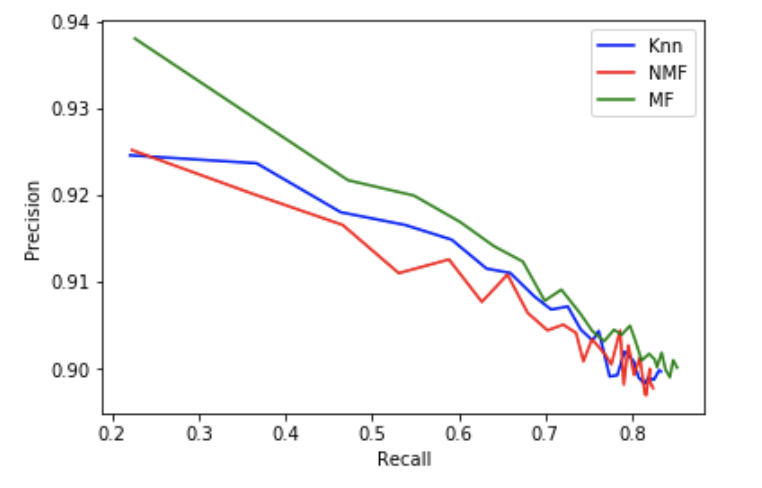
Question 38: Plot average precision (Y-axis) against t (X-axis) for the ranking obtained using MF with bias-based collaborative filter predictions. Also, plot the average recall (Y-axis) against t (X-axis) and average precision (Y-axis) against average recall (X-axis). Use optimal number of latent factors found in question 25 and sweep t from 1 to 25 in step sizes of 1. For each plot, briefly comment on the shape of the plot.



We plot the following curves for the MF algorithm using the best k=28.

The curves follow the same trend as they did for knn and NMF.

Question 39: Plot the precision-recall curve obtained in questions 36,37, and 38 in the same figure. Use this figure to compare the relevance of the recommendation list generated using k-NN, NNMF, and MF with bias predictions.



After Plotting the three Precision and Recall curves together, for all the algorithms, we see that all of them show similar performances and characteristics.