QUESTION 1: Explore the Dataset: In this question, we explore the structure of the data.

A Compute the sparsity of the movie rating dataset:

Sparsity = Total number of available ratings / Total number of possible ratings (1)

Sparsity of the dataset: 0.016999683055613623

B Plot a histogram showing the frequency of the rating values: Bin the raw 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 that fall within each bin and use this count as the height of the vertical axis for that particular bin. Comment on the shape of the histogram.

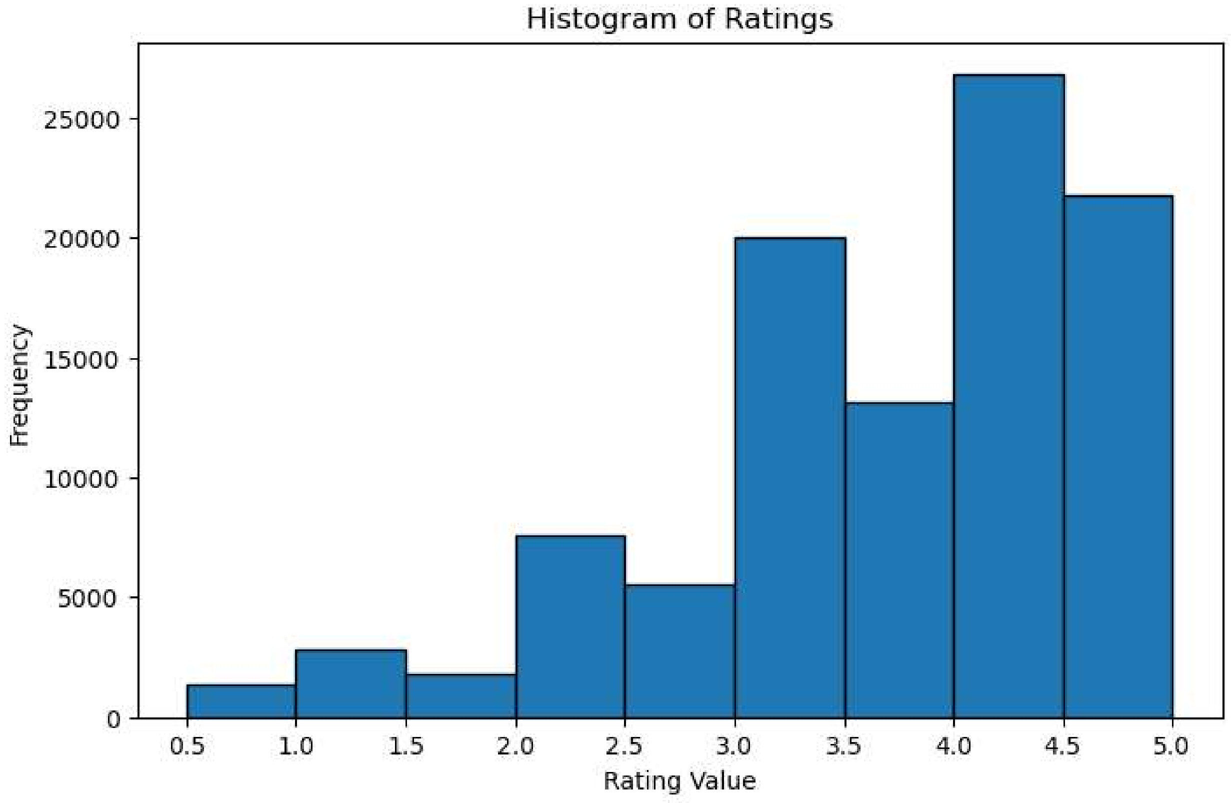

Figure 1. Histogram of Ratings

C Plot the distribution of the number of ratings received among movies: 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; ties can broken in any way. A monotonically decreasing trend is expected.

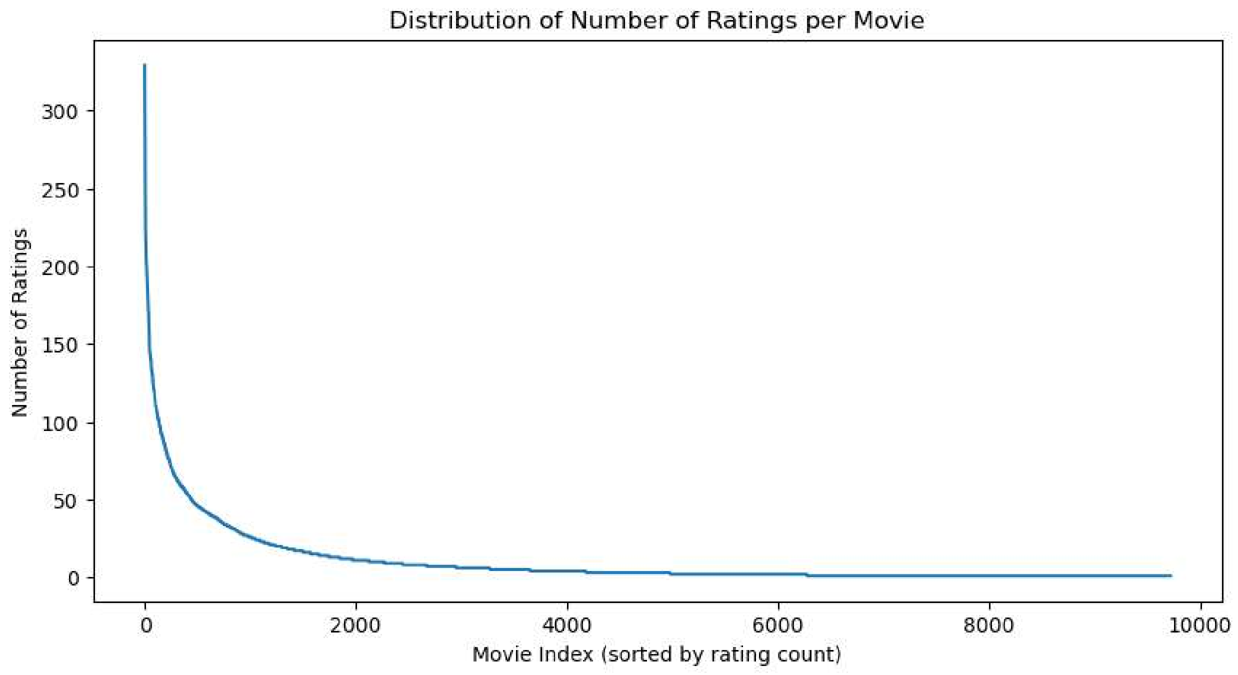

Figure 2. Distribution of the Number of Ratings Received among Movies

D Plot the distribution of ratings among users: 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 C.

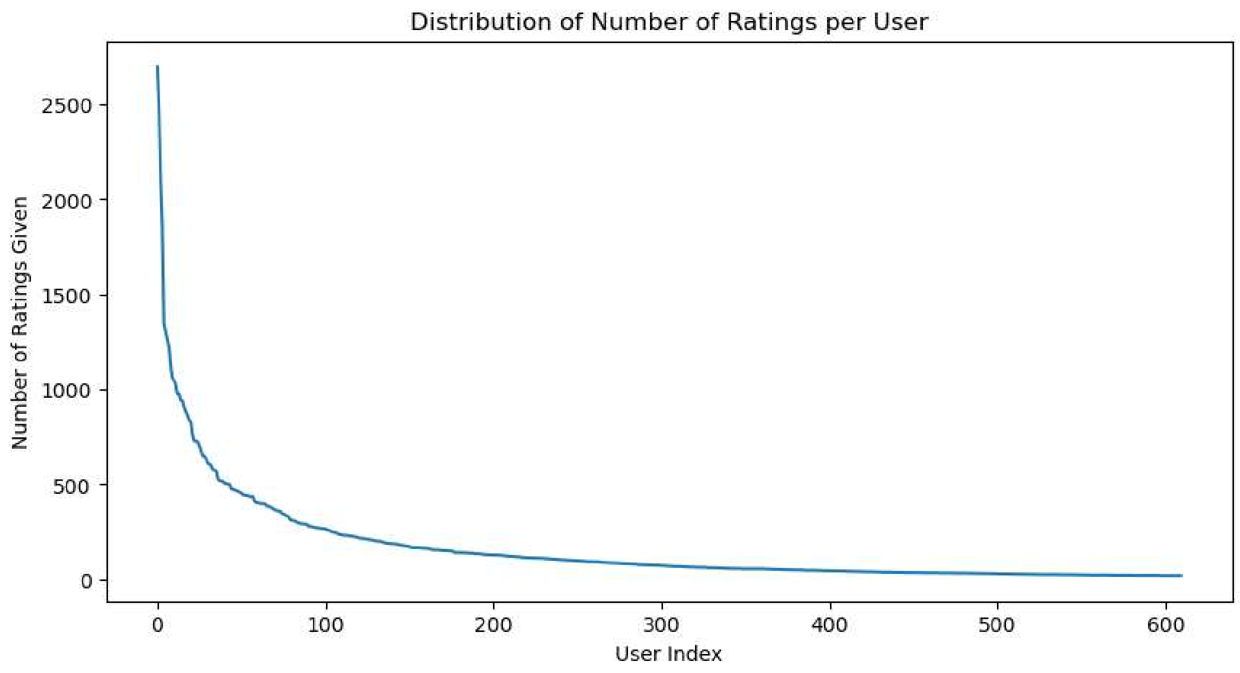

Figure 3. Distribution of Ratings among Users

E Discuss the salient features of the distributions from Questions C,D and their implications for the recommendation process.

The data shows that a few movies get most of the ratings, while many movies have very few, making it hard to recommend lesser-known films. Similarly, some users rate a lot of movies, but most users rate only a few, which makes personal recommendations tricky for new users. Since many ratings are missing, recommendation systems may mostly suggest popular movies and ignore hidden gems.

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

The histogram shows a right-skewed distribution, where most movies have low rating variance, meaning users generally agree on their scores. A few movies have high variance, indicating mixed opinions, some users rate them highly, while others give low scores. This suggests that low-variance movies are safer recommendations, while high-variance movies require more personalized recommendations.

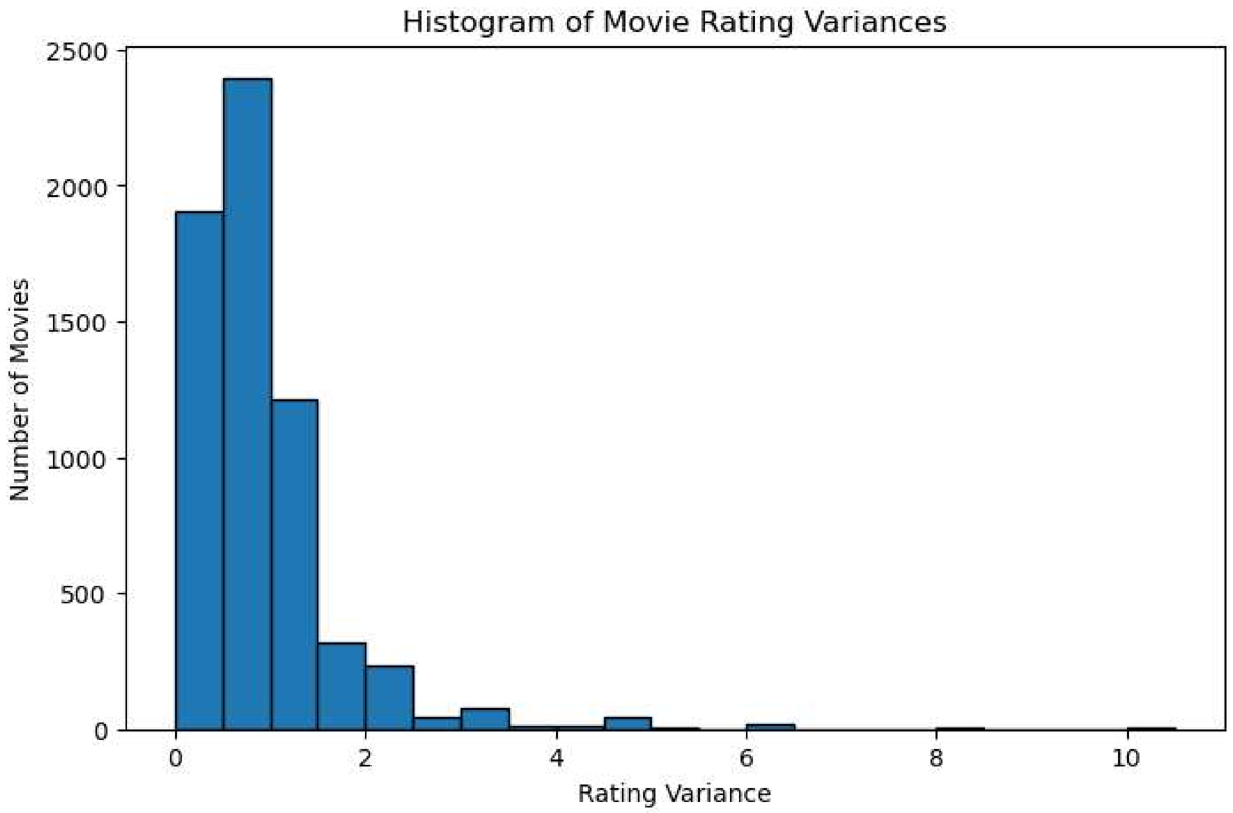

Figure 4. Histogram of Movie Rating Variances

QUESTION 2: Understanding the Pearson Correlation Coefficient:

A Write down the formula for µu in terms of Iu and ruk;

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

B In plain words, explain the meaning of Iu ∩ Iv. Can Iu ∩ Iv = ∅? (Hint: Rating matrix R is sparse)

$I_u \cap I_v$ represents the set of movies that both users u and v have rated. This intersection is important for computing similarity because Pearson correlation is only meaningful when comparing ratings for the same movies.

It is possible that $I_u \cap I_v = \emptyset$. This happens when two users have not rated any common movies. Since the rating matrix R is sparse, meaning most users rate only a small subset of movies, many user pairs will have no overlapping ratings. When this happens, Pearson correlation cannot be computed for those users.

QUESTION 3: Understanding the Prediction function: Can you explain the reason behind mean-centering the raw ratings (rvj − µv) in the prediction function? (Hint: Consider users who either rate all items highly or rate all items poorly and the impact of these users on the prediction function.)

Mean-centering helps adjust for users who always rate movies high or low, making predictions fairer. Some users might give mostly 5-star ratings, while others rate harshly with mostly 2s or 3s. Without mean-centering, these biases could mislead predictions. By subtracting each user's average rating, the model focuses on how much they like a movie compared to their usual ratings, rather than their general rating habits. This makes recommendations more accurate and personalized.

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).

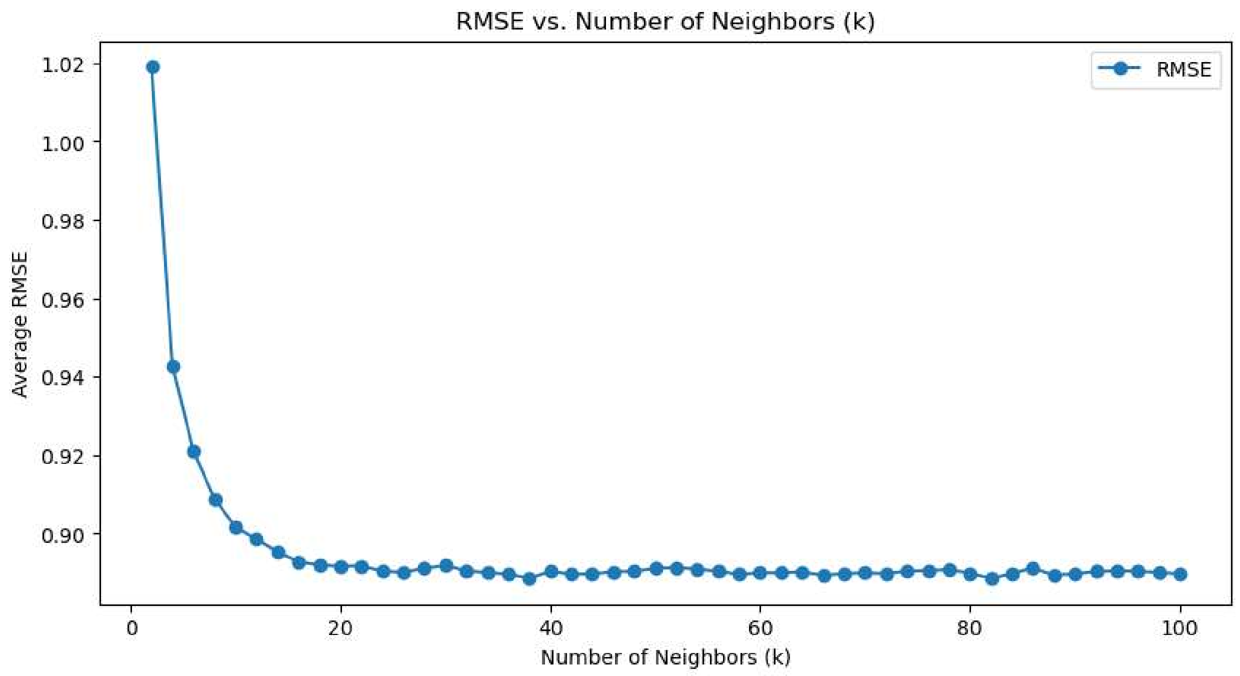

Figure 5. Average RMSE (Y-axis) against k (X-axis)

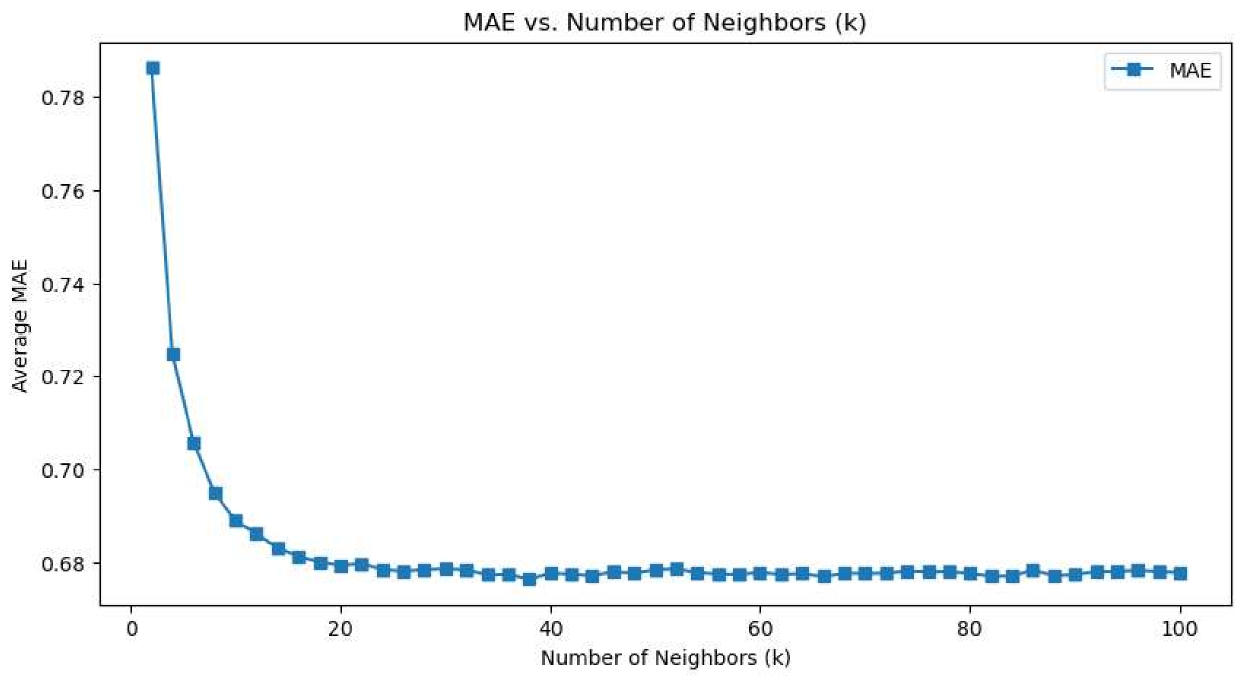

Figure 6. Average MAE (Y-axis) against k (X-axis)

QUESTION 5: Use the plot from question 4, 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.

minimum k = 20

Steady state value of average RMSE for k=20: 0.8917

Steady state value of average MAE for k=20: 0.6793

QUESTION 6: Within EACH of the 3 trimmed subsets in the dataset, design (train and validate):

A k-NN collaborative filter on the ratings of the movies (i.e Popular, Unpopular or High-Variance) and evaluate each of the three models' 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.

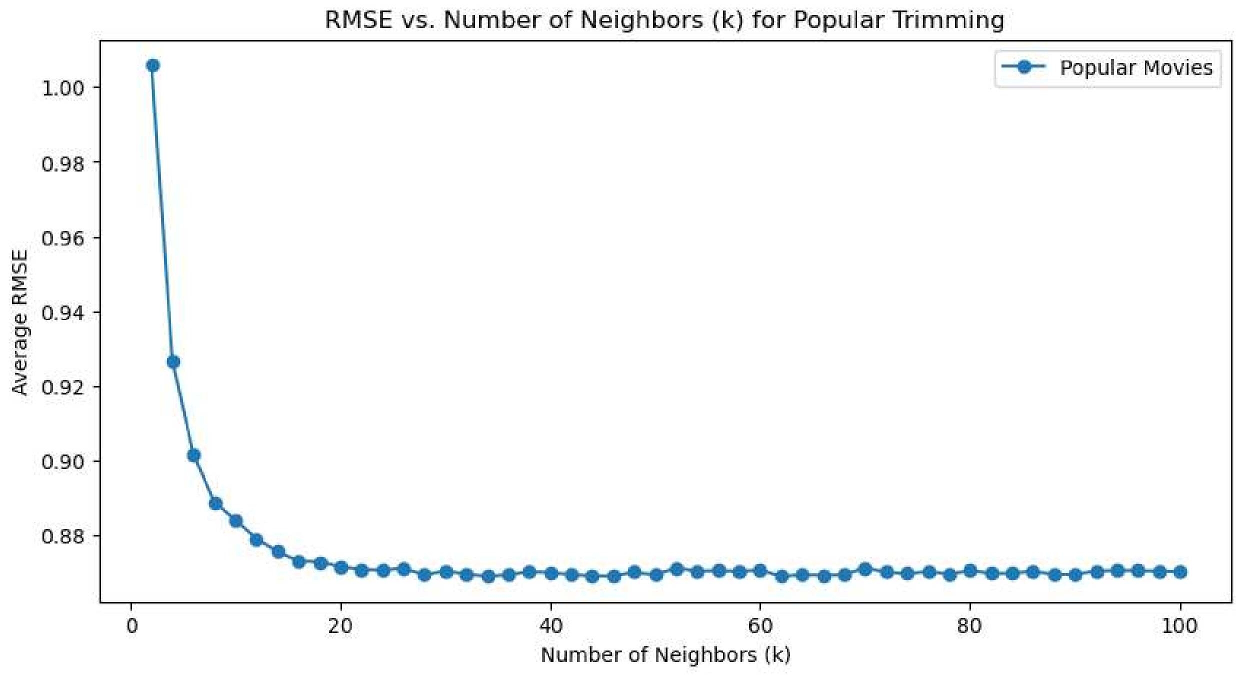

Figure 7. Average RMSE against k for Popular Trimming

Minimum RMSE for Popular Movies: 0.8690

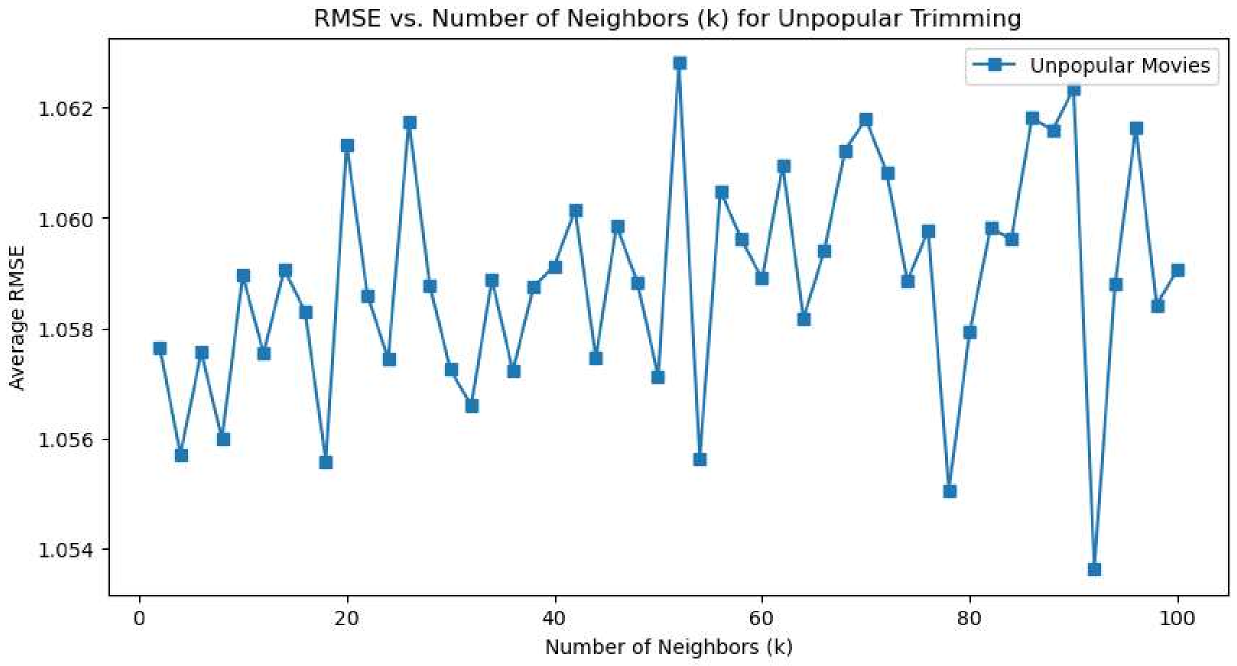

Figure 8. Average RMSE against k for Unpopular Trimming

Minimum RMSE for Unpopular Movies: 1.0536

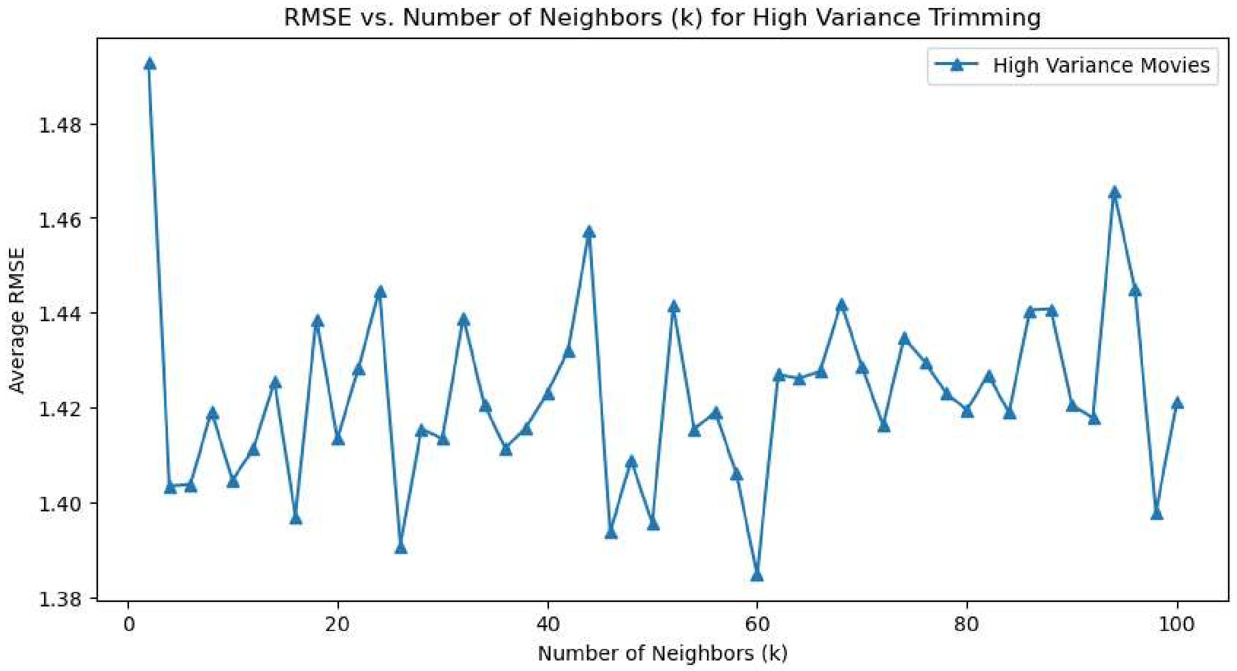

Figure 9. Average RMSE against k for High Variance Trimming

Minimum RMSE for High Variance Movies: 1.3847

• **Plot the ROC curves for the k-NN collaborative filters for threshold**

values [2.5, 3, 3.5, 4]. These thresholds are applied only on the ground truth labels in held-out validation set. For each of the plots, also report the area under the curve (AUC) value. You should have 4 × 4 plots in this section (4 trimming options – including no trimming times 4 thresholds) - all thresholds can be condensed into one plot per trimming option yielding only 4 plots.

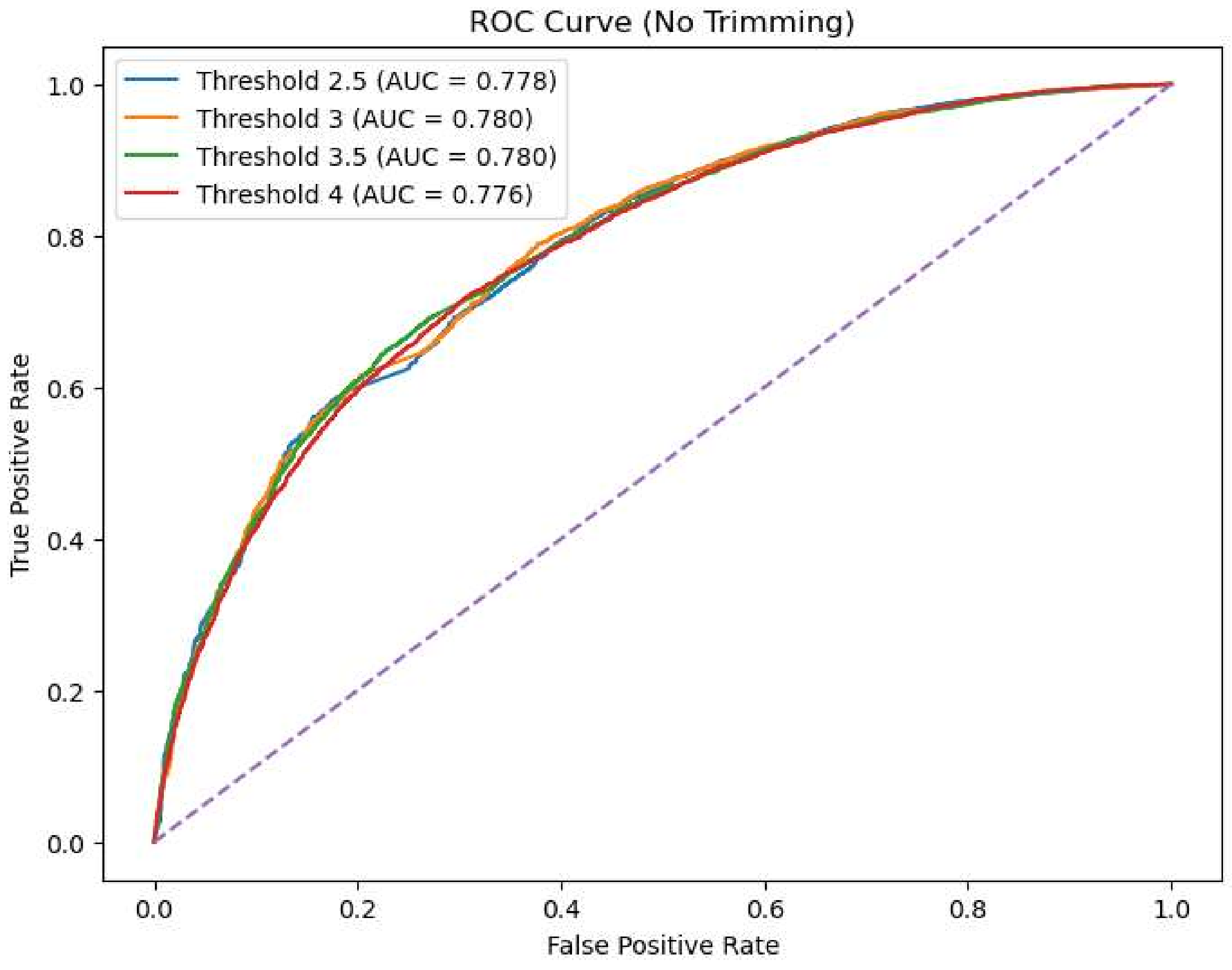

Figure 10. ROC Curve for No Trimming

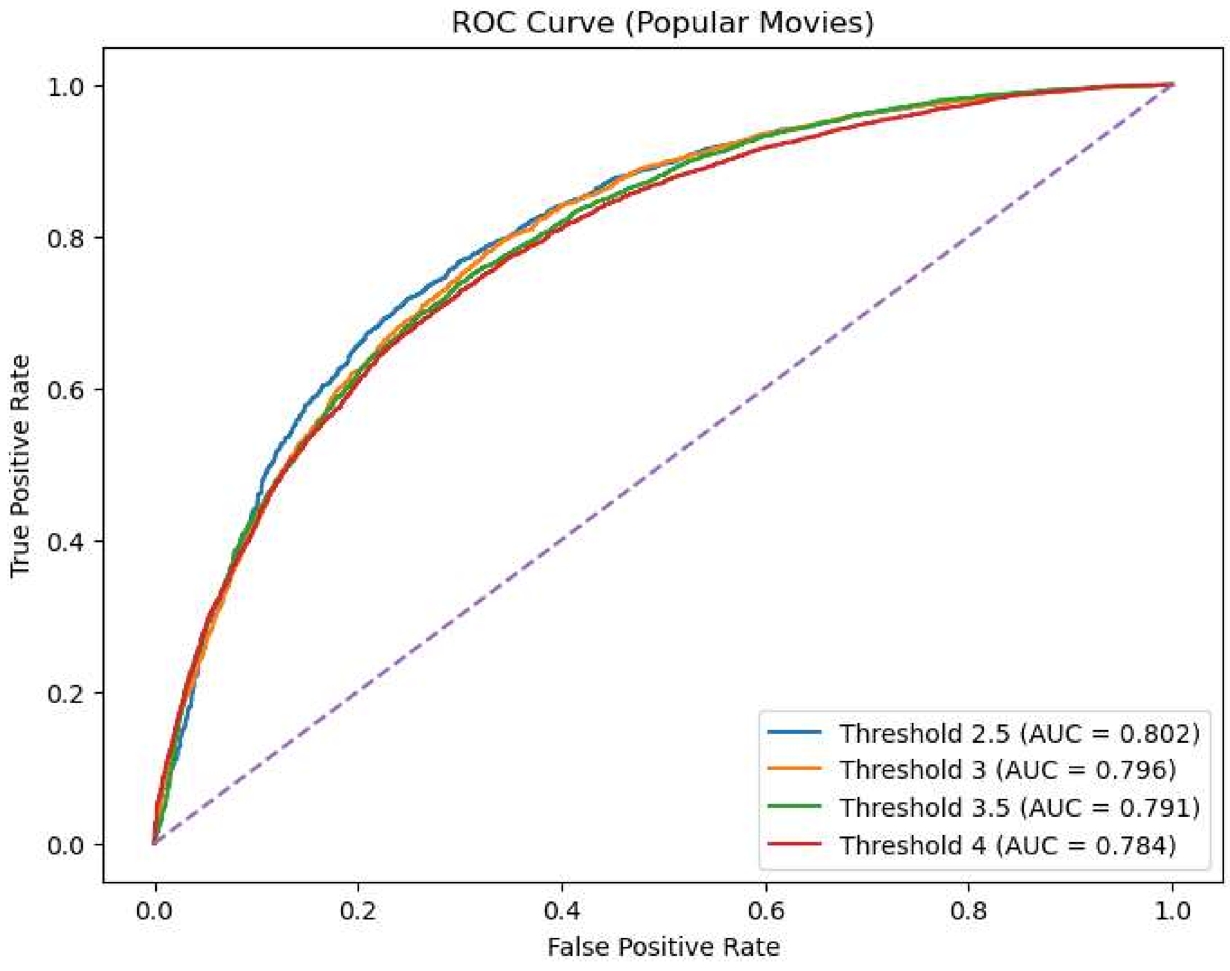

Figure 11. ROC Curve for Popular Trimming

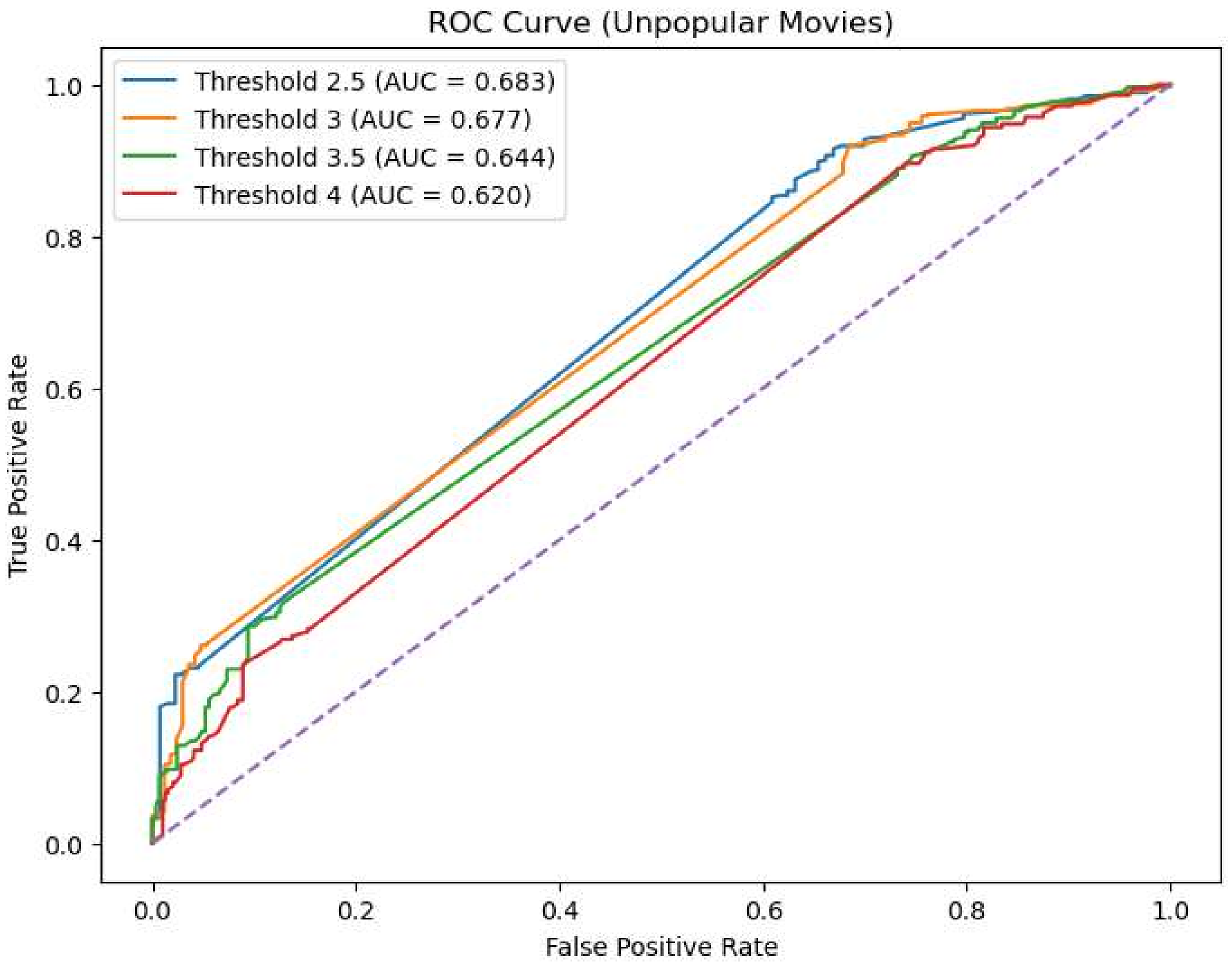

Figure 12. ROC Curve for Unpopular Trimming

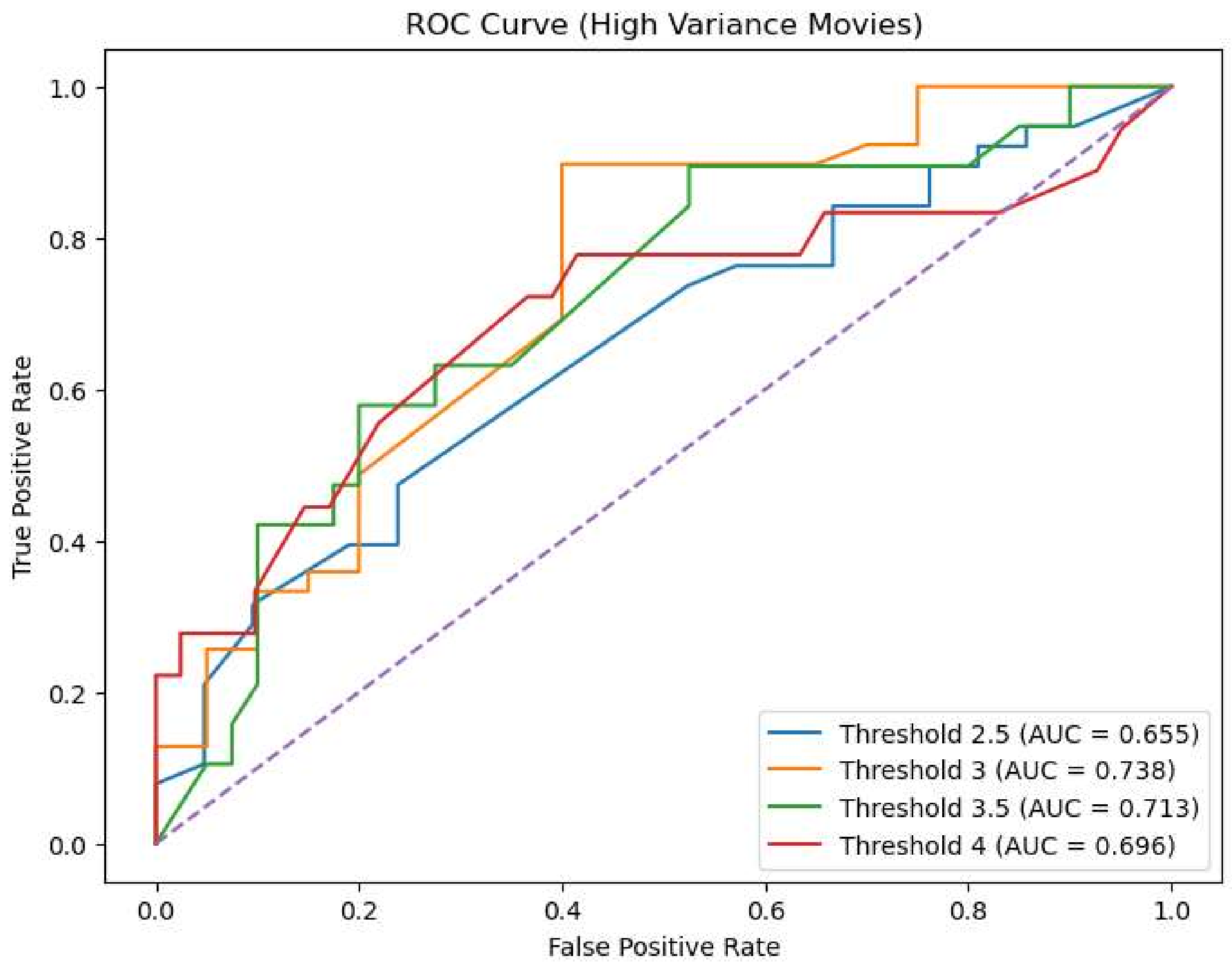

Figure 13. ROC Curve for High Variance Trimming

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 in Equation (5) is not convex, because it involves the product of two matrices U and V, making the objective function non-convex due to bilinearity.
If we fix U, we can rewrite the objective function in matrix form as

$$\min_{V} \sum_{i=1}^{m} \sum_{j=1}^{n} W_{ij}(r_{ij} - (UV^T)_{ij})^2$$

For each user i, this simplifies to solving the least-squares problem:

$$\min_{v_j} \sum_{j=1}^{n} W_{ij}(r_{ij} - u_i v_j{}^T)^2$$

QUESTION 8: Designing the NMF Collaborative Filter:

A Design a NMF-based 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 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. If NMF takes too long, you can increase the step size. Increasing it too much will result in poorer granularity in your results. 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.

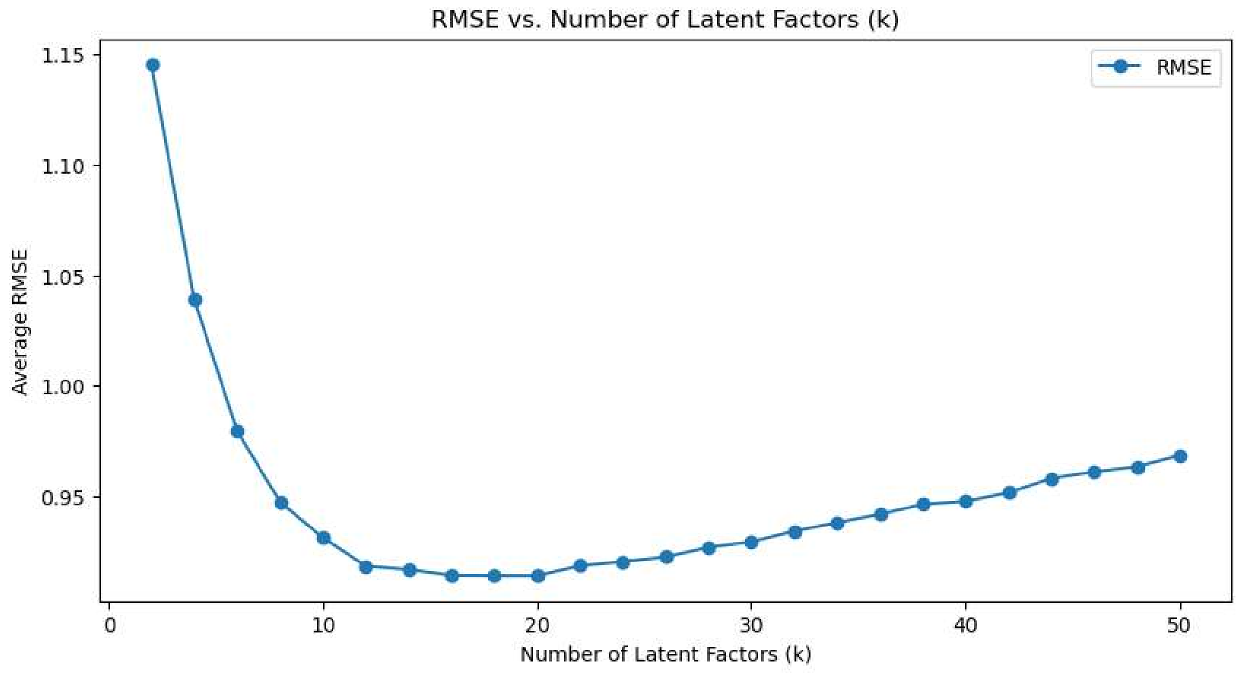

Figure 14. Average RMSE (Y-axis) against k (X-axis)

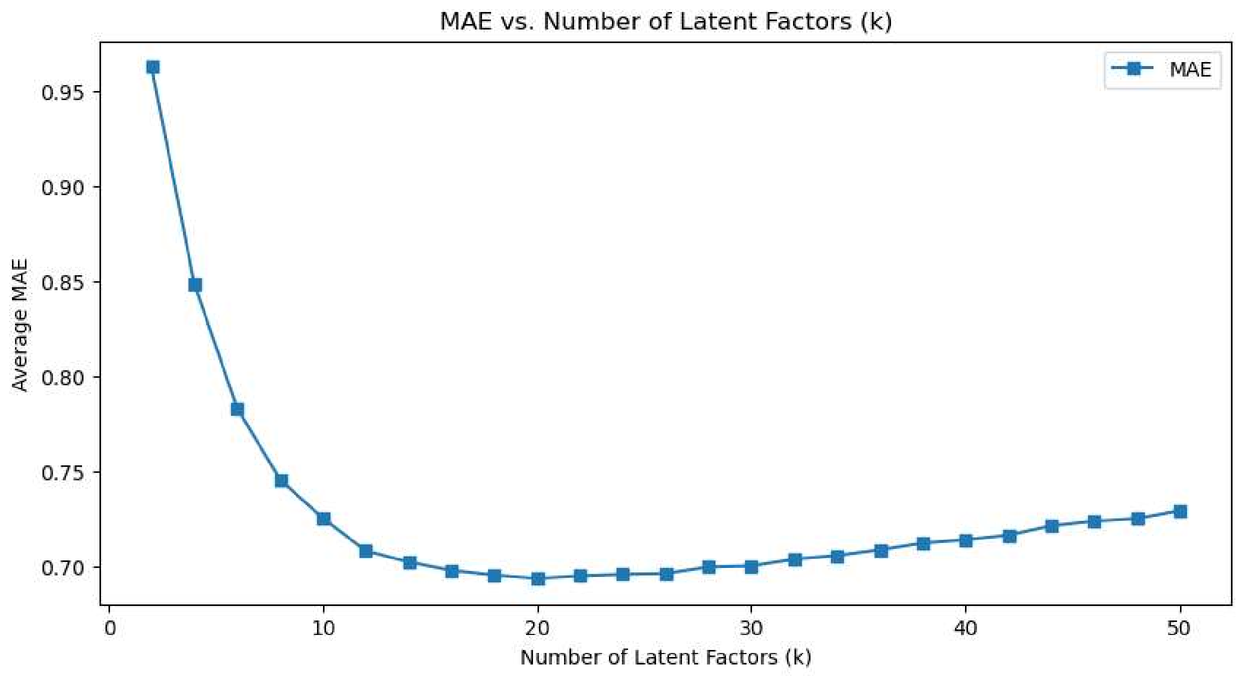

Figure 15. Average MAE (Y-axis) against k (X-axis)

B Use the plot from the previous part 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?

Optimal number of latent factors (k) based on RMSE: 20
Minimum average RMSE: 0.9141
Optimal number of latent factors (k) based on MAE: 20
Minimum average MAE: 0.6938
The optimal number of latent factors does NOT match the number of movie genres.
The optimal number of latent factors does NOT match the number of movie genres.

C Performance on trimmed dataset subsets: For each of Popular, Unpopular and High-Variance subsets -
– Design a NMF collaborative filter for each trimmed subset 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); item Report the minimum average RMSE.

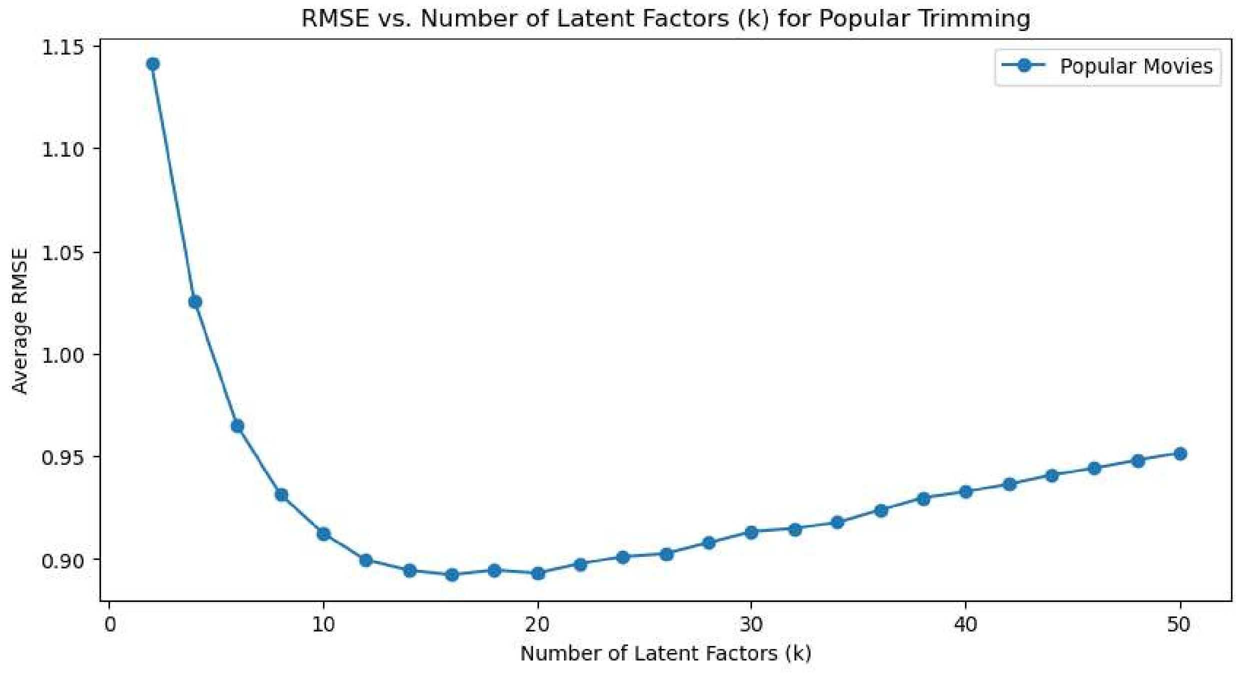

Figure 16. Average RMSE against k for Popular Trimming

Minimum RMSE for Popular Movies: 0.8923

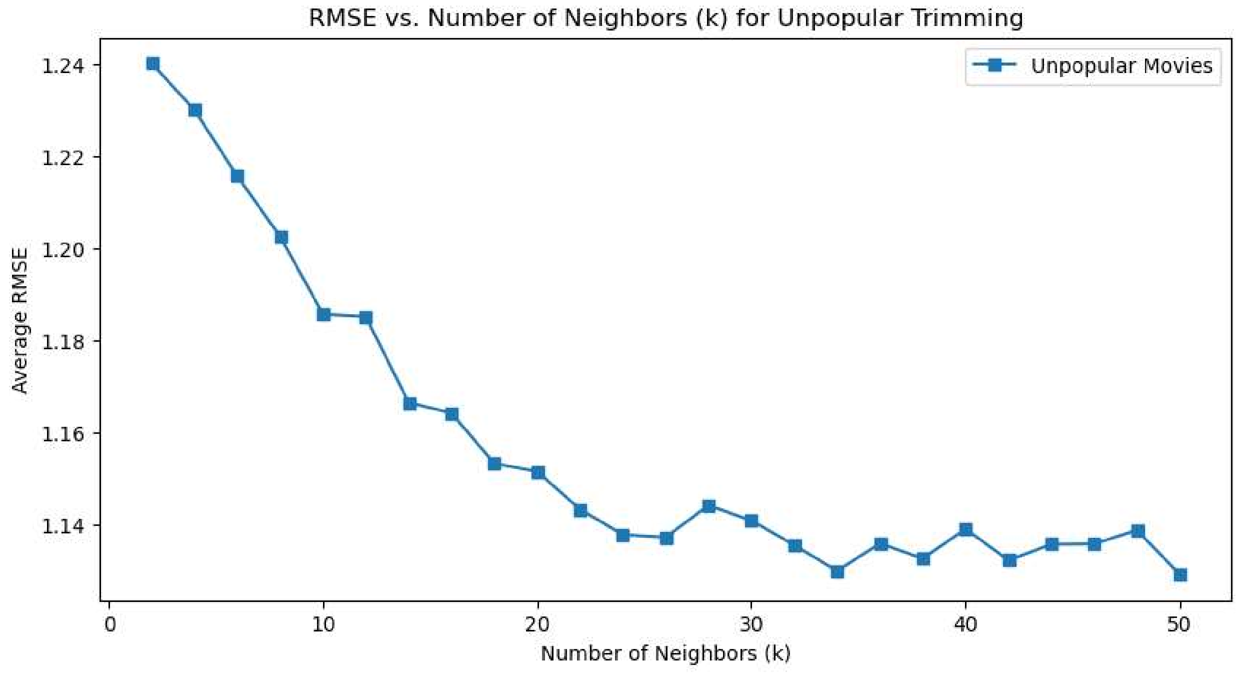

Figure 17. Average RMSE against k for Unpopular Trimming

Minimum RMSE for Unpopular Movies: 1.1291

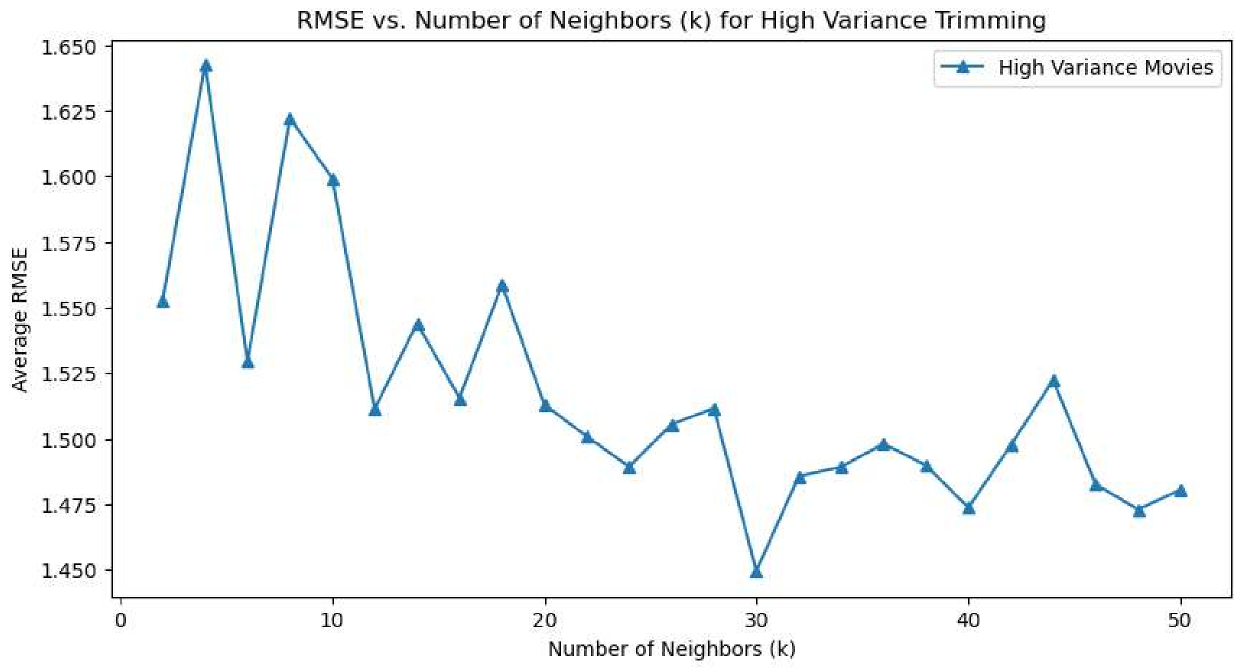

Figure 18. Average RMSE against k for High Variance Trimming

Minimum RMSE for High Variance Movies: 1.4494

• Plot the ROC curves for the NMF-based collaborative filter and also report the area under the curve (AUC) value as done in Question 6.

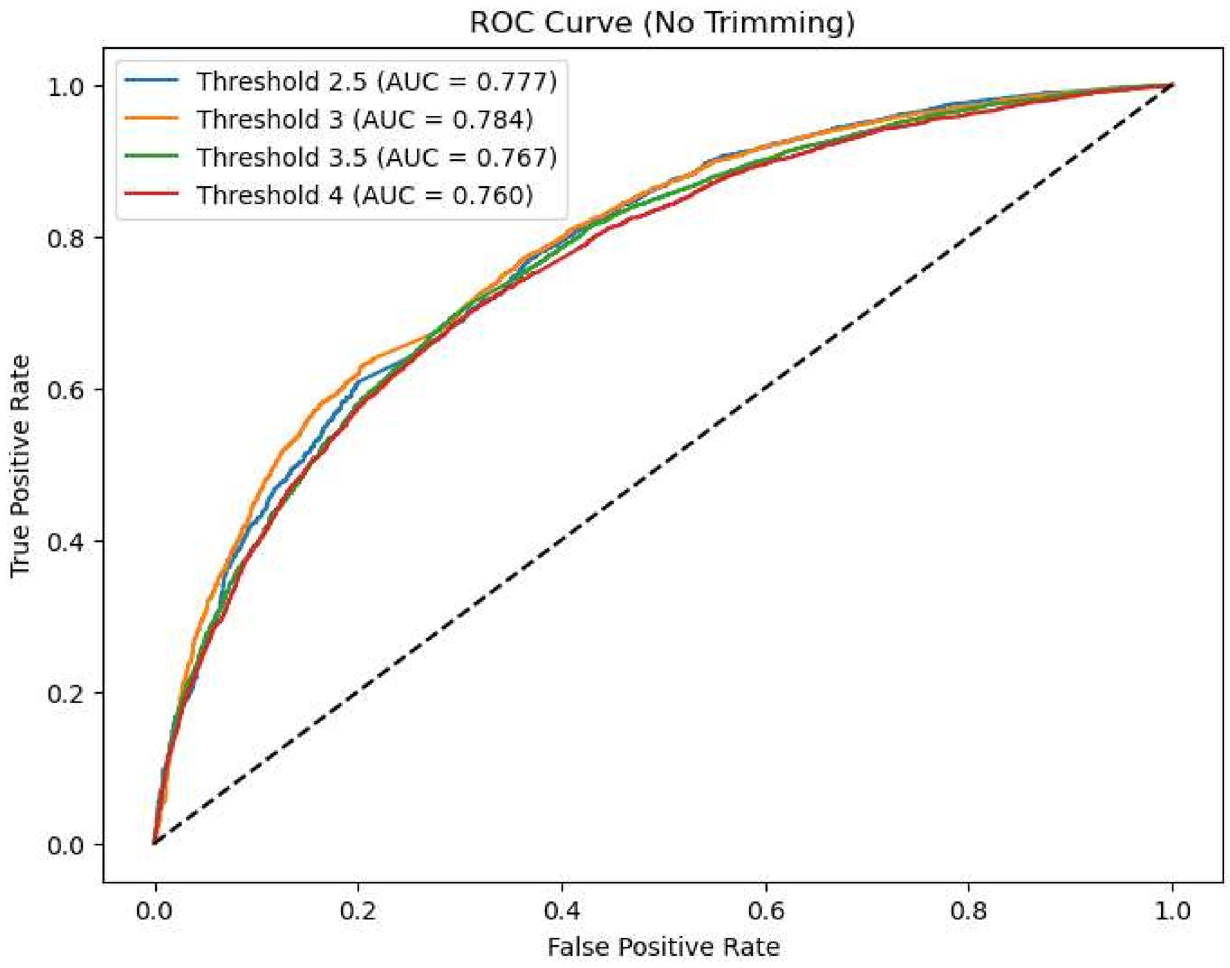

Figure 19. ROC Curve No Trimming

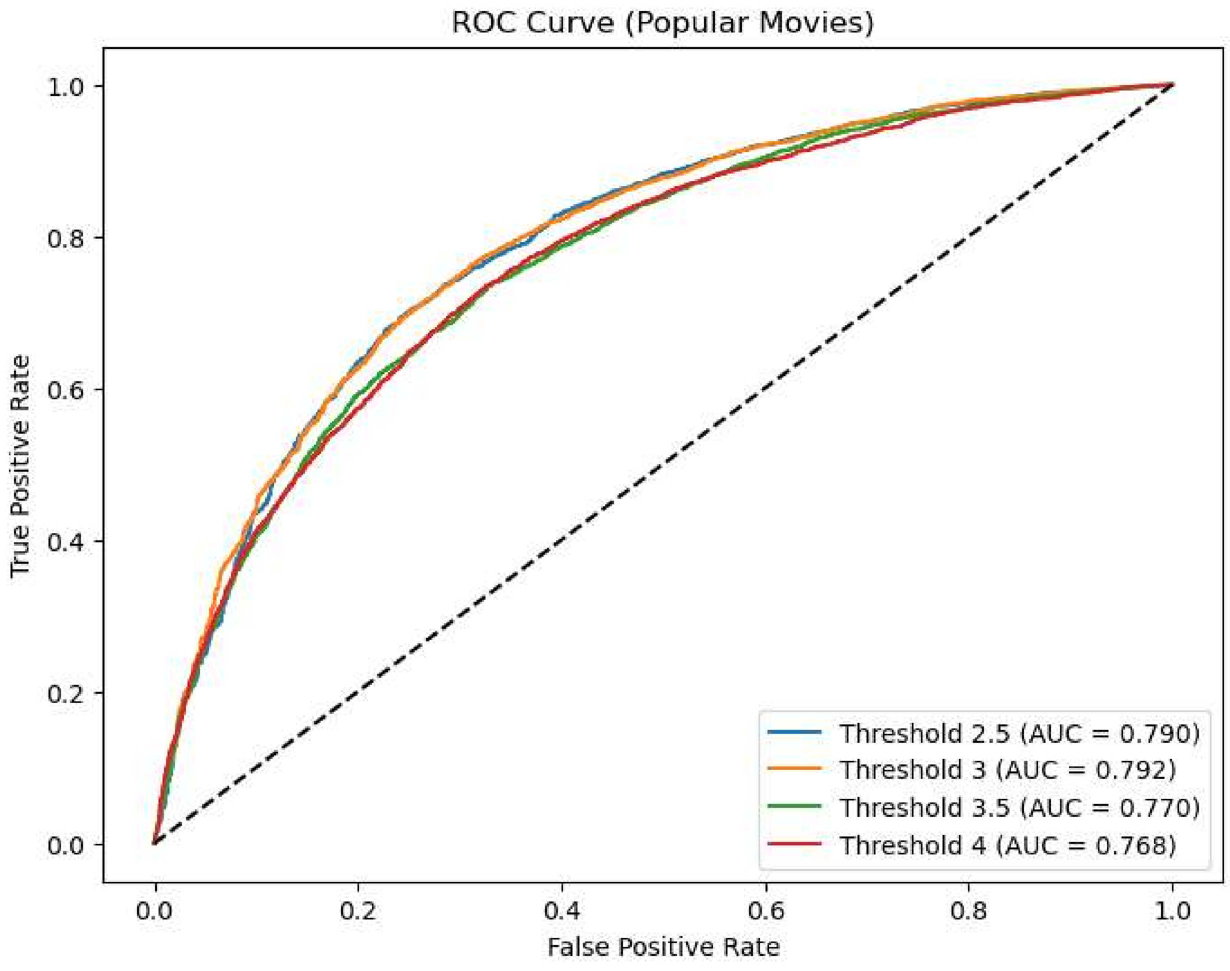

Figure 20. ROC Curve for Popular Trimming

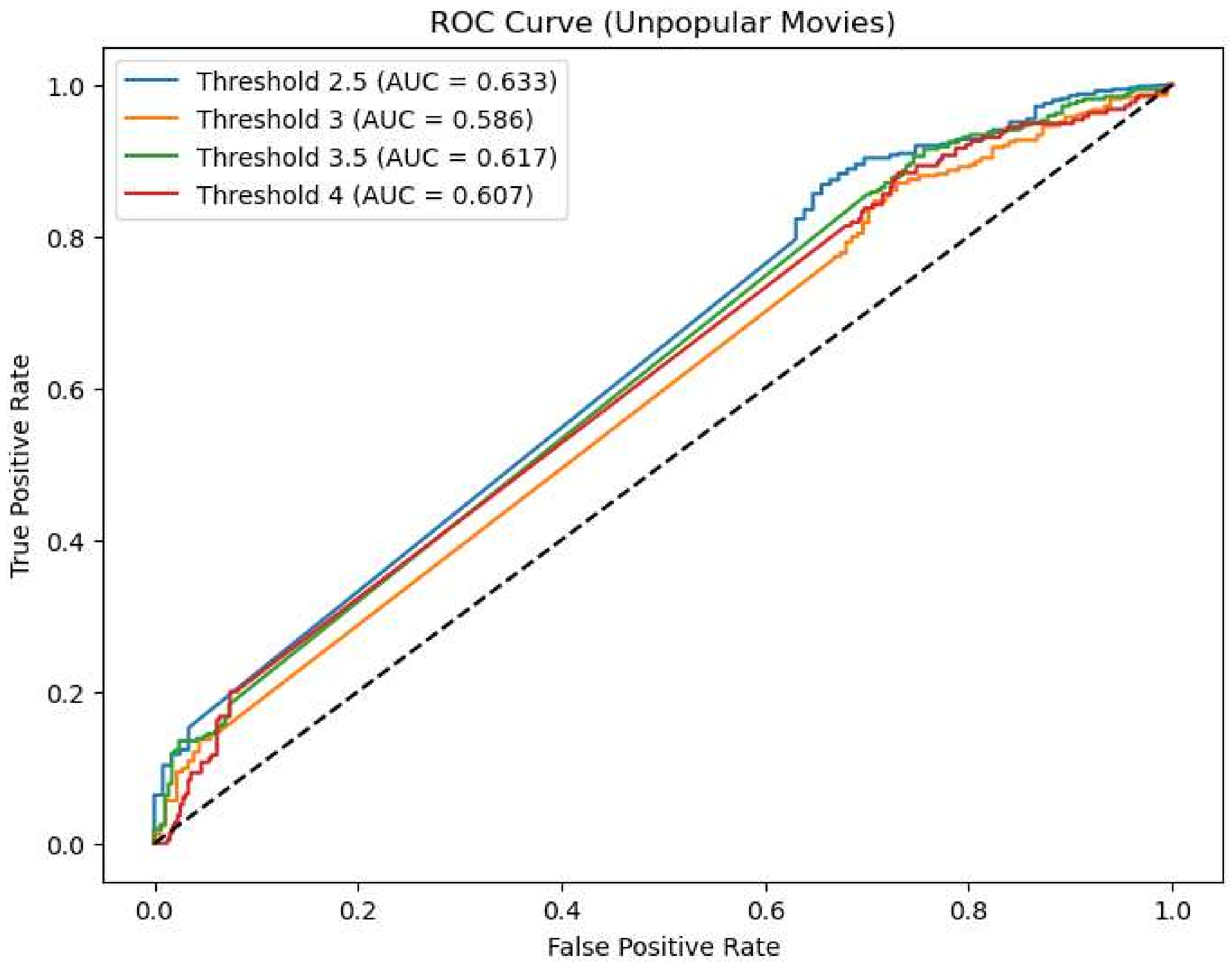

Figure 21. ROC Curve for Unpopular Trimming

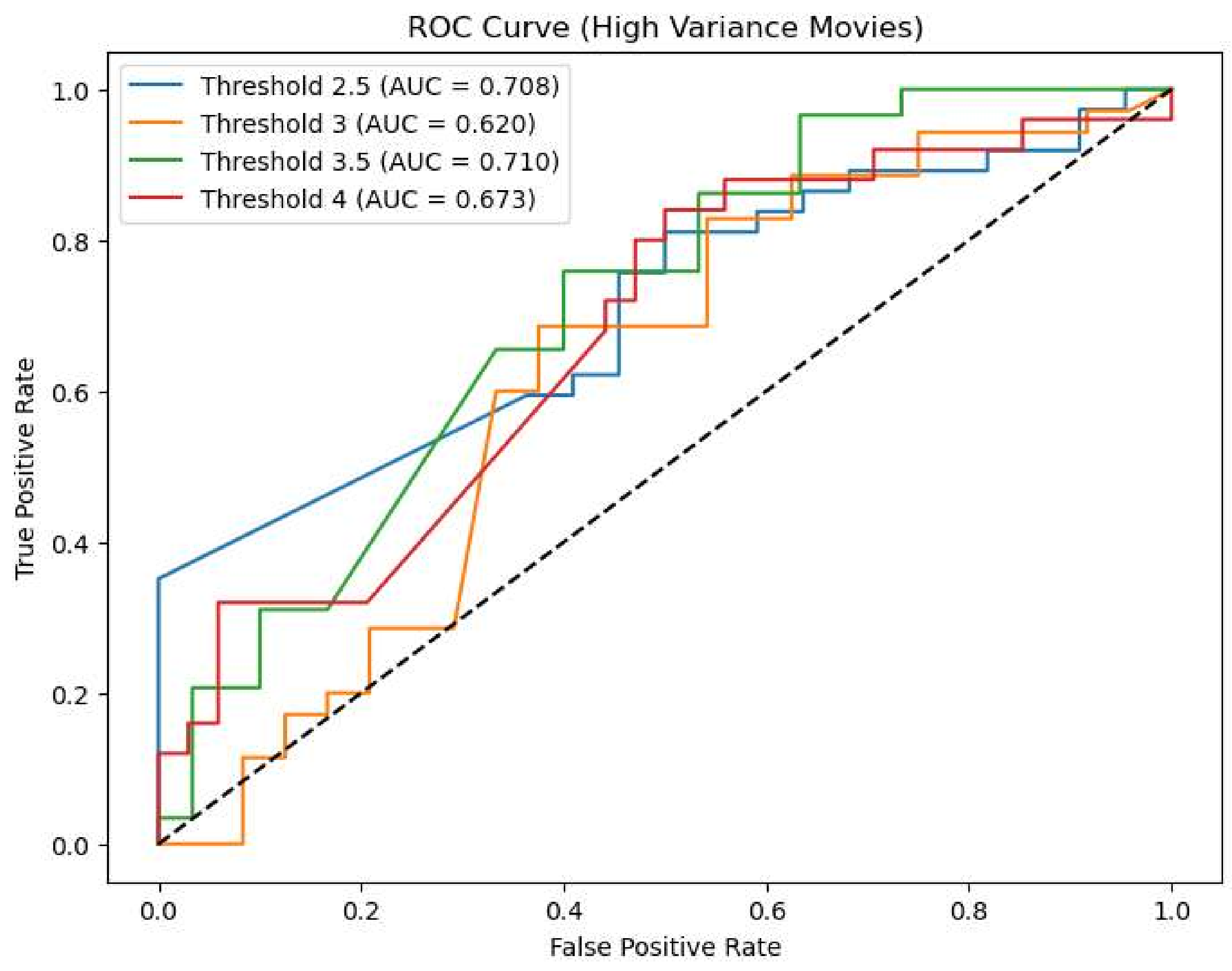

Figure 22. ROC Curve for High Variance Trimming

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 genre? Is there a connection between the latent factors and the movie genres?

NMF latent factors show a strong connection to movie genres, with some focusing on specific types of films while others capture a mix of related genres. Latent Factor 5 is mainly associated with Comedy, suggesting it groups movies that share a lighthearted or humorous tone. Latent

Factor 16 highlights Thriller, Drama, and Horror, indicating it captures films with intense storytelling and suspenseful elements. Latent Factor 17 is linked to Romance and Musical, showing a preference for emotionally driven narratives. This pattern suggests that NMF doesn't just group movies by genre labels but also identifies hidden connections based on themes, moods, or storytelling styles, helping improve personalized recommendations.

QUESTION 10: Designing the MF Collaborative Filter:

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

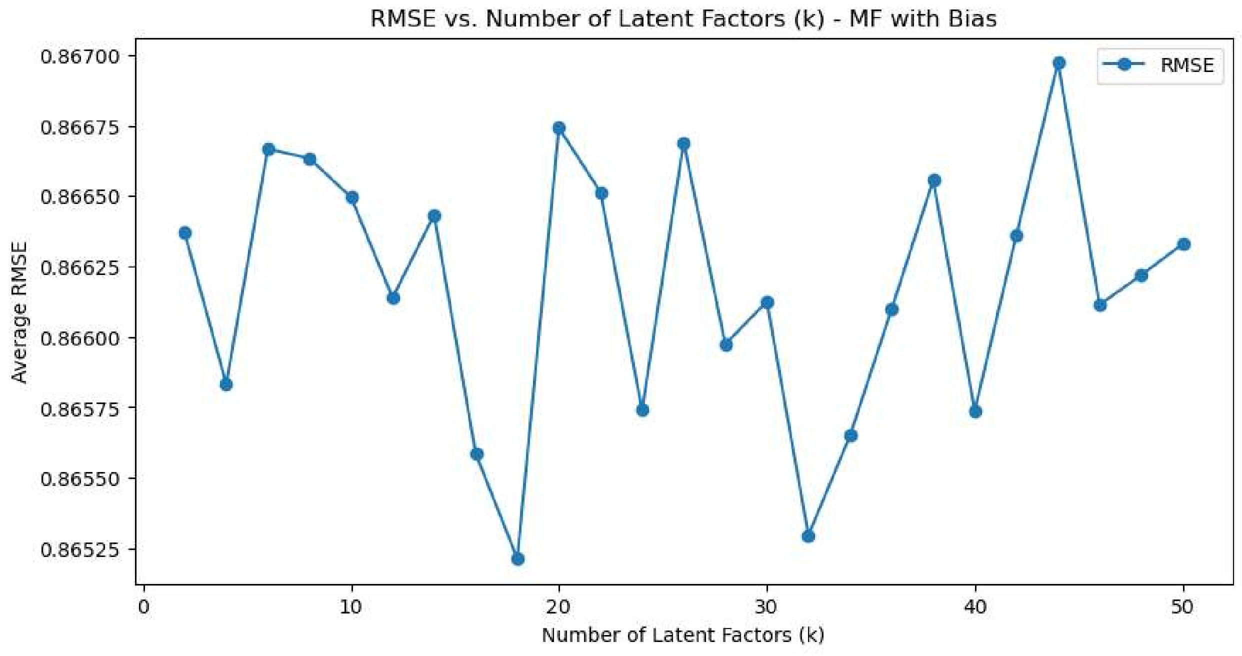

Figure 23. Average RMSE (Y-axis) against k (X-axis)

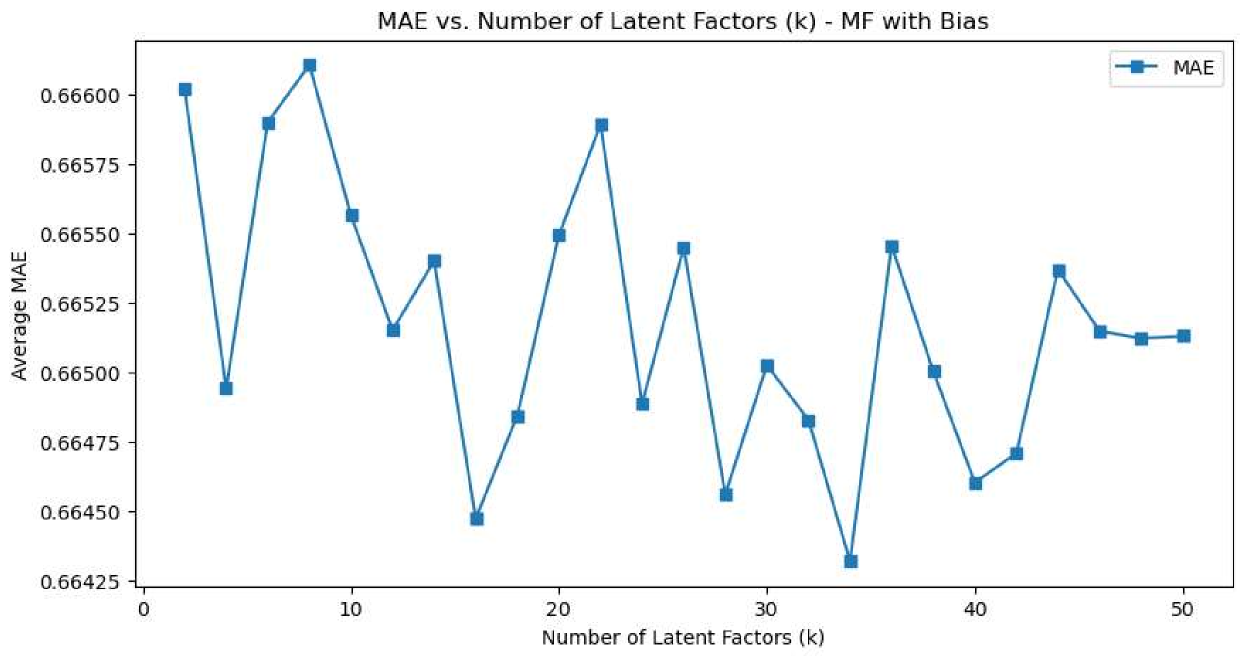

Figure 24. Average MAE (Y-axis) against k (X-axis)

B Use the plot from the previous part 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?

Optimal number of latent factors (k) based on RMSE: 18
Minimum average RMSE: 0.8652
Optimal number of latent factors (k) based on MAE: 34
Minimum average MAE: 0.6643
The optimal number of latent factors matches the number of movie genres.
The optimal number of latent factors does NOT match the number of movie genres.

C Performance on dataset subsets: For each of Popular, Unpopular and High-Variance subsets

-

– Design a MF collaborative filter for each trimmed subset 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); item Report the minimum average RMSE.

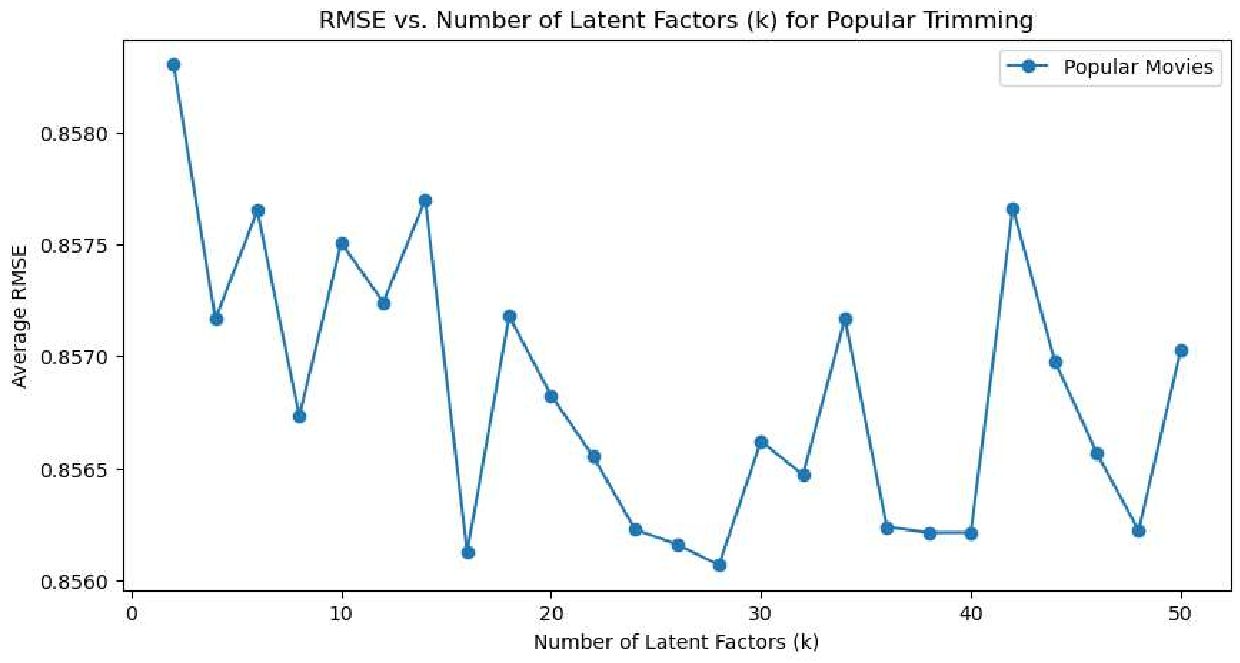

Figure 25. Average RMSE against k for Popular Trimming

Minimum RMSE for Popular Movies: 0.8561

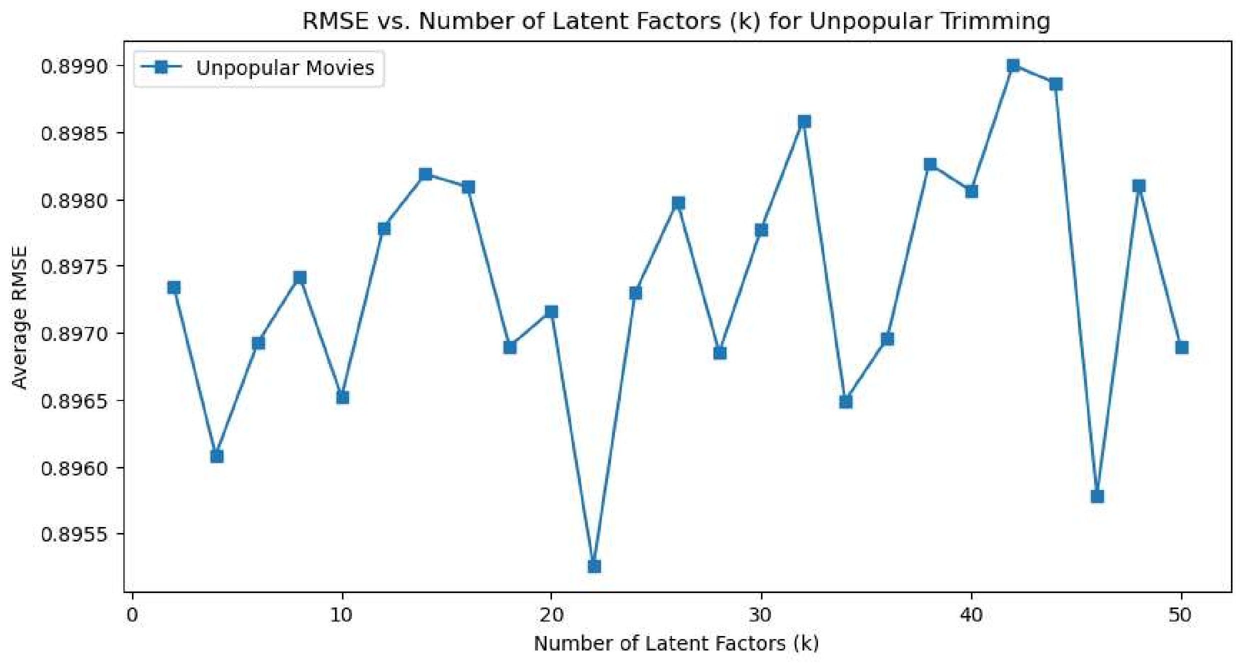

Figure 26. Average RMSE against k for Unpopular Trimming

Minimum RMSE for Unpopular Movies: 0.8953

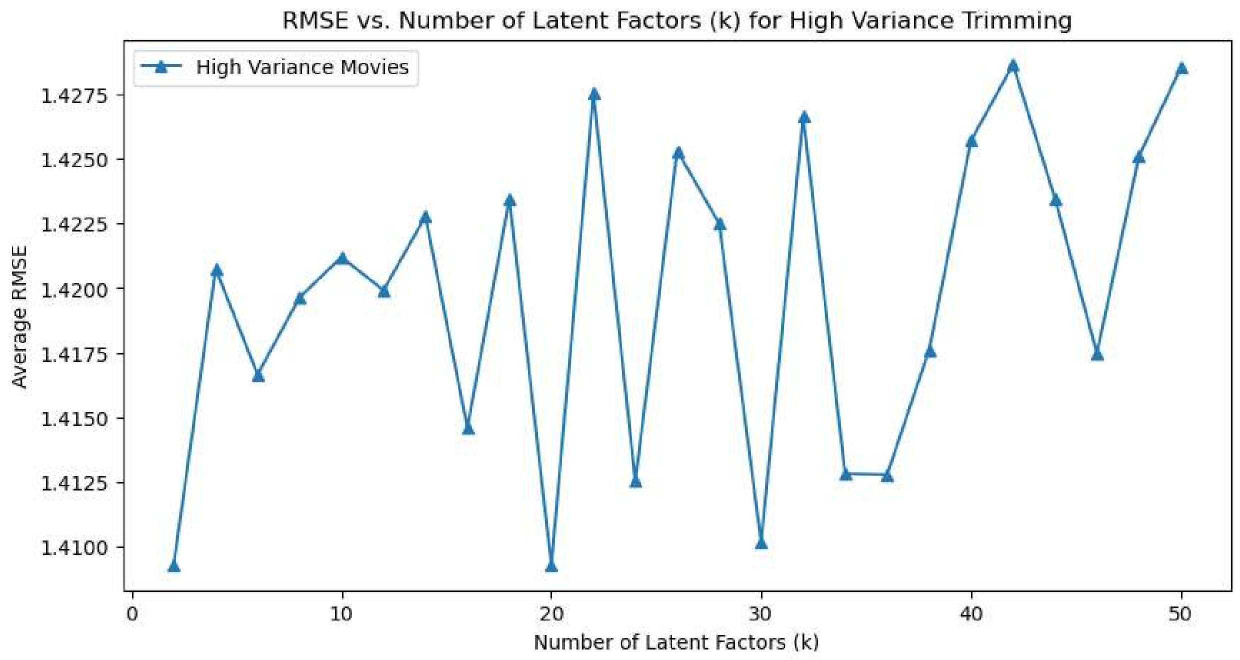

Figure 27. Average RMSE against k for High Variance Trimming

Minimum RMSE for High Variance Movies: 1.4093

• Plot the ROC curves for the MF-based collaborative filter and also report the area under the curve (AUC) value as done in Question 6.

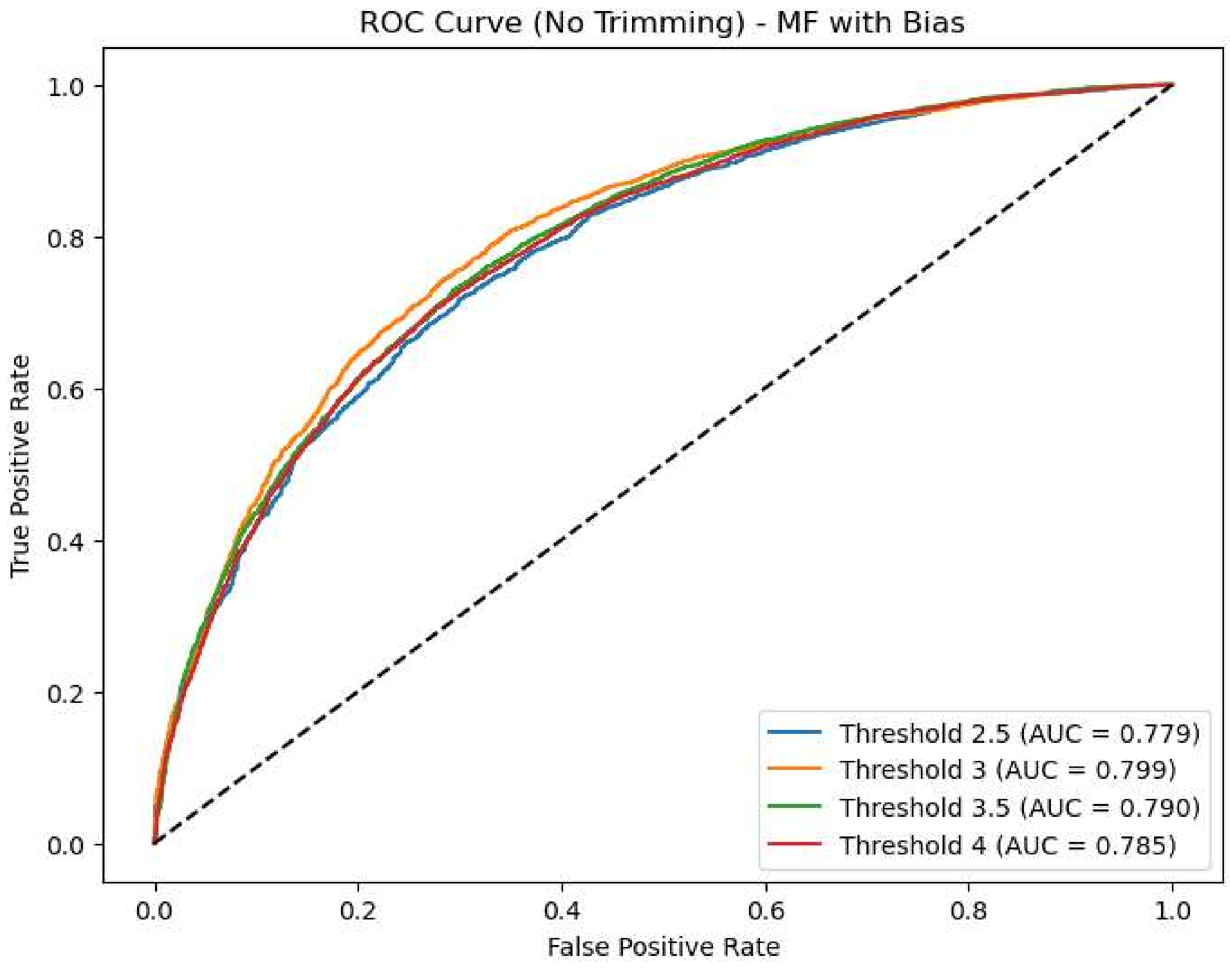

Figure 28. ROC Curve for No Trimming

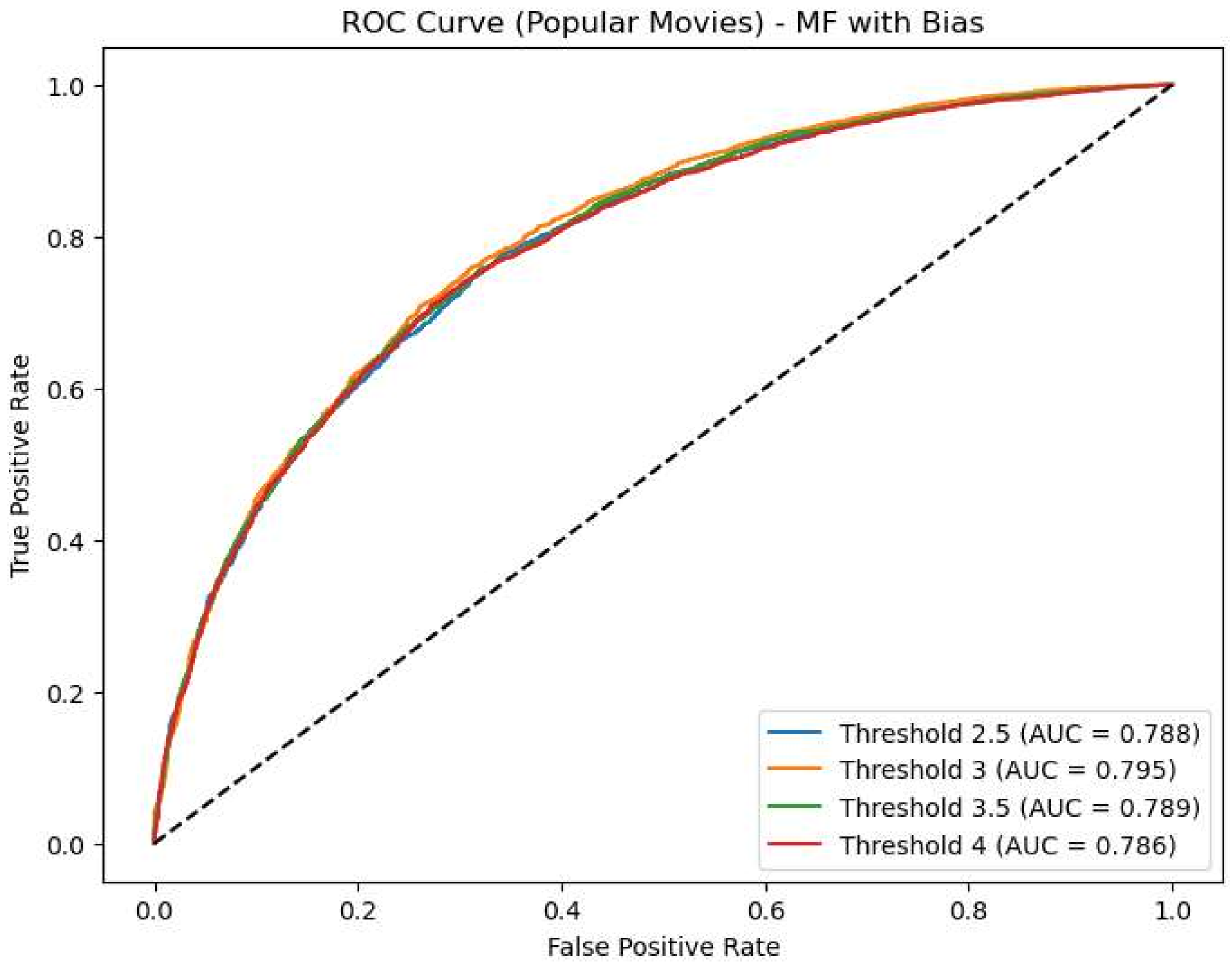

Figure 29. ROC Curve for Popular Trimming

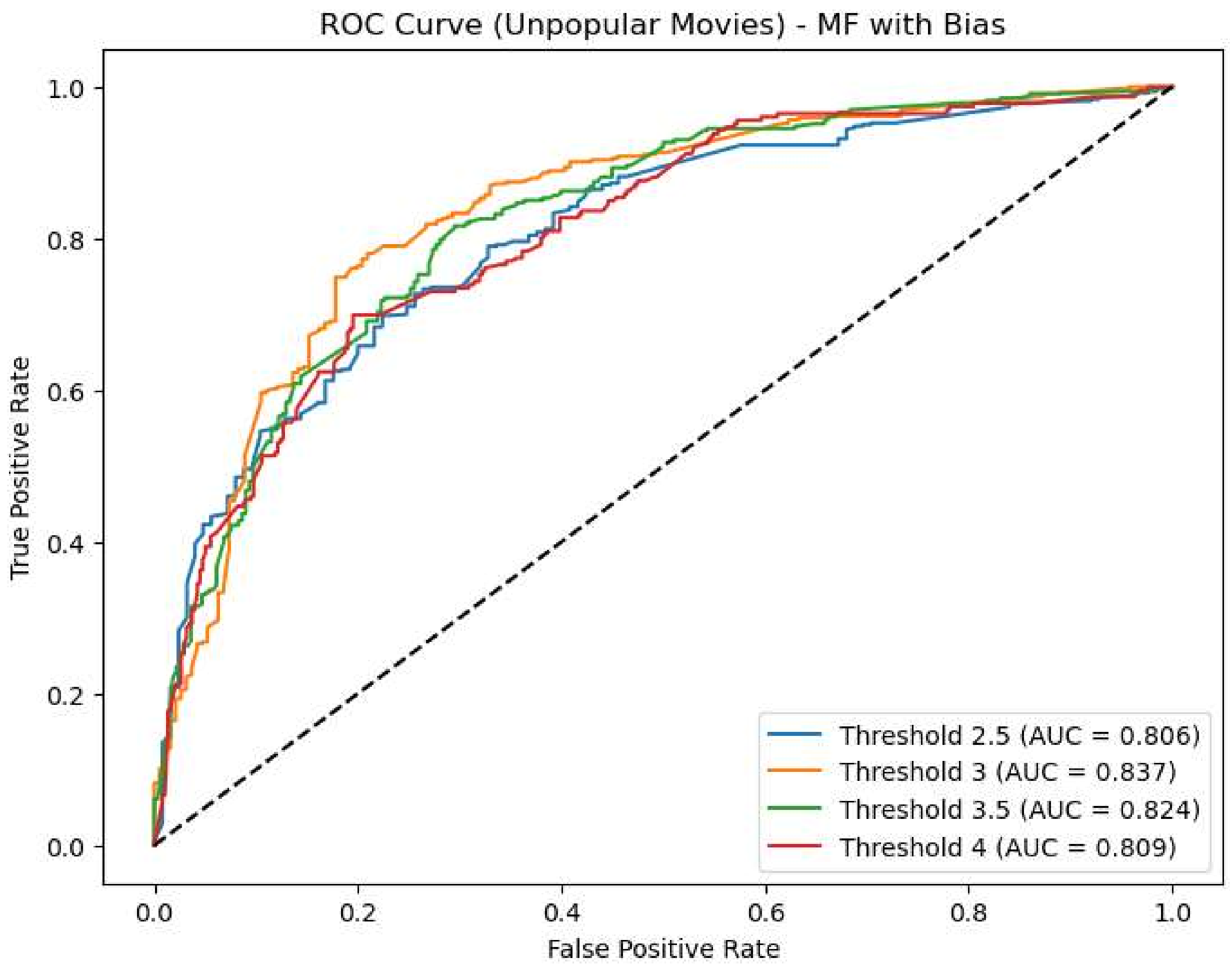

Figure 30. ROC Curve for Unpopular Trimming

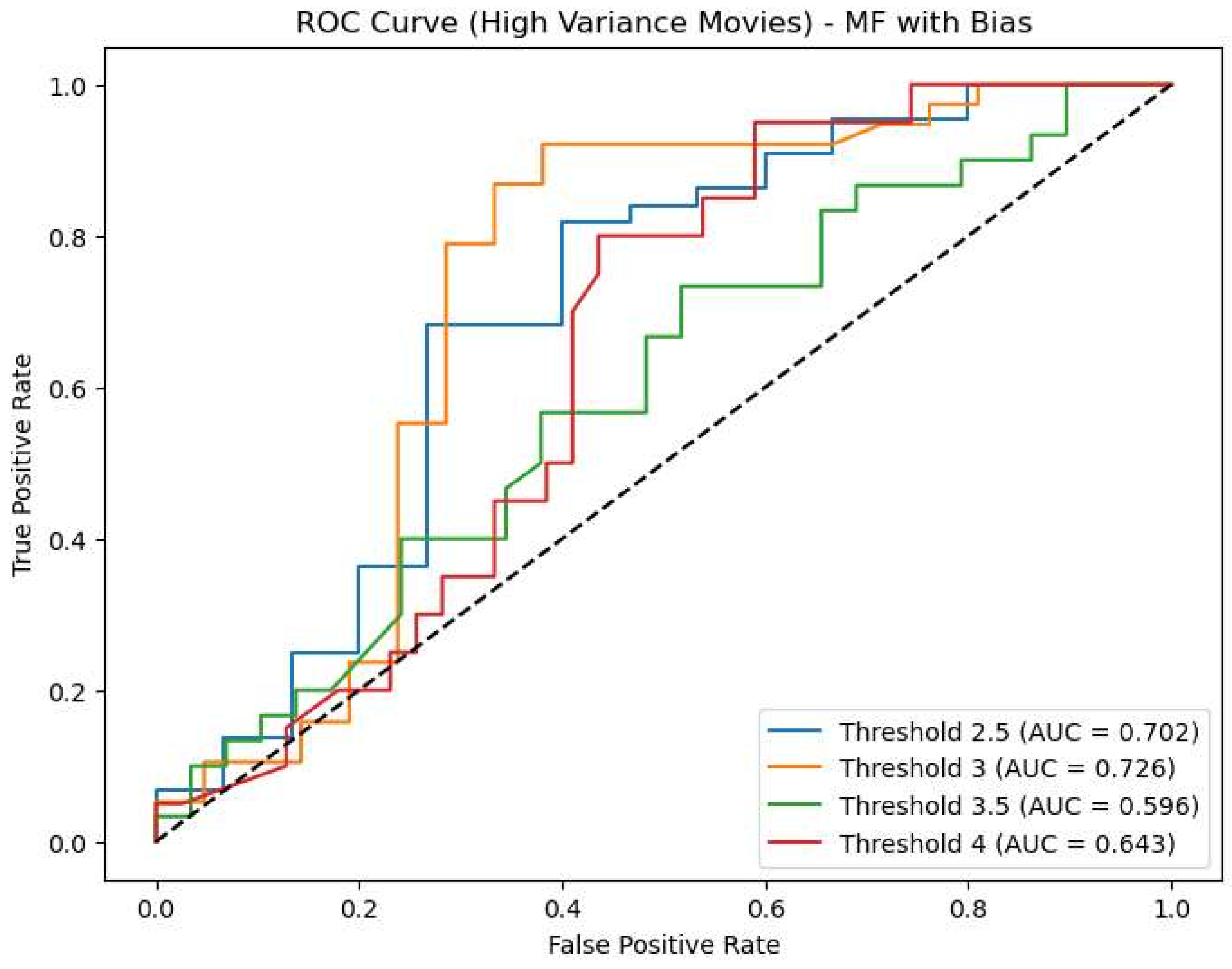

Figure 31. ROC Curve for High Variance Trimming

QUESTION 11: Designing a Naive Collaborative Filter:

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

• Performance on dataset subsets: 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.

Naïve Collaborative Filter for No Trimming: 1.1412
Naïve Collaborative Filter for Popular Movies: 1.1466
Naïve Collaborative Filter for Unpopular Movies: 1.3054
Naïve Collaborative Filter for High-Variance Movies: 1.5958

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.

The MF with Bias model performs the best, with an AUC of 0.807. This indicates that it makes the most accurate predictions. The k-NN model is slightly worse with an AUC of 0.782. This indicates that user similarities help in recommendations. The NMF model has the lowest AUC of 0.777. This indicates that it is less effective. So, in general, MF with Bias is the best model.

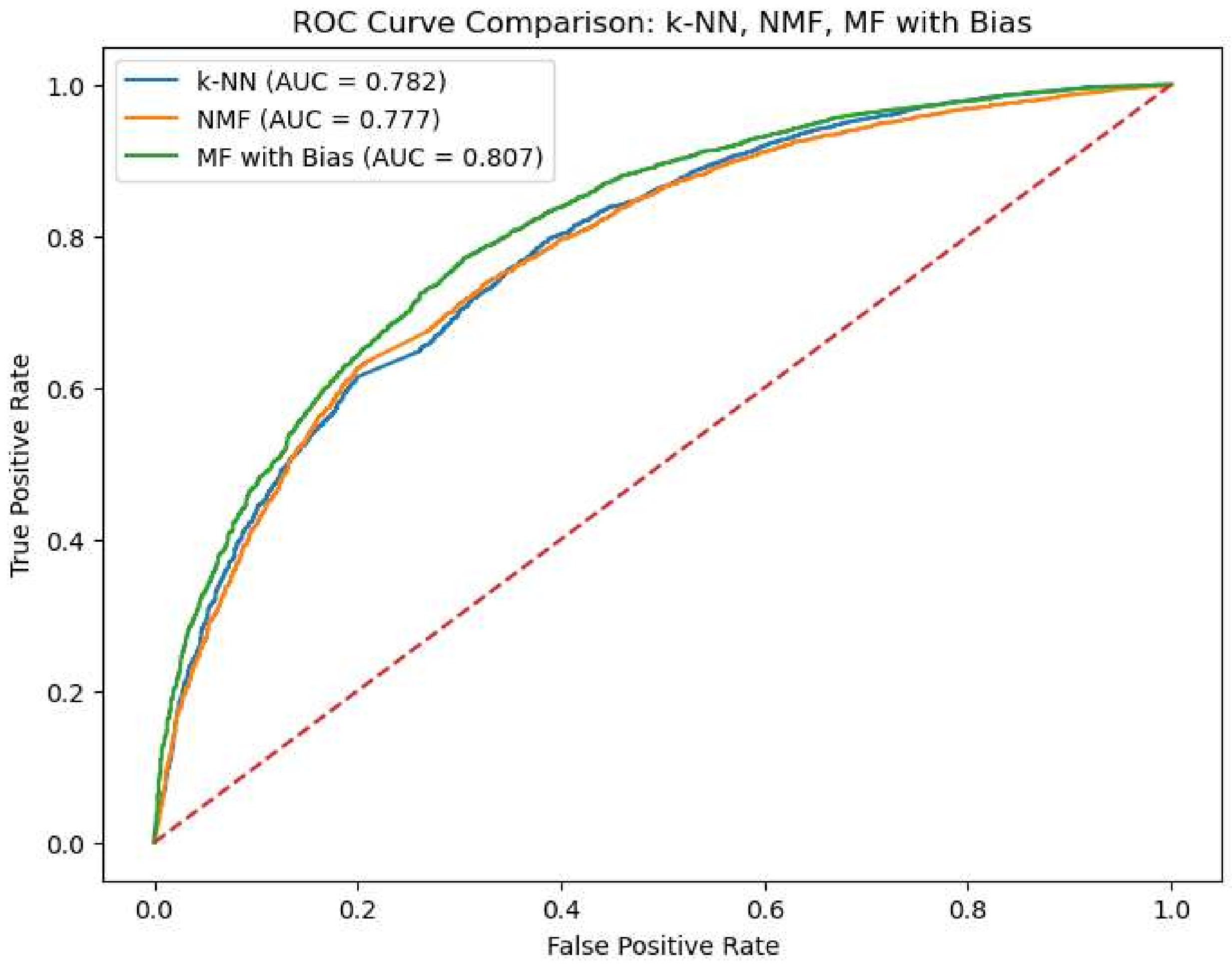

Figure 32. Model Comparison ROC Curve

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.

Number of unique queries in total: 10000

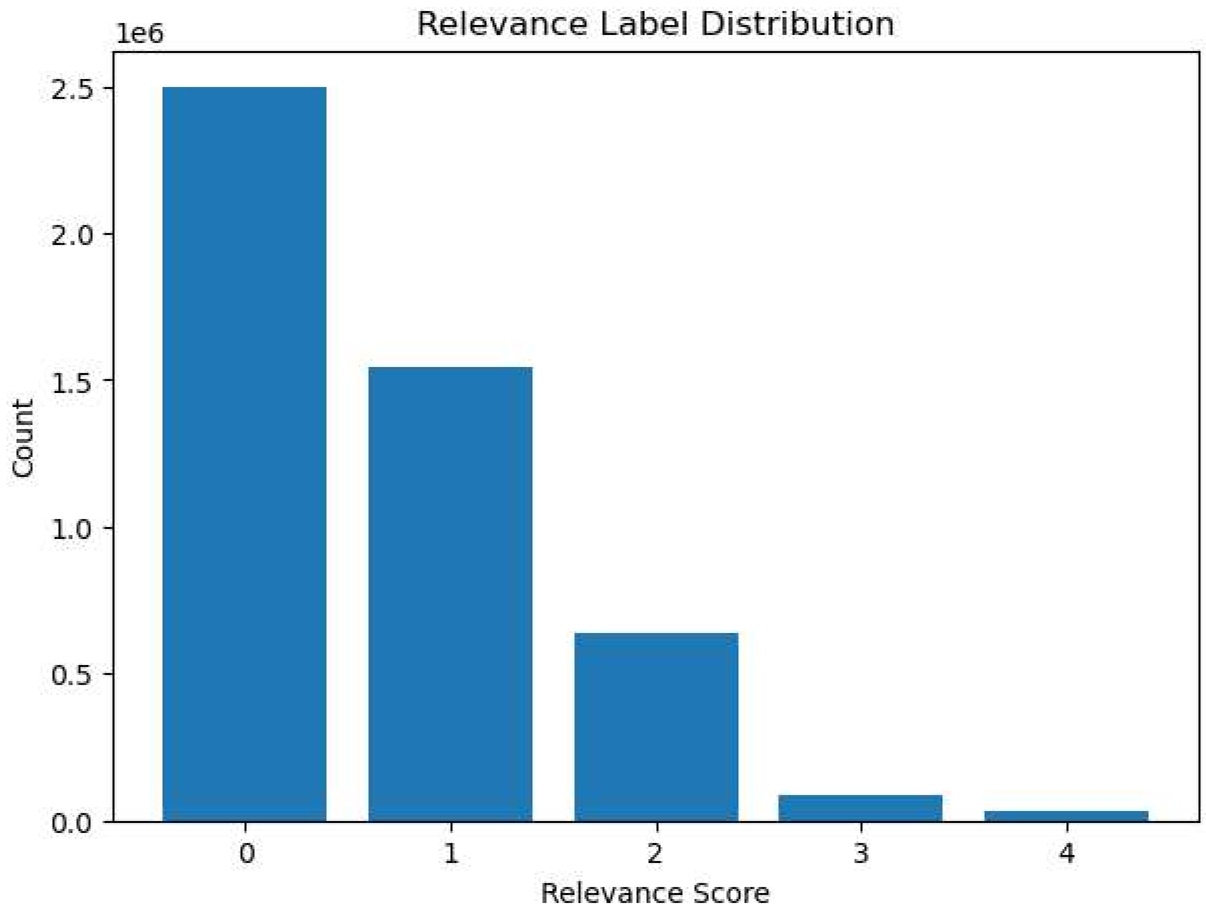

Figure 33. Distribution of Relevance Label

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.

Fold 1 - nDCG@3: 0.4565, nDCG@5: 0.4633, nDCG@10: 0.4829
Fold 2 - nDCG@3: 0.4539, nDCG@5: 0.4573, nDCG@10: 0.4768
Fold 3 - nDCG@3: 0.4491, nDCG@5: 0.4583, nDCG@10: 0.4759
Fold 4 - nDCG@3: 0.4612, nDCG@5: 0.4664, nDCG@10: 0.4877
Fold 5 - nDCG@3: 0.4696, nDCG@5: 0.4714, nDCG@10: 0.4904

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. Please use model.booster .feature importance(importance type='gain') as demonstrated here for retrieving importance score per feature. You can also find helper code in the provided notebook.

```
Top 5 Features for Fold 1:
    Feature   Importance
Feature_133 23856.702951
  Feature_7  4248.546391
Feature_107  4135.244450
 Feature_54  4078.463226
Feature_129  3635.037024

Top 5 Features for Fold 2:
    Feature   Importance
Feature_133 23578.908250
  Feature_7  5157.964912
 Feature_54  4386.669757
Feature_107  4094.012172
Feature_129  4035.070673

Top 5 Features for Fold 3:
    Feature   Importance
Feature_133 23218.075441
 Feature_54  4991.303372
Feature_107  4226.807395
Feature_129  4059.752514
  Feature_7  3691.792320

Top 5 Features for Fold 4:
    Feature   Importance
Feature_133 23796.899673
  Feature_7  4622.622978
 Feature_54  3883.481706
Feature_129  3356.846980
```

```
Feature_128  3207.575537

Top 5 Features for Fold 5:
    Feature   Importance
Feature_133 23540.942354
  Feature_7  4794.945172
 Feature_54  4079.608554
Feature_107  3514.835752
Feature_129  3209.058444
```

QUESTION 16: Experiments with Subset of Features:
For each of the five provided folds:

• Remove the top 20 most important features according to the computed importance score in the question 15. Then train a new LightGBM model on the resulted 116 dimensional query-url data. Evaluate the performance of this new model on the test set using nDCG. Does the outcome align with your expectations? If not, please share your hypothesis regarding the potential reasons for this discrepancy.

The outcome aligns with my expectations. After removing the most important features, the score went worse. This is due to those removed features will help the model predict more accurately.

```
Removed Top 20 Features
Fold 1 - nDCG@3: 0.3797, nDCG@5: 0.3850, nDCG@10: 0.4084
Fold 2 - nDCG@3: 0.3739, nDCG@5: 0.3820, nDCG@10: 0.4045
Fold 3 - nDCG@3: 0.3824, nDCG@5: 0.3900, nDCG@10: 0.4116
Fold 4 - nDCG@3: 0.3820, nDCG@5: 0.3928, nDCG@10: 0.4121
Fold 5 - nDCG@3: 0.3843, nDCG@5: 0.3922, nDCG@10: 0.4167
```

• Remove the 60 least important features according to the computed importance score in the question 15. Then train a new LightGBM model on the resulted 76 dimensional query-url data. Evaluate the performance of this new model on the test set using nDCG. Does the

outcome align with your expectations? If not, please share your hypothesis regarding the potential reasons for this discrepancy.

The outcome aligns with my expectations. After removing the least important features, the score did not change too much. This is due to those removed features will not make a huge impact to the model's prediction, since they are likely noisy.

Removed Least 60 Features

Fold 1 - nDCG@3: 0.4543, nDCG@5: 0.4627, nDCG@10: 0.4820

Fold 2 - nDCG@3: 0.4573, nDCG@5: 0.4603, nDCG@10: 0.4773

Fold 3 - nDCG@3: 0.4498, nDCG@5: 0.4586, nDCG@10: 0.4774

Fold 4 - nDCG@3: 0.4606, nDCG@5: 0.4673, nDCG@10: 0.4889

Fold 5 - nDCG@3: 0.4702, nDCG@5: 0.4734, nDCG@10: 0.4908