Project 3
Recommender Systems

Junrui Zhu 606530444

Liwei Tan 206530851

Tianxiang Xing 006530550

QUESTION 1: Explore the Dataset

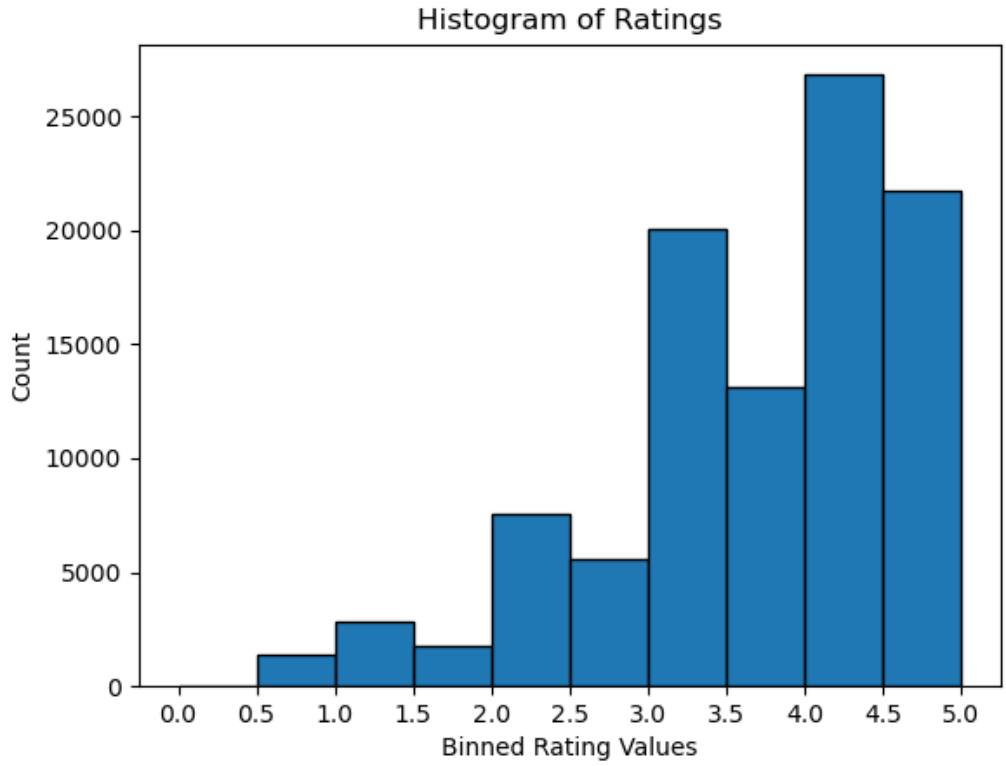
A. Compute the sparsity of the movie rating dataset:

i. Total number of available ratings is 100836.

ii. Total number of users is 610, and total number of movies that have been rated is 9724, so the total number of possible ratings is 610 * 9724 = 5931640.

iii. The sparsity is 100836 / 5931640 = 0.0170

B. Plot a histogram showing the frequency of the rating values:

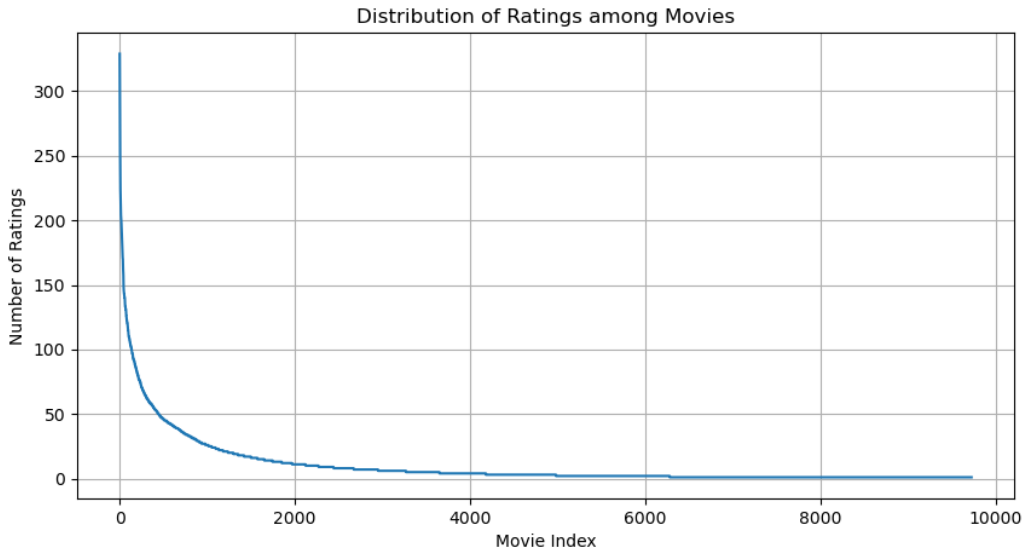


i. The majority of ratings are clustered around 4.0 - 4.5, indicating that higher ratings

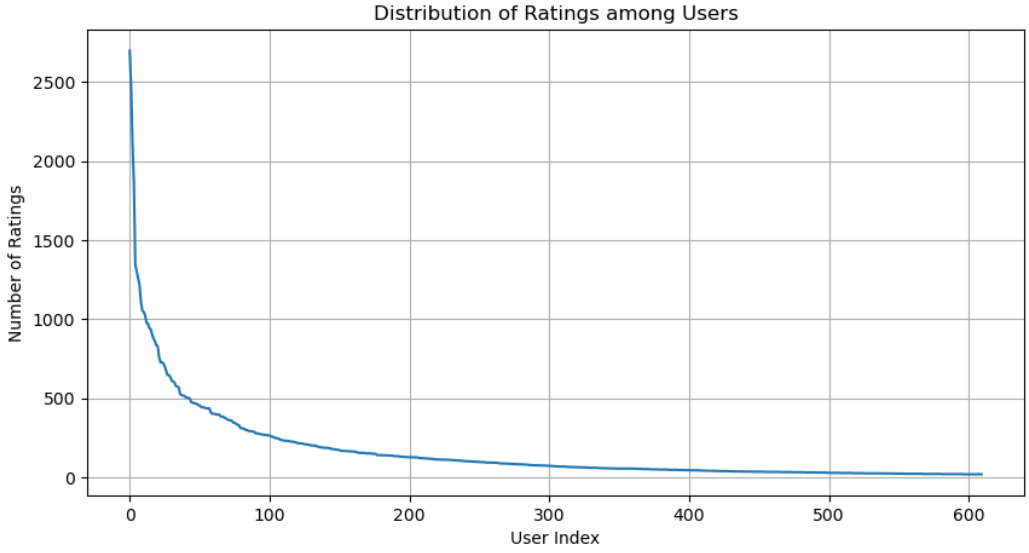
are more common.

ii. Ratings below 2.5 are extremely rare, showing that the rating dataset has a overall positive bias.

C. Plot the distribution of the number of ratings received among movies



D. Plot the distribution of ratings among users



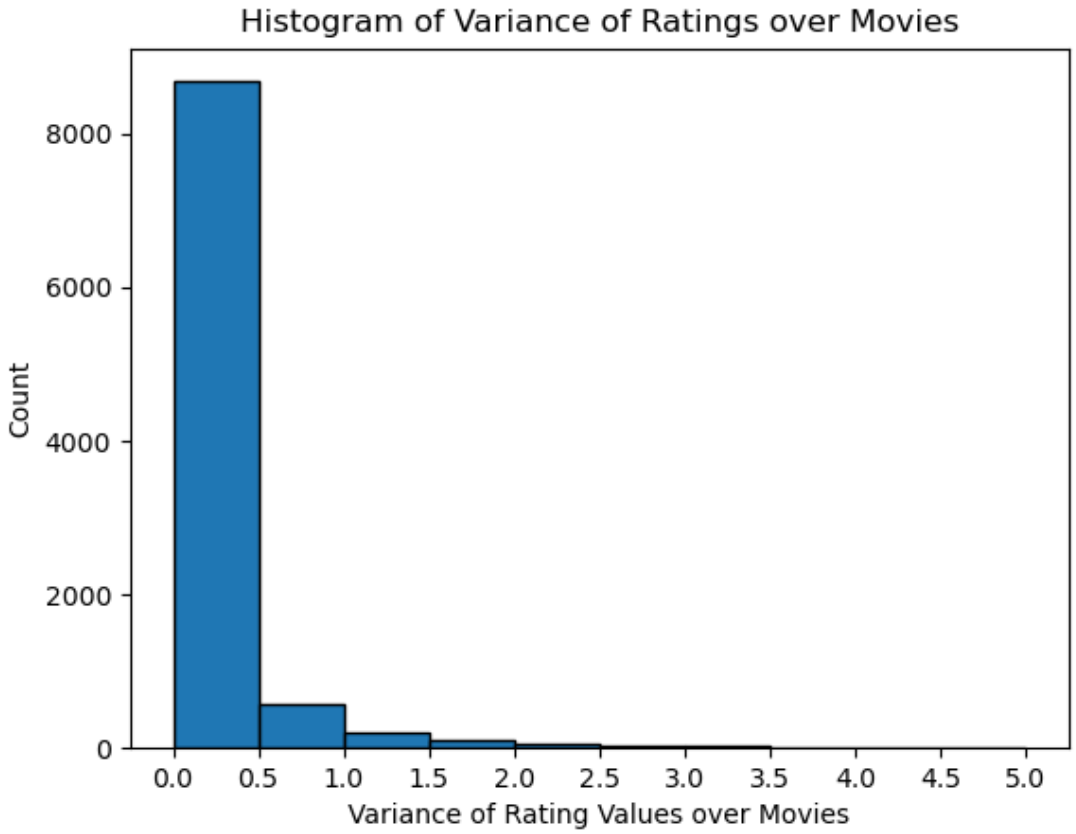
E. Discuss the salient features of the distributions and implications

i. Both distributions exhibit a long-tail pattern, where a small subset of movies and users account for the majority of ratings. A few movies receive a disproportionately high number of ratings, indicating that a small portion of movies dominate user attention. Similarly, a small group of users contributes most ratings, suggesting that

highly active users influence the dataset heavily.

ii. The imbalances shown in distributions can impact the recommendation systems by causing overfitting to popular movies and active users.

F. Compute the variance of the rating values received by each movie



The majority of variance are clustered around 0.0 to 0.5, very few of them exceed 0.5. Besides, the distribution of variance shows a monotonously decreasing trend.

QUESTION 2: Understanding the Pearson Correlation Coefficient

A. Write down the formula for μ_u in terms of I_u and r_{uk}

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

$|I_u|$ is the cardinality of I_u

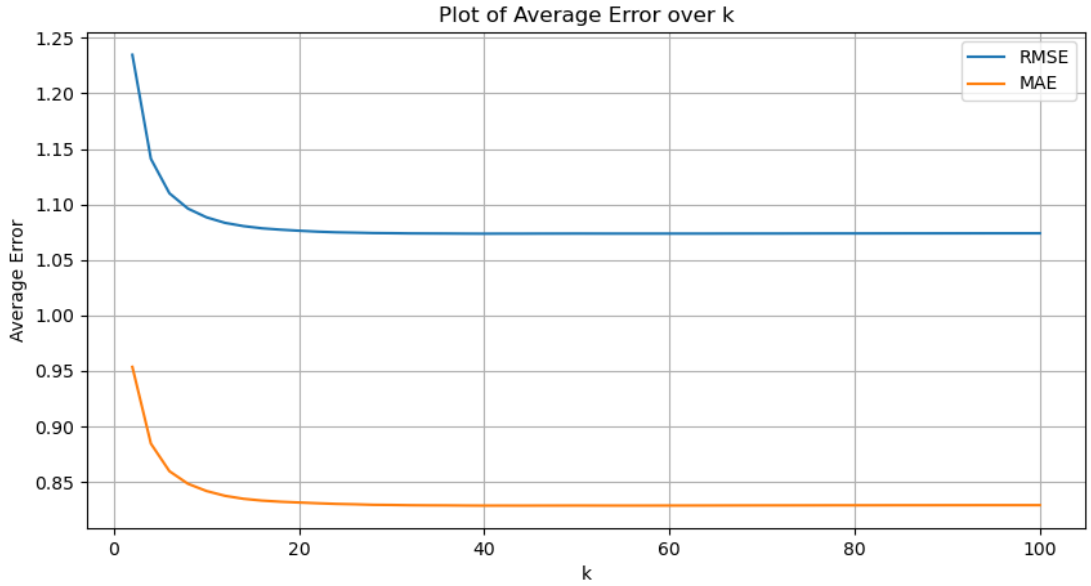
B. In plain words, explain the meaning of $I_u \cap I_v$. Can $I_u \cap I_v = \emptyset$?

The $I_u \cap I_v$ represents movies that are rated by both user u and v. $I_u \cap I_v = \emptyset$ is possible, indicating that user u and v have no movies rated in common.

QUESTION 3: Understanding the Prediction function: Can you explain the reason behind mean-centering the raw ratings $(r_{vj} - \mu_v)$ in the prediction function?

Mean-centering the raw ratings helps account for individual user biases in rating behavior. Different users usually have different baseline tendency, and by subtracting user's mean rating, we can focus on the user's relative preference for an item compared to their typical ratings.

QUESTION 4: Design a k-NN collaborative filter to predict the ratings of the movies in the original 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).



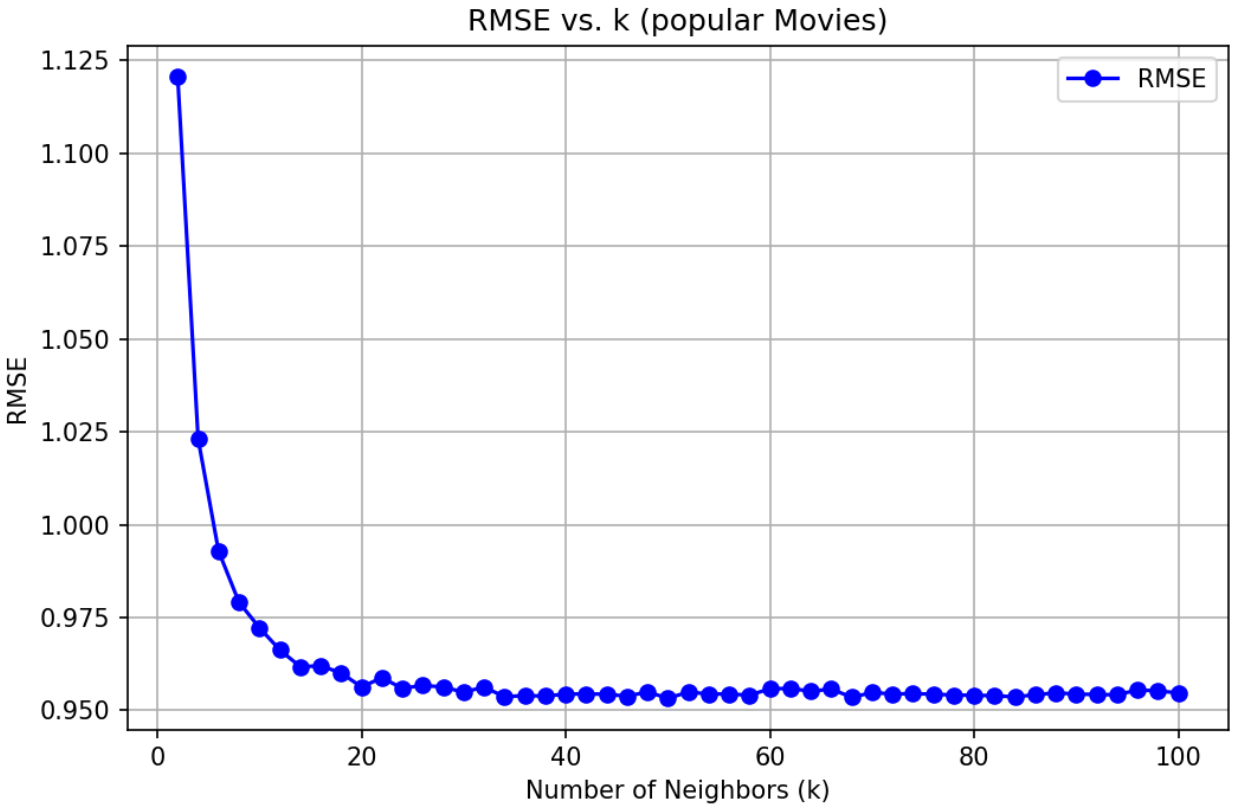
QUESTION 5: Use the plot from question 4, to find a 'minimum k'

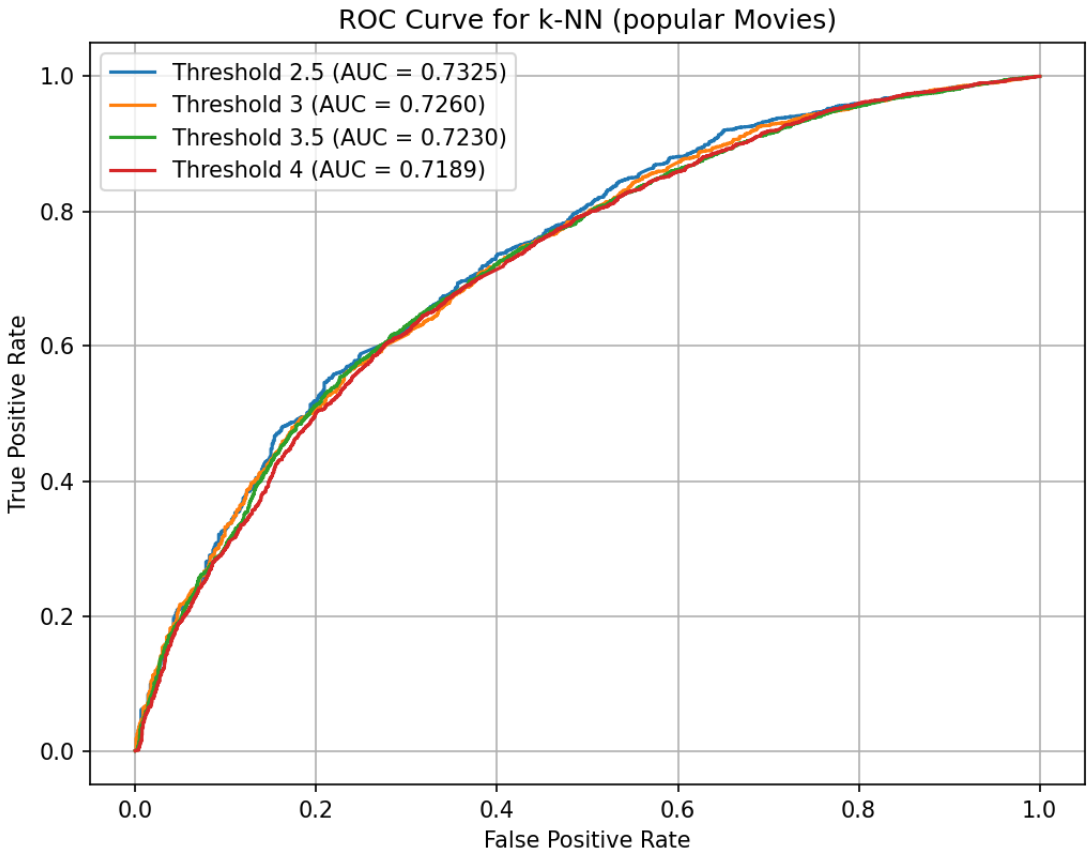
i. We set the minimum k as the smallest value for which the reduction rates in both MAE and RMSE are less than 0.0001.

ii. The minimum k obtained is 15, at which the steady state values of average RMSE is 1.0739 and that of average MAE is 0.8291.

QUESTION 6: Within EACH of the 3 trimmed subsets in the dataset, design (train and validate) k-NN collaborative filter on the ratings of the movies and evaluate each of the three models' performance using 10-fold cross validation

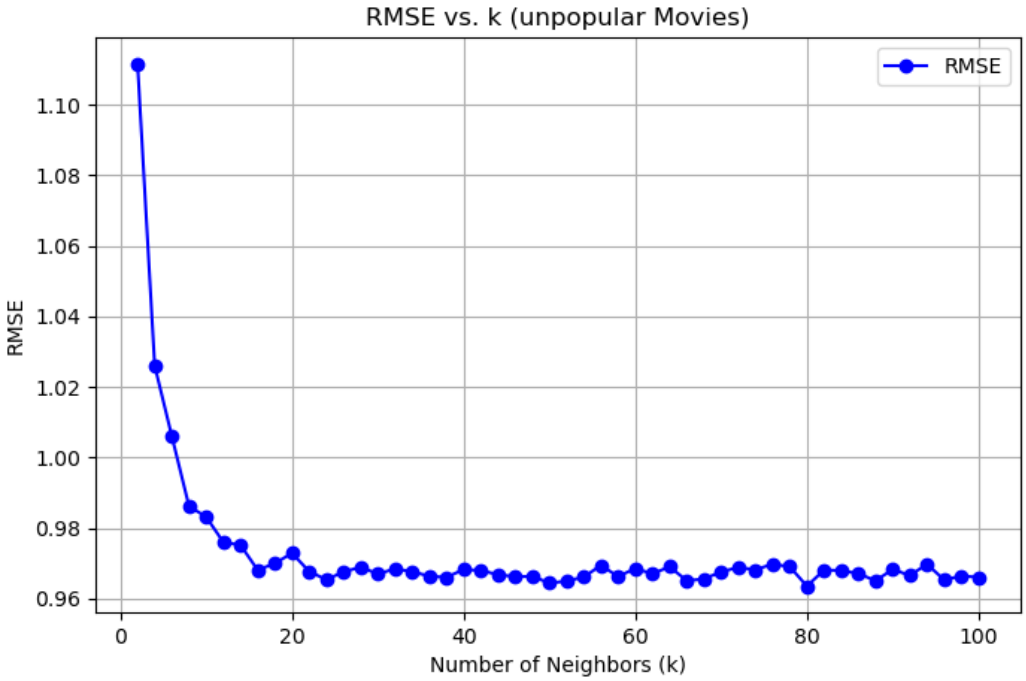
i. For popular movies, the best k is 50, with RMSE of 0.9532.

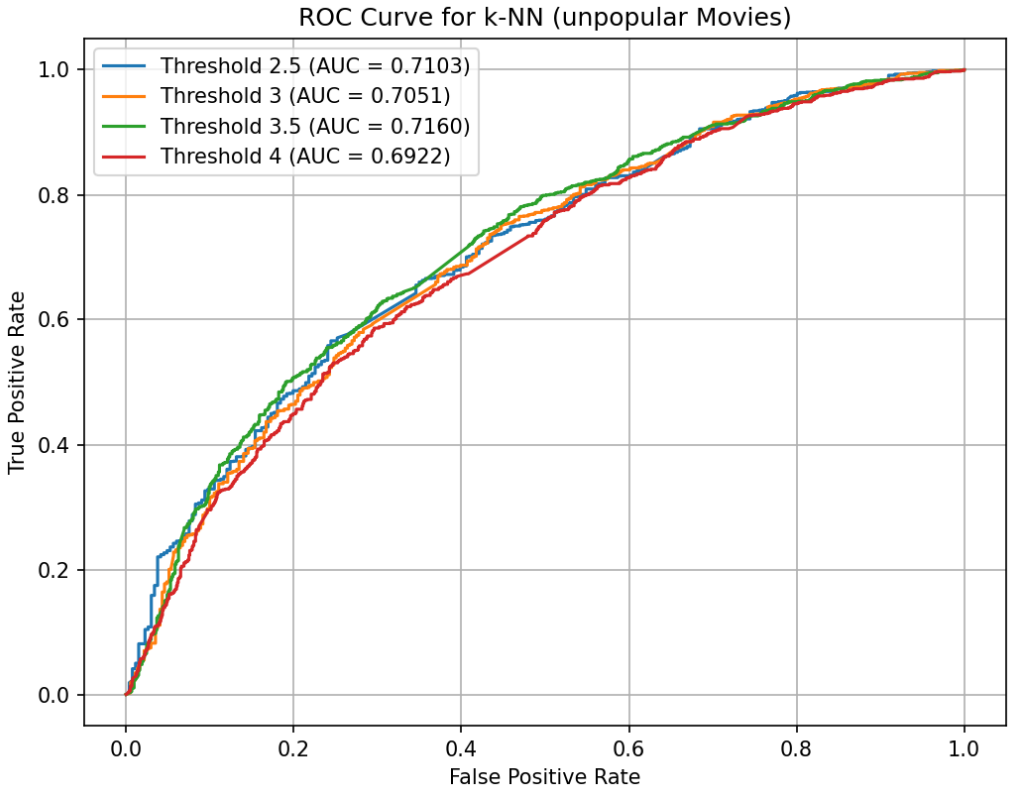




Threshold	AUC value
2.5	0.7325
3.0	0.7260
3.5	0.7230
4.0	0.7189

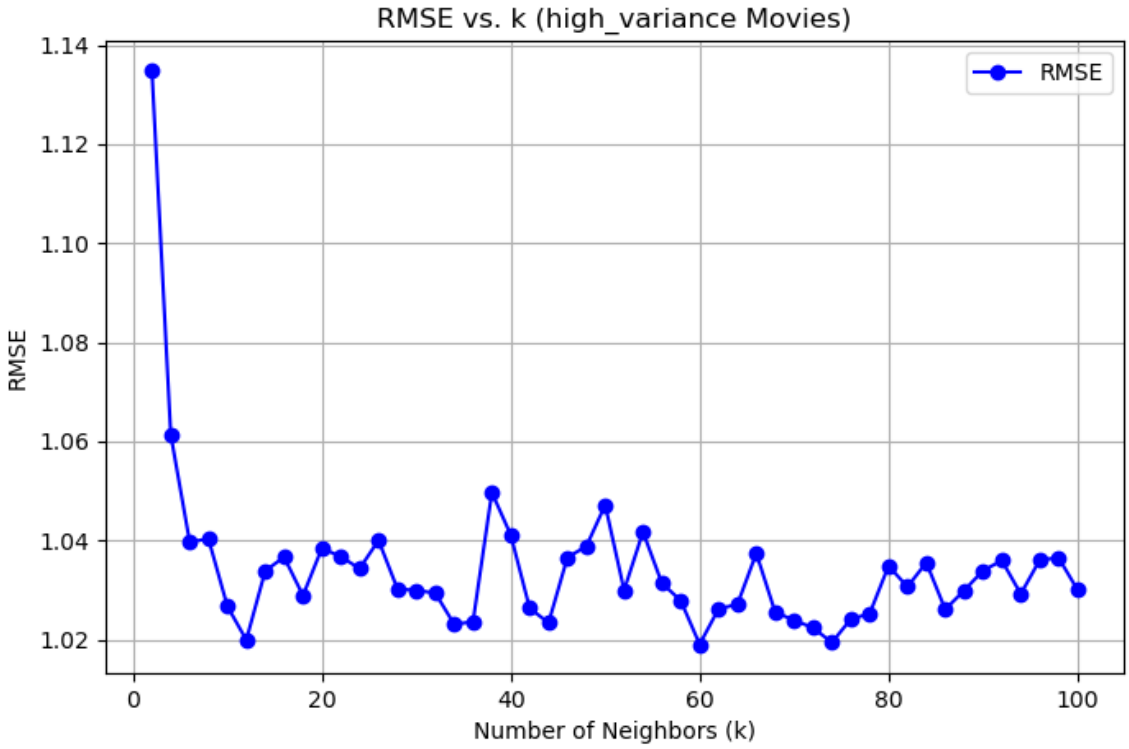
ii. Ratings For unpopular movies, the best k is 80, with RMSE of 0.9635.

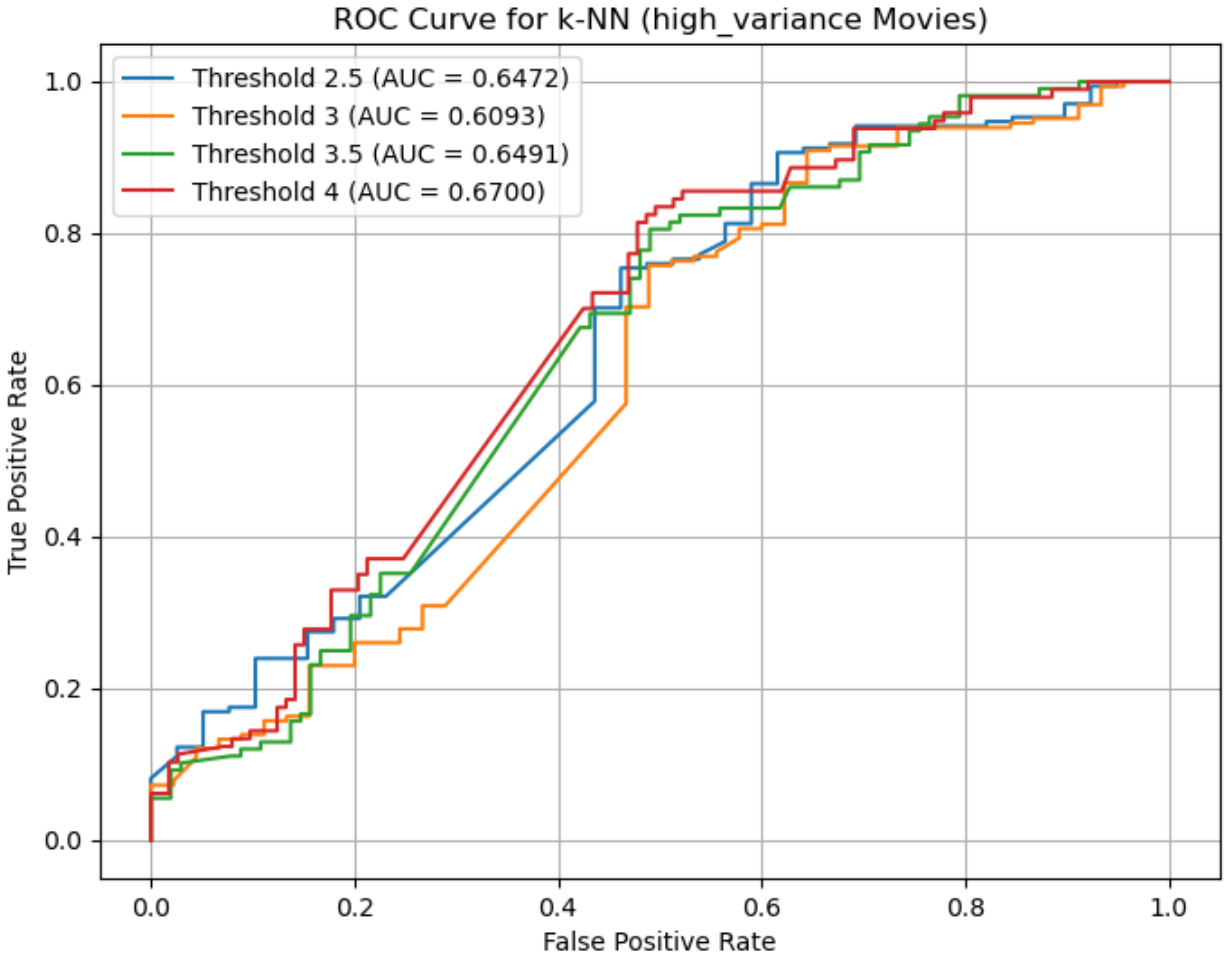




Threshold	AUC value
2.5	0.7103
3.0	0.7051
3.5	0.7160
4.0	0.6922

iii. For High Variance Movies, the best k is 60, with RMSE of 1.0190.





Threshold	AUC value
2.5	0.6472
3.0	0.6093
3.5	0.6491
4.0	0.6700

QUESTION 7: Understanding the NMF cost function: 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 given by equation 5 is not convex. Because it contains bilinear terms involving both U and V.

Fixing U transforms the problem into a least-squares regression problem for V, which can be solved using standard methods like normal equations or gradient descent.

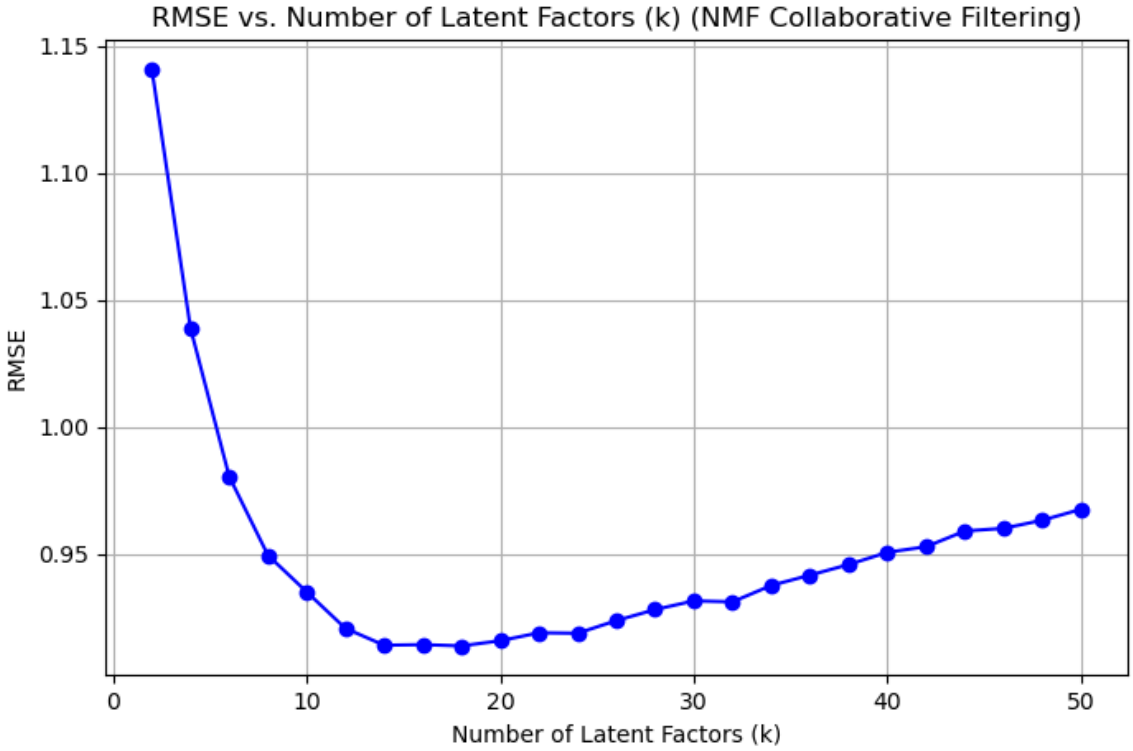
This leads to an alternating optimization approach (ALS - Alternating Least Squares). We can fix U, solve for V using least squares, then fix V, solve for U using least squares. We repeat the steps until convergence.

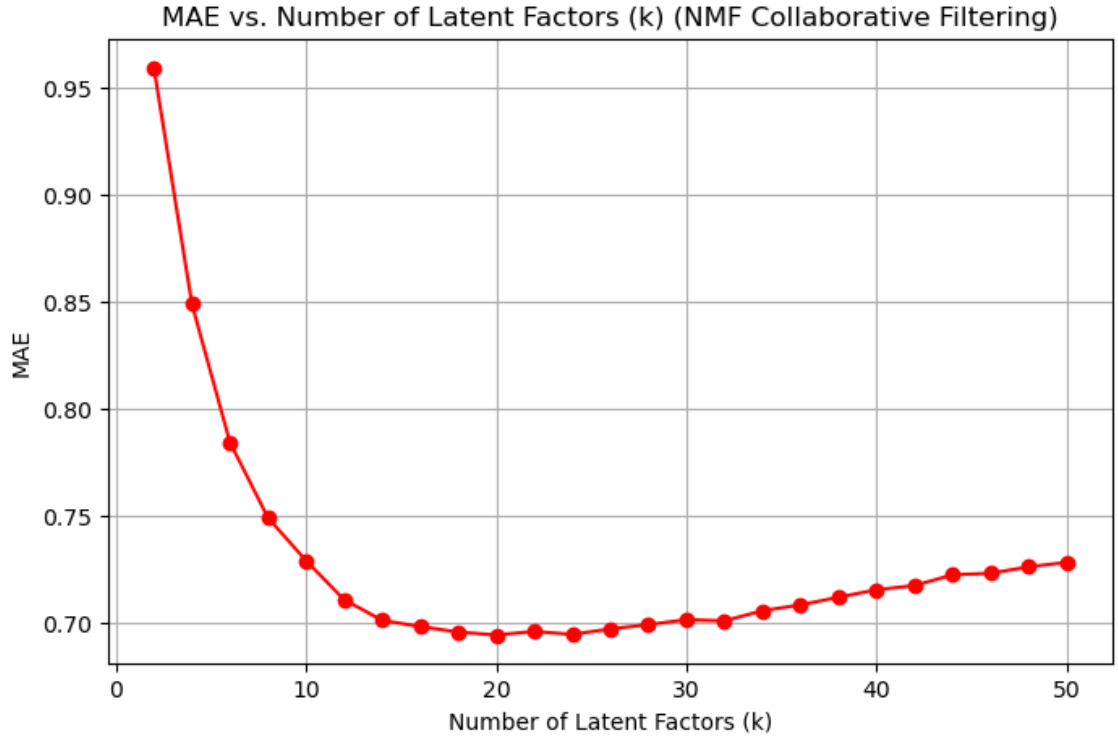
QUESTION 8: Designing the NMF Collaborative Filter

A. NMF-based collaborative filter

Optimal k (min RMSE) = 18, Minimum RMSE = 0.9139

Optimal k (min MAE) = 20, Minimum MAE = 0.6947





B.

Optimal k (min RMSE) = 16, Minimum RMSE = 0.9130

Optimal k (min MAE) = 22, Minimum MAE = 0.6934

Number of unique movie genres: 20

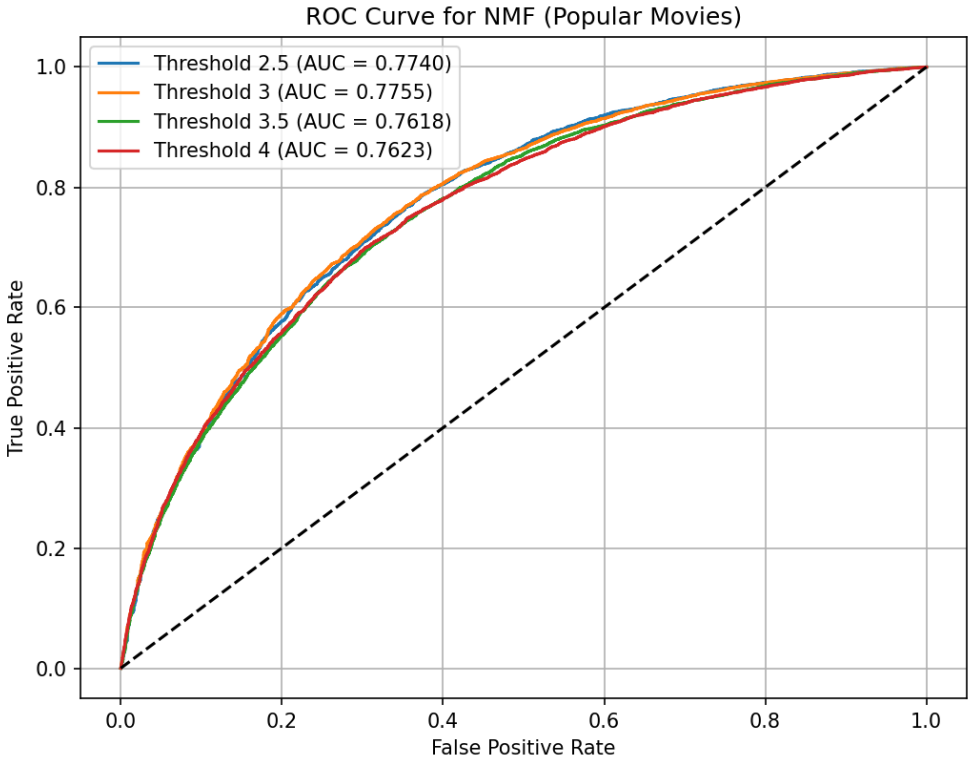
The optimal number of latent factors is different from the number of movie genres.

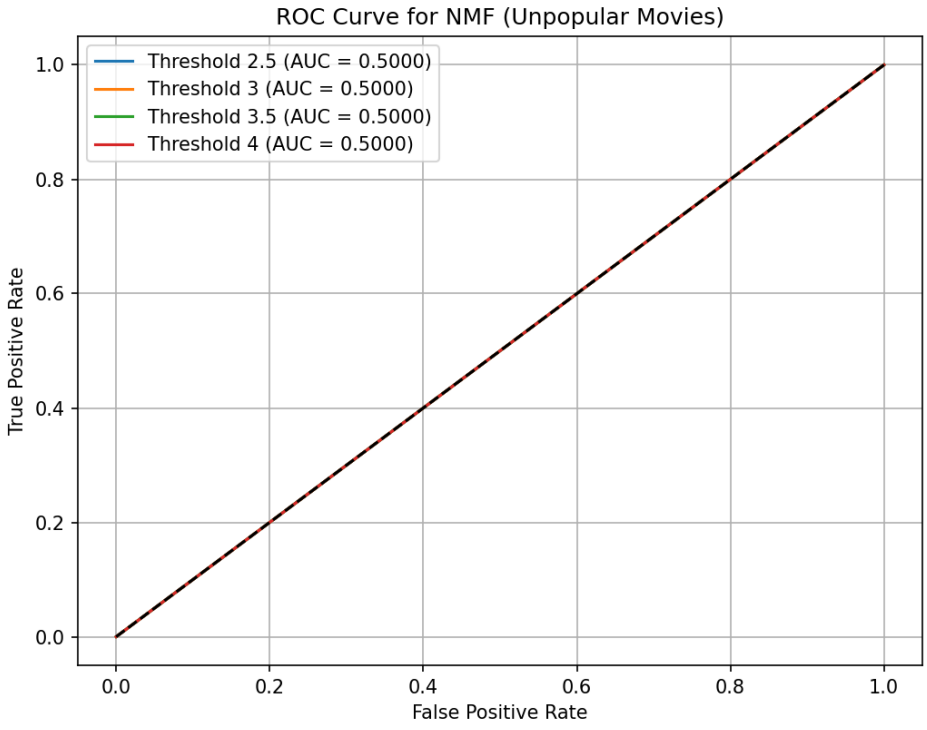
C.

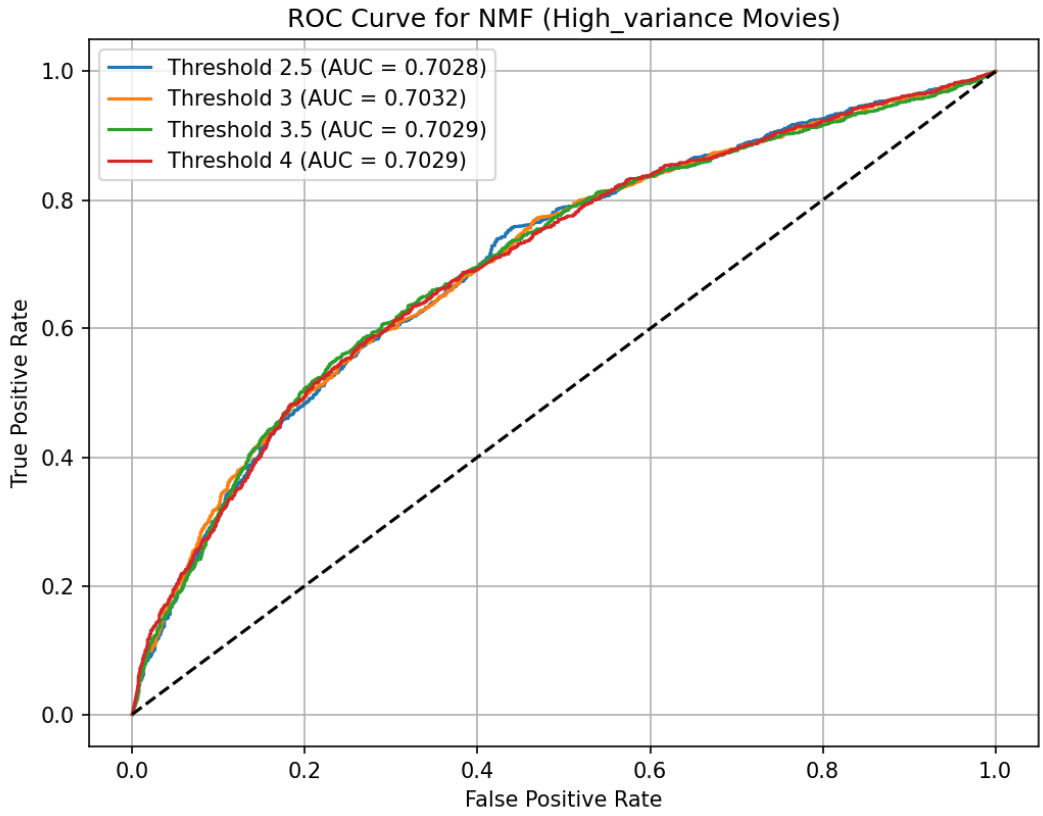
Optimal k for popular movies: 16, Minimum RMSE: 0.8711

Optimal k for unpopular movies: 32, Minimum RMSE: 1.1241

Optimal k for high_variance movies: 44, Minimum RMSE: 1.1906







QUESTION 9: Interpreting the NMF model: 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 genres? Is there a connection between the latent factors and the movie genres?

According to the results obtained by NMF decomposition, each latent factor is highly correlated with a specific type of movie. This indicates that NMF has learned user preference patterns and can categorize movies into different subject categories.

Below is a detailed analysis of Latent Factors 1, 11, and 20 as examples.

- Latent Factor 1:

  Movie ID: 5418, Genres: Action, Mystery, Thriller

  Movie ID: 5349, Genres: Action, Adventure, Sci-Fi, Thriller

Movie ID: 51662, Genres: Action, Fantasy, War, IMAX

Movie ID: 8636, Genres: Action, Adventure, Sci-Fi, IMAX

Movie ID: 6333, Genres: Action, Adventure, Sci-Fi, Thriller

Movie ID: 3793, Genres: Action, Adventure, Sci-Fi

Movie ID: 6365, Genres: Action, Adventure, Sci-Fi, Thriller, IMAX

Movie ID: 8644, Genres: Action, Adventure, Sci-Fi, Thriller

Movie ID: 6539, Genres: Action, Adventure, Comedy, Fantasy

Movie ID: 1527, Genres: Action, Adventure, Comedy, Sci-Fi

Dominant Genres: Action (10), Adventure (8), Sci-Fi (7), Thriller (5), IMAX (3), Fantasy (2), Comedy (2), Mystery (1), War (1)

This latent factor is mainly related to the "action + adventure + sci-fi" genre, which usually contains exciting fight scenes, futuristic technology elements, hero characters, etc.

Many movies also have an IMAX label, which may mean that the factor leans towards high-budget, big-budget sci-fi action blockbusters (like Marvel movies, Star Wars franchises).

This factor has almost no drama or romance, indicating that it primarily describes high-intensity action scenes and visuals-driven films.

- Latent Factor 11:

Movie ID: 5903, Genres: Action, Sci-Fi, Thriller

Movie ID: 1278, Genres: Comedy, Fantasy

Movie ID: 54259, Genres: Adventure, Comedy, Fantasy, Romance

Movie ID: 2985, Genres: Action, Crime, Drama, Sci-Fi, Thriller

Movie ID: 7387, Genres: Action, Drama, Horror

Movie ID: 4105, Genres: Fantasy, Horror, Thriller

Movie ID: 551, Genres: Animation, Children, Fantasy, Musical

Movie ID: 2, Genres: Adventure, Children, Fantasy

Movie ID: 3868, Genres: Action, Comedy, Crime, Romance

Movie ID: 1201, Genres: Action, Adventure, Western

Dominant Genres: Action (5), Fantasy (5), Thriller (3), Comedy (3), Adventure (3), Sci-Fi (2), Romance (2), Crime (2), Drama (2), Horror (2), Children (2), Animation (1), Musical (1), Western (1)

This factor has a mixed characteristic of several genres, but it can still be seen that "action +

fantasy" is the dominant type. It also has some similarities with latent factor 1 (e.g. action and thriller), but it also contains some children's movies, animated films, and horror movies.

- Latent Factor 20:

  Movie ID: 232, Genres: Comedy, Drama, Romance

  Movie ID: 21, Genres: Comedy, Crime, Thriller

  Movie ID: 441, Genres: Comedy

  Movie ID: 1719, Genres: Drama

  Movie ID: 348, Genres: Comedy

  Movie ID: 1912, Genres: Comedy, Crime, Drama, Romance, Thriller

  Movie ID: 1179, Genres: Crime, Drama, Film-Noir

  Movie ID: 2395, Genres: Comedy, Drama

  Movie ID: 39, Genres: Comedy, Romance

  Movie ID: 1635, Genres: Drama

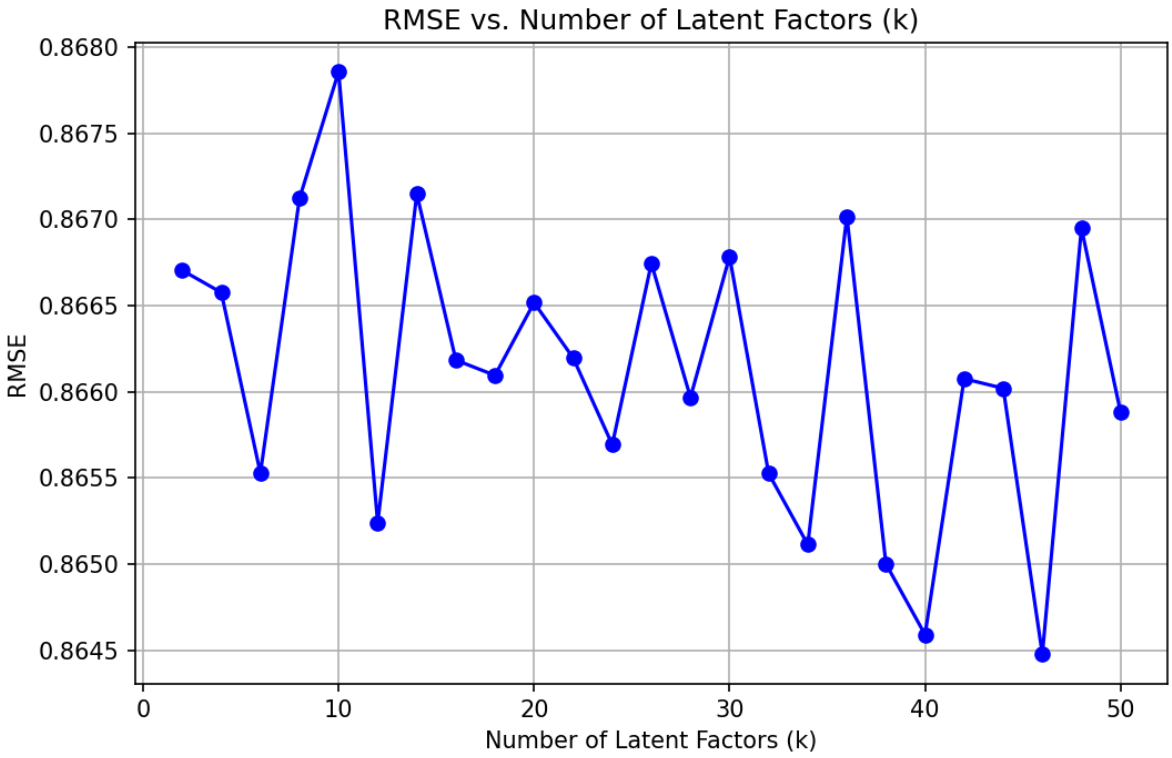
  Dominant Genres: Comedy (7), Drama (6), Romance (3), Crime (3), Thriller (2), Film-Noir (1)

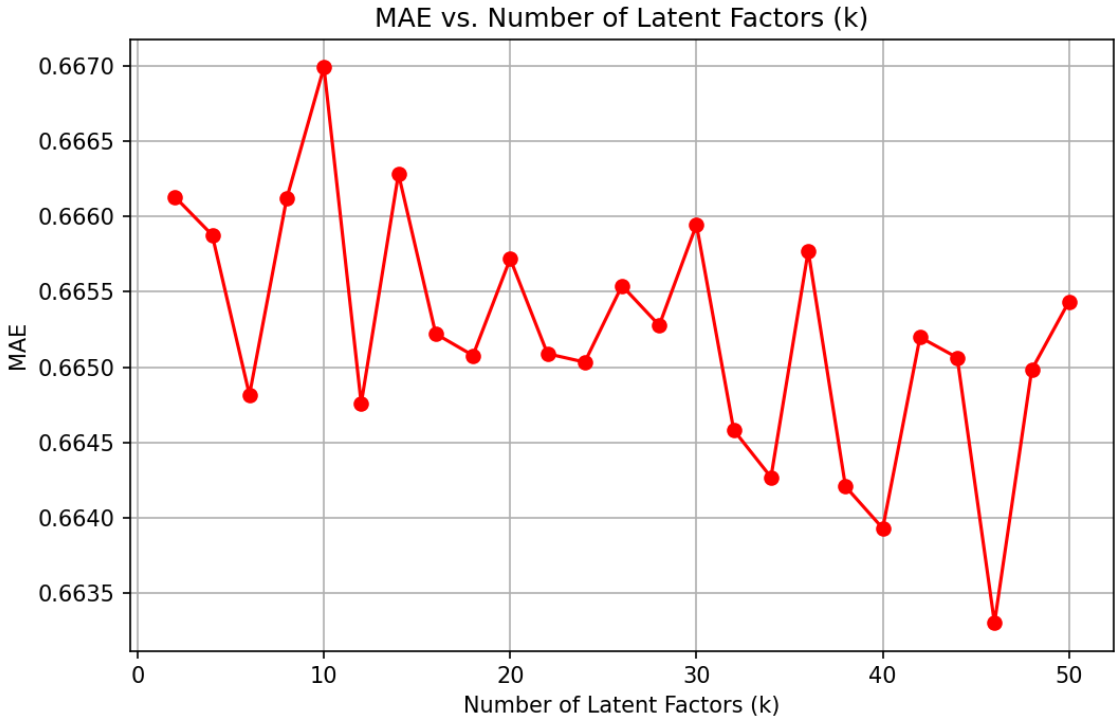
This factor is mainly made up of comedy, drama, and romance films, indicating that it mainly describes films that are lighthearted and humorous, with social-emotional elements. It is very different from the first two and does not have any action, adventure, or sci-fiction movies. This factor may represent films that are closer to everyday life and social relationships, rather than visually driven blockbusters.

QUESTION 10: Designing the MF Collaborative Filter

A.

Optimal k (min RMSE) = 46, Minimum RMSE = 0.8645, Minimum MAE = 0.6633





B.

Optimal k (Min RMSE): 20, Minimum RMSE: 0.8645

Optimal k (Min MAE): 46, Minimum MAE: 0.6637

Number of unique movie genres: 20

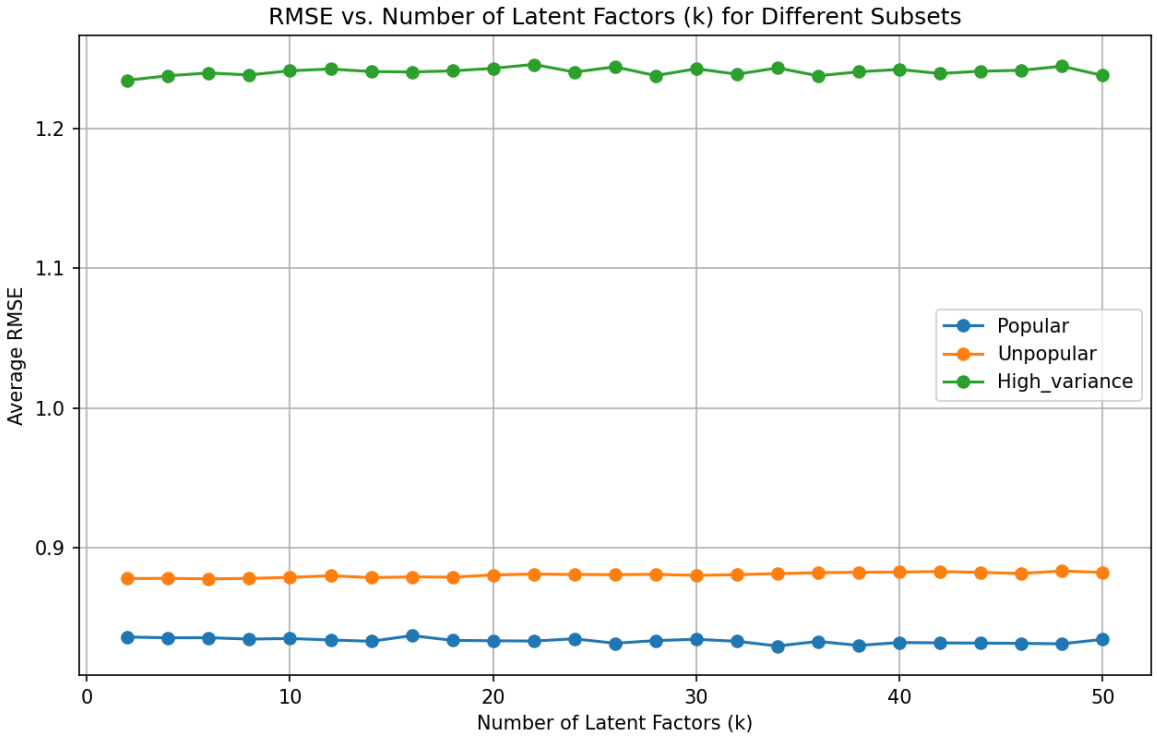
The optimal number of latent factors (20 or 46) matches the number of movie genres (20).

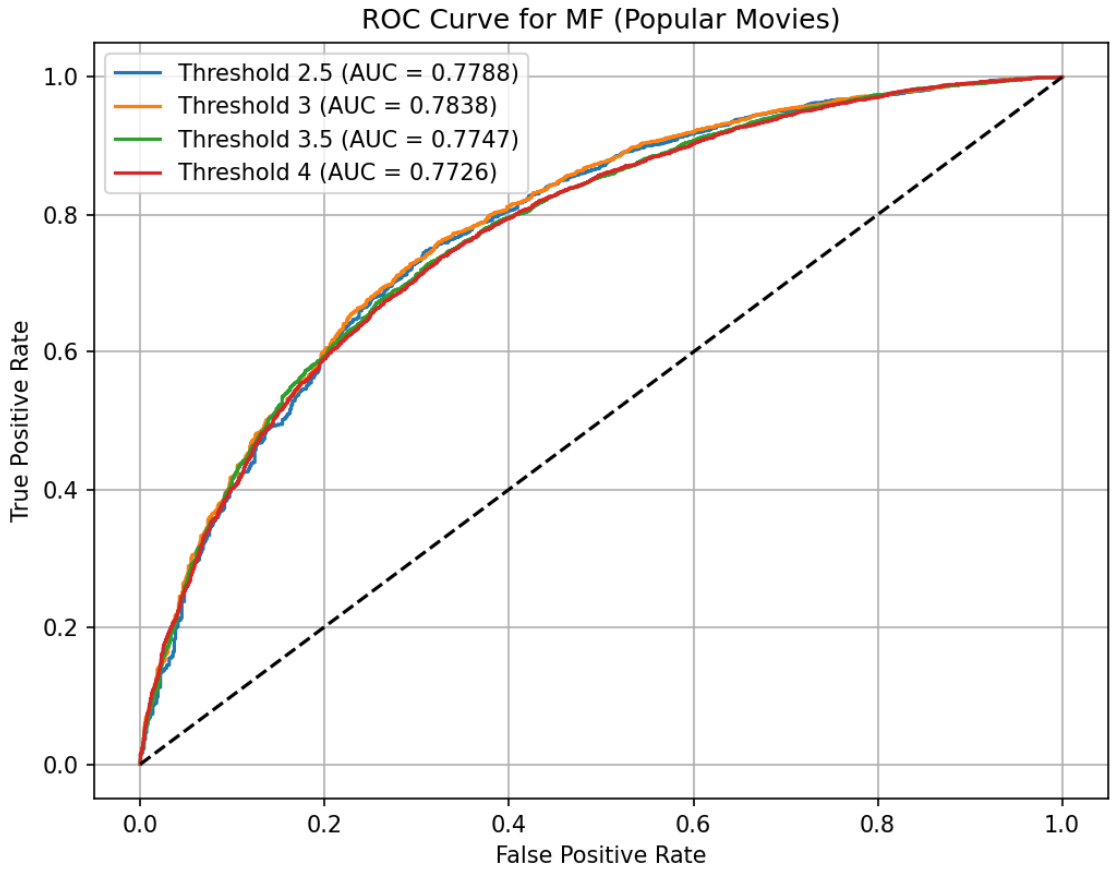
C.

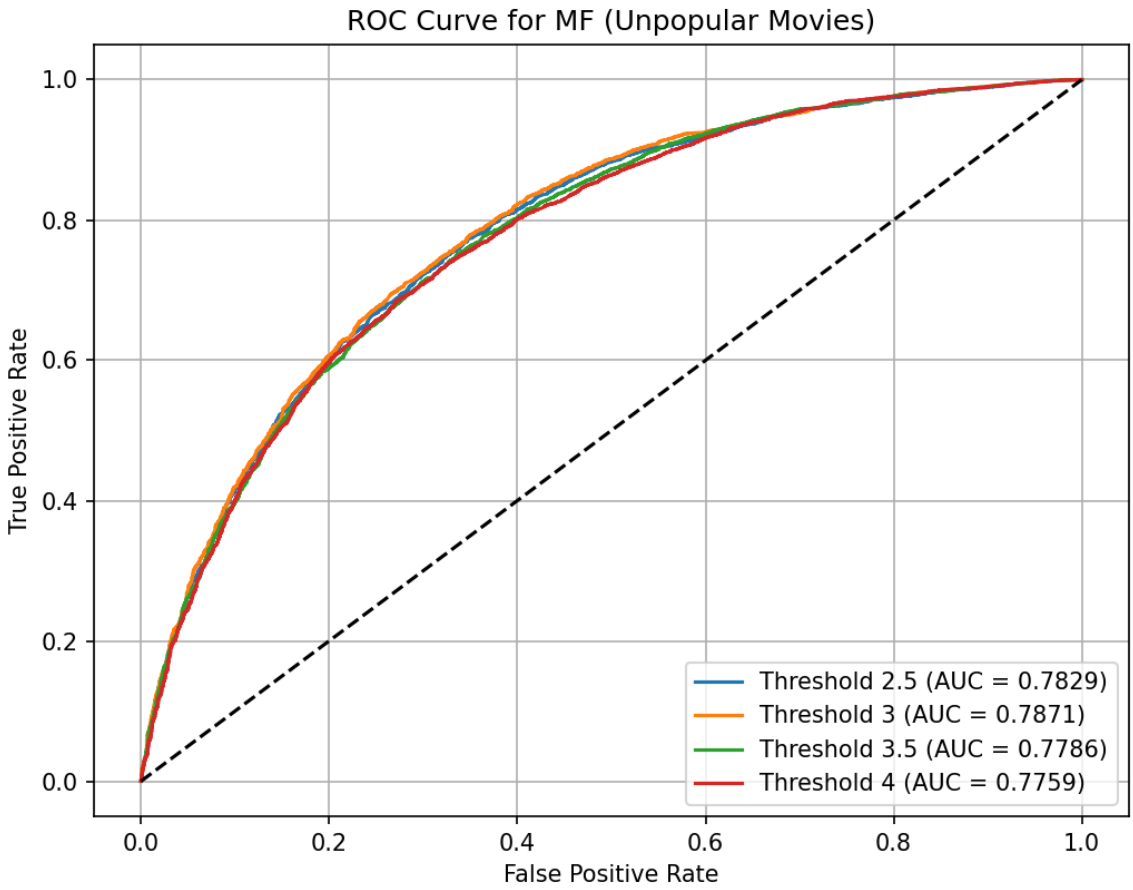
Optimal k for popular dataset: 34, Minimum RMSE: 0.8298

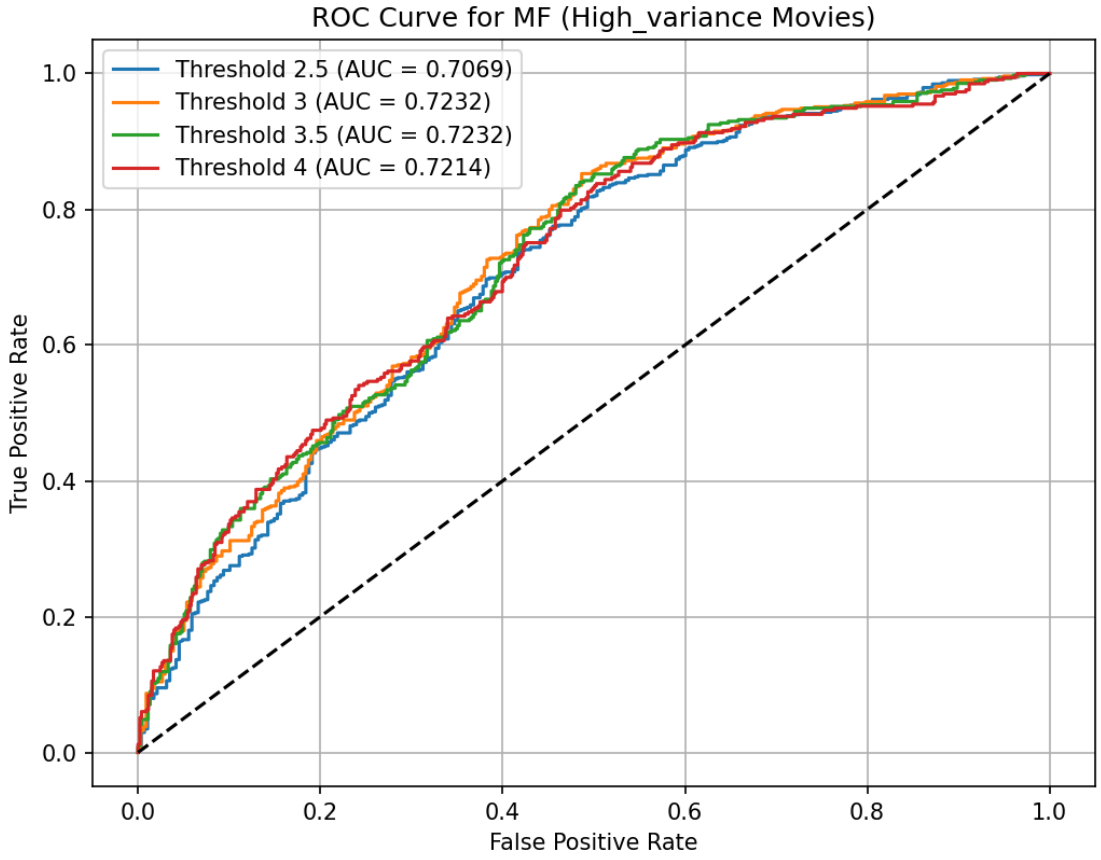
Optimal k for unpopular dataset: 6, Minimum RMSE: 0.8778

Optimal k for high_variance dataset: 2, Minimum RMSE: 1.2346






ROC Curve for MF (Unpopular Movies)
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
Threshold 2.5 (AUC = 0.7829)
Threshold 3 (AUC = 0.7871)
Threshold 3.5 (AUC = 0.7786)
Threshold 4 (AUC = 0.7759)


ROC Curve for MF (High_variance Movies)
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
Threshold 2.5 (AUC = 0.7069)
Threshold 3 (AUC = 0.7232)
Threshold 3.5 (AUC = 0.7232)
Threshold 4 (AUC = 0.7214)

QUESTION 11: For each of Popular, Unpopular and High-Variance test subsets, design a naive collaborative filter for each trimmed 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

i. Find the popular, unpopular and high-variance datasets based on their definition.

ii. Compute the cross-fold RMSE as follows:

RMSE of the oringinal dataset: 1.1781

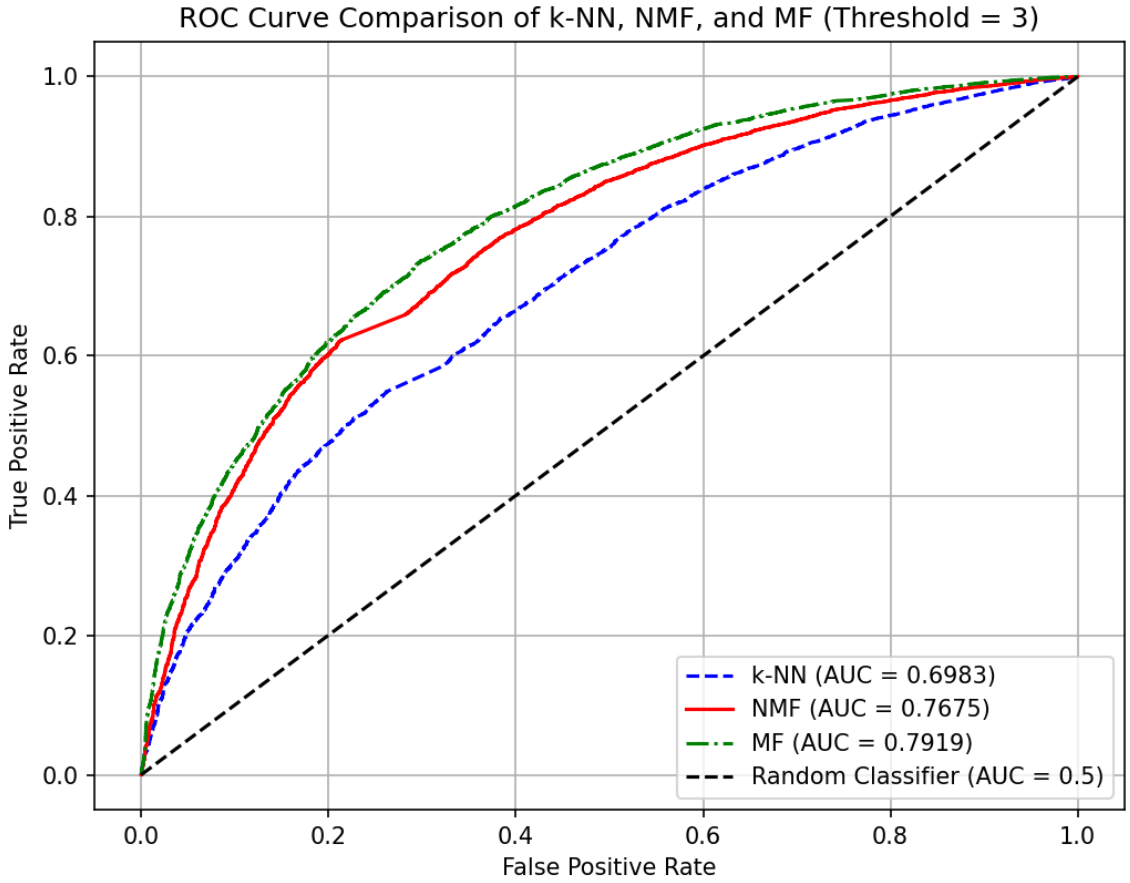
RMSE of popular dataset: 1.1722

RMSE of umpopular dataset: 2.7826

RMSE of high variance dataset: 3.2018

QUESTION 12: Comparing the most performant models across architecture: Plot the best ROC curves (threshold = 3) for the k-NN, NMF, 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.

Methods	Best k	AUC value
KNN	50	0.6983
NMF	20	0.7675
MF	20	0.7919

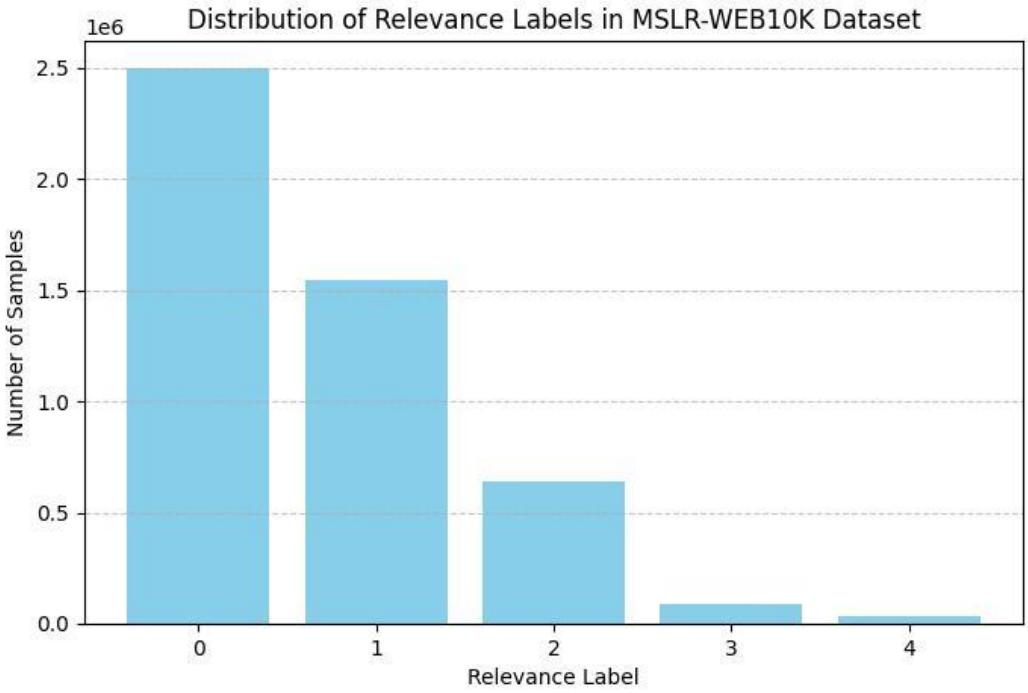


From the comparison above, we obtain that MF (SVD) is the best-performing model with the highest AUC, making it the most effective at predicting movie ratings. NMF performs well but is slightly inferior to MF, likely due to its constraints on non-negativity. k-NN performs the worst among the three, indicating that a simple neighborhood-based approach is not as effective as matrix factorization techniques.

QUESTION 13: Data Understanding and Preprocessing: Use the provided helper code for loading and pre-processing Web10k data. Print out the number of unique queries in total and show distribution of relevance labels.

i. The number of unique queries in total: 10000

ii. The distribution of relevance labels is shown in the figure below:



Comment: Most of the labels have low relevance.

QUESTION 14: LightGBM Model Training: For each of the five provided folds, train a LightGBM model using the 'lambdarank' objective. After training, evaluate and report the model's performance on the test set using nDCG@3, nDCG@5 and nDCG@10.

i. Configuration of the network:

```
params = {
            'objective': 'lambdarank',
            'metric': 'ndcg',
            'device': 'gpu',  # Use GPU for faster training
            'gpu_platform_id': 0,
            'gpu_device_id': 0,
            'learning_rate': 0.1,
            'num_leaves': 31,
            'min_data_in_leaf': 20,
            'verbose': -1,
            'lambda_l1': 1.0,
            'lambda_l2': 1.0
        }
```

Set early stopping after 5 rounds without improvement on validation set.

Set boost_round = 800.

ii. The performance on test set:

Performance on Fold1 test set:

nDCG@3: 0.5354

nDCG@5: 0.5349

nDCG@10: 0.5393

Performance on Fold2 test set:

nDCG@3: 0.5403

nDCG@5: 0.5367

nDCG@10: 0.5392

Performance on Fold3 test set:

nDCG@3: 0.5309

nDCG@5: 0.5317

nDCG@10: 0.5388

Performance on Fold4 test set:

nDCG@3: 0.5427

nDCG@5: 0.5428

nDCG@10: 0.5484

Performance on Fold5 test set:

nDCG@3: 0.5428

nDCG@5: 0.5448

nDCG@10: 0.5499

iii. Comments: Our network has an approximate 0.72 training score across all 5 folders and 3 nDCGs. The reason of bad performance on test set is, perhaps, that the high relevance labels are too rare, making the model unable to perform on test set.

QUESTION 15: Result Analysis and Interpretation: For each of the five provided folds, list top 5 most important features of the model based on the importance score.

```
       ===== Fold1 - Top 5 Features =====
  feature_name  importance_gain  importance_split
0   Column_133     22509.302441               257
1    Column_54     14422.273663               218
2   Column_129      6566.901497               814
3   Column_128      4828.602377               472
4    Column_14      4605.773697               406

       ===== Fold2 - Top 5 Features =====
  feature_name  importance_gain  importance_split
0   Column_133     22282.439733               273
1    Column_54     15825.311511               255
2   Column_129      7057.400825               917
3   Column_130      5328.208504              1253
4   Column_128      4406.970275               524

       ===== Fold3 - Top 5 Features =====
  feature_name  importance_gain  importance_split
0   Column_133     22420.718331               296
1    Column_54     13112.206020               221
2   Column_129      6768.141324               953
3   Column_128      5197.381736               460
4   Column_130      5021.586955              1221

       ===== Fold4 - Top 5 Features =====
  feature_name  importance_gain  importance_split
0   Column_133     22891.101171               273
1    Column_54     15846.240428               256
2   Column_129      5640.414552               915
3   Column_128      4834.048760               476
4   Column_130      4510.946808              1140

       ===== Fold5 - Top 5 Features =====
  feature_name  importance_gain  importance_split
0   Column_133     21986.197452               267
1    Column_54     15900.570613               234
2   Column_129      5494.489304               929
3   Column_130      5408.566897              1200
4   Column_128      5275.257707               529
```

QUESTION 16: Experiments with Subset of Features

A) 16. 1

i. We use the same network hyperparameters as in question 14 to train this network.

ii. The performance after removing top 20 features:

```
===== Performance after removing top 20 features (Fold1) =====
                        nDCG@3:  0.4428
                        nDCG@5:  0.4443
                        nDCG@10: 0.4520

===== Performance after removing top 20 features (Fold2) =====
                        nDCG@3:  0.4423
                        nDCG@5:  0.4434
                        nDCG@10: 0.4533

===== Performance after removing top 20 features (Fold3) =====
                        nDCG@3:  0.4416
                        nDCG@5:  0.4435
                        nDCG@10: 0.4568

===== Performance after removing top 20 features (Fold4) =====
                        nDCG@3:  0.4465
                        nDCG@5:  0.4481
                        nDCG@10: 0.4586

===== Performance after removing top 20 features (Fold5) =====
                        nDCG@3:  0.4384
                        nDCG@5:  0.4426
                        nDCG@10: 0.4575
```

iii. Comments: This is exactly what we expected, as the nDCG score drops by about 0.1, which is pretty significant. This is because most of the top features are removed, making the remained dataset hard to learn.

B) 16. 2

i. We use the same network hyperparameters as in question 14 to train this network.

ii. Performance after removing 60 least important features:

```
===== Performance after removing bottom 60 features (Fold1) =====
                        nDCG@3:  0.5220
                        nDCG@5:  0.5213
                        nDCG@10: 0.5255

===== Performance after removing bottom 60 features (Fold2) =====
```

```
                            nDCG@3:  0.5180
                            nDCG@5:  0.5151
                            nDCG@10: 0.5194

      ===== Performance after removing bottom 60 features (Fold3) =====
                            nDCG@3:  0.5113
                            nDCG@5:  0.5103
                            nDCG@10: 0.5162

      ===== Performance after removing bottom 60 features (Fold4) =====
                            nDCG@3:  0.5157
                            nDCG@5:  0.5157
                            nDCG@10: 0.5237

      ===== Performance after removing bottom 60 features (Fold5) =====
                            nDCG@3:  0.5348
                            nDCG@5:  0.5307
                            nDCG@10: 0.5337
```

iii. Comments: We can see that after removing 60 least important features, there's hardly any change on the test performance. Although this seems unusual, this phenomenon is understandable given the characteristic of the dataset.

Since nDCG emphasizes top-ranked documents, features affecting lower-ranked results may not show significant performance differences when removed.

The lack of improvement is likely due to the global importance bias, limited redundancy, and local significance of the removed features.