ECE 219
Large Scale Data Mining: Models and Algorithms
Prof. Vwani Roychowdhury
UCLA, Department of ECE
Spring 2021

Project 3: Collaborative Filtering

Kenny Huynh, 805436804
Shu-Yun Ku, 405515493
Ryan Li, 704142455
Osama Hassen, 105644793

QUESTION 1:
Compute the sparsity of the movie rating dataset, where sparsity is defined by equation 1: Sparsity = (Total number of available ratings) / (Total number of possible ratings)

<u>Answer:</u>
We computed the sparsity as defined by the given equation. The total number of available ratings is equal to the total number of ratings in the ratings dataset. The total number of possible ratings is the product of the number of unique users and the number of unique movies in the dataset. Using these numbers, we found that the sparsity is 0.016999683055613623. This relatively low value for sparsity indicates that our movie rating dataset is quite sparse. This sparseness is expected because most users would have viewed only a small fraction of the entire catalog of available movies. As a result, most of the ratings are unspecified and thus we have a sparse matrix with a lot of “zero” values.

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

<u>Answer:</u>

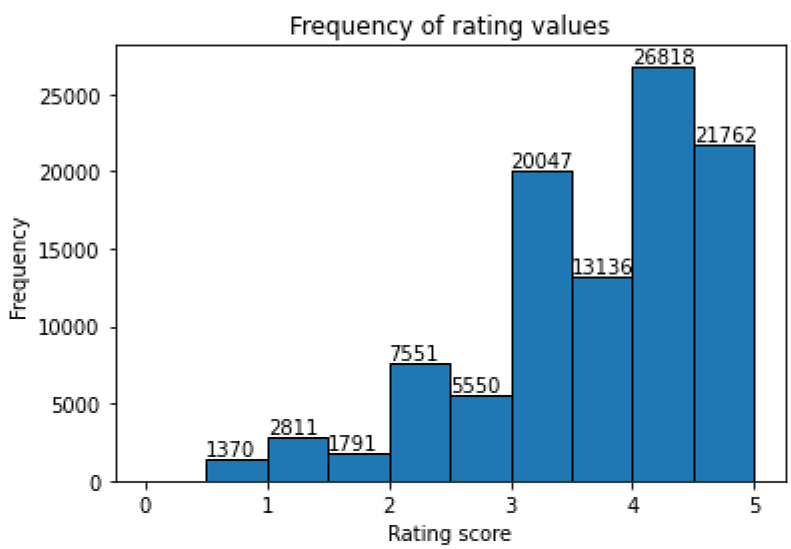

Figure 1: Frequency of Rating Values

The histogram shows that the majority of ratings were between 3.0 and 5.0. These results indicate that the users may in general have a bias to rate movies more positively. One explanation for this bias is that users are more psychologically and emotionally enthused to leave positive ratings on movies they like than to leave negative ratings on movies they did not like. It’s also possible that users generally don’t like being too critical or negative and therefore inflate their ratings. It could also be that users tend to only watch movies they are interested in and have a natural inclination towards, and thus watch fewer movies that they would be inclined to give a negative rating to. Whatever the reason, this dataset has more “positive” ratings (above 2.5) than “negative” ratings (below 2.5).

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 broken in any way. A monotonically decreasing curve instead of a histogram is expected.

Answer:
We plotted the distribution as described by the question, and we saw a monotonically decreasing curve as expected. The results indicate that out of the roughly 10,000 movies in this dataset, there are a few popular movies that receive a lot of ratings while the majority of movies have relatively few ratings. After the top 2,000 or so movies, the number of ratings received by the remaining movies is close to zero.

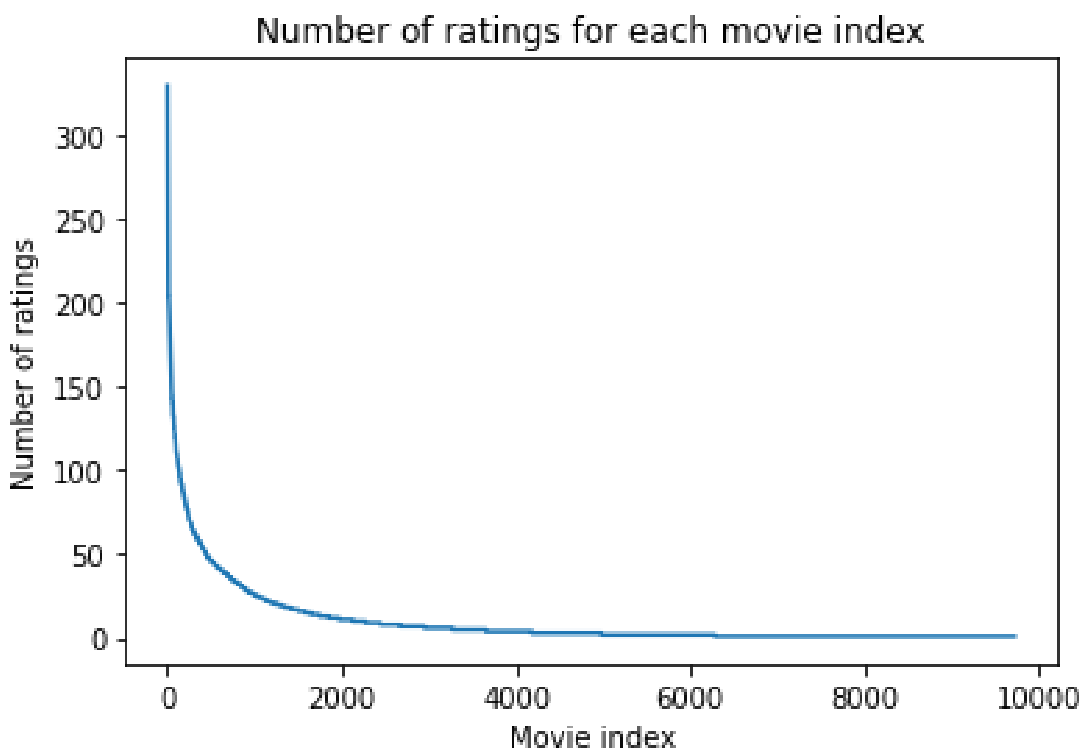

Figure 2: Number of Ratings for each Movie Index

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 has rated. The requirement of the plot is similar to that in Question 3.

<u>Answer:</u>

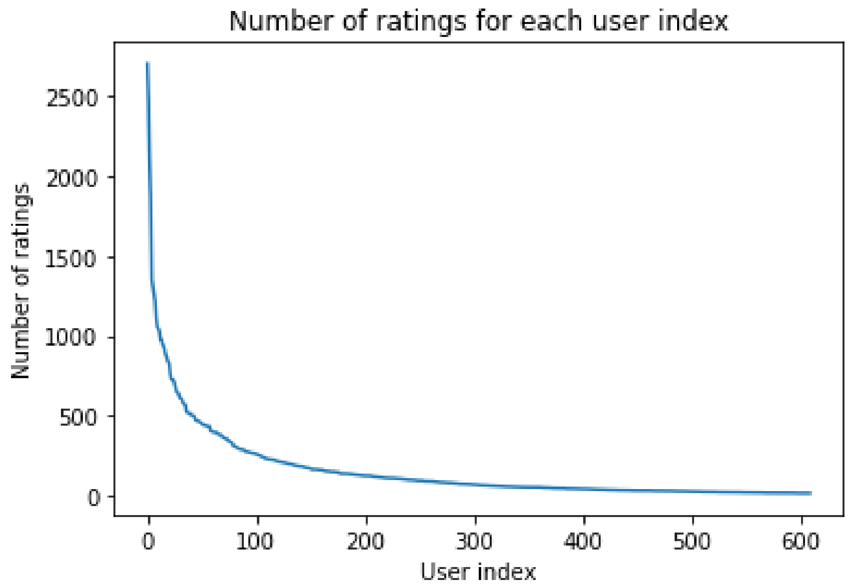

Figure 3: Number of Ratings for each User Index

Similar to the plot in Question 3, we see that the number of ratings falls off quite rapidly after the top few users. This indicates that a few users are quite active in giving out ratings for movies while the majority of the user base has only given out a few ratings. As a result, a recommendation system based on user similarity could be heavily influenced by these most active users since the large majority of the ratings in our dataset come from them. This is not necessarily crippling to our recommendation system, but is something that we should keep in mind as we build our model.

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

<u>Answer:</u>
From our results in Question 3 seen in Figure 2, we see that a small number of movies received a majority of the ratings. A corollary to that is that most of the movies received a very small number of ratings. This implies that the rating matrix R is sparse, and that heavy regularization may need to be added to the recommendation process to prevent overfitting and false links. Additionally, for the movies that have a large number of ratings (the “popular” movies), the recommendation system will have more actual data points to perform calculations on, and therefore be more likely to accurately predict whether or not to recommend these “popular” movies to a given user.

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.

Answer:

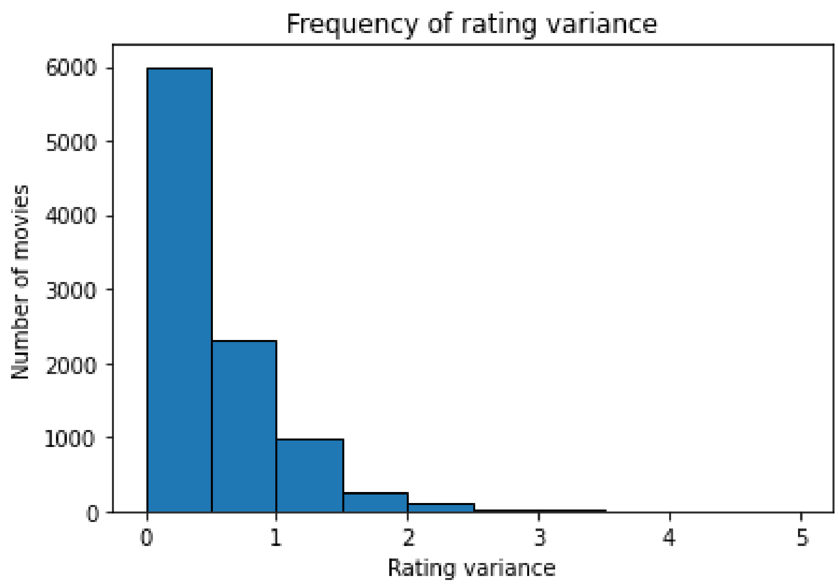

Figure 4: Frequency of Rating Variance

The histogram shows the rating variance for nearly all movies lies between 0 and 2 with the vast majority of movies having a rating variance between 0 and 1. This means that the variance in ratings is generally small and that the users generally rate the movies in about the same range. This is good for our recommendation system because if two users rated one movie about the same, the low variance means that it is likely that these same two users would rate another movie about the same as well. As such, movies rated highly by one of these users would likely be good recommendation targets for the other user and vice versa.

QUESTION 7:
Write down the formula for μ_u in terms of I_u and r_{uk}

Answer:

$$\mu_u = \frac{\sum_{k \in I_u} r_{uk}}{|I_u|}$$

QUESTION 8:
In plain words, explain the meaning of $I_u \cap I_v$. Can $I_u \cap I_v = \varnothing$ (Hint: Rating matrix *R* is sparse)

Answer:
The quantity $I_u \cap I_v$ corresponds to movies that have been rated by both user u and user v. Since the rating matrix R is sparse, it is not unlikely that $I_u \cap I_v = \emptyset$. There are so many movies in the dataset that it is possible that user u and user v did not watch any of the same movies. Or, even if they watched some of the same movies, it's possible that one user left a rating for that movie but the other did not, which would still give a result of $I_u \cap I_v = \emptyset$.

QUESTION 9:
Can you explain the reason behind mean-centering the raw ratings ($r_{vj} - \mu_v$) 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)

Answer:

Mean-centering the raw ratings in the prediction function helps reduce bias and remove extreme data points. For example, users who either rate all items highly or poorly are usually giving extreme opinions which are biased and can be considered noisy. Therefore, we can make a more accurate prediction if we mean-center the ratings.

Put another way, mean-centering helps to account for and remove the effect of users' rating styles on the dataset. For example, we may have a user u that tends to be highly critical in their ratings and rates most movies between 1-3. Meanwhile, user v tends to be overly generous in their ratings and rates most movies between 3 and 5. If we only look at the raw ratings, the similarity in their movie preferences might be lost due to the difference in their raw ratings. In this case, it's possible that a user w with a similarly overly generous rating style (between 3 and 5) but different movie preferences might be considered more similar to user v than user u because user v and user w might have more of the same raw ratings. However, by mean-centering all the ratings, we can remove some of the bias that comes from different rating styles (highly critical or overly generous) to see the underlying movie preferences of users. We would be able to identify that 3 is a "favorable" rating from user u and in some sense is equivalent to a rating of 5 from user v. Even though the raw scores of these two users for this particular movie are different, the mean-centered ratings indicate that they probably both liked this movie. A similar logic applies for "unfavorable" ratings from the two users, 1 for user u and 3 for user v.

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).

Answer:

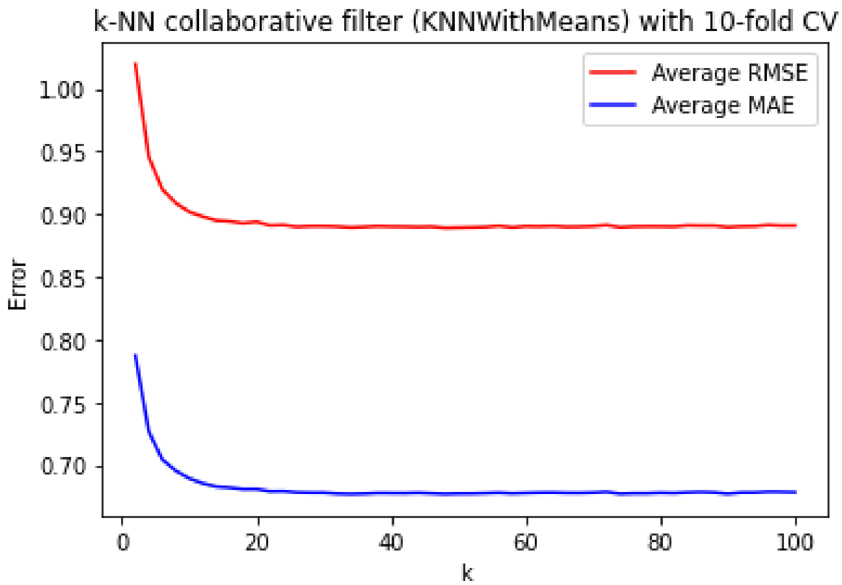

Figure 5: k-NN Collaborative Filter (kNN with Means) with 10 fold CV

As detailed by the question, we designed a k-NN collaborative filter, swept k from 2 to 100 in step sizes of 2, and did a 10-fold cross validation. From the results displayed above, we can see that the average RMSE and average MAE both drop relatively quickly for the first few values of k. This drop indicates that the error is reduced as k increases. However, there seems to be a point of diminishing returns after about k=20.

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

Answer:
We picked the "minimum k" to be k=22 since both average RMSE and average MAE reach their respective steady state values at around that k value. After that point, increasing k does not give us a significant decrease in average RMSE or average MAE.
Steady-state average RMSE = 0.8902 | Steady-state average MAE = 0.6782

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.

Answer:

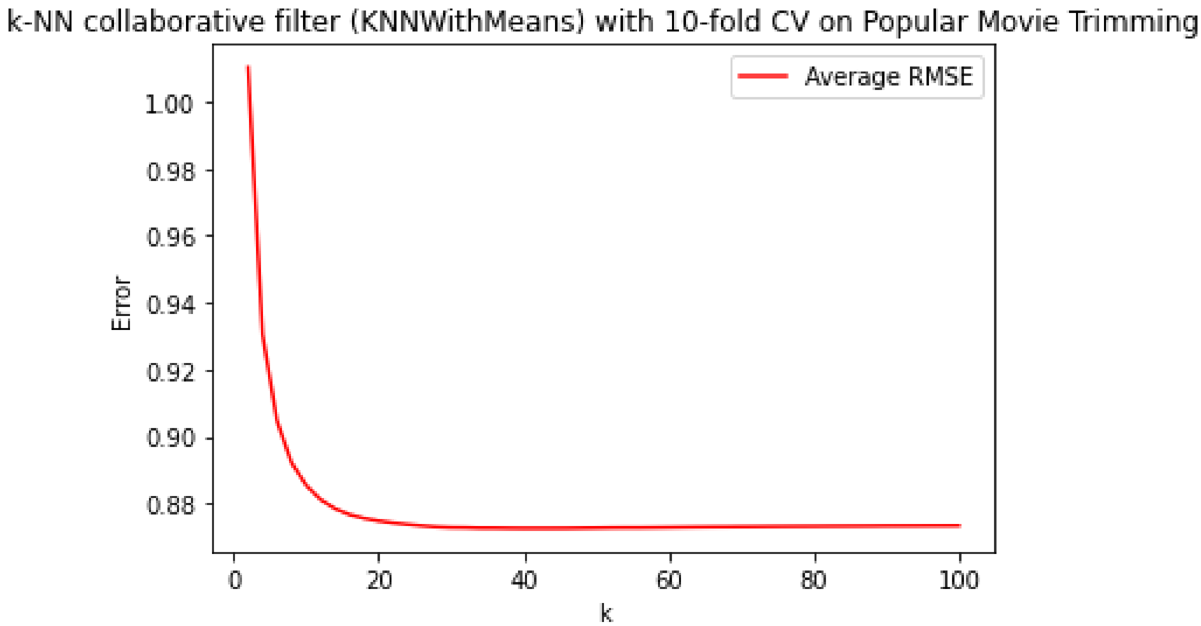

Figure 6: k-NN Collaborative Filter (kNN with Means) with 10 fold CV using Popular Movie Trimming

The results when predicting on a test set with popular movie trimming bear a similar shape to the results when predicting on a test set containing the original data with no trimming, but with slightly better performance or accuracy. The average RMSE falls rapidly as k increases for the first few values of k. After a certain point though, there seems to be diminishing returns with no significant decrease in average RMSE as k increases. In our testing, we found the minimum average RMSE using Popular Movie Trimming was 0.8725297543179018, which happened at k=40. However, after somewhere in the range of k=30 to k=34, the average RMSE values are very similar for all subsequent values of k.

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.

Answer:

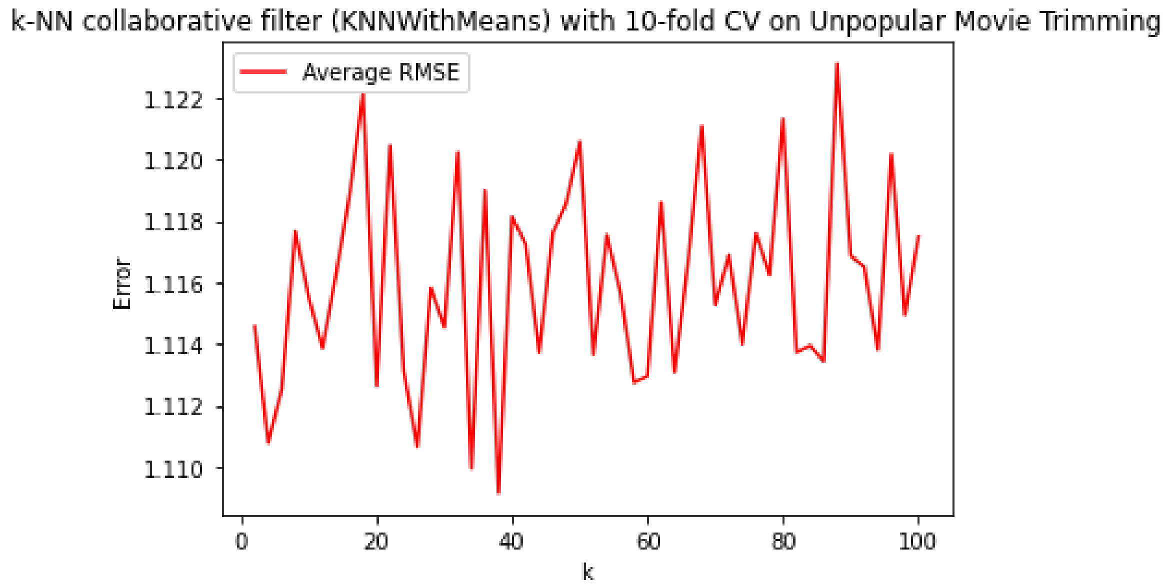

Figure 7: k-NN Collaborative Filter (kNN with Means) with 10 fold CV on Unpopular Movie Trimming

In our testing, the minimum average RMSE using Unpopular Movie Trimming was 1.109157853482604, which happened at k=38. From the figure above, we can see that the average RMSE using unpopular movie trimming does not have a clear trend and fluctuates in a range between 1.109 and 1.123 as we sweep k from 2 to 100. This differs from what we observed using the original data or using popular movie trimming, where the average RMSE generally decreased as k increased, until we reached a point of diminishing returns.

Also, the error values seen using unpopular movie trimming are much higher than what we observed using the original dataset or using popular movie trimming. This makes sense because by performing unpopular movie trimming, we inherently only keep movies that have received 2 or fewer ratings. This makes the ratings matrix even more sparse. Any user in the test set with an actual rating for a particular movie will have at most 1 other actual rating in their k nearest neighbors that could be used to calculate a prediction or recommendation. Because of this, the recommendation system has little to go off of and will thus make a less reliable recommendation. More data points would likely cause the recommendation system to be more accurate because more calculations could be made and cross-checked, but the increased sparseness here makes that not possible.

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.

<u>Answer:</u>

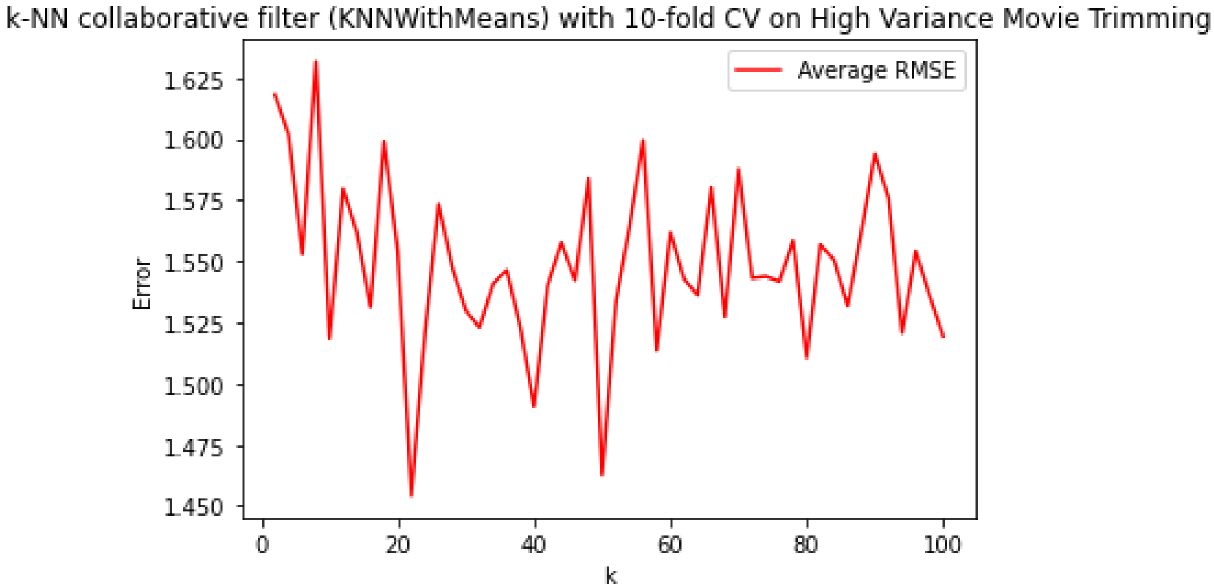

Figure 8: k-NN Collaborative Filter (kNN with Means) with 10 fold CV on High Variance Movie Trimming

In our testing, the minimum average RMSE using High Variance Movie Trimming was 1.454116371848785, which occurred at k=22. Similar to what we saw when using unpopular movie trimming, there does not appear to be a discernible trend in the average RMSE when using high variance movie trimming.

Moreover, the error values here are higher than anything we have seen so far. This makes sense because by using high variance movie trimming, we inherently pick the movies for which the ratings have a high variance and therefore we would expect the error values to be high. With a high variance in ratings, the ratings of the k-nearest neighbors to a user u would have a larger range. This causes the adjustment term in the prediction function that takes into account the ratings of the k-nearest neighbors to be less useful and perhaps even misleading when attempting to predict the missing ratings.

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.

<u>Answer:</u>

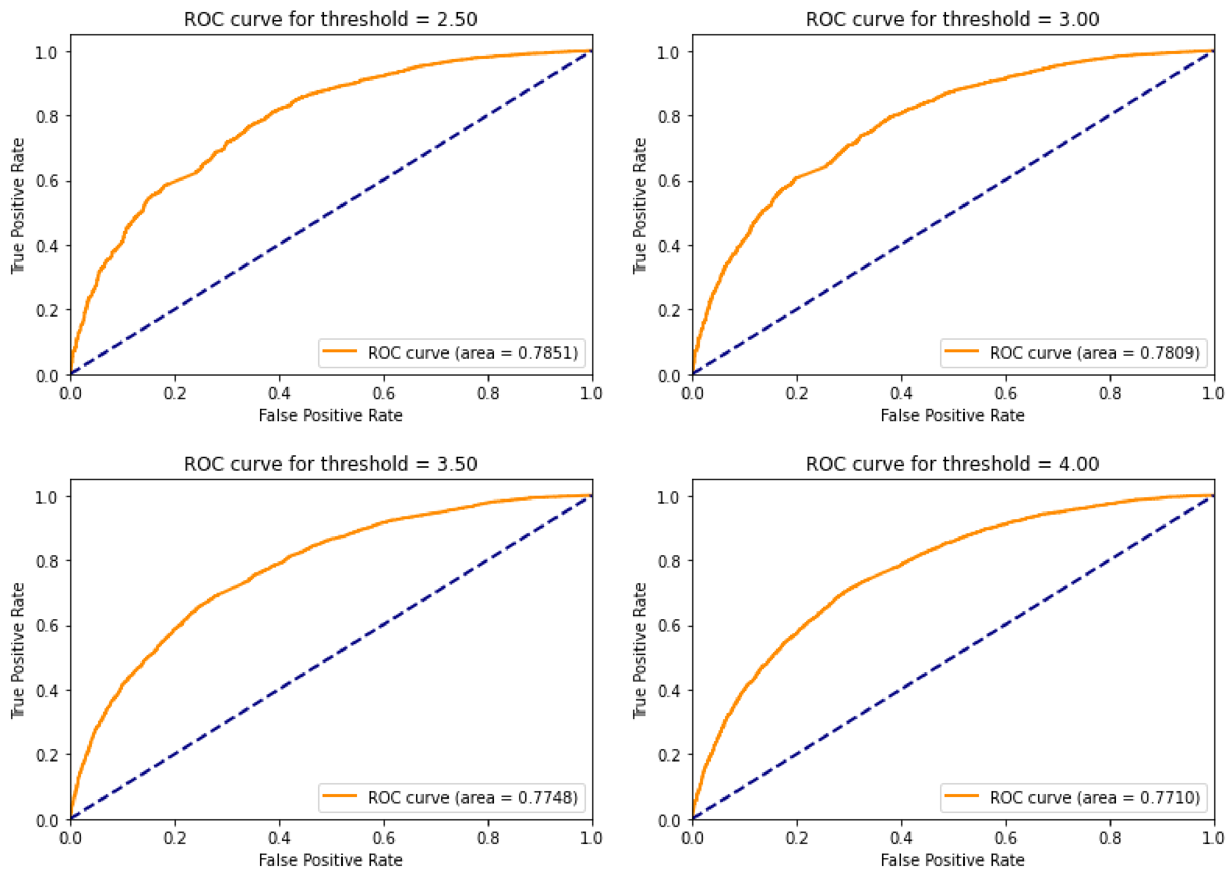

Figure 9: ROC Curves for the k-NN Collaborative Filter Using Different Thresholds

Using k=22, which is the k-value we picked in Question 11, we plotted the ROC curves for different thresholds. Here, threshold=2.50 had the highest area under the curve (NOTE: The threshold that has the highest area under the curve may change depending on the random_state used).

Threshold	Area Under the Curve (AUC)
2.5	0.7851
3.0	0.7809
3.5	0.7748
4.0	0.7710

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.

Answer:
The optimization problem given by equation 5 is:

$$\underset{U,V}{\text{minimize}} \quad \sum_{i=1}^{m}\sum_{j=1}^{n} W_{ij}(r_{ij} - (UV^T)_{ij})^2$$

Given that we are trying to find a minimum in this optimization problem, if this matrix function were convex, its Hessian matrix would be positive semi-definite. Therefore, we will calculate the Hessian matrix (second derivative) of this matrix function and see if it is positive semi-definite or not.

For simplicity, we will consider the case where m = n = 1 (The results will extend out and apply to larger values of m and n as well). Then all our matrices (W, r, U, V) are 1x1 scalars. We will also assume here that W = 1 (The values in the W matrix are by definition either 0 or 1. Since we are dealing with 1x1 scalars now, a value of W = 0 would make the whole equation equal 0, which is a trivial and uninteresting case. The cases where W = 1 (where there was an actual rating by an actual user) are the cases we are particularly interested in). With these simplifications, the matrix function in the optimization problem can be written as:

$$L(U,V) = (r - UV)^2 = r^2 - 2rUV + U^2V^2$$

Calculating the Gradient and Hessian matrix of this function L with respect to both U and V, we have:

$$\nabla L(U,V) = \begin{bmatrix} \frac{\partial L}{\partial U} \\ \frac{\partial L}{\partial V} \end{bmatrix} = \begin{bmatrix} -2rV + 2V^2U \\ -2rU + 2U^2V \end{bmatrix}$$

$$\nabla^2 L(U,V) = \begin{bmatrix} \frac{\partial^2 L}{\partial U^2} & \frac{\partial^2 L}{\partial U\,\partial V} \\ \frac{\partial^2 L}{\partial V\,\partial U} & \frac{\partial^2 L}{\partial V^2} \end{bmatrix} = \begin{bmatrix} 2V^2 & -2r + 4UV \\ -2r + 4UV & 2U^2 \end{bmatrix}$$

The determinant of the Hessian matrix is:

$$
\begin{aligned}
|\nabla^2 L(U,V)| &= 4U^2V^2 - \ (-2r + 4UV)^2 \\
&= 4U^2V^2 - (4r^2 - 16rUV + 16U^2V^2\ ) \\
&= 4[U^2V^2 - r^2 + 4rUV - 4U^2V^2] \\
&= 4[-r^2 + 4rUV - 3U^2V^2] \\
&= -4[r^2 - 4r(UV) + 3(UV)^2] \\
&= -4[r - 3(UV)][r - (UV)]
\end{aligned}
$$

This equation is not always positive or nonnegative for all values of r, U, V > 0 (We can assume U, V > 0 because we are attempting to find a non-negative matrix factorization. And we can assume r > 0 because all the ratings are positive (between 0.5 and 5)).
For example, if r=1, U=1, V=1, this equation evaluates to = -4[-2][0] = 0
If r=5, U=1, V=1, this equation evaluates to = -4[2][4] = -32
If r=5, U=1, V=2, this equation evaluates to = -4[-1][3] = 12

The determinant of a positive semidefinite matrix would always evaluate to be greater than or equal to 0. However, since that is not always the case for the determinant of the Hessian matrix we have found, we can conclude that our Hessian matrix is not positive semidefinite, and therefore our original matrix equation is <u>NOT CONVEX</u>.

Now, given a fixed U, we would like to formulate this optimization problem as a least-squares problem so that it can be solved using an alternating least-squares (ALS) algorithm. Using the following notation:

$$r_j = \ \left[r_{1j}, \dots, r_{mj}\right]^T ;\ V_j = \ \left[V_{1j}, \dots, V_{kj}\right]^T ;\ W_j = \ diag\left\{W_{1j}, \dots, W_{mj}\right\} ; \quad (j = 1, \dots n)$$

We can rewrite the optimization problem as (minimizing over V):

$$L(V) = \sum_{j=1}^{n} W_j\left(r_j - UV_j\right)^2 = \ \sum_{j=1}^{n} \left(r_j - UV_j\right)^T W_j(r_j - UV_j)$$

Since we are interested in finding the minimum, we want the terms in the summation to be as close to 0 as possible. The two major terms in the summation, which are the two terms in parentheses, are essentially equivalent to each other since they are just the transposes of each other. Therefore, finding the condition that causes one of the terms to be as close to 0 as possible is sufficient. Taking the second term, and folding in the weighting term, W_j, the condition we want to solve for is:

$$W_j\left(r_j - UV_j\right) = \ 0$$

Solving for V_j, we get:

$$
\begin{aligned}
W_j r_j - W_j U V_j &= \ 0 \\
W_j r_j &= W_j U V_j \\
U^T W_j r_j &= U^T W_j U V_j \\
V_j &= (U^T W_j U)^{-1} U^T W_j r_j \ ; \quad (j = 1, \dots n)
\end{aligned}
$$

The equation we got for V_j is the formula for the (Weighted) Least-Squares Estimator. Therefore, for a fixed U, we have formulated this optimization problem as a least-squares problem.

Similarly, if V is fixed, then we can solve for U using a least-squares method as follows:

$$U_i = (V^T W_i V)^{-1} V^T W_i r_i \ ; \quad (i = 1, \dots m)$$

Where:

$$r_i = [r_{i1}, \dots, r_{in}]^T \ ; \ U_i = [U_{i1}, \dots, U_{ik}]^T \ ; \ W_i = diag\{W_{i1}, \dots, W_{in}\} \ ; \quad (i = 1, \dots, m)$$

With these two equations, we can then solve this NNMF problem using alternating least-squares (ALS) as the optimization algorithm. First, we would keep U fixed and solve for V. Then, in the next step, we would keep V fixed and solve for U. In each step, we are solving a least-squares problem, until eventually the solution converges.

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-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.

Answer:

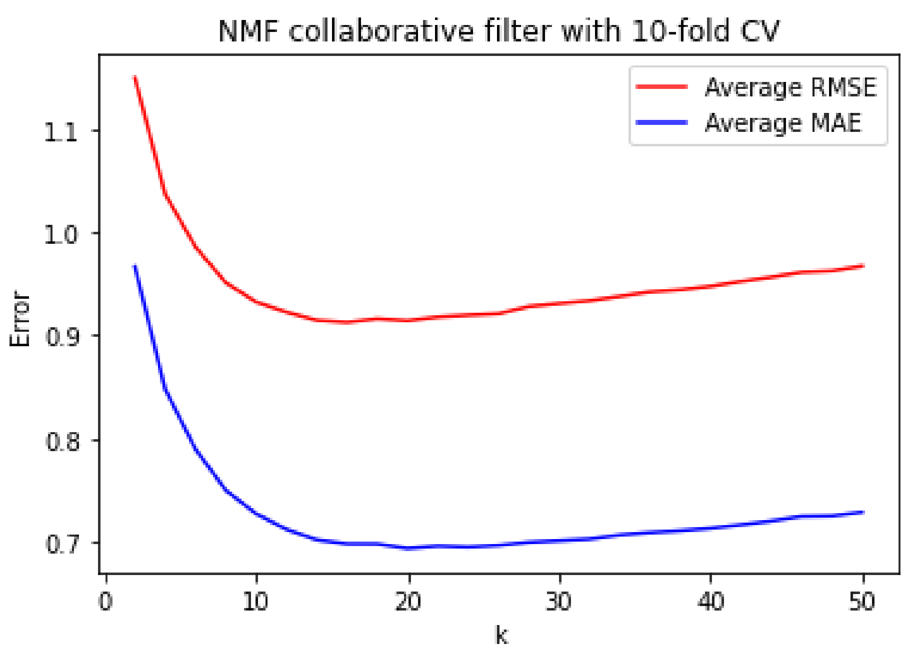

Figure 10: NMF Collaborative Filter with 10 Fold CV

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?

Answer:
From our results in question 17, the minimum average RMSE was 0.912733, which happened at k=16. The minimum average MAE was 0.693586, which happened at k=20. The number of movies in the dataset as mentioned by the README.txt that came with the dataset is 19. Thus, the optimal number of latent factors is roughly about the same as the number of movie genres. Since we have been using minimum average RMSE as our main measure of error across the different questions, we will pick k=16 to minimize the minimum average RMSE.

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.

Answer:

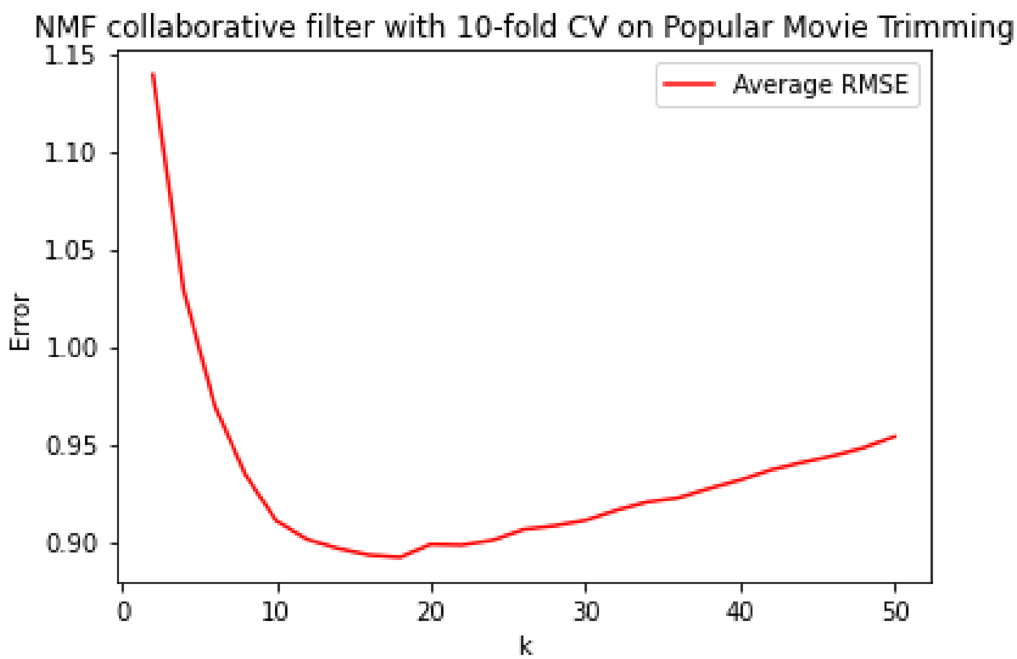

Figure 11: NNMF Collaborative Filter with 10 Fold CV on Popular Movie Trimming

The minimum average RMSE using NNMF and Popular Movie Trimming was 0.8920038, which happened at k=18. The shape of the curve is similar to NNMF without any trimming. Notably, using popular movie trimming caused the minimum average RMSE to go down slightly here compared to NNMF without any trimming. This matches what we saw earlier in Questions 11 and 12, where popular movie trimming also slightly reduced the minimum average RMSE when we were using a KNNWithMeans collaborative filter.

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

Answer:

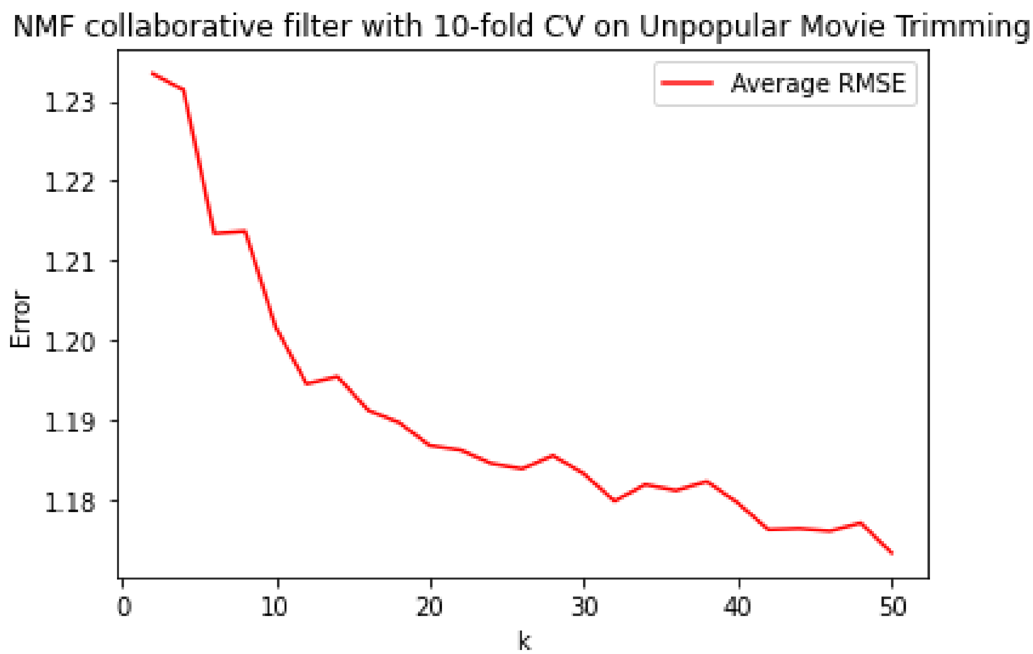

Figure 12: NNMF Collaborative Filter with 10 Fold CV on Unpopular Movie Trimming

The minimum average RMSE using NNMF and Unpopular Movie Trimming was 1.173233, which happened at k=50. Here, we observe that in general, the average RMSE appears to decrease with increasing k. This is in contrast to what we saw when we used a KNNWithMeans collaborative filter with unpopular movie trimming. In that case, there did not seem to be any discernible pattern in average RMSE with respect to k. It is also worth noting that comparing both cases where a NNMF collaborative filter was used, popular movie trimming produced a lower minimum average RMSE than unpopular movie trimming did.

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

Answer:

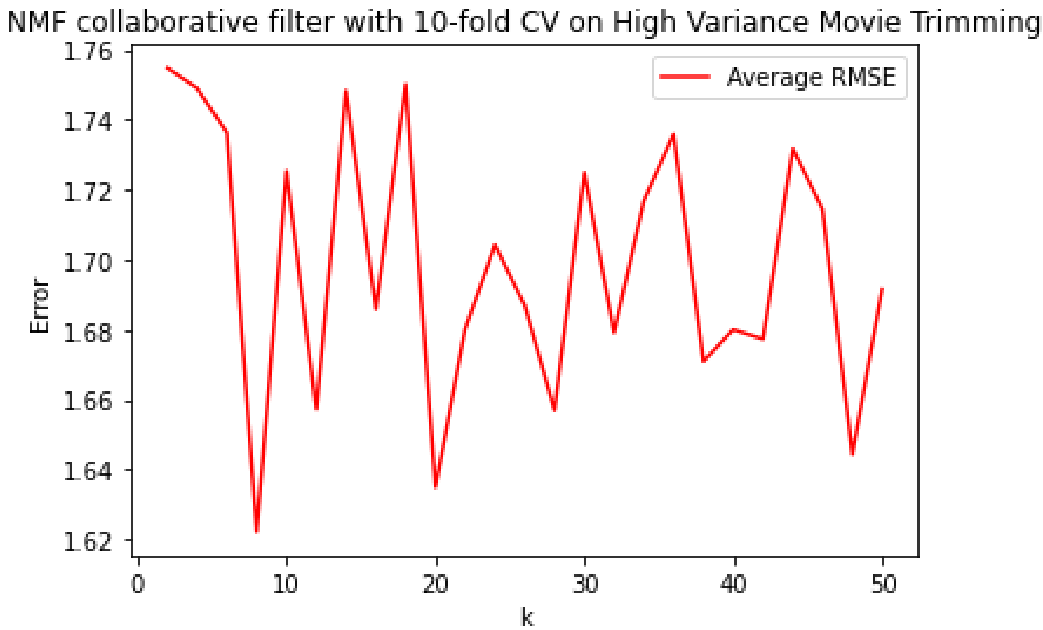

Figure 13: NNMF Collaborative Filter with 10 Fold CV on High Variance Movie Trimming

The minimum average RMSE using NNMF and High Variance Movie Trimming was 1.622177, which happened at k=8. There does not seem to be a pattern in average RMSE with respect to k. Also, comparing all cases where an NNMF collaborative filter was used, high variance movie trimming had a higher average RMSE than both popular movie trimming and unpopular movie trimming. These observations line up with what we had observed when using high variance movie trimming on a KNNWithMeans collaborative filter.

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.

Answer:

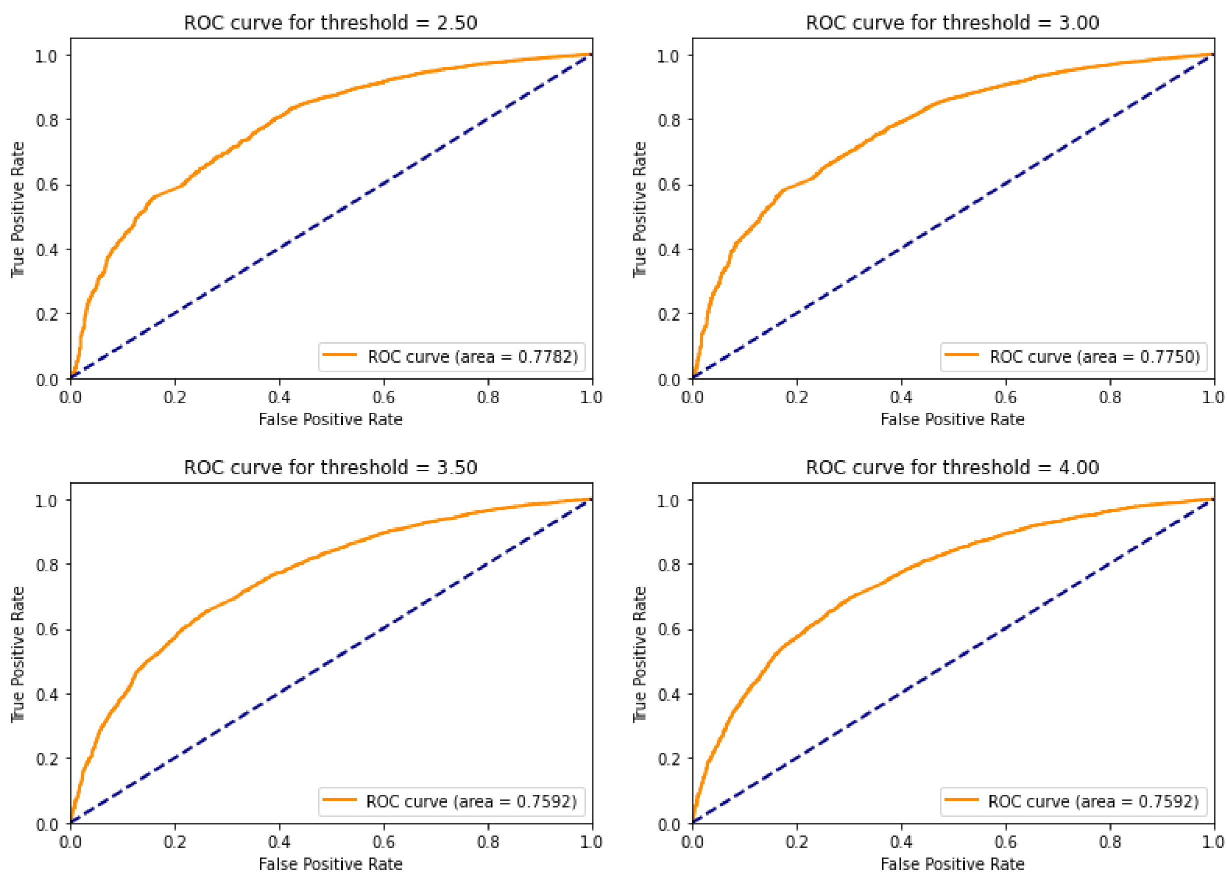

Figure 14: ROC Curves for the NNMF Collaborative Filter Using Different Thresholds

Using the optimal number of latent factors we found in Question 18, which was n_factors=16, we plotted the ROC curves using an NNMF collaborative filter for different thresholds. Here, threshold=2.50 had the highest area under the curve (NOTE: The threshold that has the highest area under the curve may change depending on the random_state used).

Threshold	Area Under the Curve (AUC)
2.5	0.7782
3.0	0.7750
3.5	0.7592
4.0	0.7592

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?

Answer:
The top 10 movies in each column of V (or movie-latent factor) tend to belong to a small collection of genres. This result leads us to believe that there is some connection between the latent factors and the movie genres. While the mapping is not strictly one-to-one, we can observe that certain latent factors have one or two genres that are heavily represented in the top 10 movies of that movie-latent factor (or column in V). Some example columns are given below with the genre(s) that are most heavily represented in the top 10 movies of those columns, especially relative to the other columns. In general, we can conclude that each latent factor represents a small collection of genres that form something approximating a principal component in the data set. This suggests that there is some natural correlation or grouping between the heavily represented genres in each latent factor.

Column 3 (Romance)	Column 5 (Action, Adventure)	Column 9 (Sci-Fi, Thriller)
Drama|Musical|Romance Action|Crime|Drama|Thriller Horror|Thriller Comedy|Romance Adventure|Animation|Children|Comedy|Fantasy|Musical|Romance Action|Horror|Thriller Action|Adventure|Comedy Comedy|Drama|Romance Comedy|Drama|Romance Drama	Action|Adventure|Thriller Drama|War Comedy Comedy|Crime|Thriller Drama|Mystery Drama|Romance Action|Adventure Drama Action|Animation|Crime|Sci-Fi|Thriller Adventure|Comedy|Drama|Romance	Sci-Fi|Thriller Comedy Crime|Drama|Thriller Drama Sci-Fi|Thriller Action|Adventure|Animation|Fantasy Drama Comedy|Drama|Musical Drama|Thriller Adventure|Comedy|Sci-Fi
Column 10 (Comedy)	Column 12 (Children, Animation)	Column 15 (Drama)
Comedy Comedy|Drama Children|Comedy|Drama Comedy Crime|Film-Noir Animation|Comedy Crime Comedy Drama|Mystery|Sci-Fi Action|Animation|Comedy|Horror|Thriller	Animation|Comedy Adventure|Children|Drama Drama Comedy|Romance Drama|Romance Action|Adventure|Children|Comedy Adventure|Animation|Children Comedy|Drama|Musical|Romance Comedy|Romance Horror	Drama Drama Comedy Sci-Fi Action|Comedy|War Drama Drama Comedy|Drama|Thriller Animation|Comedy|Fantasy Horror

QUESTION 24:
Design a MF with bias 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 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.

Answer:

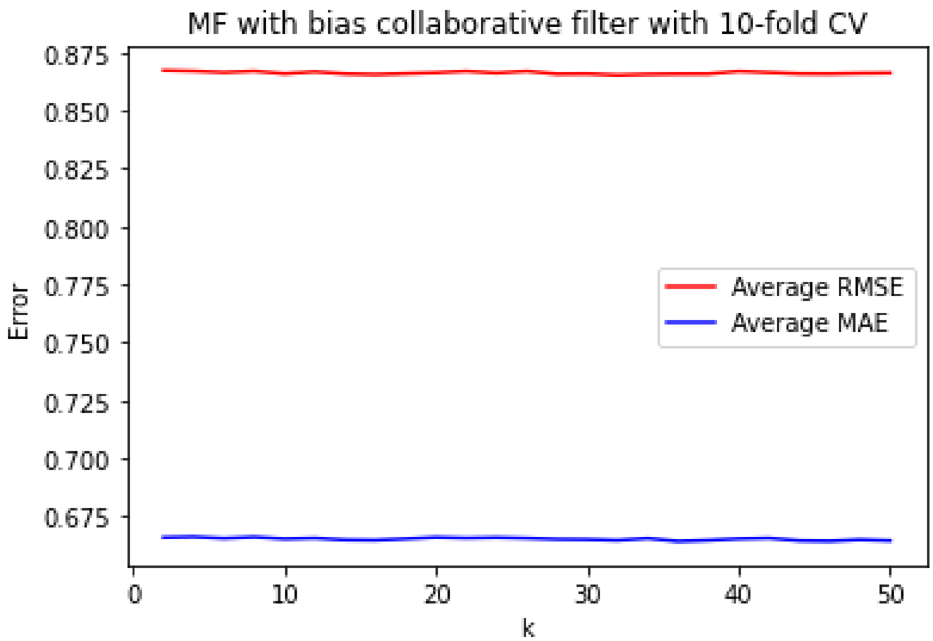

Figure 15: MF with Bias Collaborative Filter with 10 Fold CV

There isn’t much variation in error across the k values in both RMSE and MAE averages.

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.

<u>Answer:</u>

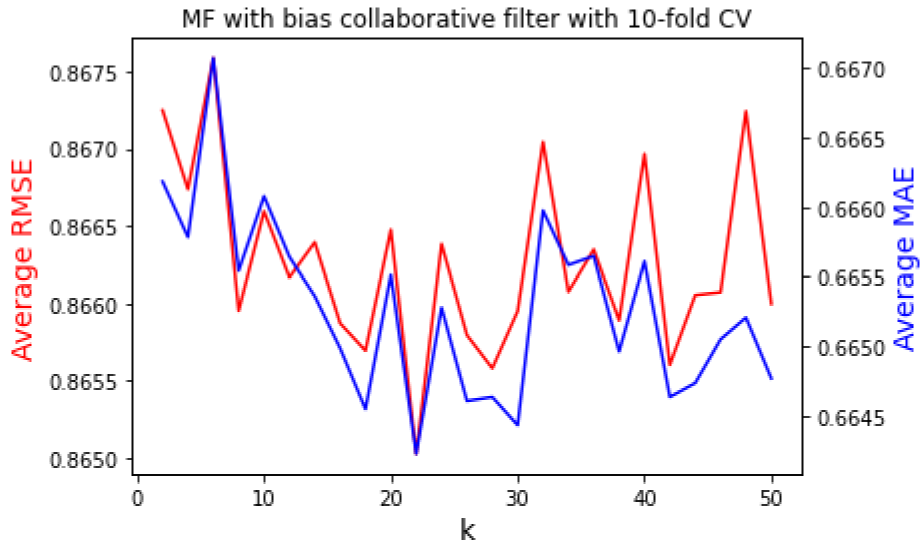

Figure 16: MF with Bias Collaborative Filter with 10 Fold CV - different y-axis

“Zooming in” on the two curves as shown in figure 22 above shows a similar trend of the average RMSE and average MAE across the k values. There is an initial downward trend, with **both having a minimum average at k=22**, then a slight increase and plateau.

Minimum average RMSE is 0.8652083390716919
Minimum average MAE s 0.6642001956535919

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

Answer:

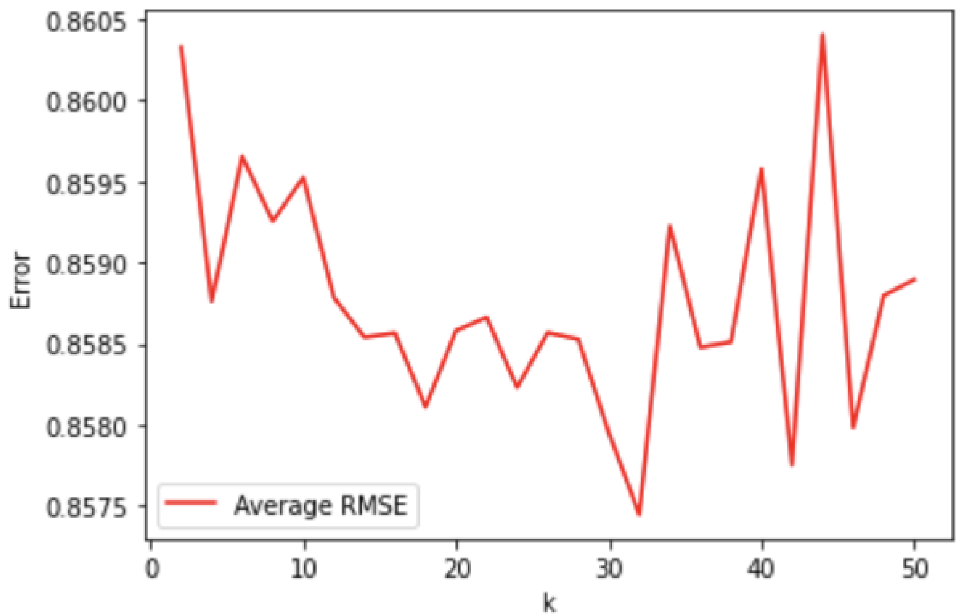

Figure 17: MF with Bias Collaborative Filter with 10 Fold on Popular Movie Trimming

Minimum average RMSE for Popular Movie Trimming is 0.8574484812310328
As seen before, trimming the dataset to movies with >2 reviews slightly improves our minimum average RMSE from ~0.865 with the entire dataset with bias to ~0.857 with only the popular movies. In fact, the average RMSE across all k-values in the popular movies set is lower than the minimum average RMSE of the entire dataset and has little variation with k.

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 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

Answer:

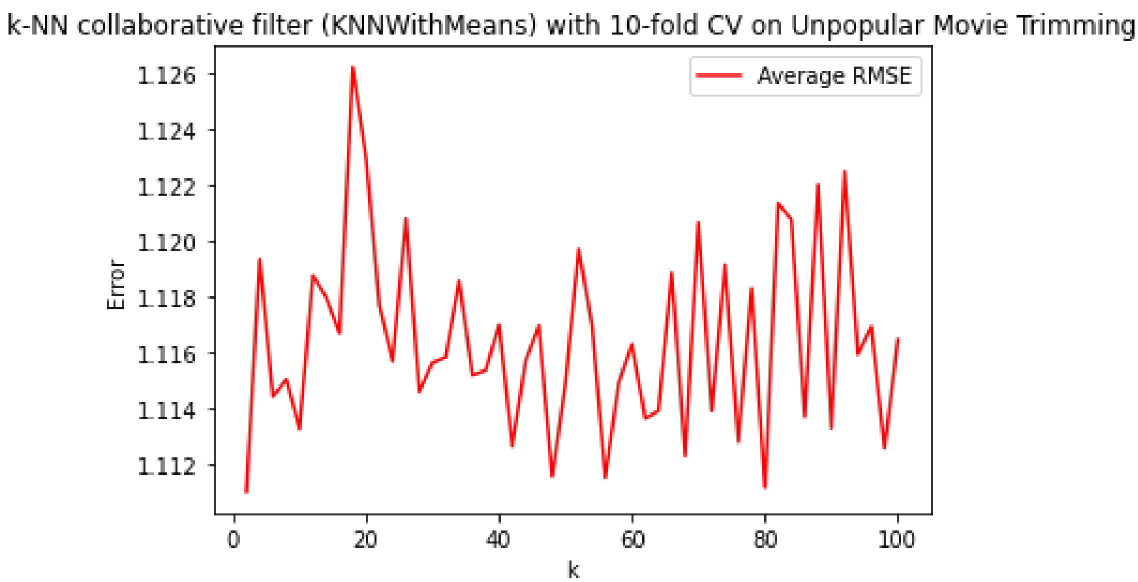

Figure 18: MF with Bias Collaborative Filter with 10 Fold on Unpopular Movie Trimming

Minimum average RMSE for Unpopular Movie Trimming is 0.9706730162042637
Again, as seen before, trimming to unpopular movies gives us a higher Average RMSE across the board.

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

Answer:

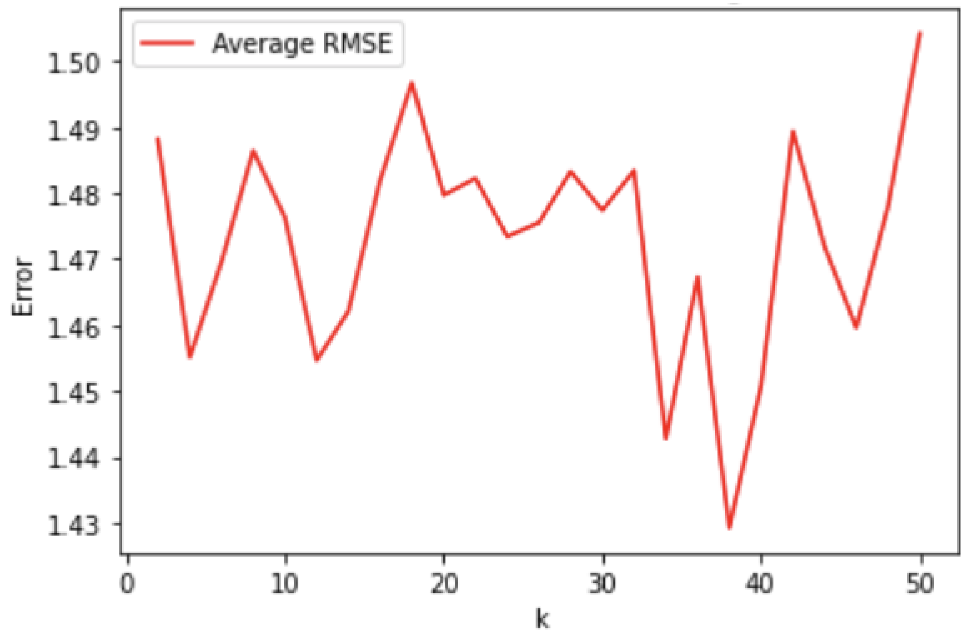

Figure 19: MF with Bias Collaborative Filter with 10 Fold on High Variance Movie Trimming

Minimum average RMSE for High variance Movie Trimming is 1.4292730381587788
Trimming to high variance movies dataset yields the worst (highest) RMSE as expected intuitively and explained earlier.

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.

Answer:
The following ROC curves were made using the optimal number of latent factors found in question 25 (k=22):

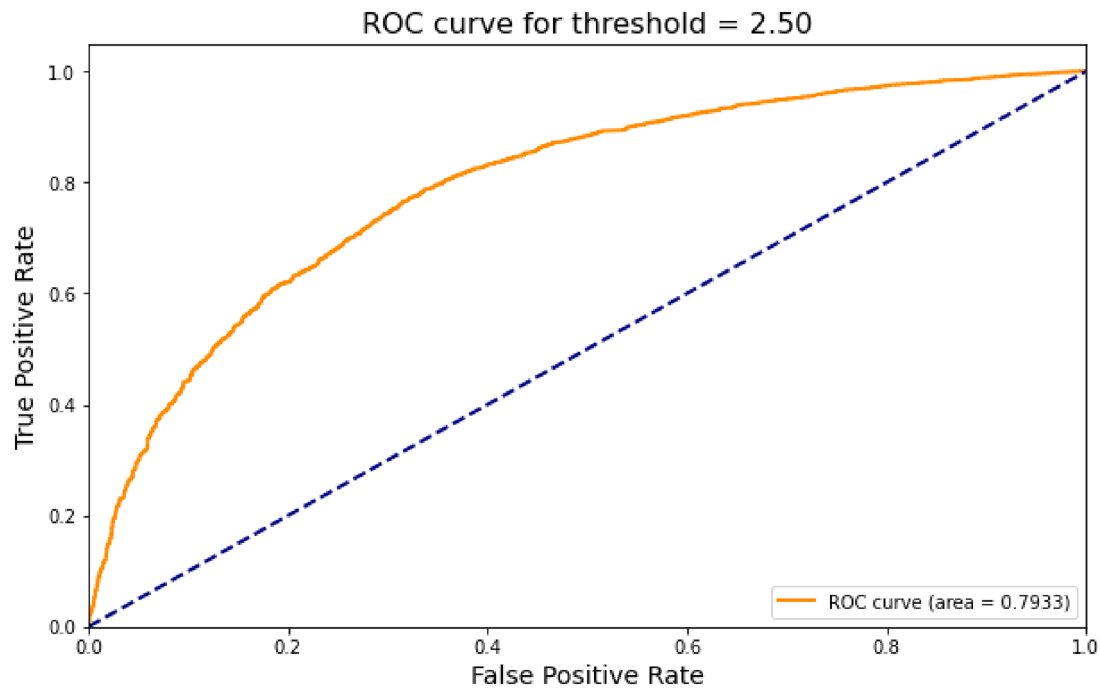

Figure 20: ROC Curve for Threshold = 2.50

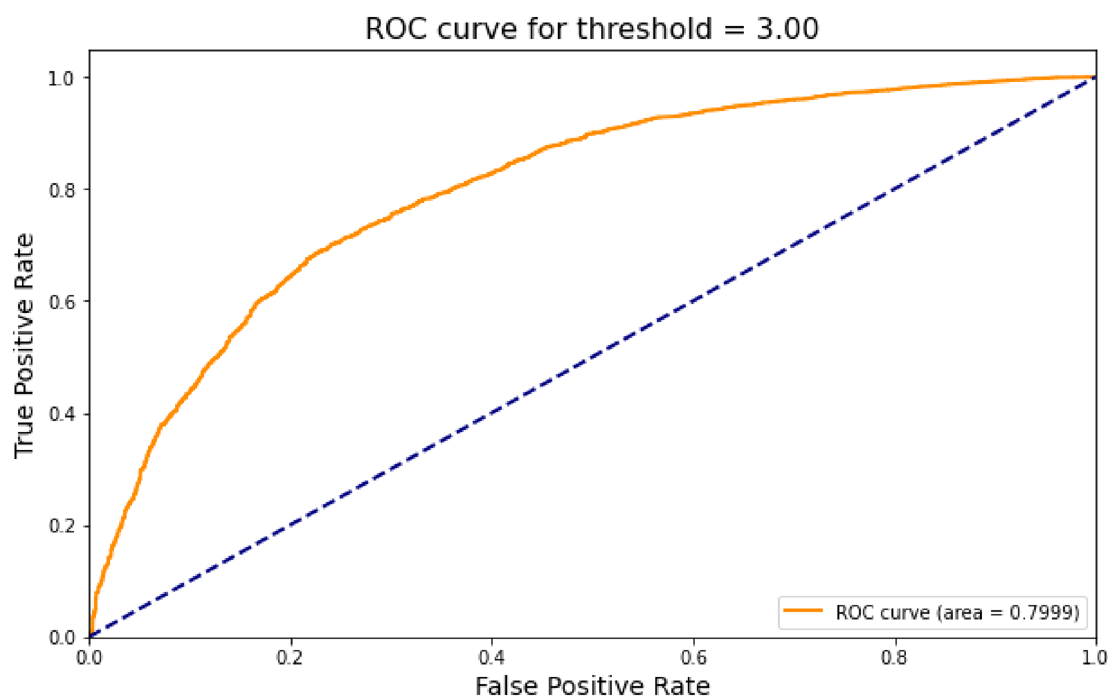

Figure 21: ROC Curve for Threshold = 3.00

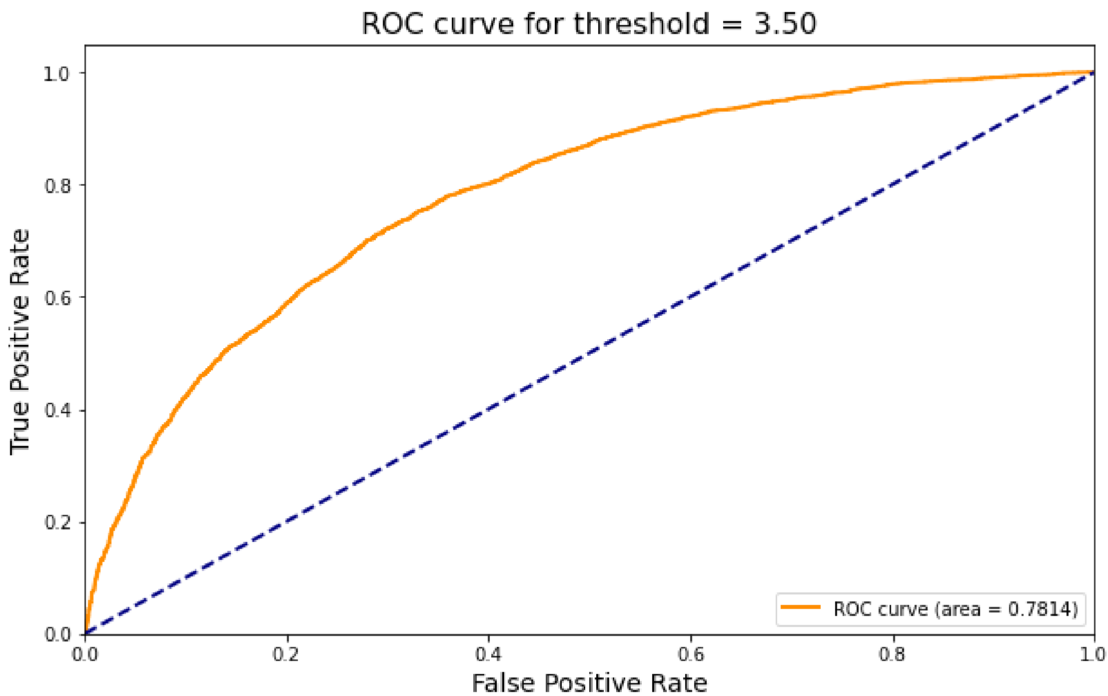

Figure 22: ROC Curve for Threshold = 3.50

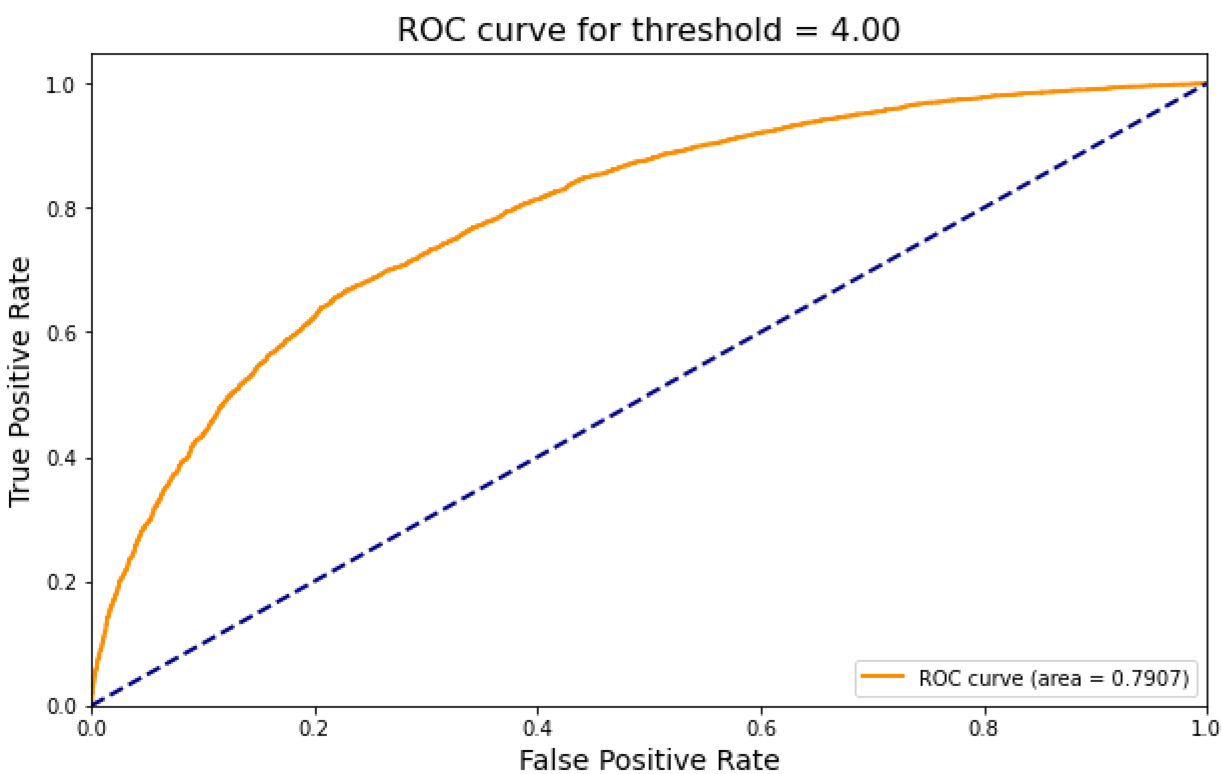

Figure 23: ROC Curve for Threshold = 4.00

Below is a summary of the Area Under the Curve for the ROC Curves above for Matrix Factorization with bias collaborative filter:

Threshold	Area Under the Curve (AUC)
2.5	0.7933
3.0	0.7999
3.5	0.7814
4.0	0.7907

QUESTION 30:
Design a naive collaborative filter to predict the ratings of the movies in the MovieLens dataset 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.
Note that in this case, when performing the cross-validation, there is no need to calculate μ_i's for the training folds each time. You are only asked to use a single set of μ_i's calculated on the entire dataset and validate on 10 validation folds.

<u>Answer:</u>
Average RMSE for naive collaborative filter is 0.9347033620738958

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.

<u>Answer:</u>
Average RMSE for naive collaborative filter with Popular Movie Trimming is 0.9323124182963666. The error is slightly lower than the Average RMSE for the entire dataset as expected.

QUESTION 32:
Design a naive 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. Compute the average RMSE by averaging the RMSE across all 10 folds. Report the average RMSE.

<u>Answer:</u>
Average RMSE for naive collaborative filter with Unpopular Movie Trimming is 0.9714753648149612. The error is somewhat higher than the Average RMSE for the entire dataset as expected.

QUESTION 33:
Design a naive 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. Compute the average RMSE by averaging the RMSE across all 10 folds. Report the average RMSE.

Answer:
Average RMSE for naive collaborative filter with High Variance Movie Trimming is 1.4807566584608713. The error is much higher than the Average RMSE for the entire dataset as expected.

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.

Answer:

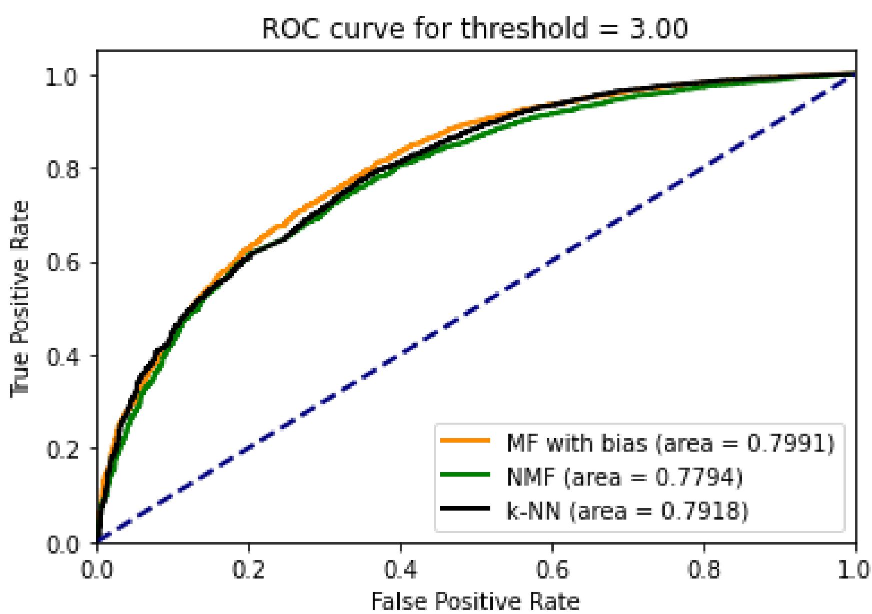

Figure 24: ROC Curve for Threshold = 3.00

MF with bias seems to have the smoothest curve and has the largest area under the curve when threshold = 3. This means that MF with bias is the best among the three collaborative filters at predicting the ratings of the movies.

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.

Answer:

$$Precision(t) = \frac{|S(t) \cap G|}{|S(t)|}$$

Precision is the fraction of items that the user liked out of the items that were recommended to the user, it is used to look at whether our recommendations were useful to the user.

$$Recall(t) = \frac{|S(t) \cap G|}{|G|}$$

Recall is the fraction of recommended items that the user likes out of all the items that the user likes, it is used to evaluate how many of the recommendations actually recalled what the user likes.

QUESTION 36:
Plot average precision (Y-axis) against *t* (X-axis) for the ranking 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.

Answer:

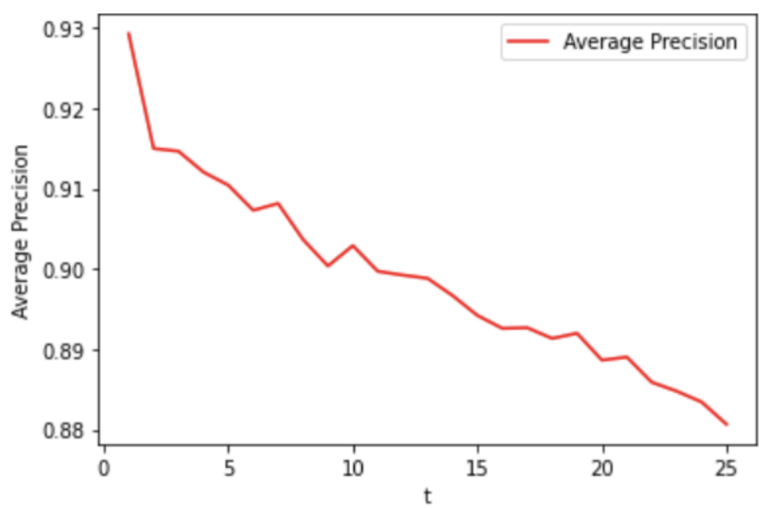

Figure 25: k-NN Collaborative Filter (kNN with Means) with 10 Fold CV for Average Precision

As t increases, the average precision decreases generally. There is an inverse relationship between t and average precision.

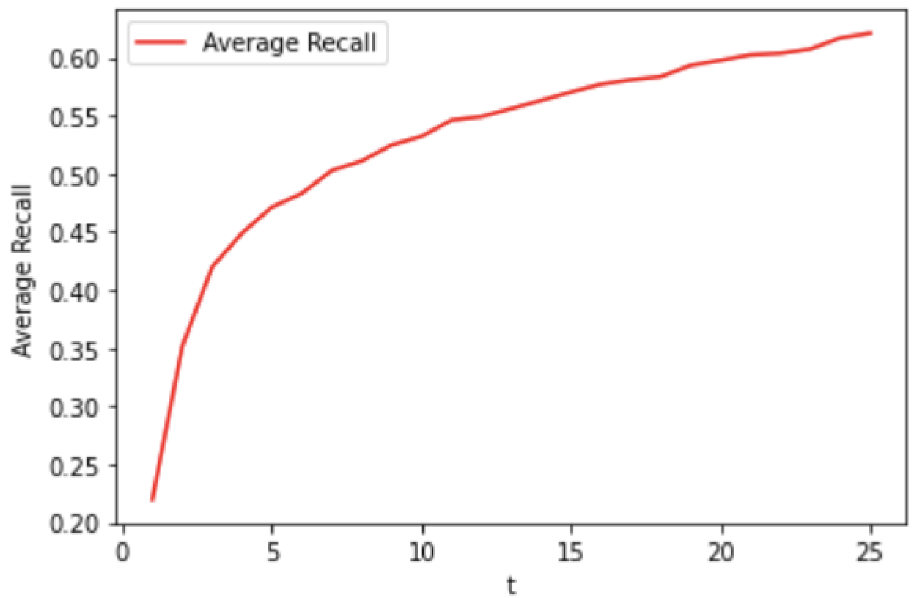

Figure 26: k-NN Collaborative Filter (kNN with Means) with 10 Fold CV for Average Recall

As t increases, the average recall increases. There is a positive, increasing relationship between t and average recall.

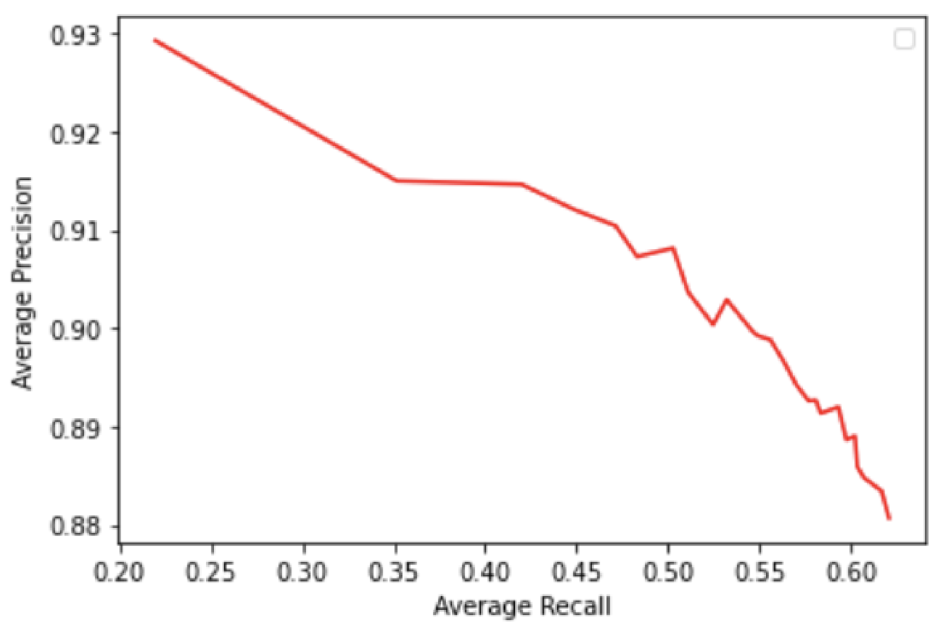

Figure 27: k-NN Collaborative Filter (kNN with Means) with 10 Fold CV for Average Precision VS Average Recall

As average recall increases, the average precision decreases. There is an inverse relationship between the two variables. This shows that there is a trade-off. Both variables cannot be maximized simultaneously and there needs to be a compromise.

QUESTION 37:
Plot average precision (Y-axis) against *t* (X-axis) for the ranking 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.

Answer:

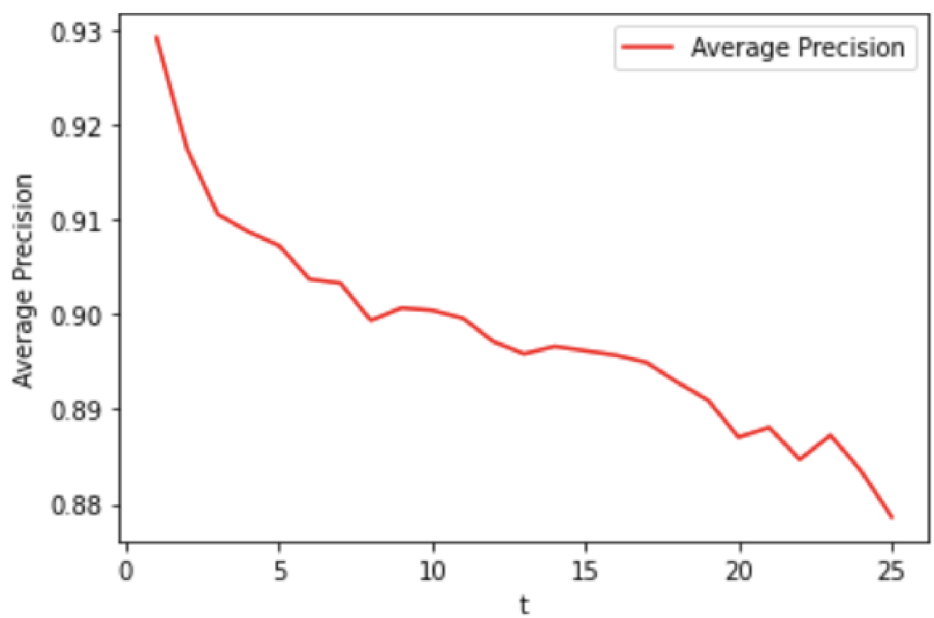

Figure 28: NMF Collaborative Filter with 10 Fold CV for Average Precision

As t increases, the average precision decreases generally. There is an inverse relationship between t and average precision.

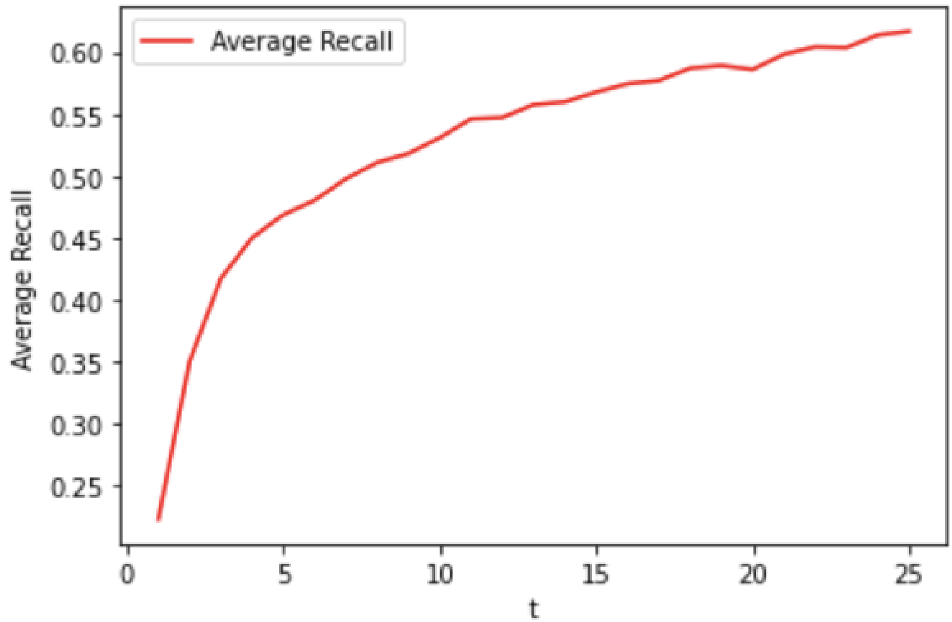

Figure 29: NMF Collaborative Filter with 10 Fold CV for Average Recall

As t increases, the average recall increases. There is a positive, increasing relationship between t and average recall.

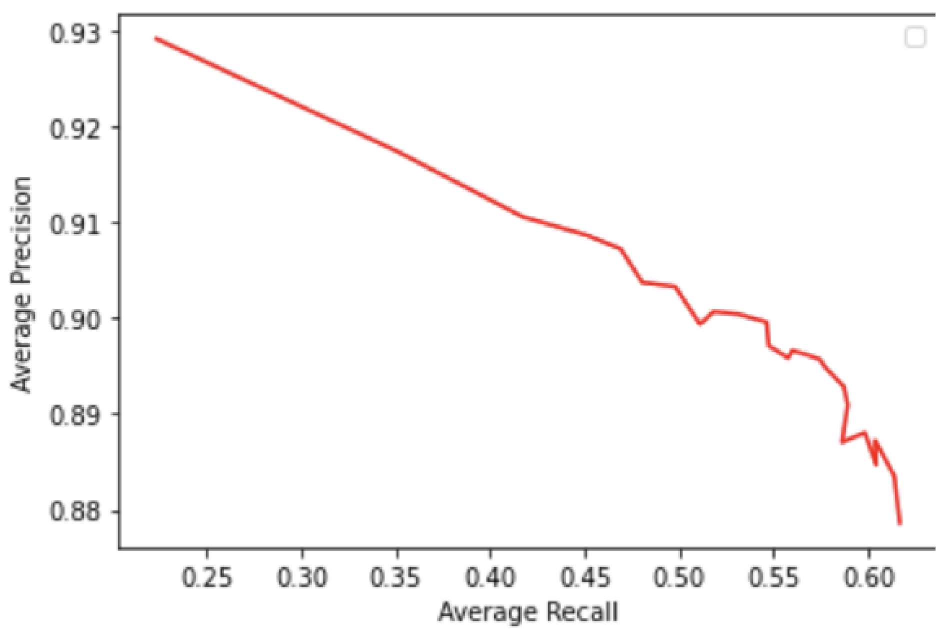

Figure 30: NMF Collaborative Filter with 10 Fold CV for Average Recall vs Average Precision

As average recall increases, the average precision decreases. There is an inverse relationship between the two variables. This shows that there is a trade-off. Both variables cannot be maximized simultaneously and there needs to be a compromise.

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.

Answer:

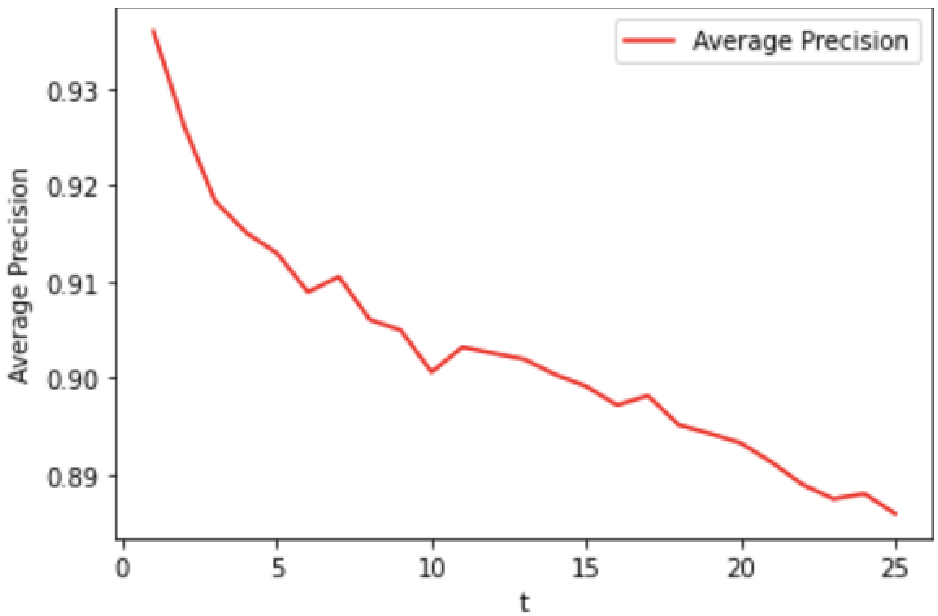

Figure 31: MF Collaborative Filter with 10 Fold CV for Average Precision

As t increases, the average precision decreases generally. There is an inverse relationship between t and average precision.

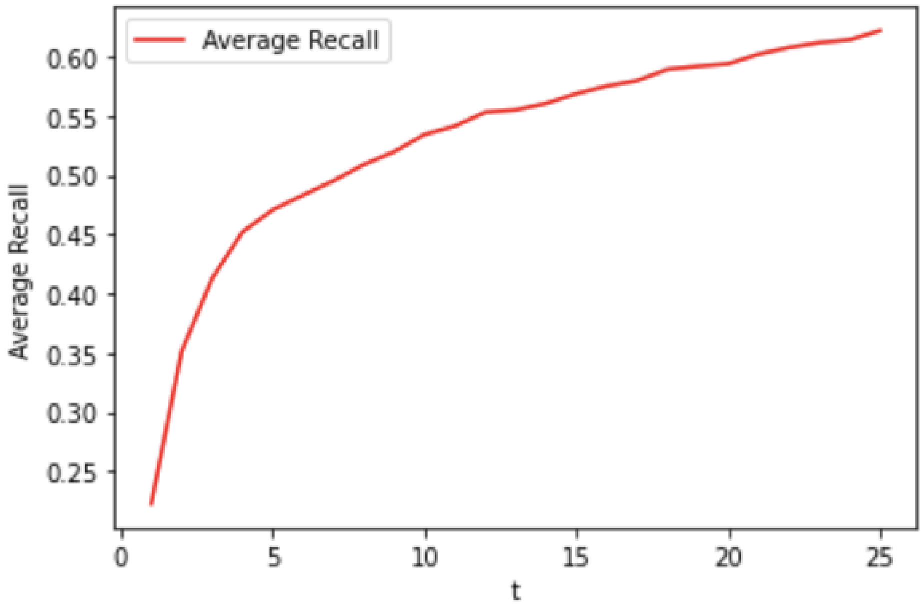

Figure 32: MF Collaborative Filter with 10 Fold CV for Average Recall

As t increases, the average recall increases. There is a positive, increasing relationship between t and average recall.

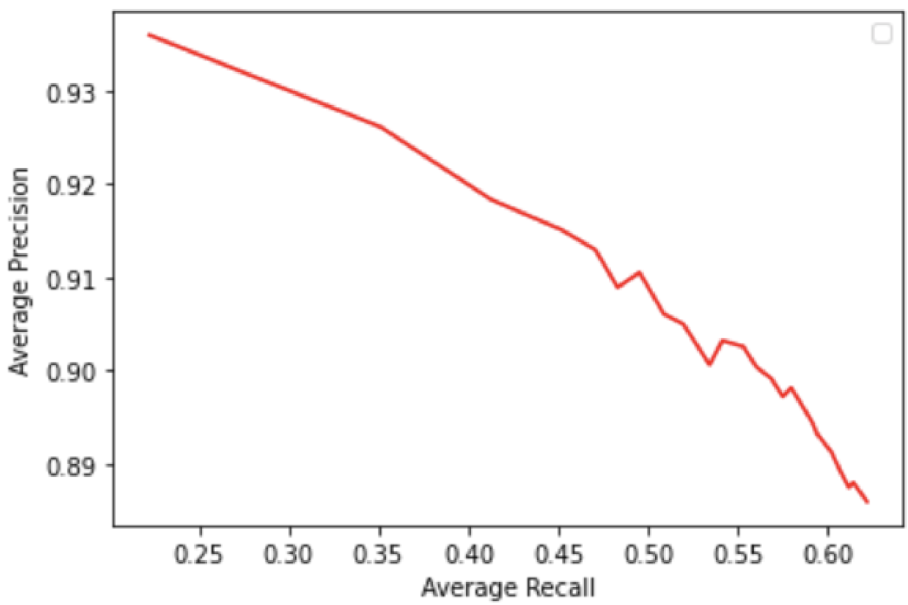

Figure 33: MF Collaborative Filter with 10 Fold CV for Average Recall vs Average Precision

As average recall increases, the average precision decreases. There is an inverse relationship between the two variables. This shows that there is a trade-off. Both variables cannot be maximized simultaneously and there needs to be a compromise.The trend of the above plots is similar to Question 36 and Question 37, however, the plot for MF seems to be smoother.

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.

<u>Answer:</u>

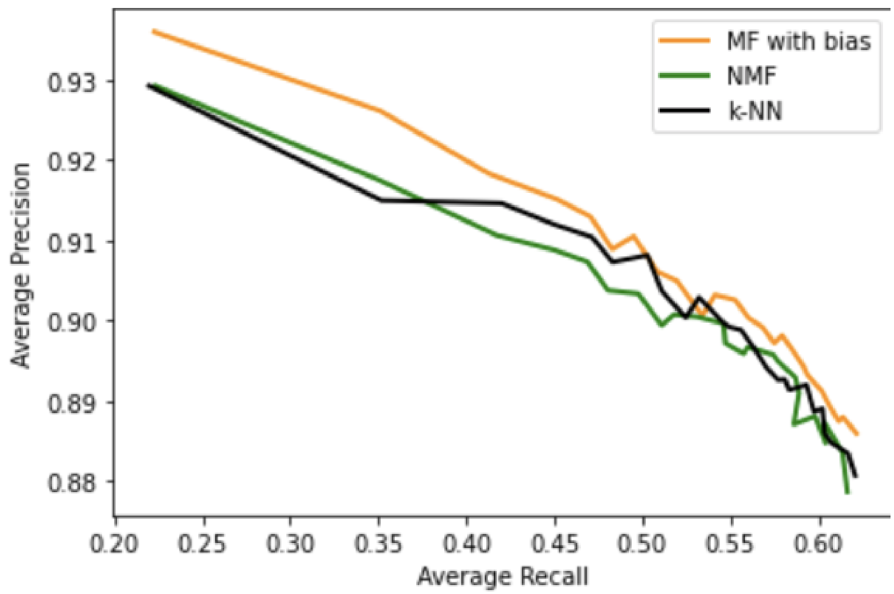

Figure 34: Comparison of k-NN, NMF, and MF with Bias

The precision-recall curves for all three prediction algorithms show an inverse relationship between average precision and recall. However, MF with bias shows the best performance since it has the smallest slope. This means that average recall can be increased with only small changes in the average precision.