Project 3: Collaborative Filtering

Lumi Huang, Christian Warloe, Ke Zhao, Landi Luo

1 Introduction

Recommender systems play a major role in today's e-commerce industry. Many e-commerce and retail companies are leveraging the power of data and boosting sales by implementing recommender systems on their websites. The basic idea behind recommender systems is that they aim to predict users' interests and recommend items that quite likely are interesting for them, based on the users' previous data (explicit user ratings after watching a movie or listening to a song, implicit search engine queries and purchase histories, or from other knowledge about the users/items themselves). Sites like Spotify, YouTube or Netflix use that data in order to suggest playlists or to make video/movie recommendations.

2 Collaborative Filtering Models

In this project, we will build a recommendation system using collaborative filtering methods. Collaborative filtering models use *user-item interactions* data, such as ratings or number of purchases. We will implement and analyze the performance of two types of collaborative filtering methods:

1. Neighborhood-based collaborative filtering
2. Model-based collaborative filtering

3 MovieLens dataset

In this project, we used the `MovieLens` dataset to build a recommendation system to predict the ratings of the movies in the dataset. The dataset can be downloaded using the following link:

http://files.grouplens.org/datasets/movielens/ml-latest-small.zip

This dataset (ml-latest-small) describes 5-star rating and free-text tagging activity from MovieLens, a movie recommendation service. It contains 100836 ratings and 3683 tag applications across 9742 movies. These data were created by 610 users between March 29, 1996 and September 24, 2018. This dataset was generated on September 26, 2018.

We first built the ratings matrix R as an *m x n* matrix containing *m* users (rows) and *n* movies (columns). The shape of the R matrix is (610, 9724). We first analyzed and visualized some properties of the MovieLens dataset, before moving onto the collaborative filtering implementation.

Question 1: Sparsity

We computed the sparsity of the movie rating dataset, where sparsity is defined as the number of available ratings divided by the number of possible ratings:

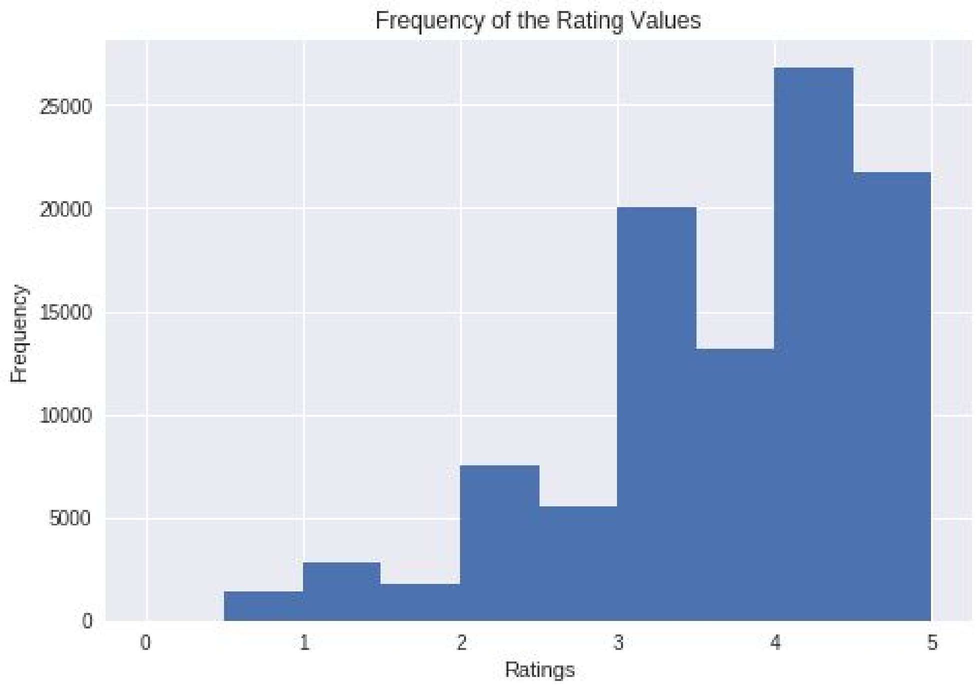
$$sparsity = \frac{\text{Total number of available ratings}}{\text{Total number of possible ratings}} = 0.01699968$$

Question 2: Histogram of Rating Values

Figure 1 shows the frequency of the rating values. Since ratings are made on a 5-star scale, with half-star increments (0.5 stars - 5.0 stars), we binned the rating values into intervals of width 0.5 and used the binned rating values as the x-axis. We used the number of entries in the ratings matrix R with rating values in the binned intervals as the y-axis.

From the histogram below, we can see that most ratings lie between 3 and 5. It shows that users in our dataset tend to give higher ratings. This information might be helpful when converting the observed ratings (continuous scale) to a binary scale. For example, in question 15, we were asked to threshold the observed ratings. If the observed rating is greater than the threshold value, then we set it to 1 (like). If the observed rating is less than the threshold value, then we set it to 0 (dislike). Because our ratings are mostly located between 3 and 5, adjusting the threshold to higher values may help on transforming the observed continuous data into like and dislike (binary scale).

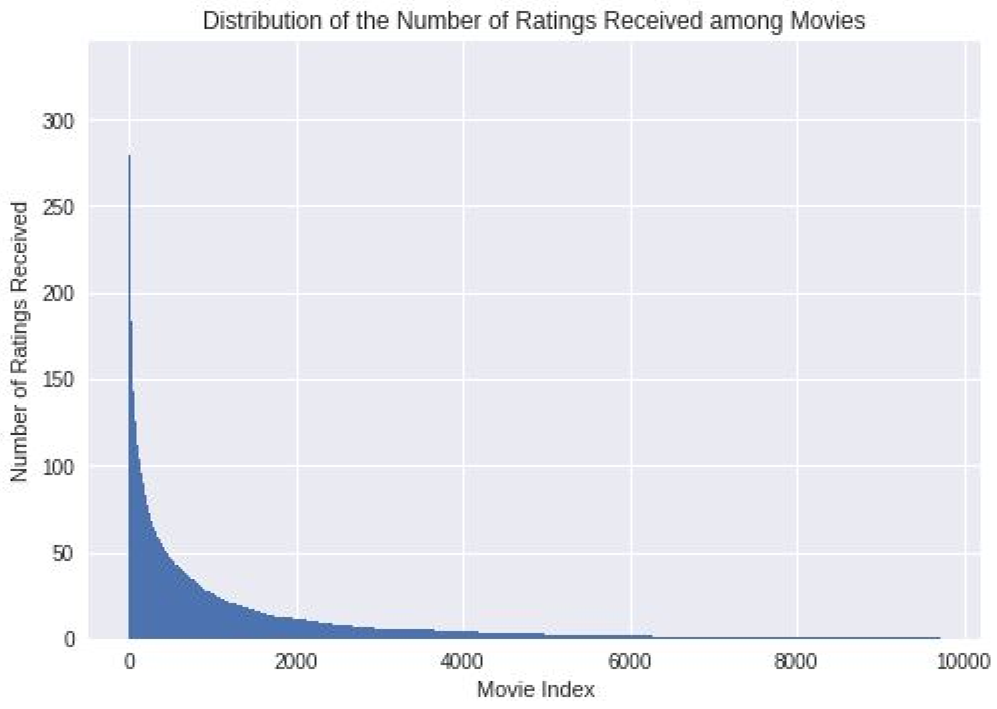
Figure 1



Question 3: Distribution of Number of Ratings Received Among Movies

Figure 2 plots the distribution of the number of ratings received among movies. The x-axis is the movie index ordered by decreasing frequency and the y-axis is the number of ratings the movie has received. For example, the movie that has the largest number of ratings has index 1. The distribution looks like a monotonically decreasing curve. We observe that the highest number of ratings for a movie is between 250-300 ratings. But the majority of the ~10,000 movies in the dataset have less than 100 ratings. Movie index 5000 (around the halfway mark) has only 2 ratings, so we see that around half of the 9742 movies only have 2 or less ratings.

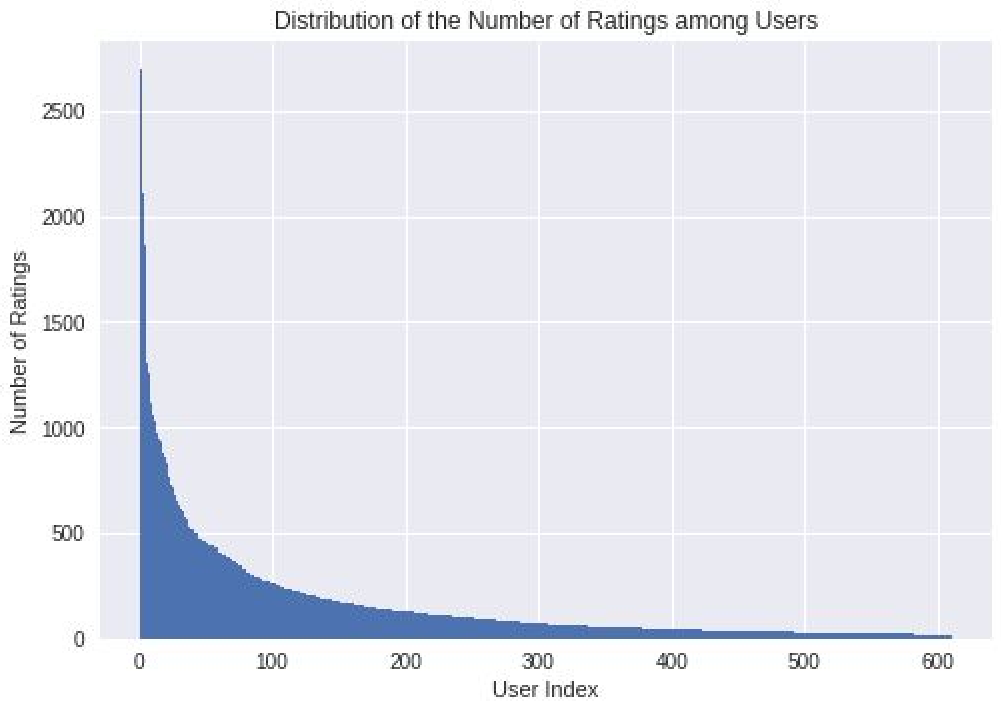
Figure 2



Question 4: Distribution of Ratings Among Users

In addition to looking at the distribution of ratings among movies, we also plotted the distribution of ratings among users (Figure 3). The x-axis displays the user index ordered by decreasing frequency, and the y-axis is the number of movies the user have rated. The plot follows a similar distribution as Figure 2, also a monotonically decreasing curve. User ID 414 gave the most ratings among all the users, with a frequency of 2698 ratings given. However, among the 610 users, most of them have given less than 100 ratings.

Figure 3



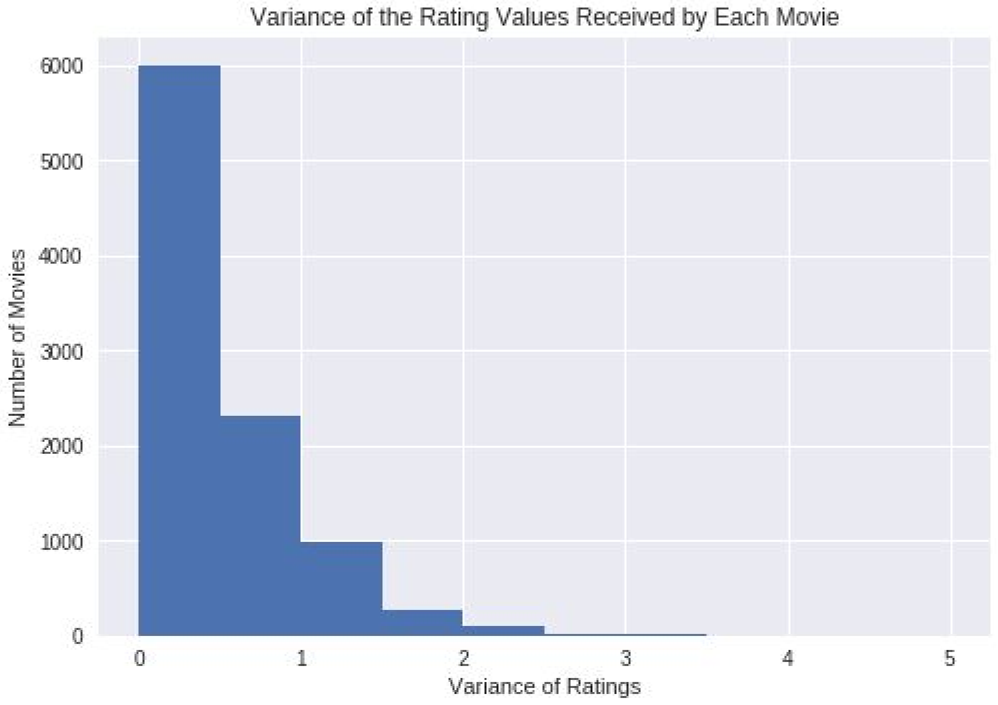
<u>Question 5: Implications of Question 3 for the Recommendation Process</u>

A salient feature of Figure 2 from Question 3 is the shape of the distribution, which looks similar to an exponentially decreasing function. Since only a few movies received a substantial amount of ratings, this implies that the ratings matrix *R* will be sparse. This corresponds with our sparsity measure in Question 1, which indicated a low number of available ratings compared to the number of possible ratings. In addition, we also observed that many of the movies received very few ratings, with around half of the movies having received only 2 or less ratings. This makes prediction of the movie ratings difficult since we have very little prior data to build the predictions on.

<u>Question 6: Variance of Rating Values by Movie</u>

We computed the variance of the rating values received by each movie and plotted a histogram (Figure 4). The x-axis is the binned variance values, binned into intervals of width 0.5. The y-axis is the number of movies with the variance values in the binned intervals. The histogram is right-skewed, with the majority (~6000) of the movies having a variance of between 0 and 0.5. This indicates that most movies receive similar ratings, without a lot of variability in the ratings they receive.

Figure 4



4 Neighborhood-Based Collaborative Filtering

Neighborhood-based collaborative filtering models are based on the fact that similar users display similar patterns of rating behavior, and similar items receive similar ratings. The basic idea behind 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.
2. *Item-based models:* similar items are rated in a similar way by the same user.

We will implement user-based collaborative filtering in this project. We will use the Pearson-correlation coefficient to compute the similarity between users:

$$Pearson(u,v) = \frac{\sum_{k\in I_u\cap I_v}(r_{uk}-\mu_u)(r_{vk}-\mu_v)}{\sqrt{\sum_{k\in I_u\cap I_v}(r_{uk}-\mu_u)^2}\sqrt{\sum_{k\in I_u\cap I_v}(r_{vk}-\mu_v)^2}}$$

I_u : Set of item indices for which ratings have been specified by user u
I_v : Set of item indices for which ratings have been specified by user v
μ_u: Mean rating for user u computed using her specified ratings
r_{uk}: Rating of user u for item k

Question 7: Mean Rating for User *U*

The mean rating for user *u* (denoted as μ_u) computed using his/her specified ratings can be calculated using the formula below:

μ_u = Mean rating for user u computed using her specified ratings
I_u = Set of item indices for which ratings have been specified by user u
r_{uk} = Rating of user u for item k

$$\mu_u = \frac{1}{I_u} \sum_{k \in I_u} r_{uk}$$

Question 8:

$I_u \cap I_v$ is the intersection of sets of item indices for which ratings have been specified by user u and user v. That is, it is the set of movies that have been rated by both user u and user v. $I_u \cap I_v = \varnothing$ can be true. This is because R is sparse, and it is possible that user u and user v have specified ratings for completely different movies. If none of the movies are rated by both user u and user v, then $I_u \cap I_v$ is a null set.

Question 9: Prediction Function

We define the prediction function for user-based neighborhood model as the equation below, where $\hat{r}_{uj}$ is the predicted rating of user u for item j:

$$\hat{r}_{uj} = \mu_u + \frac{\sum_{v \in P_u} Pearson(u,v)(r_{vj} - \mu_v)}{\sum_{v \in P_u} |Pearson(u,v)|}$$

We mean center the raw ratings $(r_{vj} - \mu_v)$ in the prediction function because different users have different rating standards. Some users may have higher standard, and they would probably give lower ratings. By mean-centering the raw ratings $(r_{vj} - \mu_v)$ in the prediction function, we can account for the systematic higher/lower ratings by each user.

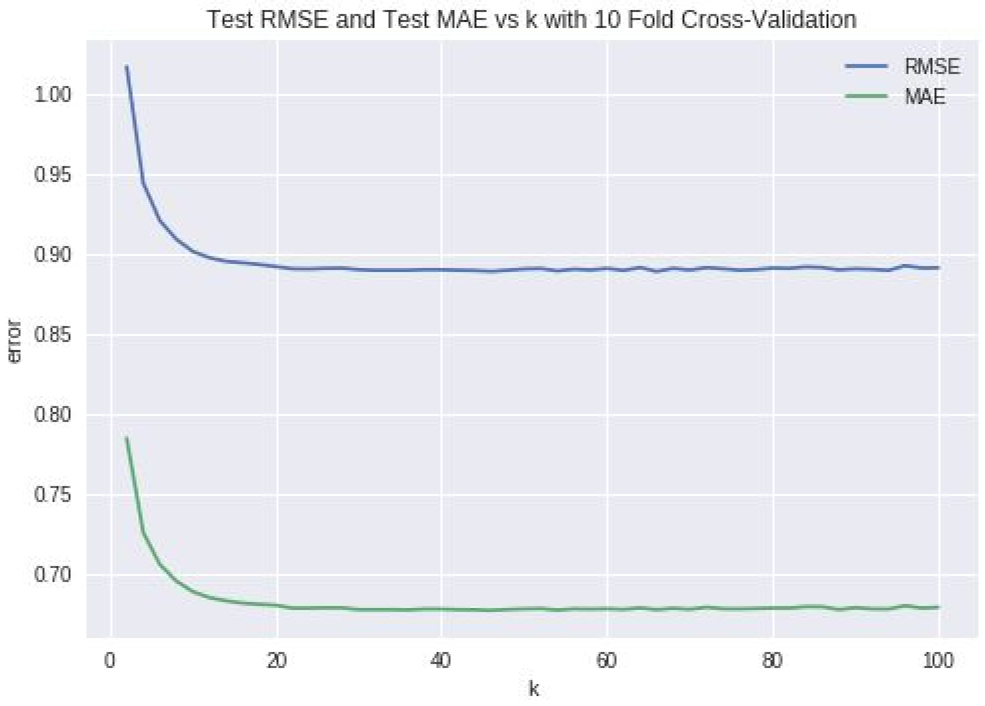
Question 10: Design k-NN Collaborative Filter and Test via Cross-Validation

In this part, we designed a k-NN collaborative filter to predict the ratings of the movies in the MovieLens dataset and evaluated its performance using 10-fold cross validation. We used the Pearson-correlation function as the similarity metric. We sweeped k (number of neighbors) from 2 to 100 in step sizes of 2, and for each k we computed the average RMSE (root mean squared error) and average MAE (mean absolute error) obtained by averaging the RMSE and MAE across all 10 folds. Figure 5 plots the average RMSE (Y-axis) against k (X-axis) and average MAE (Y-axis) against k (X-axis).

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_j)^2} \qquad \mathrm{MAE} = \frac{1}{n}\sum_{j=1}^{n}|y_j - \hat{y}_j|$$

We observed that the two measures follow the same trend; however, the RMSE tends to give higher values as compared to the MAE. This is caused in part by the way the two measures are calculated; MAE measures the average magnitude of the errors in a set of predictions, without considering their direction, and all individual differences have equal weight. The RMSE is the square root of the average squared errors; since the errors are squared before they are averaged, the RMSE gives a relatively high weight to large errors.

Figure 5



<u>**Question 11: Minimum k**</u>

Based on Figure 5, the MAE reaches its minimum error at k=46 and RMSE reaches its minimum value at k=66. It appears that around **k=20**, the average RMSE and average MAE converge to their steady-state values. From k=20 onwards, increasing k above the minimum value does not result in a significant decrease in the average RMSE or average MAE. The steady-state value of the average RMSE is **0.89**, and the steady-state value of the average MAE is **0.65**.

4-2 Collaborative Filter Performance on Trimmed Test Sets

In this part of the project, we analyzed the performance of the k-NN collaborative filter in predicting the movie ratings using trimmed test sets. The test set can be trimmed in many ways, but we will consider three methods:

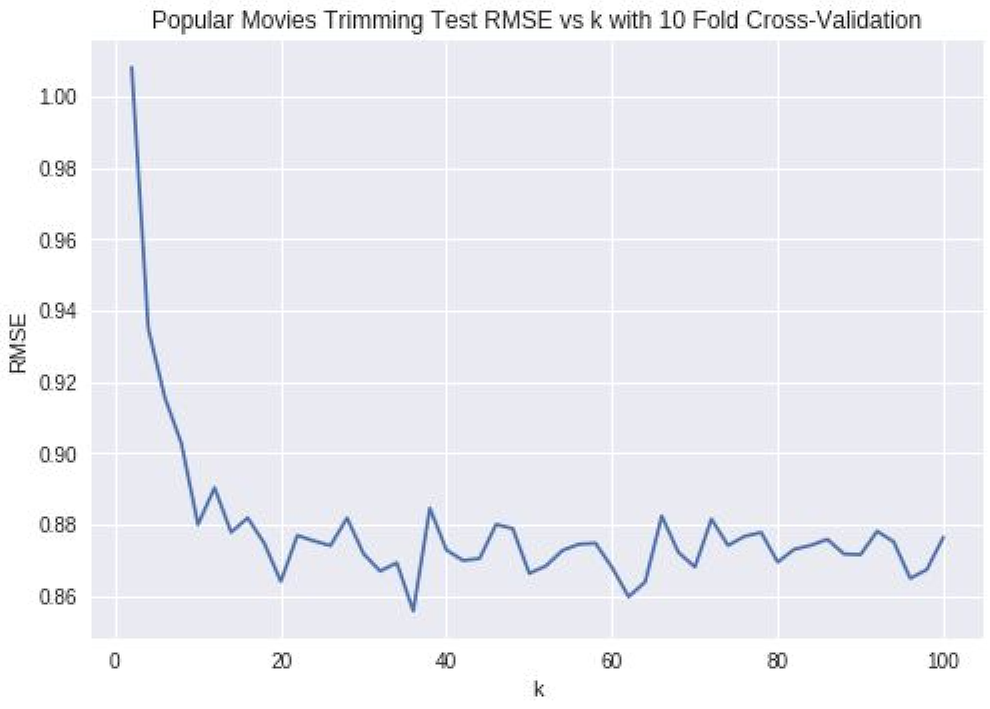
1. Popular movie trimming
2. Unpopular movie trimming
3. High variance movie trimming

Question 12: Popular Movie Trimmed Test Set

To create the popular movie trimmed test set, we trimmed the test set to contain movies that have received more than 2 ratings. Thus, we deleted movies in the test set that received less than or equal to 2 ratings in the entire dataset, and we do not predict the rating of those movies using the trained filter.

We designed a k-NN collaborative filter to predict the ratings of the movies in the popular movie trimmed test set and evaluated its performance using 10-fold cross validation, following the same method as in Question 10. It appears that the RMSE varies between 0.85 and 0.88 from k=20 onwards. The minimum RMSE is 0.8557.

Figure 6



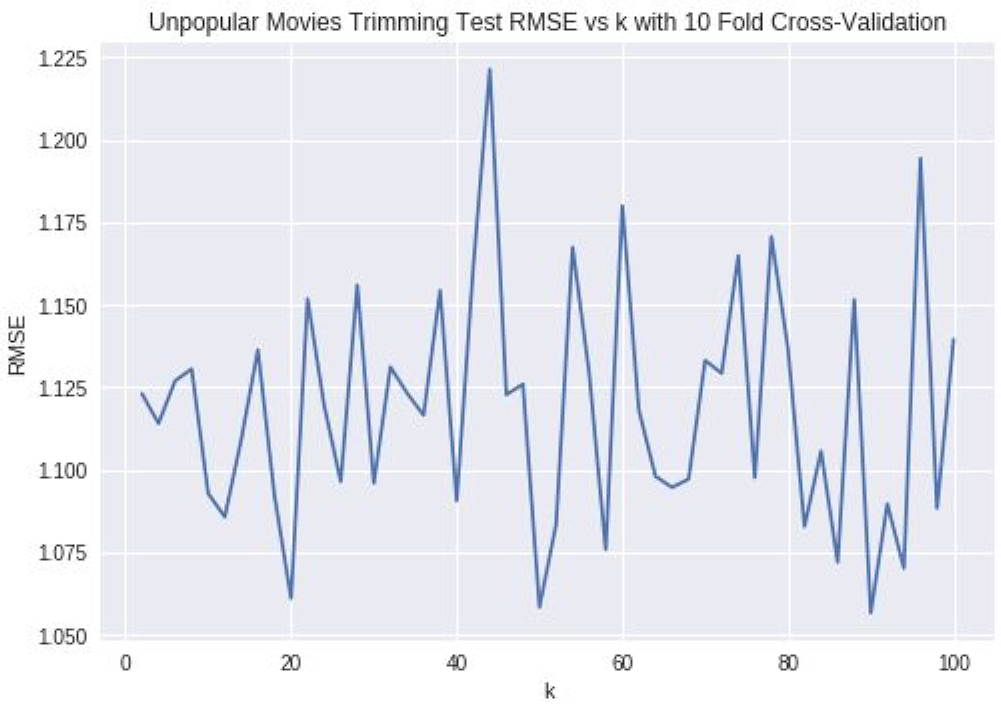
```
Minimum RMSE: 0.8557
```

Question 13: Unpopular Movie Trimmed Test Set

To create the unpopular movie trimmed test set, we trimmed the test set to contain movies that have received less than or equal to 2 ratings. Thus, if a movie in the test set has received more than 2 ratings in the entire dataset, then we deleted that movie from the test set and did not predict the rating of that movie using the trained filter.

We then designed a k-NN collaborative filter to predict the ratings of the movies in the unpopular movie trimmed test set and evaluated its performance using 10-fold cross validation following the same method as Question 10. Figure 7 is a plot of the average RMSE for k between 0 and 100. It appears that the RMSE does not reach a steady-state value, as the values vary significantly from k=0 to k=100. This may be because unpopular movies generally received very few ratings, thus contributing to increased error in the predictions. The minimum average RMSE is 1.0567.

Figure 7



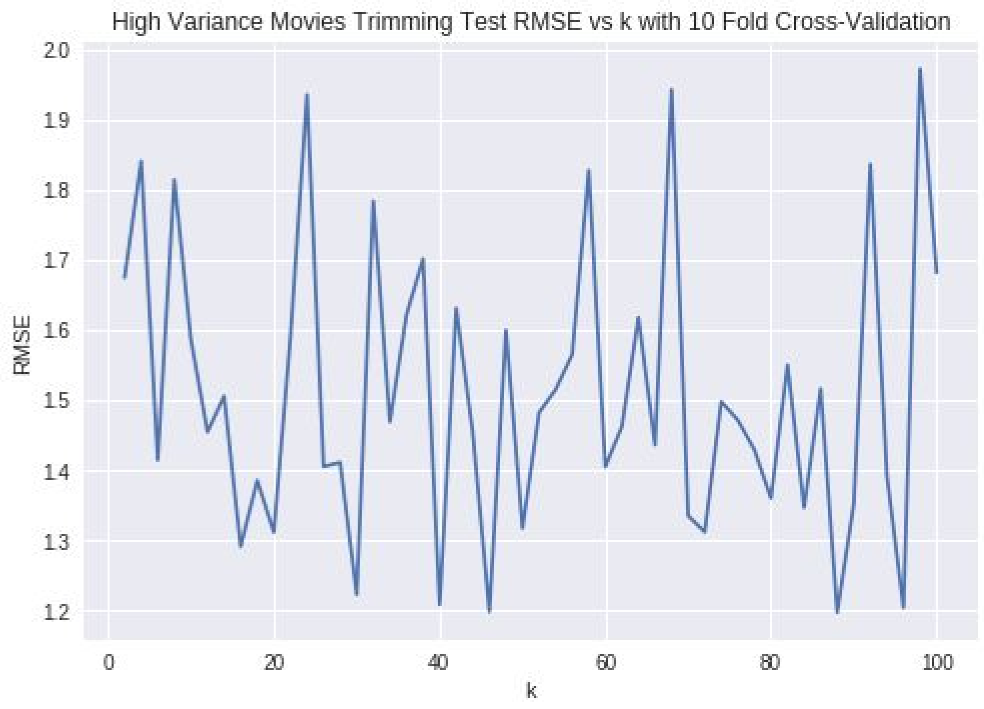
```
Minimum RMSE: 1.0567
```

Question 14: High Variance Movie Trimmed Test Set

To create the high variance movie trimmed test set, we trimmed the test set to contain movies that have variance (of the ratings values received) of at least 2, and that have received at least 5 ratings in the entire dataset. Thus, we deleted movies that had variance less than 2 or had received less than 5 ratings in the entire dataset, and we did not predict the rating of those movies using the trained filter.

We designed a k-NN collaborative filter to predict the ratings of the movies in the high variance movie trimmed test set and evaluated its performance using 10-fold cross validation, following the same method as Question 10. Figure 8 is a plot of the average RMSE (Y-axis) against k (X-axis). Similar to Figure 7, it appears that the RMSE does not reach a steady-state value, as the values vary significantly from k=0 to k=100. This makes sense since we are only using movies with high variance in the ratings, so we expect higher variability in the prediction performance. The minimum average RMSE is 1.1978.

Figure 8

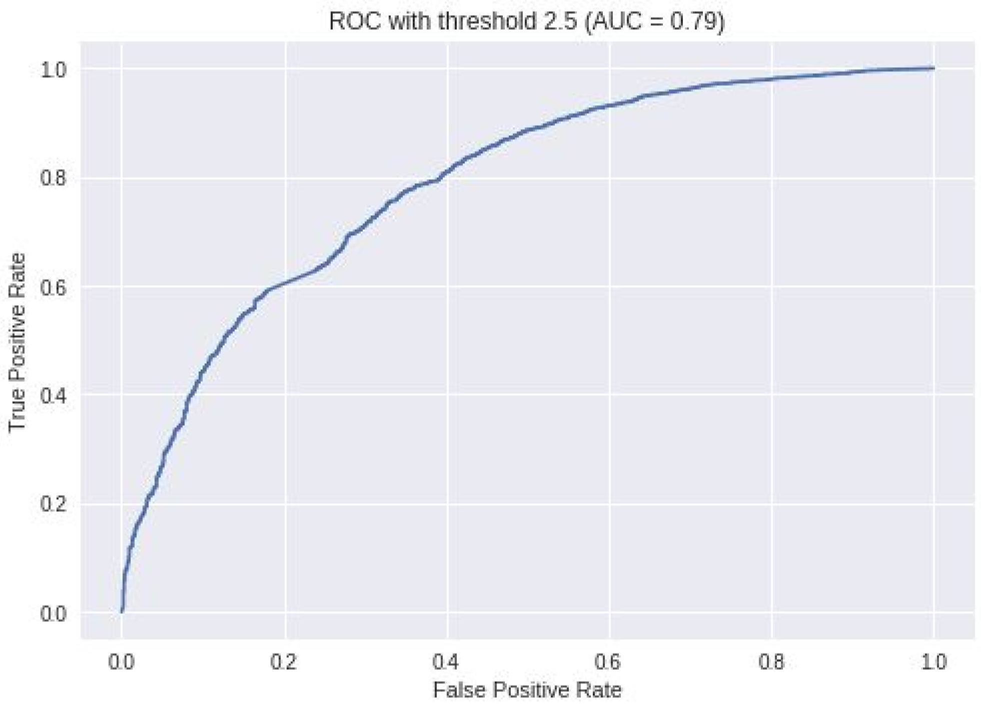


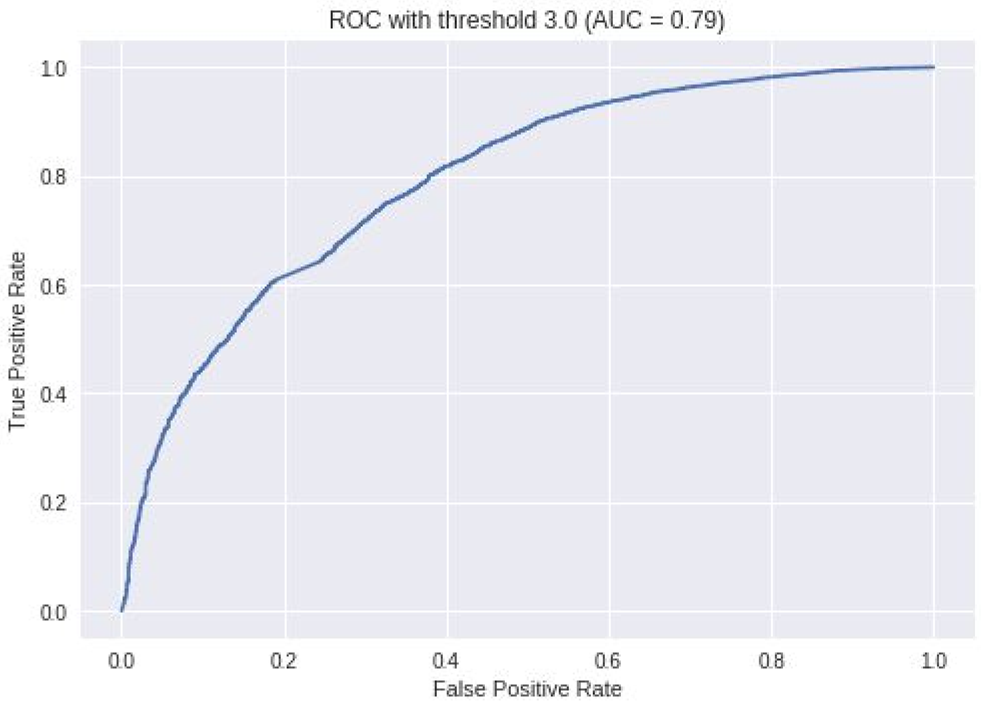
```
Minimum RMSE: 1.1978
```

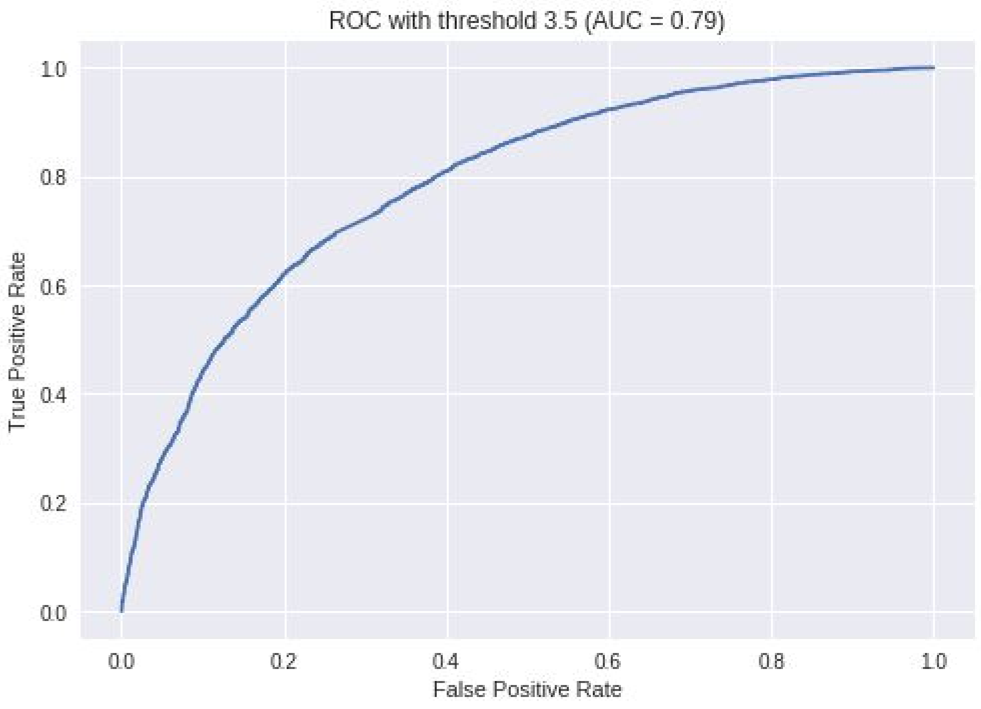
Question 15: Performance Evaluation Using ROC Curve

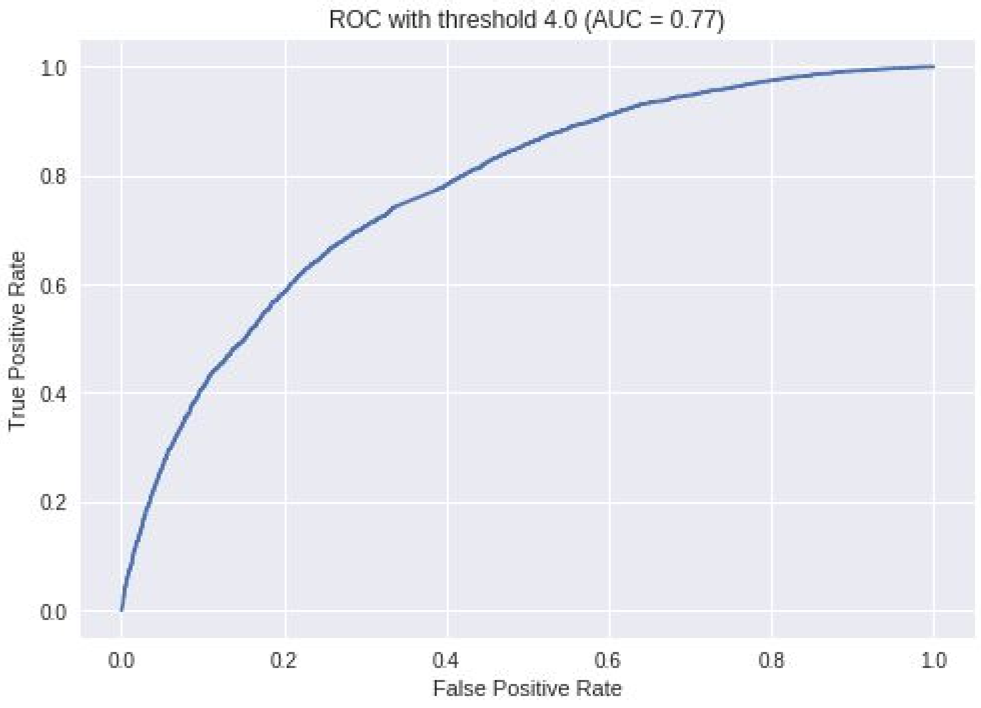
We used ROC curves as a measure of the relevance of the items recommended to the user. We plotted the ROC curves for the k-NN collaborative filter designed in question 10 for threshold values [2.5, 3, 3.5, 4]. We used k=25. The ROC curves are shown below (Figure 9), along with the area under the curve (AUC) values. For threshold values 2.5, 3, and 3.5 the AUC is 0.79. The AUC for threshold value 4 is 0.77, slightly lower.

Figure 9


ROC with threshold 2.5 (AUC = 0.79)
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0


ROC with threshold 3.0 (AUC = 0.79)
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0


ROC with threshold 3.5 (AUC = 0.79)
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0


ROC with threshold 4.0 (AUC = 0.77)
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0

5 Model-Based 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.

A function f is said to be convex at an interval if, for all pairs of points on the $f(x)$ graph, the line segment that connects these two points passes above the $f(x)$curve. This optimization problem given by equation 5 is not convex because the Hessian for the equation

$$minimize_{U,V} \sum_{i=1}^{m} W_{ij}(r_{ij} - (UV^T)_{ij})^2$$

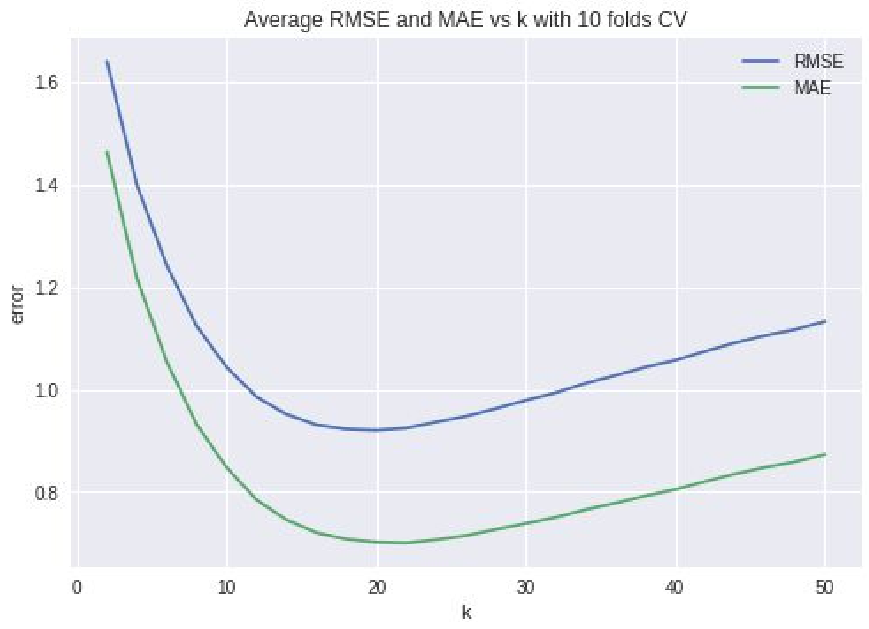
is not positively-defined.

Given U is fixed, we can treat U as a parameter of the variable V, and UV^T here can be seen as an estimate of the ground truth r_{ij}. Therefore, we can formulate it as a least square problem.

Question 17: NNMF Based Collaborative Filter

We designed a NNMF-based collaborative filter to predict the ratings of the movies in the MovieLens dataset and evaluated its performance using 10-fold cross-validation. We sweeped k (number of latent factors) from 2 to 50 in step sizes of 2, and for each k we computed 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). We used the default value for the regularization parameter (0.02), and the default learning rate of 0.005.

Figure 10



Question 18: Optimal Number of Latent Factors

Using the plot from question 17, we found the optimal number of latent factors. The optimal number of latent factors is the value of k that gives the minimum average RMSE or the minimum average MAE. The minimum average RMSE and MAE are reported below:

```
the minimum average RMSE is 0.9203250882912071
the minimum average MAE is 0.7004655573935168
```

The minimum average RMSE is achieved at k=20, and the corresponding MAE at k=20 is 0.702190. The minimum average MAE is achieved at k=22, with a corresponding RMSE value of 0.924433.

There are 18 movie genres listed below:

- Action
- Adventure
- Animation
- Children's
- Comedy
- Crime
- Documentary
- Drama
- Fantasy
- Film-Noir
- Horror
- Musical
- Mystery
- Romance
- Sci-Fi
- Thriller
- War
- Western

The optimal number of latent factors using minimum average RMSE is 20, and it is 22 using minimum average MAE. Thus, the number of optimal latent factors is not the same as the number of movie genres.

5.2.3 NNMF filter performance on trimmed test set

Question 19:

A NNMF collaborative filter was designed to predict the ratings of the movies in the popular movie trimmed test set and its performance was evaluated using 10-fold cross validation. We swept k (number of latent factors) from 2 to 50 in step sizes of 2, and for each k the average RMSE obtained by averaging the RMSE across all 10 folds was computed. The plot of average RMSE (Y-axis) against k (X-axis) was shown in Figure 5.1. The minimum average RMSE = 0.8997.

As the number of latent factors (k) increases, the average RMSE decreases and reaches its minimum when k is between 10 and 20, and starts to increase. The curve is smooth probably because there exists low variance in the popular movie trimmed test set.

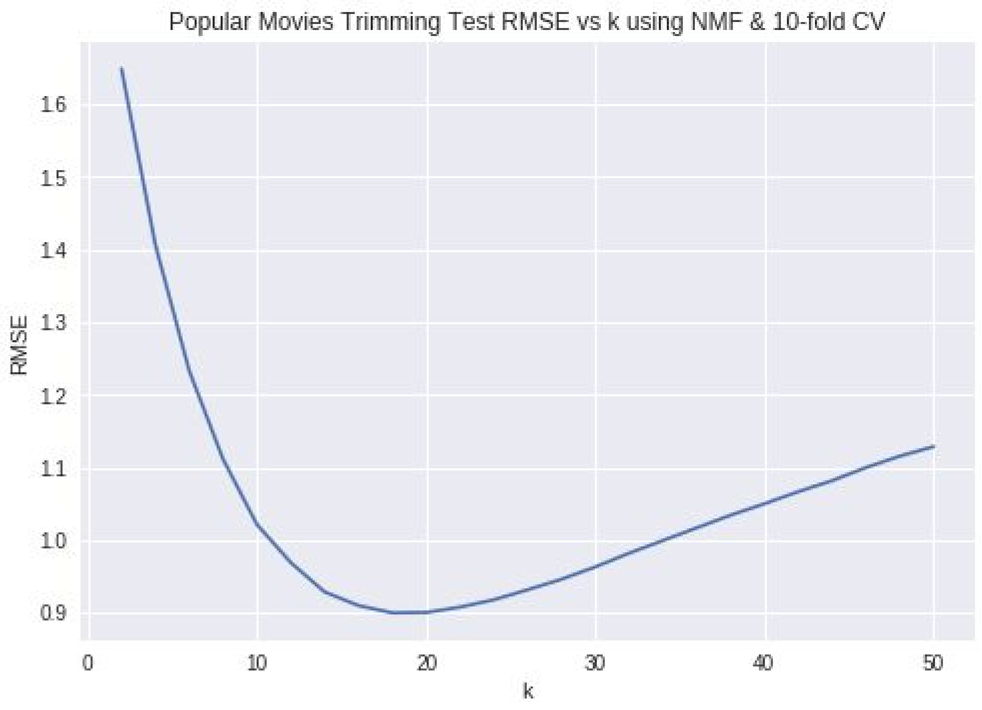


Figure 5.1

Question 20:
A NNMF collaborative filter was designed to predict the ratings of the movies in the unpopular movie trimmed test set and evaluate its performance was evaluated using 10-fold cross validation. We swept k (number of latent factors) from 2 to 50 in step sizes of 2, and for each k the average RMSE obtained by averaging the RMSE across all 10 folds was computed. The plot of average RMSE (Y-axis) against k (X-axis) was shown in Figure 5.2 The minimum average RMSE = 1.1749.

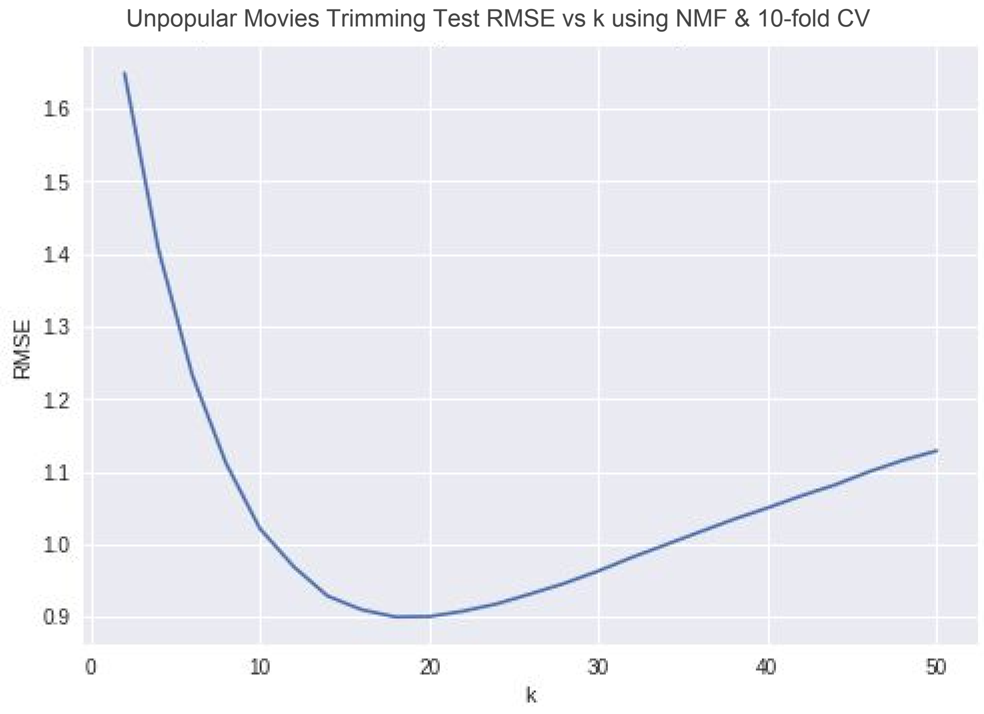


Figure 5.2

Question 21:
A NNMF collaborative filter was designed to predict the ratings of the movies in the high variance movie trimmed test set and evaluate its performance was evaluated using 10-fold cross validation. We swept k (number of latent factors) from 2 to 50 in step sizes of 2, and for each k the average RMSE obtained by averaging the RMSE across all 10 folds was computed. The plot of average RMSE (Y-axis) against k (X-axis) was shown in Figure 5.3. The minimum average RMSE = 1.6333.

As the number of latent factors increases, the average RMSE decreases dramatically when k is from 2 to 10. Then it starts to fluctuate and reaches its minimum when k = 30. The curve is not smooth probably because there exists more variance in this high variance movie trimmed test set.

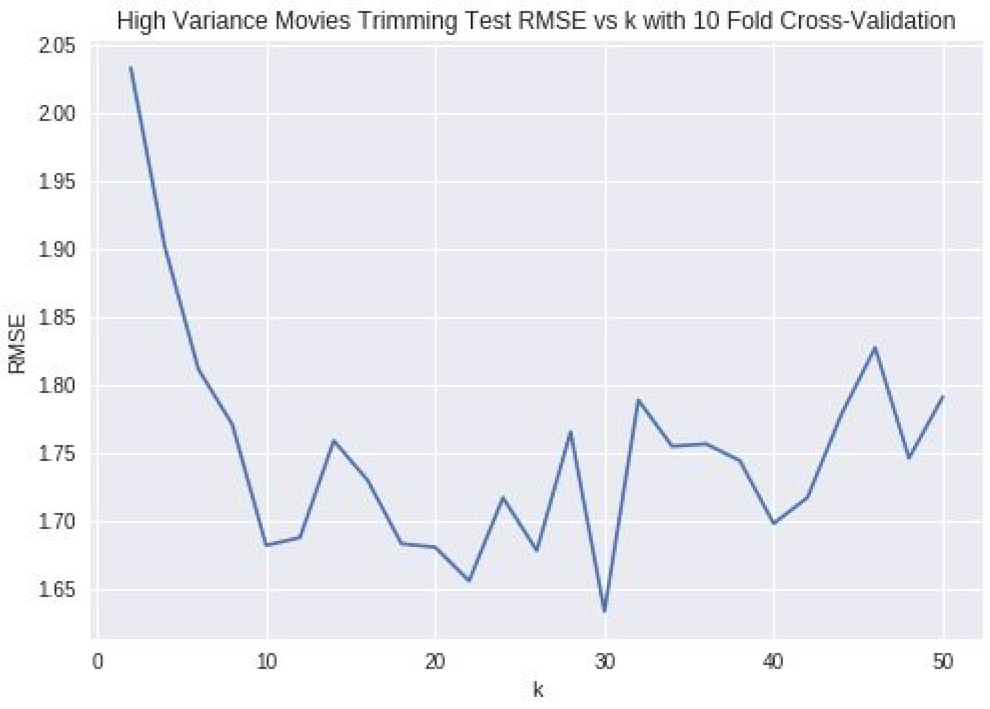


Figure 5.3

5.2.4 Performance evaluation using ROC curve

Question 22:

The plots of ROC curves for the NNMF-based collaborative filter designed in question 17 for threshold values [2.5,3,3.5,4] were shown in Figure 5.4. The optimal number of latent factors found in question 18 was used for the ROC plotting. The area under the curve (AUC) value was reported.

When the threshold value = 3.0, the AUC = 0.78, which is the highest AUC value among all 4 values. Based on the outcome of AUC, the threshold value = 3.0 have better results.

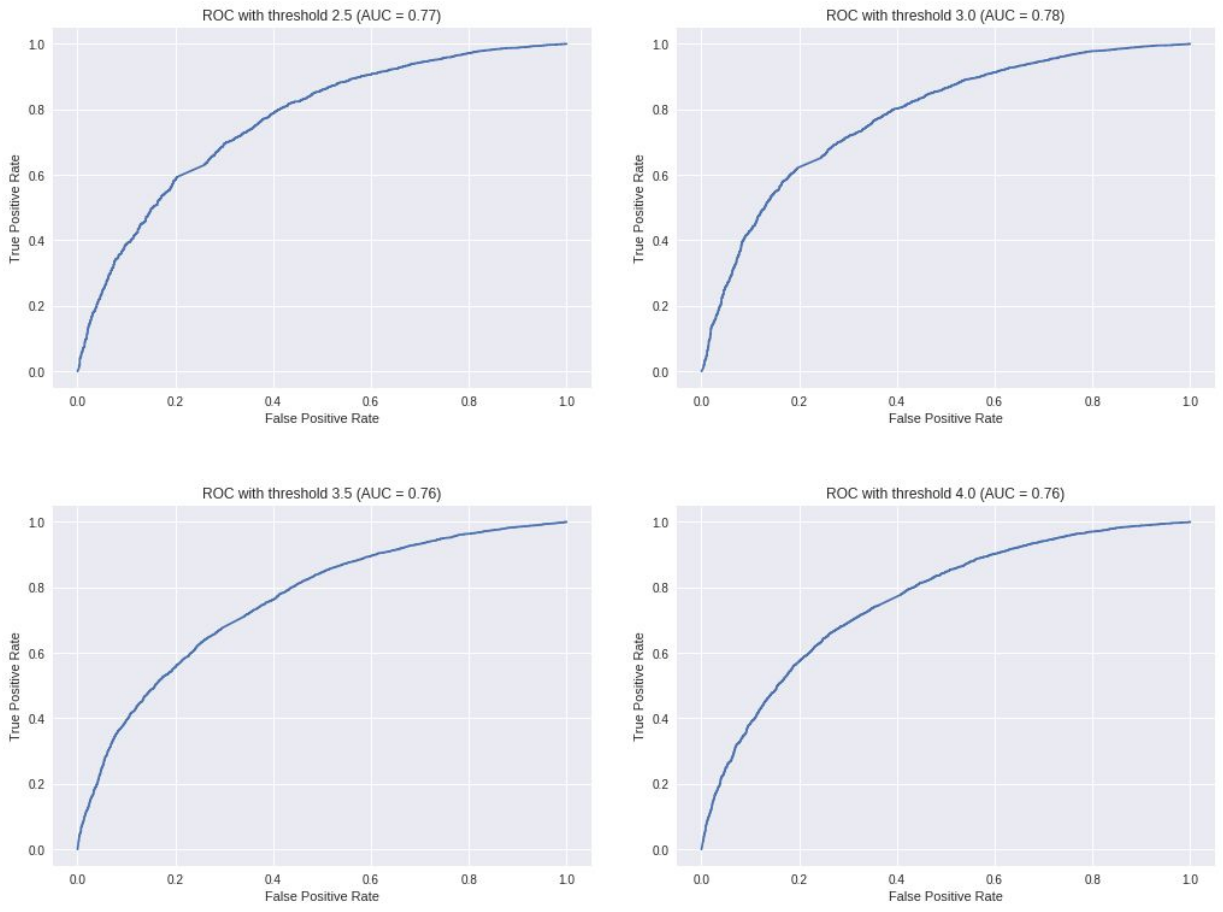


Figure 5.4

5.2.5 Interpretability of NNMF

Question 23:

The Non-negative matrix factorization on the ratings matrix R was performed 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, the movies were sorted in descending order and the genres of the top 10 movies were reported, which were shown in Table 5.1.

Drama is the most popular genre among the top 10 movies. Based on the outcome below, there is not a clear connection between the latent factors and the movie genres, which is a bit different from what we expected. However, we do see that there seems to be some correlation between the two as most rows have 2-3 genres overrepresented in the top 10. Based on this fact, it seems likely that movie genres play a small part in user ratings, but there are likely stronger factors that influence user ratings which are being captured in these latent factors.

In row 0 among the top 10 movies, 5 movies share the same genre in Drama; 3 movies share the same genre in Thriller.
In row 4 among the top 10 movies, 5 movies share the same genre Drama; 5 movies share the same genre in Comedy.
In row 9 among the top 10 movies, 5 movies share the same genre in Drama; 3 movies share the same genre in Thriller; 3 movies share the same genre in Sci-Fi; 3 movies share the same genre in Action.
In row 14 among the top 10 movies, 3 movies share the same genre in Action.
In row 19 among the top 10 movies, 4 movies share the same genre in Comedy; 3 movies share the same genre in Drama; 3 movies share the same genre in Thriller.

Table 5.1

Genres in row 0	Genres in row 4	Genres in row 9	Genres in row 14	Genres in row 19
Horror|Thriller	Comedy|Drama|Romance	Drama|Fantasy|Romance	Crime|Thriller	Action|Adventure|Animation|Children|Fantasy|Sci-Fi
Action|Drama|Thriller	Action|Comedy	Action|Adventure|Drama|Romance|Thriller	Action|Adventure|Fantasy	Comedy|Horror|Sci-Fi
Horror	Action|Adventure|Thriller	Adventure|Comedy|Fantasy|Sci-Fi	Comedy|Horror|Sci-Fi	Thriller
Documentary	Animation|Children	Action|Sci-Fi	Action|Comedy|Crime|Romance	Documentary
Drama|Romance	Comedy|Crime|Drama|Thriller	Drama	Drama	Comedy|Fantasy|Horror|Thriller
Action|Drama|Romance	Animation|Fantasy|Thriller	Comedy|Drama|Romance	Comedy|Romance	Crime|Drama|Thriller
Drama	Action|Drama	Drama|Sci-Fi	Western	Animation|Drama
Comedy	Comedy|Romance	Horror|Thriller	Action|Adventure|Fantasy	Horror
Crime|Drama|Thriller	Drama	Action|Crime|Thriller	Thriller	Comedy|Musical|Romance
Adventure|Animation|Children|Comedy	Comedy|Drama	Horror|Mystery	Horror|Western	Action|Comedy|Crime|Drama

5.3 Matrix factorization with bias (MF with bias)

In MF with bias, we modify the cost function (equation 6) by adding bias term for each user and item. With this modification, the optimization formulation of MF with bias is given by the following equation,

$$\underset{U,V,b_u,b_i}{\text{minimize}} \quad \sum_{i=1}^{m}\sum_{j=1}^{n} W_{ij}(r_{ij} - \hat{r}_{ij})^2 + \lambda \|U\|_F^2 + \lambda \|V\|_F^2 + \lambda \sum_{u=1}^{m} b_u^2 + \lambda \sum_{i=1}^{n} b_i^2$$

In the above formulation, b_u is the bias of user u and b_i is the bias of item i, and we jointly optimize over U, V, b_u, b_i to find the optimal values.

5.3.1 Prediction function

After we have solved the optimization problem for U, V, b_u, b_i , then we can use them for predicting the ratings. The predicted rating of user i for item j, denoted by $\hat{r}_{ij}$ is given by the following equation,

$$\hat{r}_{ij} = \mu + b_i + b_j + \sum_{s=1}^{k} u_{is} \cdot v_{js}$$

where μ is the mean of all ratings, b_i is the bias of user i, and b_j is the bias of item j.

5.3.2 Design and test via cross-validation

Question 24:

A matrix factorization with bias collaborative filter was designed to predict the ratings of the movies in the MovieLens dataset and its performance was evaluated using 10-fold cross-validation. We swept k (number of latent factors) from 2 to 50 in step sizes of 2, and for each k computed the average RMSE and average MAE obtained by averaging the RMSE and MAE across all 10 folds. The plot of average RMSE (Y-axis) against k (X-axis) and the average MAE (Y-axis) against k (X-axis) were shown in Figure 5.5. The default value for the regularization parameter was used.

As the number of latent factors (k) increases, the average RMSE and average MAE fluctuate but the changes are small. The average RMSE is between 0.850 and 0.875. The average MAE is between 0 and 0.675.

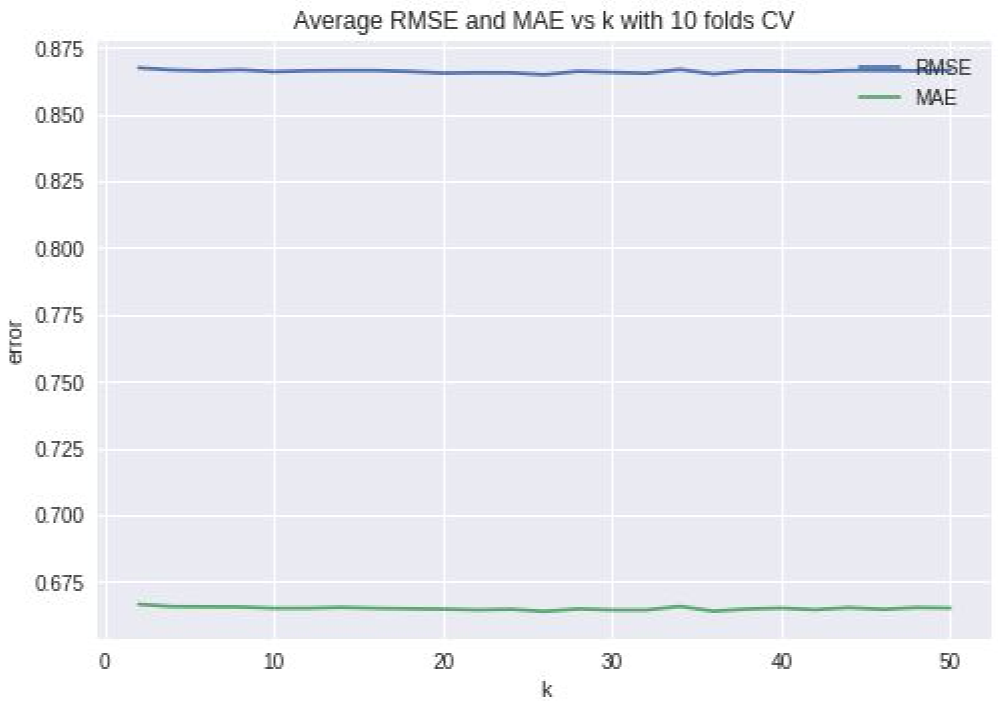


Figure 5.5

Question 25:
Using the plot from question 24, the optimal number of latent factors were found when k = 26. Optimal number of latent factors is the value of k that gives the minimum average RMSE or the minimum average MAE. The minimum average RMSE = 0.8648159309103371. The minimum average MAE = 0.6639783228987499

5.3.3 MF with bias filter performance on trimmed test set

Question 26:
A MF with bias collaborative filter was designed to predict the ratings of the movies in the popular movie trimmed test set and its performance was evaluated using 10-fold cross validation. We swept k (number of latent factors) from 2 to 50 in step sizes of 2, and for each k the average RMSE obtained by averaging the RMSE across all 10 folds was computed. The plot of average RMSE (Y-axis) against k (X-axis) was shown in Figure 5.6. The minimum RMSE = 0.8574.

As k increases, the average RMSE fluctuates dramatically.

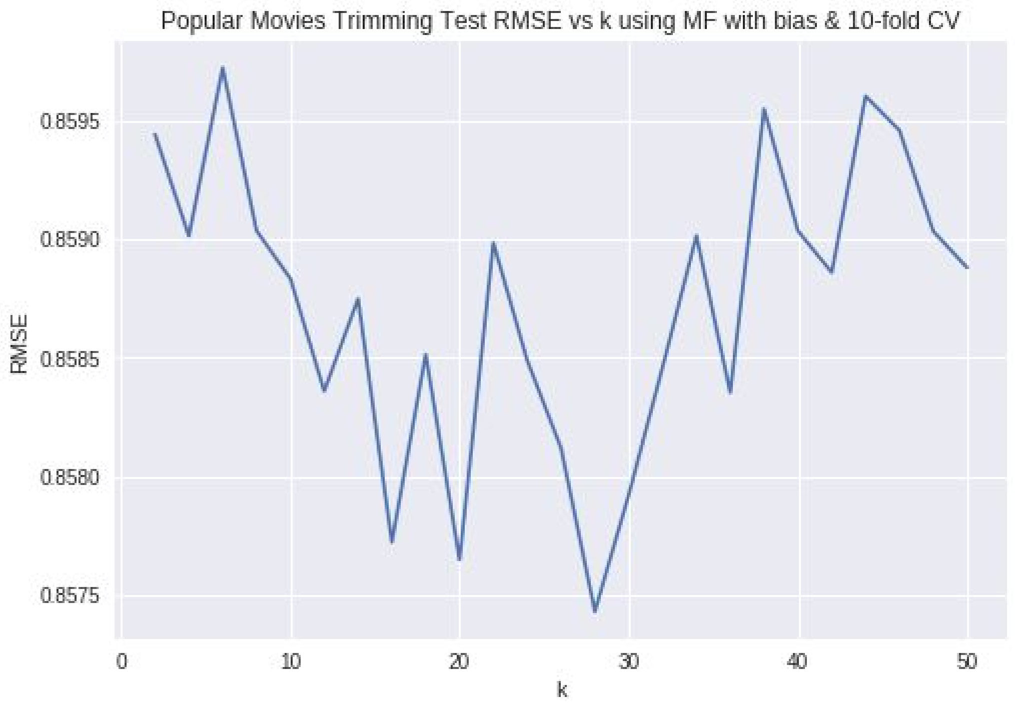


Figure 5.6

Question 27:
A MF with bias collaborative filter was designed to predict the ratings of the movies in the unpopular movie trimmed test set and its performance was evaluated using 10-fold cross validation. We swept k (number of latent factors) from 2 to 50 in step sizes of 2, and for each k the average RMSE obtained by averaging the RMSE across all 10 folds was computed. The plot of average RMSE (Y-axis) against k (X-axis) was shown in Figure 5.7. The minimum RMSE = 0.9705.

As k increases, the average RMSE fluctuates dramatically.

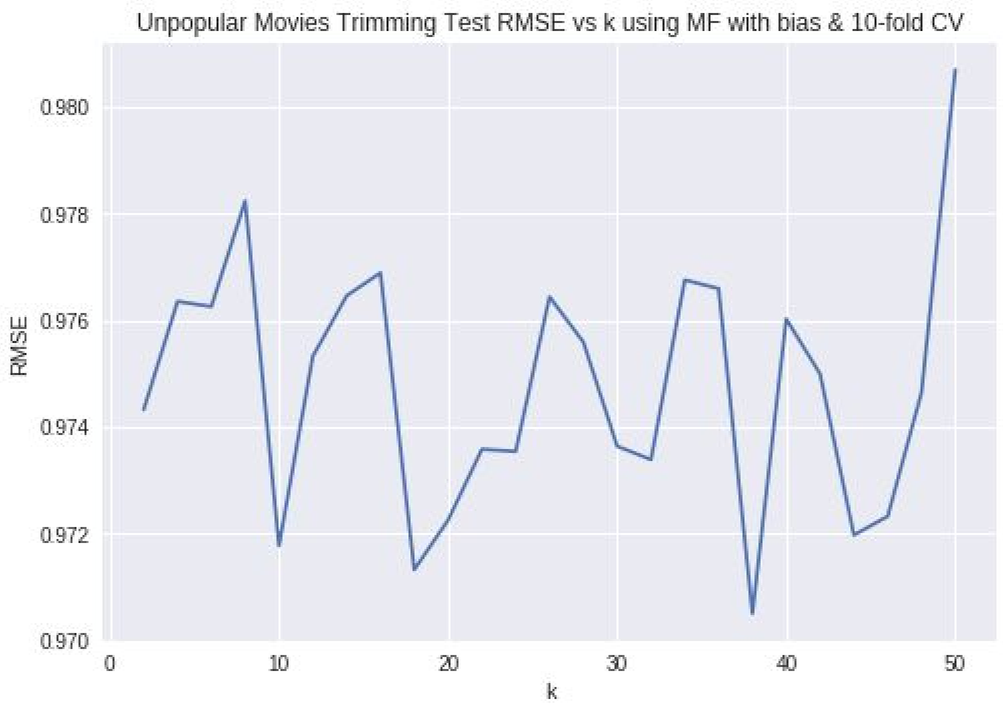


Figure 5.7

<u>Question 28:</u>
A MF with bias collaborative filter was designed to predict the ratings of the movies in the high variance movie trimmed test set and its performance was evaluated using 10-fold cross validation. We swept k (number of latent factors) from 2 to 50 in step sizes of 2, and for each k the average RMSE obtained by averaging the RMSE across all 10 folds was computed. The plot of average RMSE (Y-axis) against k (X-axis) was shown in Figure 5.8. The minimum RMSE = 1.4439.

As k increases, the average RMSE fluctuates dramatically.

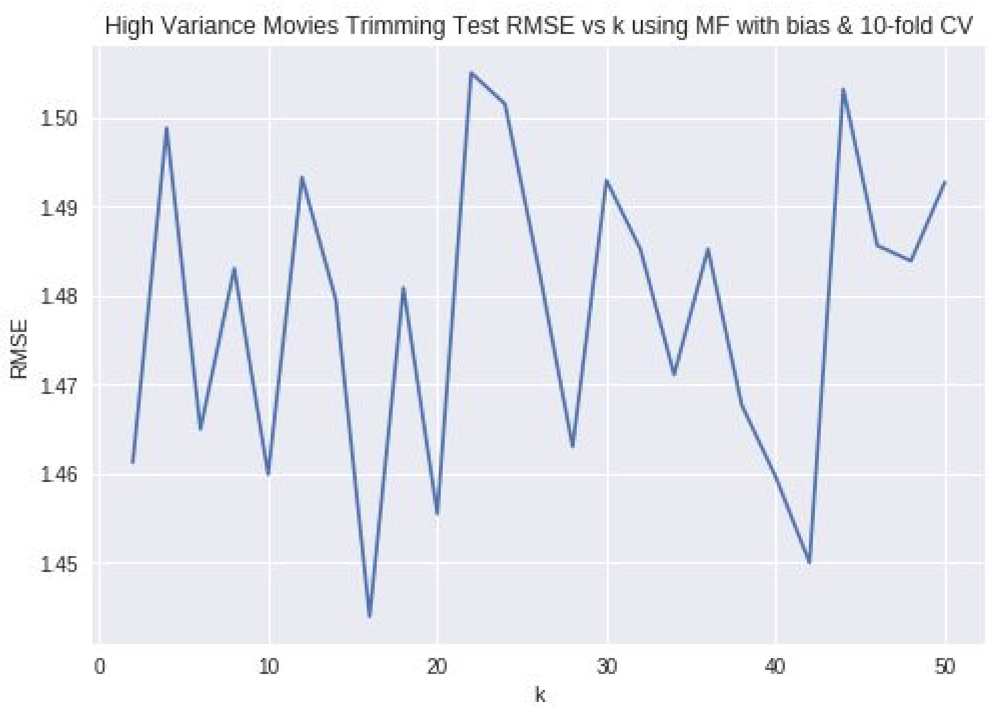


Figure 5.8

5.3.4 Performance evaluation using ROC curve

Question 29:

The plots of ROC curves for the MF with bias collaborative filter designed in question 24 were shown in Figure 5.9 for threshold values [2.5, 3, 3.5, 4]. The optimal number of latent factors found in question 25 was used, which was when k = 26.

When the threshold values = 2.5 or 3, the AUC = 0.8.
When the threshold values = 3.5 or 4, the AUC = 0.78.
Based on the outcome of AUC, the threshold values 2.5 or 3 have better results.

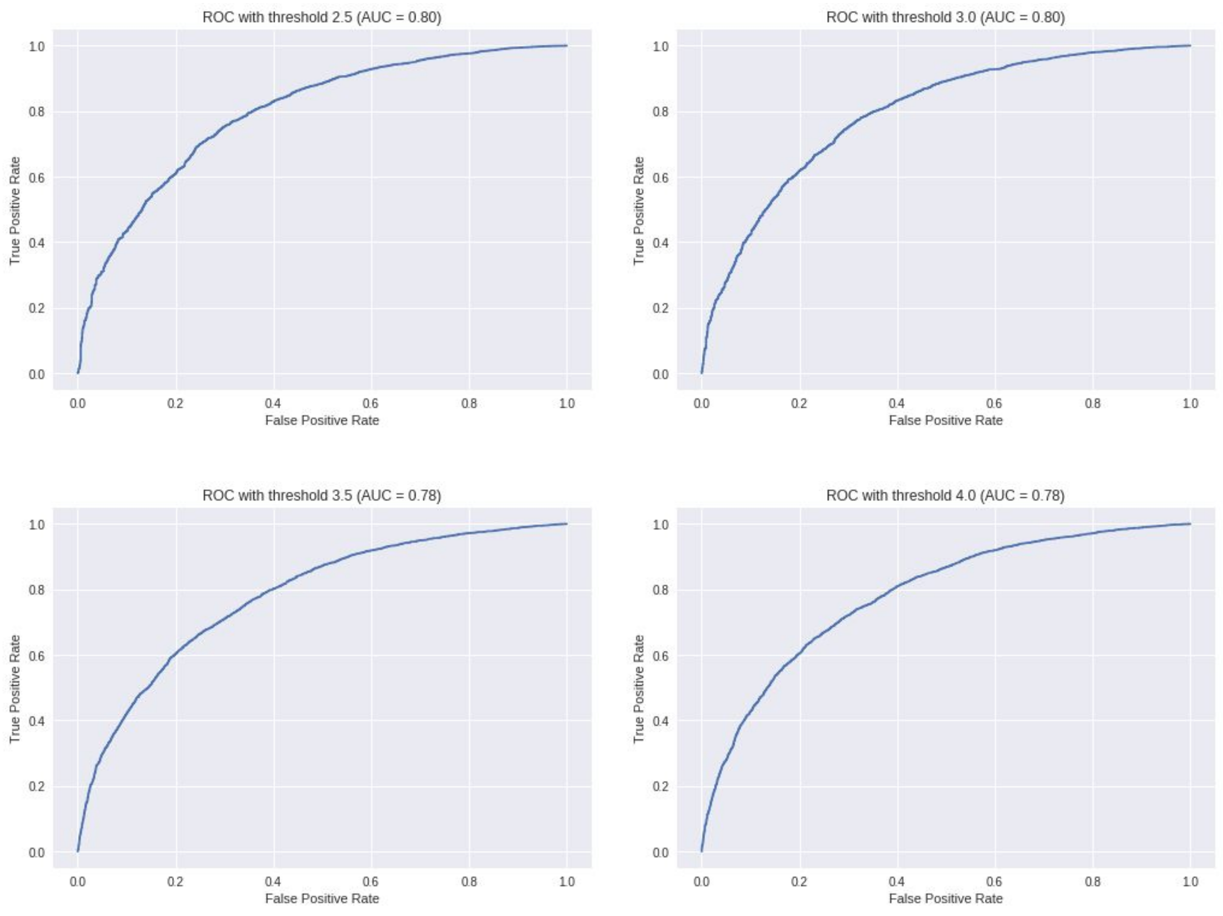


Figure 5.9

6 Naive collaborative filtering

In this part of the project, we will implement a naive collaborative filter to predict the ratings of the movies in the MovieLens dataset. This filter returns the mean rating of the user as it's predicted rating for an item.

6.1 Prediction function

The predicted rating of user i for item j, denoted by $\hat{r}_{ij}$ is given by the following equation,

$$\hat{r}_{ij} = \mu_i$$

where μ_i is the mean rating of user i.

6.2 Design and test via cross-validation

Question 30:

A naive collaborative filter was designed to predict the ratings of the movies in the MovieLens dataset and its performance was evaluated using 10-fold cross validation. The average RMSE was computed by averaging the RMSE across all 10 folds.

The average RMSE = 0.9357012529750481.

6.3 Naive collaborative filter performance on trimmed test set

Question 31 & 32 & 33:
A naive collaborative filter was designed to predict the ratings of the movies in the following 3 test sets: the popular movie trimmed test set, the unpopular movie trimmed test set and the high variance movie trimmed test set. The performance was evaluated using 10-fold cross validation. The average RMSE was computed by averaging the RMSE across all 10 folds. The average RMSEs were reported in Table 6.1.

Table 6.1

	popular movie trimmed test set	unpopular movie trimmed test set	high variance movie trimmed test set
average RMSE	0.9316774436854212	0.9423359052203419	1.0754196462687842

Compared to the outcome in Question 30, the average RMSE of the ratings in popular movie trimmed test set is lower than the ratings in the MovieLens dataset. This makes sense since there is lower variance in the ratings of popular movies.

The average RMSE of unpopular movie trimmed test set is higher than the ratings in the MovieLens dataset. This can be explained by the lower number of movies with low ratings.

The average RMSE of high variance movie trimmed test set is the highest among all three test set and is also much higher than the ratings in the MovieLens dataset in Question 30. This can be explained by the high variance in this test set.

7 Performance comparison

Question 34:
The plot of ROC curves (threshold = 3) for the k-NN, NNMF, and MF with bias based collaborative filters was shown in Figure 7.1. Use the figure to compare the performance of the filters in predicting the ratings of the movies.

The results of AUC for the k-NN, NNMF, and MF with bias based collaborative filters are very close. The MF with bias has the highest AUC = 0.79 and NNMF has the lowest AUC = 0.77.

Based on the figure, the MF with bias has the best performance and the NNMF has the poorest performance.

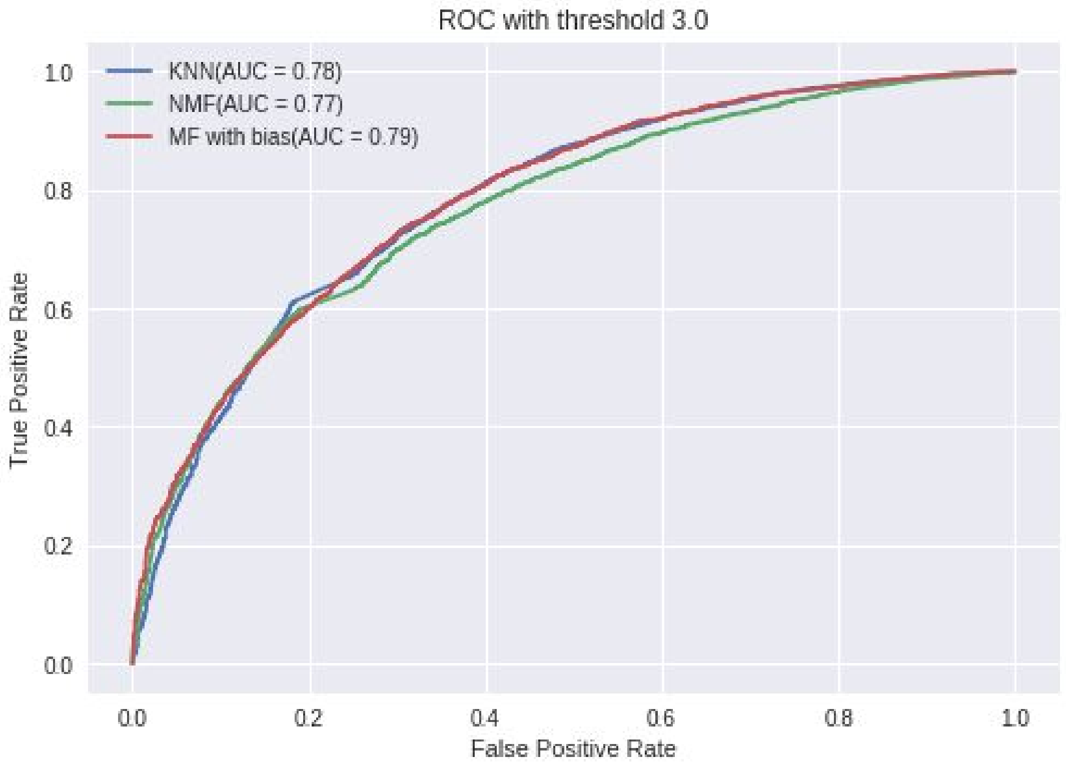


Figure 7.1

8 Ranking

Two primary ways in which a recommendation problem may be formulated are:
1. Prediction version of problem: Predict the rating value for a user-item combination
2. Ranking version of problem: Recommend the top k items for a particular user

8.2 Evaluating ranking using precision-recall curve

Precision-recall curve can be used to evaluate the relevance of the ranked list. Before stating the expressions for precision and recall in the context of ranking, let’s introduce some notation:

S(t) : The set of items of size t recommended to the user. In this recommended set, ignore (drop) the items for which we don’t have a ground truth rating.
G: The set of items liked by the user (ground-truth positives)
Then with the above notation, the expressions for precision and recall are given by equations 8 and 9 respectively

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

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

Question 35:
Precision: among all the items recommended to the user, the percentage of the correct recommendations, which are the items that were also liked by the user.
Recall: among all the items liked by the user, the percentage of the correct recommendations, which are items that were also recommended to the user.

Question 36 & 37 & 38:
The following plots were shown in Figure 8.1: the average precision (Y-axis) against t (X-axis), the average recall (Y-axis) against t (X-axis), and the average precision (Y-axis) against average recall (X-axis) for the ranking obtained using k-NN, NNMF and MF with bias collaborative filter predictions.

In general, the plots of the 3 methods of collaborative filter predictions are similar. For the plot of average precision against t, the precision decreases as t increases and the decrease is concave up. For the plot of average recall against t, as t increases, the increase in recall is similar to exponential. For the plot of average precision against recall, the precision decreases as t increases. However, the decrease is concave down for k-NN, and for NNMF and MF with bias the decrease is close to linear.

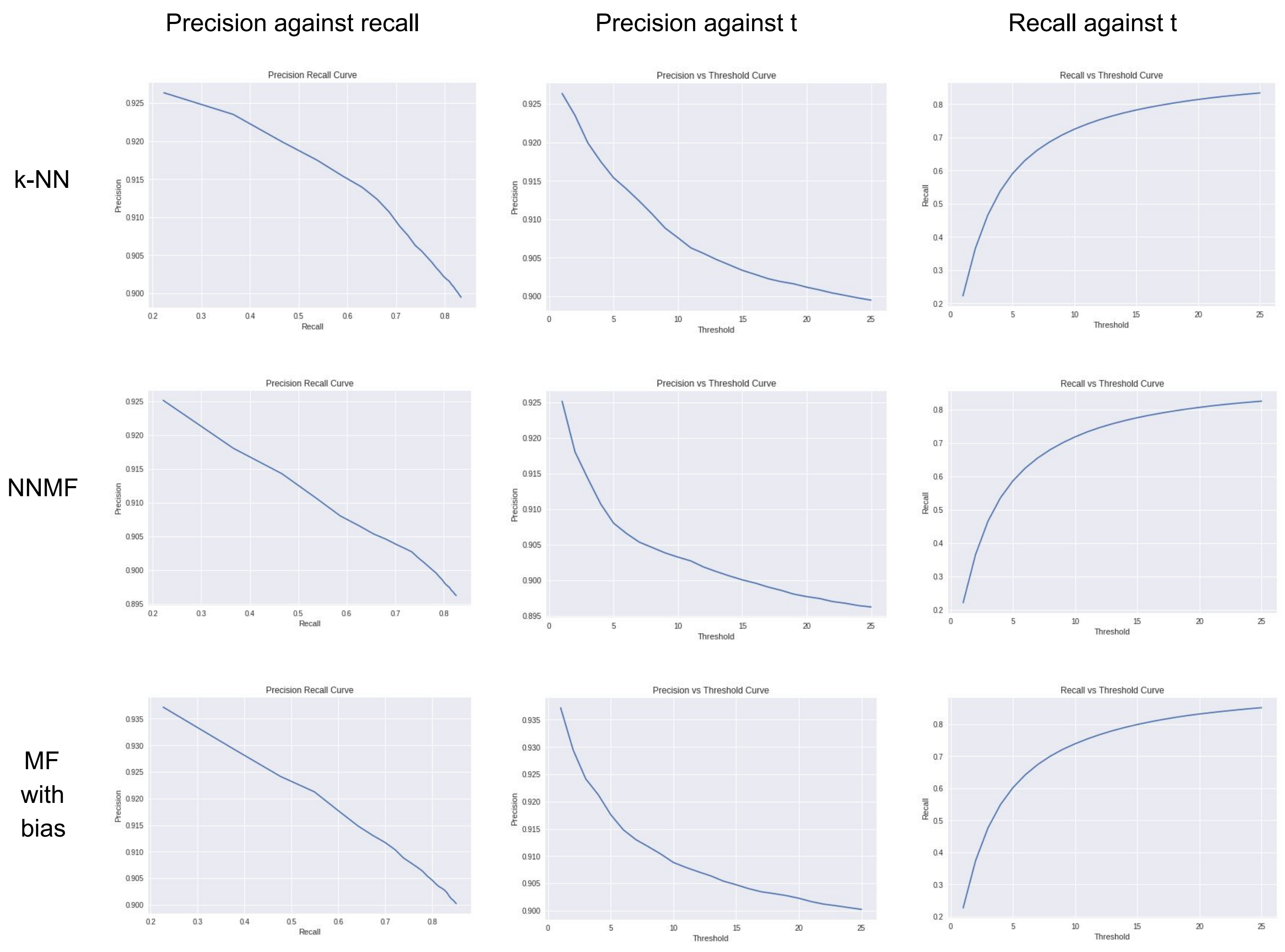


Figure 8.1

<u>Question 39:</u>

The plot of the precision-recall curve obtained in questions 36,37, and 38 was shown in Figure 8.2. The Precision-Recall curve is used to evaluate the relevance of a ranked list. In Figure 8.2 The MF with bias curve is the highest and the NMF curve is the lowest, with KNN curve in the middle. The 3 curves don't cross each other in this plot. This indicates that for any given recall, the MF with bias has the highest precision compared to NMF and KNN, and for any given precision, the MF with bias has the highest recall compared to NMF and KNN. Therefore, the MF with bias has the best performance in making recommendations compared to NMF and KNN. The NMF has the worst performance in making recommendations in this case.

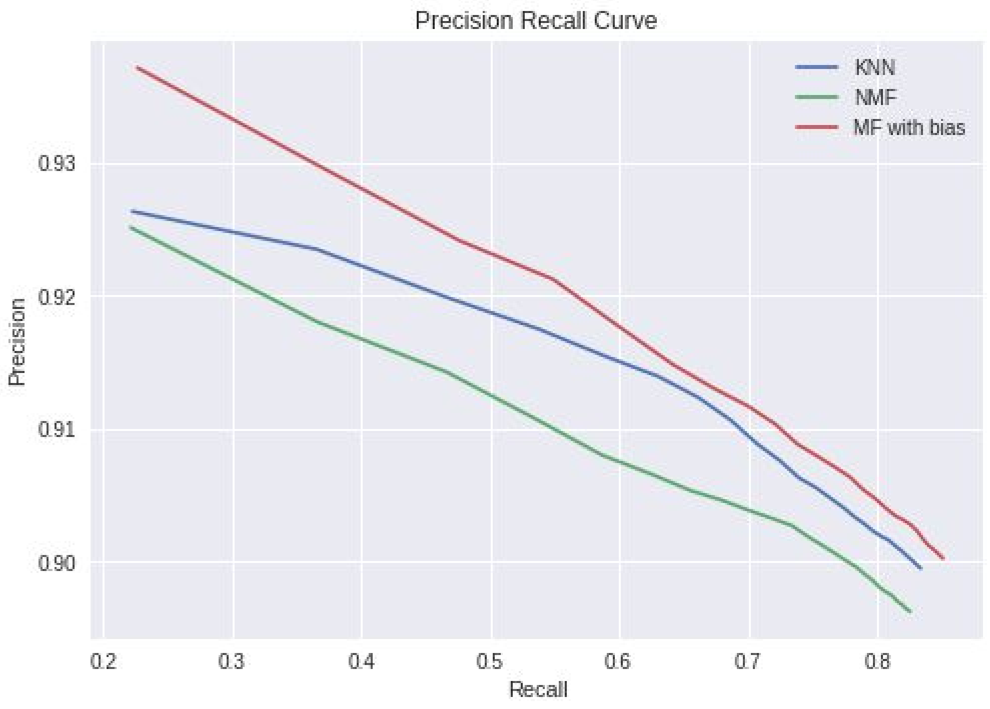

Precision Recall Curve
KNN
NMF
MF with bias
Precision
0.93
0.92
0.91
0.90
Recall
0.2
0.3
0.4
0.5
0.6
0.7
0.8

Figure 8.2