UCLA

ECE 219 - Large-Scale Data Mining: Models and Algorithms

Winter 2024

Project 3: Recommender Systems

Sunday, February 25, 2024

Author:

Gian Zignago (UID: 706294998)

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

A. **Compute the sparsity of the movie rating dataset:**

$$Sparsity = \frac{Total\ number\ of\ available\ ratings}{Total\ number\ of\ possible\ ratings}$$

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.

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 be broken in any way. A monotonically decreasing trend is expected.

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.

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

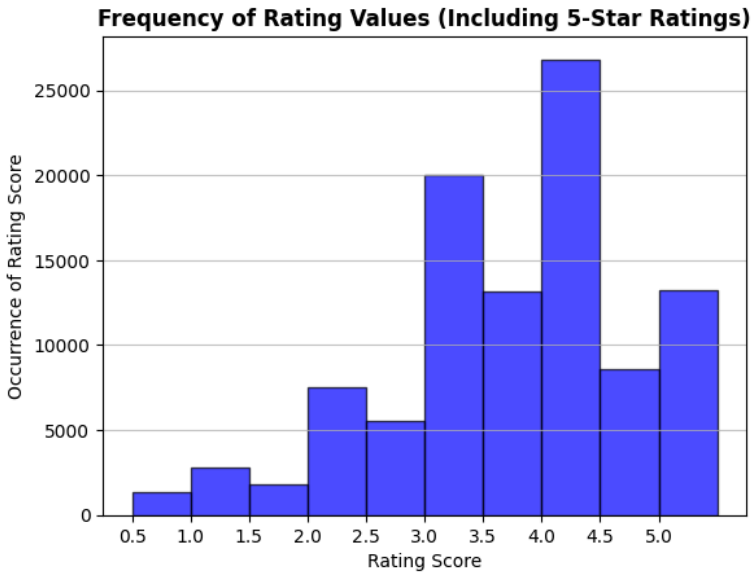
F. **Compute the variance of the rating values received by each movie:** Bin the variance values into intervals of width 0.5 and use the binned variance values as the horizontal axis. Count the number of movies with variance values in the binned intervals and use this count as the vertical axis. Briefly comment on the shape of the resulting histogram.

A. Compute the sparsity of the movie rating dataset

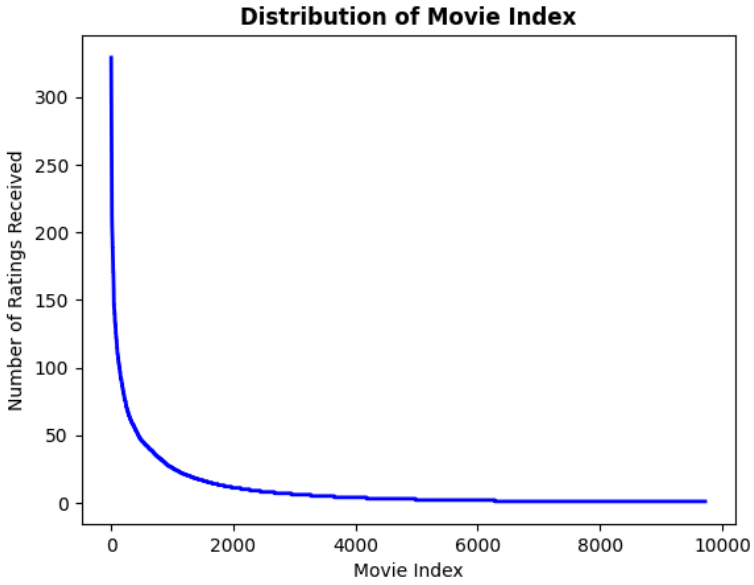
Sparsity of MovieLens Dataset: **0.016999683055613623**.

This result suggests that the provided ratings matrix is sparse. This is expected, as it is unlikely that each user rated all the movies in the dataset.

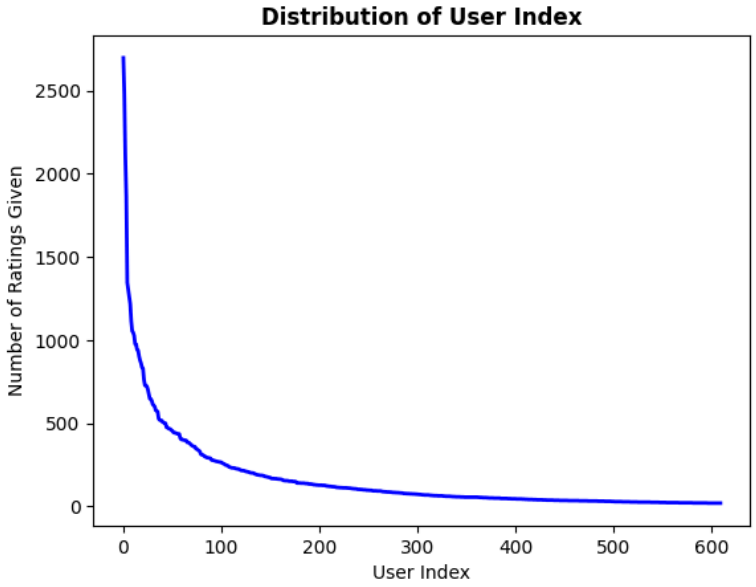
B. Plot a histogram showing the frequency of the rating values



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



D. Plot the distribution of ratings among users

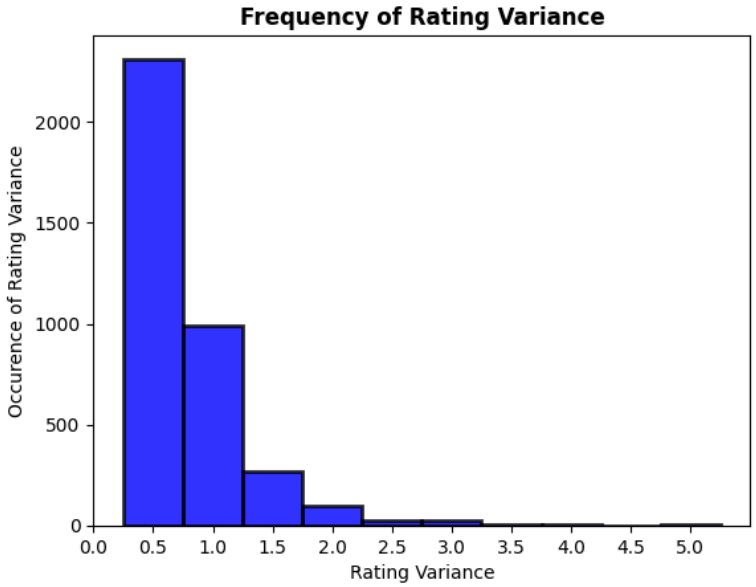


E. Discuss the salient features of the distributions

Regarding the *Distribution of Movie Index* plot in Question C, the plot typically exhibits a long tail distribution where a small number of movies receive a large number of ratings, signifying their high popularity. Most movies, in contrast, garner far fewer ratings, illustrating a skew towards specific blockbuster films that dominate user attention. In the context of a recommendation system, this presents a risk of popularity bias. This bias occurs when the system disproportionately recommends popular movies, thus neglecting films with fewer overall ratings that could appeal more strongly to specific users. Additionally, this presents the cold start problem where new movies struggle to gain notice and ratings in a system that favors films with more existing ratings.

The *Distribution of User Index* plot in Question D reveals a similar trend. A small subset of users rate a large number of movies, while the majority rate far fewer, indicating varied engagement levels among the user base. In the context of a recommendation system, users who rate more movies can disproportionately influence the system's perception of what constitutes popular or high-quality films. Also, personalization becomes more challenging for users who have rated only a few movies, as their limited data makes it harder to generate accurate, personalized recommendations. The sparsity of ratings can create difficulties in providing accurate recommendations, especially for users and movies in the long tail of the distribution.

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



QUESTION 2. Understanding the Pearson Correlation Coefficient:

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

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

A.

the formula for μ_u in terms of I_u and r_{uk} is as follows:

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

B.

$I_u \cap I_v$ means a set of movie indices where ratings have been specified by both user u and user v. $I_u \cap I_v$ can be equal to the empty set $\emptyset$, as it is possible that movies can be rated by neither user u nor user v.

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?

The process of mean-centering the raw ratings in the prediction function is meant to address the variations in rating habits among different users.

Let's consider two users, user *u* and user *v*, who share similar preferences for movies. Despite their similar tastes, their rating styles might differ significantly; for example, user *u* might generally rate movies more strictly than user *v*. In the absence of mean-centering, this discrepancy in rating behavior could lead to inaccurate predictions by the recommendation system.

By implementing the mean-centering approach as outlined in equation (3) from section 5.4, the ratings from different users are normalized, effectively mitigating the impact of individual rating tendencies. This normalization ensures that the recommendation system's predictions are more balanced and free from biases introduced by varied rating habits.

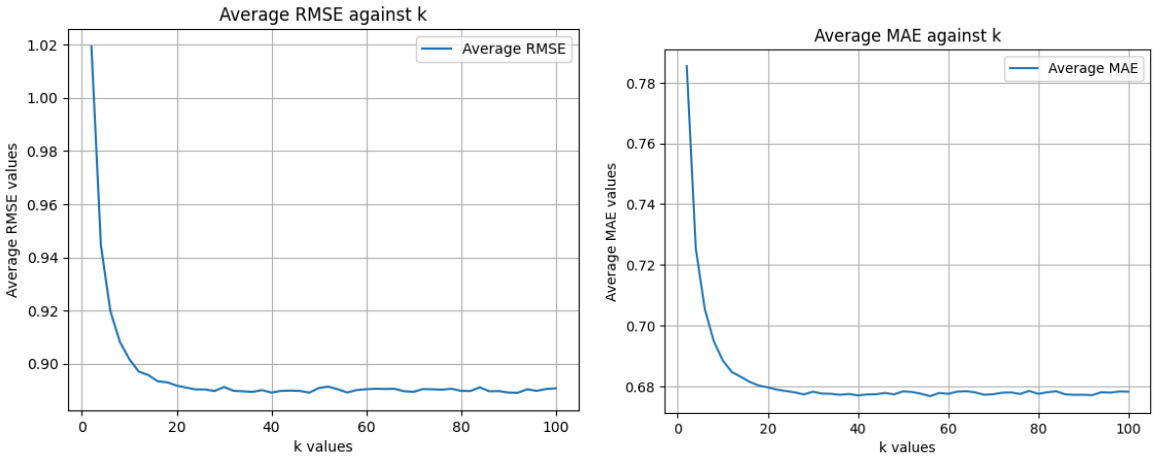
QUESTION 4. Design a k-NN collaborative filter to predict the ratings of the movies in the original dataset and evaluate its performance using 10-fold cross validation. Sweep k (number of neighbors) from 2 to 100 in step sizes of 2, and for each k compute the average RMSE and average MAE obtained by 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).

For each k the average RMSE and average MAE across all 10 folds:

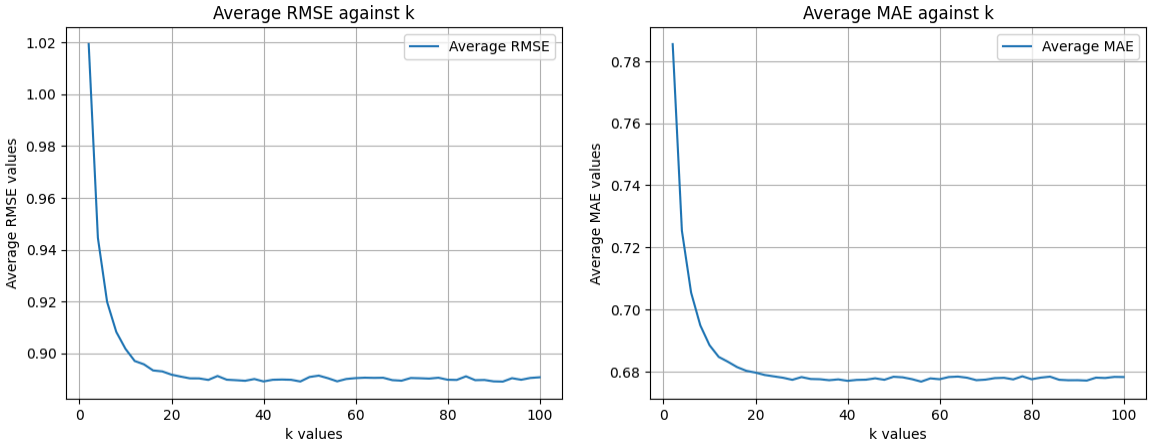
```
Average RMSE:

 [1.0191455440376331, 0.945921427174504, 0.9200198678486521,
0.9078205046261253, 0.9014900500394459, 0.8970543898490098,
0.8959340204679627, 0.8943389523765077, 0.8926287603970435,
0.8917062822567987, 0.8918878892489396, 0.8908229663708072,
0.8899187077141703, 0.8901955038511595, 0.8896306596306761,
0.8907643822478327, 0.890198780020332, 0.8902323600356989,
0.8888590131479039, 0.88949135576914, 0.8893265323143831,
0.890053565805619, 0.8899243778450001, 0.8901732333944267,
0.8894449820496504, 0.8889403426913344, 0.8903305453652275,
0.8894363100464714, 0.8897011885462336, 0.8900661076823739,
0.8890582443511164, 0.8905246060496769, 0.8900323966568859,
0.8898534132443092, 0.8904160526789253, 0.8894034396150037,
0.8906396793685879, 0.8907688114963536, 0.8906117065944217,
0.8910187474520441, 0.8903897779173484, 0.88928774896683,
```

```
0.8902230276951115, 0.8920159343495324, 0.8898854282199775,
0.8898380901081401, 0.8904967953615401, 0.8898809741879592,
0.891293906641175, 0.8918810164016401]

Average MAE:

 [0.7861180873938372, 0.7268222745467301, 0.7048579670006965,
0.6939230506910992, 0.6886863220839492, 0.684610741733521,
0.6833065192179704, 0.6816609898074052, 0.6802149302959786,
0.6795092041108901, 0.6793072085382531, 0.6789892349190805,
0.678175393779006, 0.6781282359124148, 0.677069163815757,
0.6779688601791252, 0.6777396226622279, 0.6776948875175592,
0.6766497806244252, 0.6775265220537281, 0.677327967563387,
0.6776455058563956, 0.6774850439261378, 0.6775855341517769,
0.6768122764116249, 0.6769747739790536, 0.677785762859619,
0.6774766721665569, 0.6774613038339303, 0.677648990324714,
0.6772307017371759, 0.6780735796005095, 0.677681614804601,
0.6776860708664418, 0.6777881844875011, 0.6773999673149995,
0.6783144987606494, 0.6780324045569094, 0.6784853249852248,
0.6787368193855714, 0.6782392707112352, 0.677197685274109,
0.6781368173331483, 0.67891025305105, 0.67790948564462,
0.6780217666518633, 0.6777447636767541, 0.67780988518234,
0.6788920161632296, 0.6786549564040425]
```

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'. 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 converge to a steady-state value. Please report the steady state values of average RMSE and average MAE.

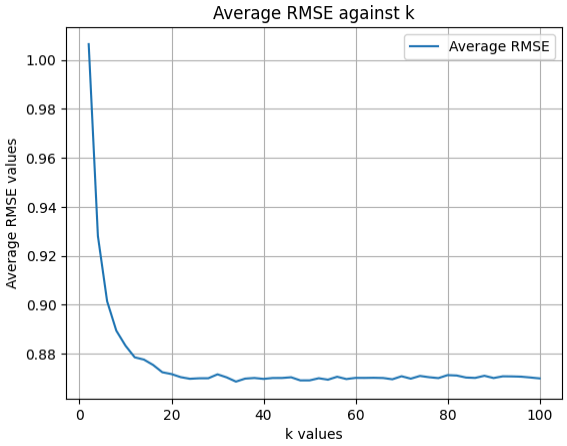


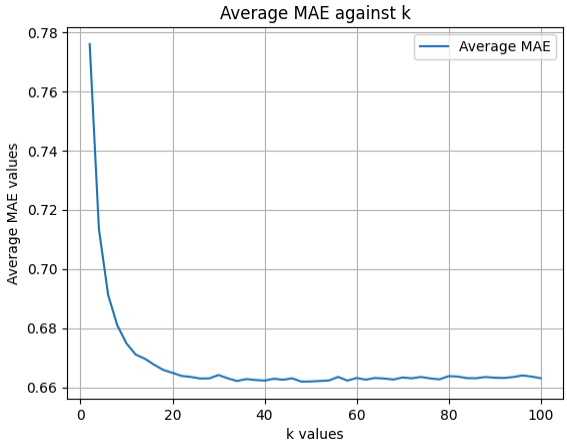
From the plot, min k = 24; steady-state RMSE is 0.890; steady-state MAE is 0.678.

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

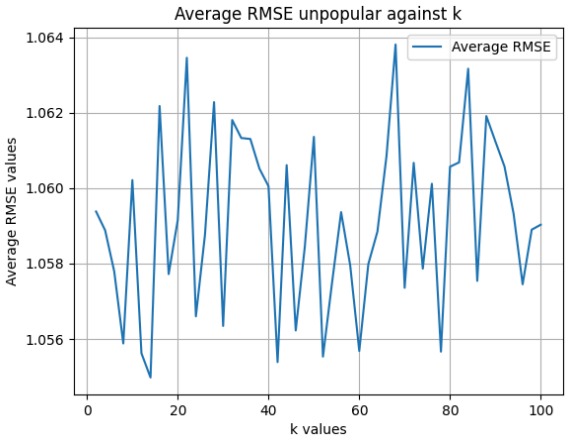
A k-NN collaborative filter on the ratings of the movies (i.e Popular, Unpopular or High-Variance) and evaluate each of the three models' performance using 10-fold cross validation:

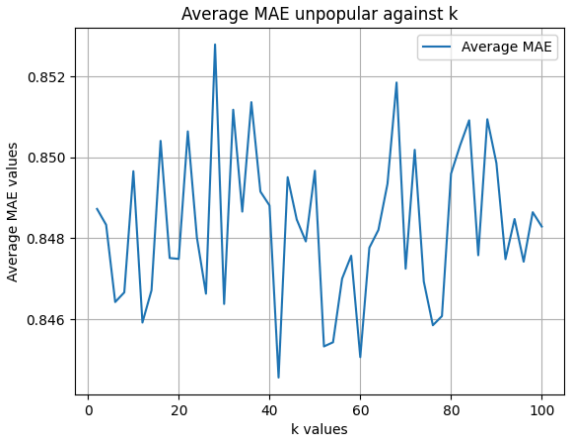
- Sweep k (number of neighbors) from 2 to 100 in step sizes of 2, and for each k compute the average RMSE obtained by averaging the RMSE across all 10 folds. Plot average RMSE (Y-axis) against k (X-axis). Also, report the minimum average RMSE.
- Plot the ROC curves for the k-NN collaborative filters 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.



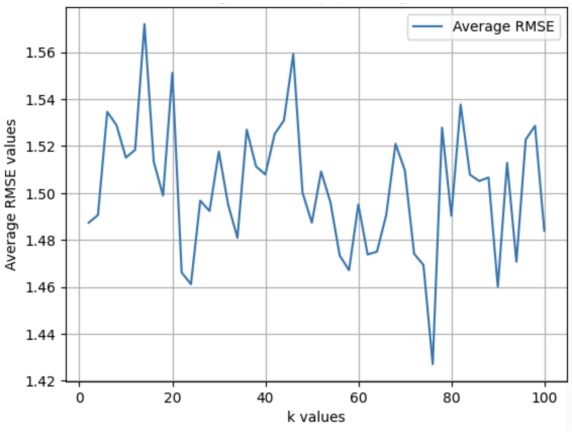


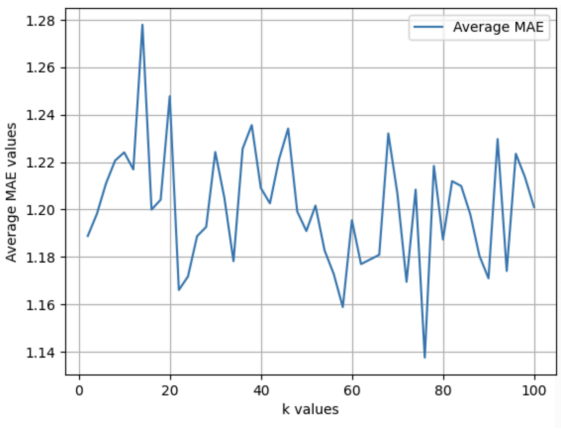
As above, the average RMSE (popular) is 0.876; the average MAE (popular) is 0.668.





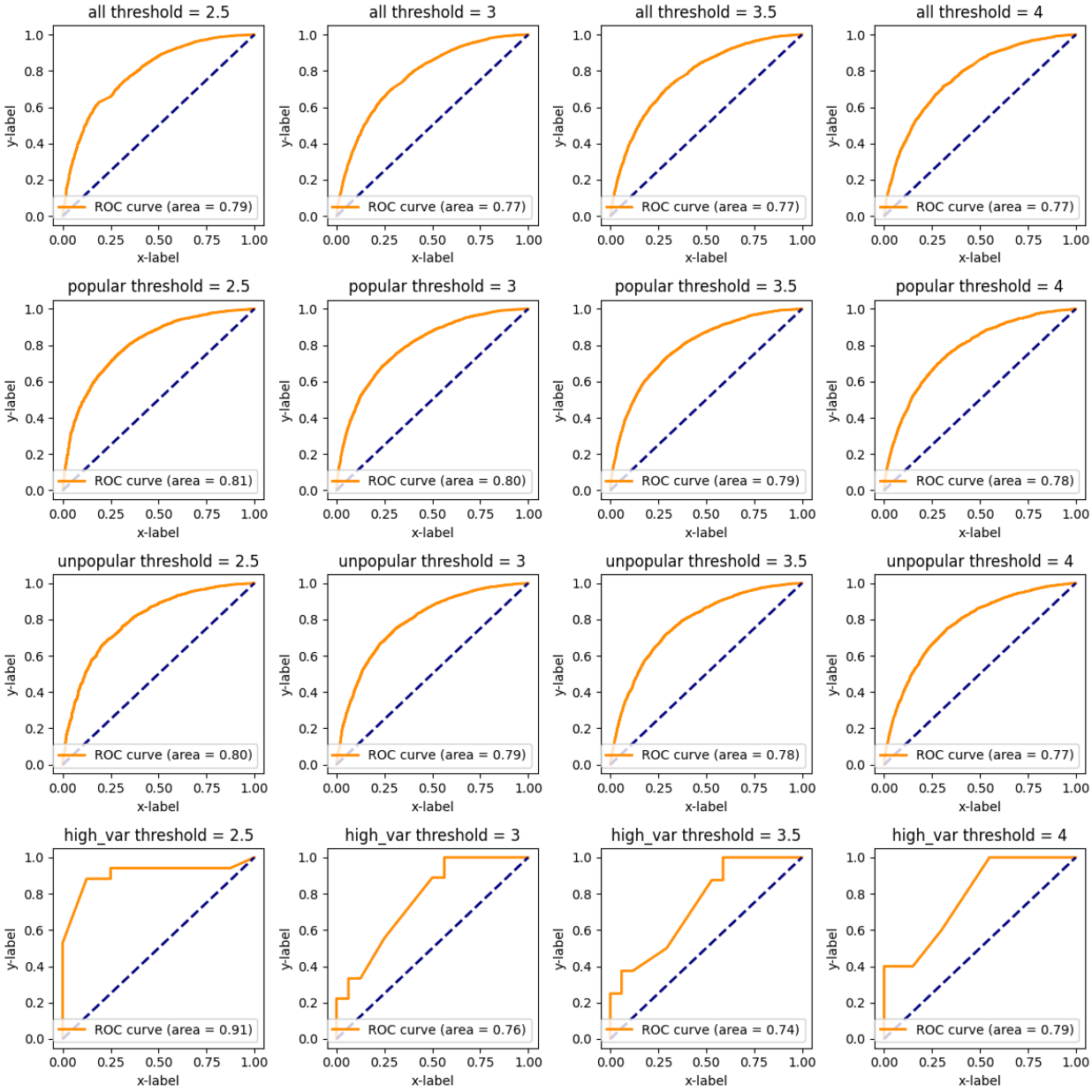
As above, the average RMSE (unpopular) is 1.059, the average MAE (unpopular) is 0.848.





As above, the average RMSE (high-var) is 1.502, the average MAE (high-var) is 1.201.

Plots:


all threshold = 2.5
ROC curve (area = 0.79)
all threshold = 3
ROC curve (area = 0.77)
all threshold = 3.5
ROC curve (area = 0.77)
all threshold = 4
ROC curve (area = 0.77)
popular threshold = 2.5
ROC curve (area = 0.81)
popular threshold = 3
ROC curve (area = 0.80)
popular threshold = 3.5
ROC curve (area = 0.79)
popular threshold = 4
ROC curve (area = 0.78)
unpopular threshold = 2.5
ROC curve (area = 0.80)
unpopular threshold = 3
ROC curve (area = 0.79)
unpopular threshold = 3.5
ROC curve (area = 0.78)
unpopular threshold = 4
ROC curve (area = 0.77)
high_var threshold = 2.5
ROC curve (area = 0.91)
high_var threshold = 3
ROC curve (area = 0.76)
high_var threshold = 3.5
ROC curve (area = 0.74)
high_var threshold = 4
ROC curve (area = 0.79)
x-label
y-label

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.

Equation 5 is not convex, because the calculation of matrix factorization is by multiplying U and V, and this process lacks convexity as the objective function is invariant to permutations and rotations. Hence there are multiple local minimums in the objective function gradient plane.

With U fixed, the formula can be transferred into $min_V \sum_{i=1}\sum_{j=1} W_{ij}[r_{ij} - (UV^T)_{ij}]^2$, $V = (UU^T)^{-1}UR$ as a weighted least squares problem.

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 (Yaxis) against k (X-axis). For solving this question, use the default value for the regularization parameter.

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?

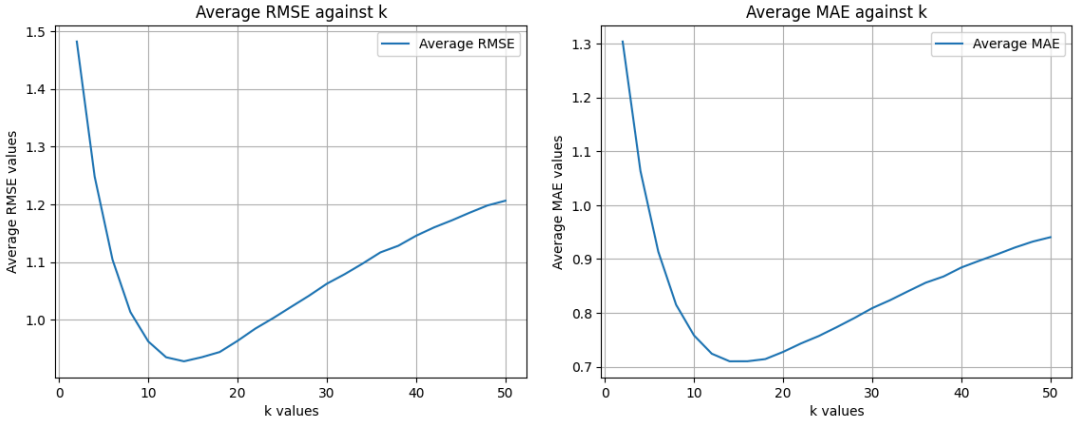
C. **Performance on trimmed dataset subsets:** For each of Popular, Unpopular and HighVariance subsets -

   – 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.

- 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.

A: **The average RMSE (Y-axis) against k (X-axis) and average MAE (Y- axis) against k (X-axis)**

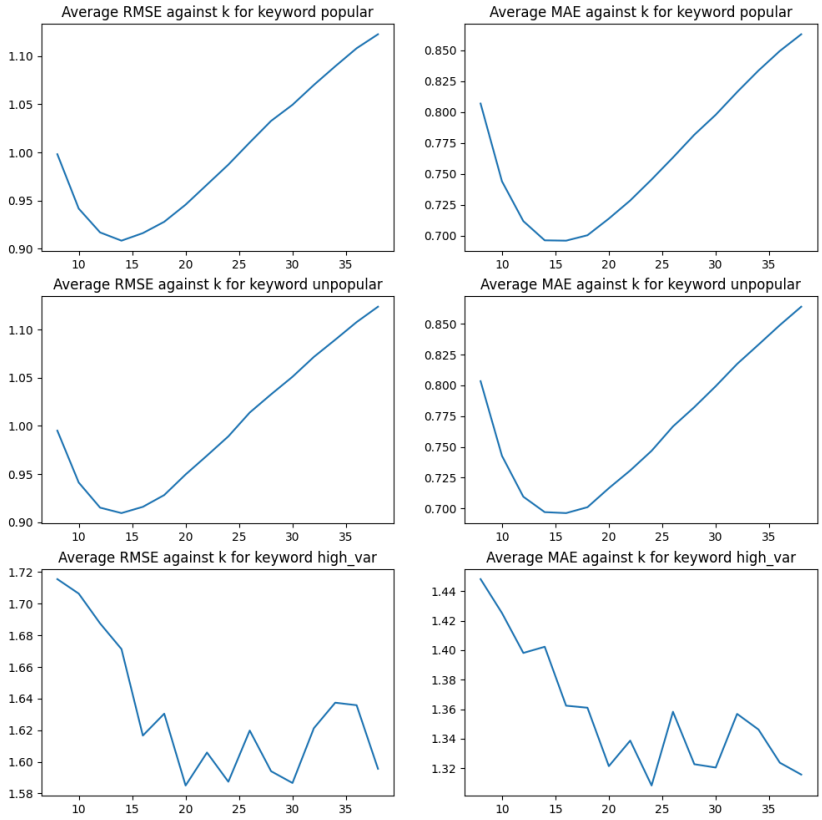


B **Please report the minimum average RMSE and MAE. Is the optimal number of latent factors the same as the number of movie genres?**

The minimum average RMSE is 0.890; the minimum average MAE is 0.678.

The optimal number of latent factors (k=16) is close to, but not the same as the number of movie genres.

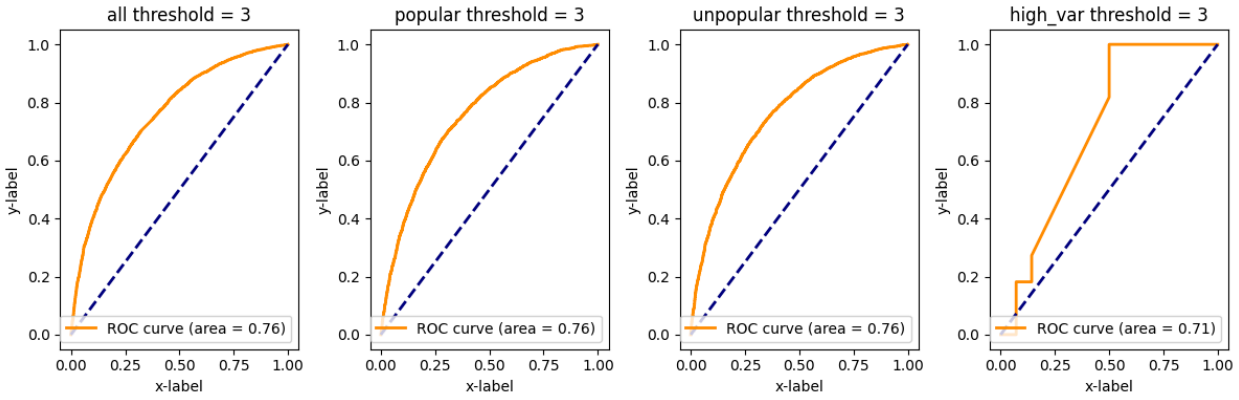
C **For each of Popular, Unpopular and HighVariance subsets: plot average RMSE (Y-axis) against k (X-axis); item Report the minimum average RMSE.**



```
min RMSE is 0.9083412425862987 for keyword popular
min RMSE is 0.9093701130788256 for keyword unpopular
min RMSE is 1.585049865863104 for keyword high_var
```

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.

The AUC value: as shown in the figures.



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?

Due to the length, the detailed results of each column can be found in the .ipynb file.

Do the top 10 movies belong to a particular or a small collection of genre?

The top 10 movies belong to a small collection of genres. For example, in column 2, all top 10 movies belong to comedy/drama.

For column 2, the top 10 movie ids:

	movieId	genres
1388	1904	Comedy|Drama
2423	3223	Drama
3048	4082	Comedy|Drama|Romance
3469	4733	Comedy
3660	5034	Drama|Romance
3987	5621	Action|Comedy
6095	42018	Comedy|Drama
6240	46572	Drama|Thriller
8110	100714	Drama|Romance
8165	102686	Comedy

For column 15, the top 10 movie ids:

	movieId	genres
1606	2148	Comedy|Fantasy|Horror
3425	4663	Comedy
3462	4721	Action|Comedy|Western
5032	7834	Comedy|Crime|Mystery|Romance
5379	8968	Action|Adventure|Comedy|Crime|Thriller
6110	42730	Drama
6371	49932	Drama|Mystery|Thriller
6489	53127	Drama|Horror|Thriller
7432	80860	Comedy|Romance
8650	120635	Action|Crime|Thriller

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

Yes, as each latent factor is related to a certain group of movies from a small collection of genres. For example, top 10 movies belong to comedy in column 2, while top 10 movies are more related to thriller/horror in column 15.

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.

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?

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

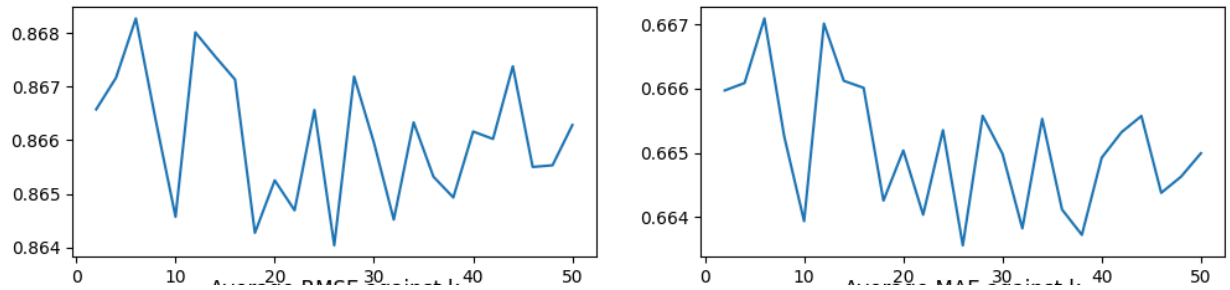
– Design a MF collaborative filter for each trimmed subset and evaluate its performance using 10-fold cross validation. Sweep k (number of latent factors) from 2 to 50 in step sizes of 2, and for each k compute the average RMSE obtained by averaging the RMSE across all 10 folds.

– Plot average RMSE (Y-axis) against k (X-axis); item Report the minimum average RMSE. 9

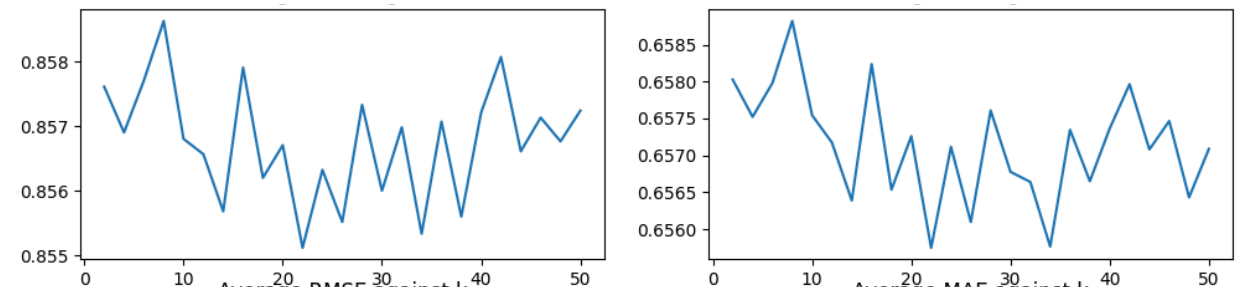
- Plot the ROC curves for the MF-based collaborative filter and also report the area under the curve (AUC) value as done in Question 6.

A *(The exact values can be obtained and printed from the codes.)*

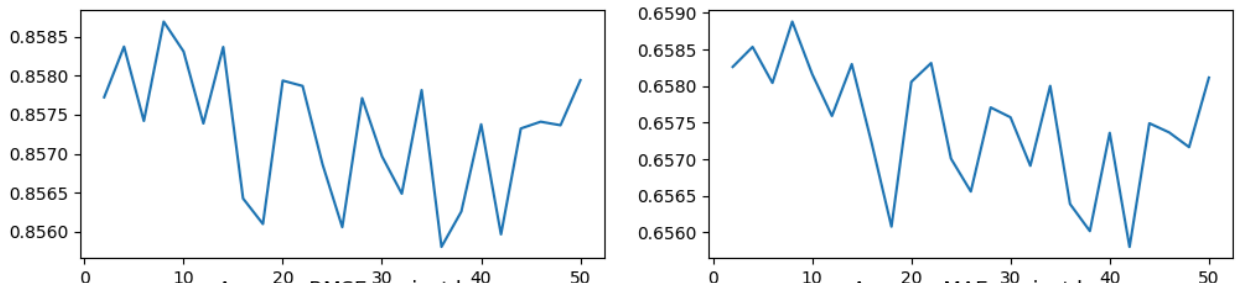
[All] Left: Average RMSE against k Right: Average MAE against k



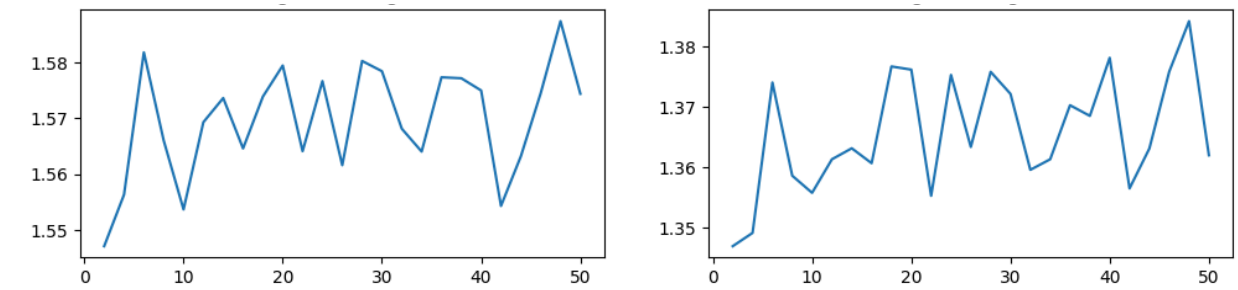
[Popular] Left: Average RMSE against k Right: Average MAE against k



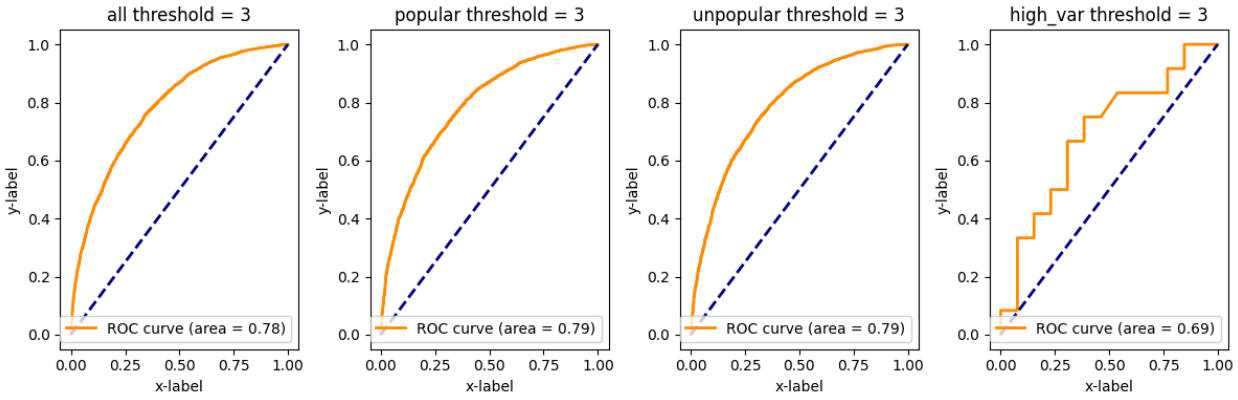
[Unpopular] Left: Average RMSE against k Right: Average MAE against k



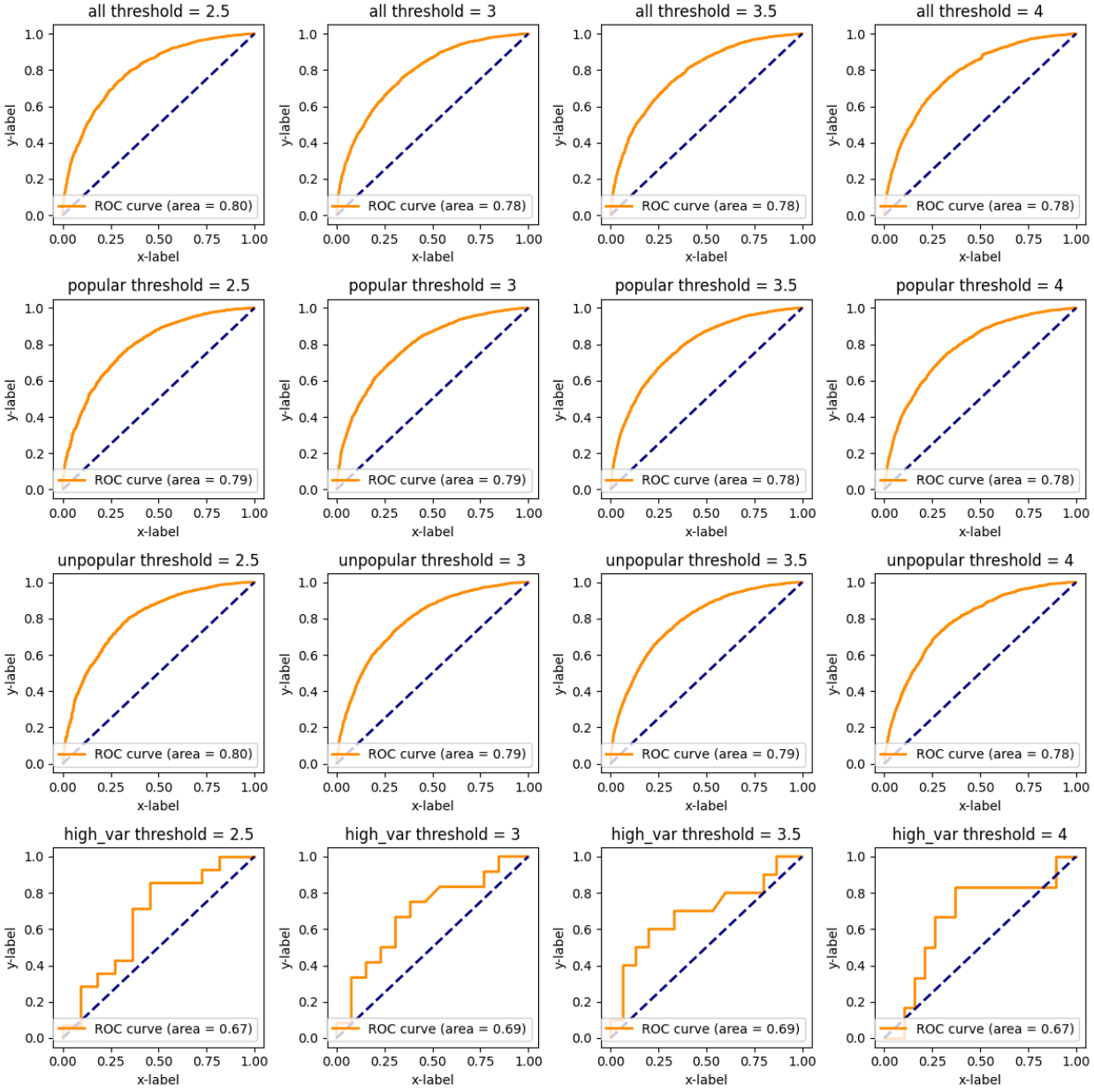
[High-var] Left: Average RMSE against k Right: Average MAE against k



B The optimal k = 24. It's also close to the number of the movie genres.



C 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.



QUESTION 11. Designing a Naïve 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.

The average RMSE (all): 1.0425;

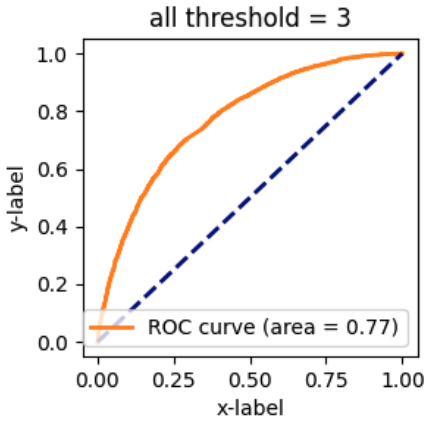
The average RMSE (popular): 1.0355;

The average RMSE (unpopular): 1.0355;

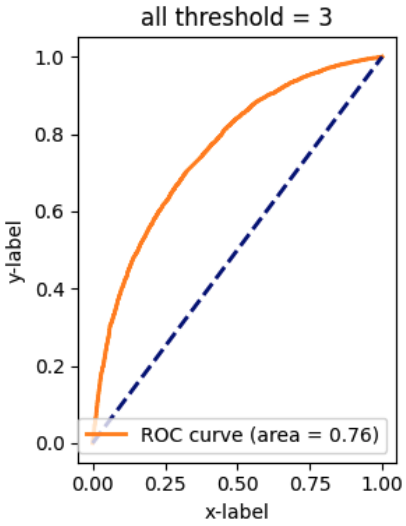
The average RMSE (high_var): 1.612.

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.

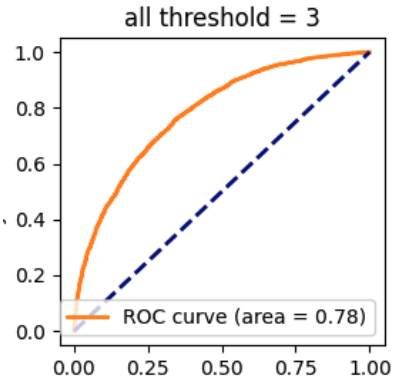
For k-NN: optimal k = 24.



For NMF: optimal k = 16.



For MF with bias: optimal k = 24.



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 Training Set: **6000**

Number of Unique Queries in Testing Set: **2000**

Distribution of Relevance Labels in Training Set: [377957 232569 95082 12658 5146]

Distribution of Relevance Labels in Testing Set: [124784 77896 32459 4450 1932]

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.

1. **Fold 1** – 723,412 data points and 25,637 total bins

   Model performance on the test set:

   - nDCG@3: 0.4564571300800643
   - nDCG@5: 0.4632890672260867
   - nDCG@10: 0.48286731451235976

2. **Fold 2** – 716,683 data points with 25,623 total bins

   Model performance on the test set:

   - nDCG@3: 0.4538895365009714
   - nDCG@5: 0.4573292117374164
   - nDCG@10: 0.4767546810011047

3. **Fold 3** – 719,111 data points in this fold, with a total of 25,659 bins

   Model performance on the test set:

   - nDCG@3: 0.4490681494620125
   - nDCG@5: 0.4583480538865081
   - nDCG@10: 0.47589507831078093

4. **Fold 4** – 718,768 data points and had 25,631 total bins

   Model performance on the test set:

   - nDCG@3: 0.461178820507814
   - nDCG@5: 0.4663860127875315
   - nDCG@10: 0.487724614983737

5. **Fold 5** – 722,602 data points and 25,501 total bins

   Model performance on the test set:

   - nDCG@3: 0.46963442883961365
   - nDCG@5: 0.4714315145908388
   - nDCG@10: 0.49035928048966515

Across all folds, the model demonstrated consistent performance, with slight variations in the nDCG scores. The nDCG@10 scores were consistently higher than nDCG@5 and nDCG@3 scores in all folds, indicating better model performance when considering a larger set of top-ranked items. The fifth fold showed the highest performance. The number of data points and total bins varied slightly across the folds, which may have influenced the model's performance.

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.

1. **Fold 1**:
    - Feature 134: Importance Score - 23856.702950954437
    - Feature 8: Importance Score - 4248.546391487122
    - Feature 108: Importance Score - 4135.244449853897
    - Feature 55: Importance Score - 4078.463216304779
    - Feature 130: Importance Score - 3635.03702378273

2. **Fold 2**:
    - Feature 134: Importance Score - 23578.90825009346
    - Feature 8: Importance Score - 5157.964912414551
    - Feature 55: Importance Score - 4386.669756650925
    - Feature 108: Importance Score - 4094.0121722221375
    - Feature 130: Importance Score - 4035.0706725120544

3. **Fold 3**:
    - Feature 134: Importance Score - 23218.075441122055
    - Feature 55: Importance Score - 4991.3033719062805
    - Feature 108: Importance Score - 4226.807395458221
    - Feature 130: Importance Score - 4059.7525141239166
    - Feature 8: Importance Score - 3691.792320251465

4. **Fold 4**:
    - Feature 134: Importance Score - 23796.899673223495
    - Feature 8: Importance Score - 4622.622978448868
    - Feature 55: Importance Score - 3883.4817056655884
    - Feature 130: Importance Score - 3356.8469800949097
    - Feature 129: Importance Score - 3207.5755367279053

5. **Fold 5**:
    - Feature 134: Importance Score - 23540.94235444069
    - Feature 8: Importance Score - 4794.9451723098755
    - Feature 55: Importance Score - 4079.608554124832
    - Feature 108: Importance Score - 3514.8357515335083
    - Feature 130: Importance Score - 3209.0584440231323

Across all folds, Feature 134 consistently emerged as the most important, with the highest importance scores in each fold. Other features such as 8, 55, 108, and 130 also frequently appeared in the top 5 list, although their rankings and scores varied among the folds.

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 resulting 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.
- Remove the 60 least important features according to the computed importance score in the question 15. Then train a new LightGBM model on the resulting 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.

1. **Removing Top 20 Most Important Features**
    - Fold 1: nDCG Score - 0.4083636029390886
    - Fold 2: nDCG Score - 0.4045026694861529
    - Fold 3: nDCG Score - 0.4116363812695088
    - Fold 4: nDCG Score - 0.4121071637228934
    - Fold 5: nDCG Score - 0.4166871494621703

    The removal of the top 20 most important features led to a noticeable decrease in nDCG scores across all folds. This outcome aligns with expectations, as removing important features diminishes the model's ability to make accurate predictions.

2. **Removing 60 Least Important Features**
    - Fold 1: nDCG Score - 0.4819713060930259
    - Fold 2: nDCG Score - 0.4772534003341443
    - Fold 3: nDCG Score - 0.4774361560299901
    - Fold 4: nDCG Score - 0.48888147783549574
    - Fold 5: nDCG Score - 0.4908165844880891

    The removal of the 60 least important features resulted in comparatively higher nDCG scores. The removal of less important features likely reduced noise and irrelevant information, leading to a more focused and effective model.

Hypothesis for Discrepancy

This outcome from removing the 60 least important features did not align with our initial expectations. We initially expected the nDCG scores to be higher than in the case of removing the 20 most important features, but we did not expect the performance improvement to be so drastic. It is clear now that because these 60 features were providing little to no value, or even introducing noise, their removal led to a more streamlined and effective model. The removals also prevented overfitting to these less important features, and removing them helped in generalizing the model better. The remaining features were more than sufficient to capture the essential patterns in the data, resulting in such high nDCG scores.