Large Scale Data Mining: Models and Algorithms

ECE 219 Spring 2023

Instructor: Professor Vwani Roychowdhury

UCLA, Department of ECE

Project 3 Report: Recommender Systems

Group Member

Syed R Nawshad	*StudentID* : 805871517
Huijun S Hao	*StudentID* : 105863846
Chao Zhang	*StudentID* : 904506124

May 21, 2023

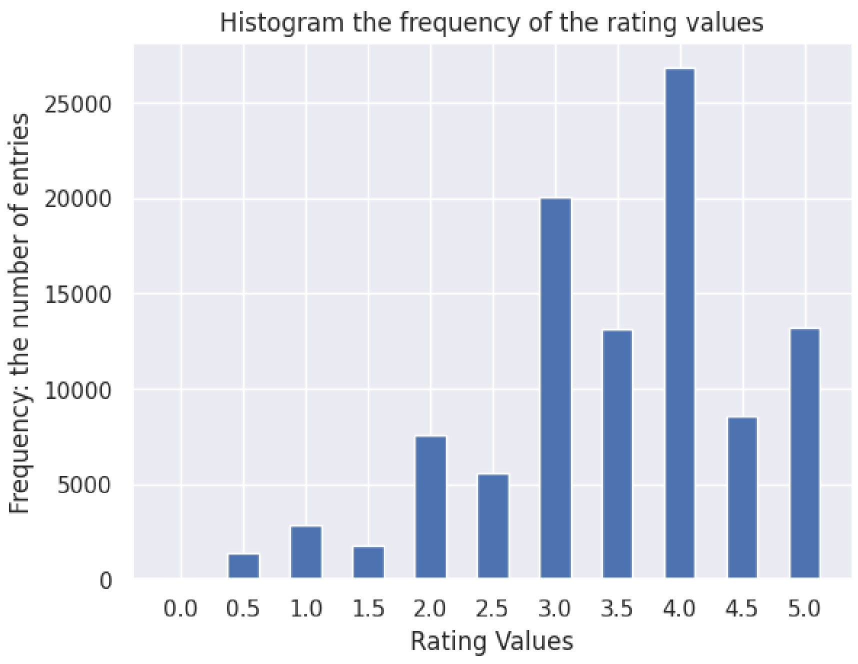
4. Dataset

QUESTION 1: Explore the Dataset

A. Compute the sparsity of the movie rating dataset.

$$Sparsity = \frac{Totalnumberofavailableratings}{Totalnumberofpossibleratings} = \frac{100836}{610*9742} = 0.016968273253211548$$

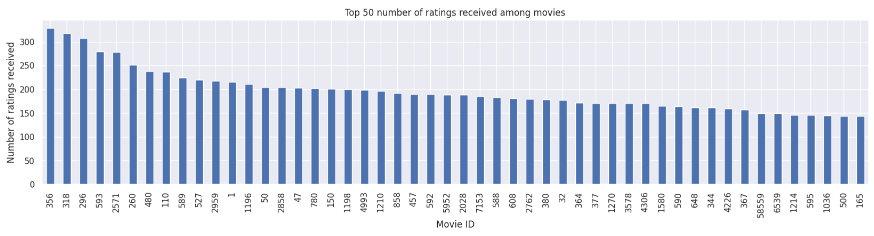
B. Plot a histogram showing the frequency of the rating values.

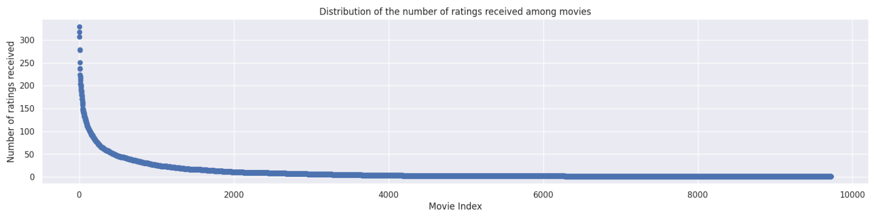


According to the plot above, the most rated movies have ratings ranging from 3 to 5. Many users will most likely rate their favorite popular movie and will not rate an unpopular movie.

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

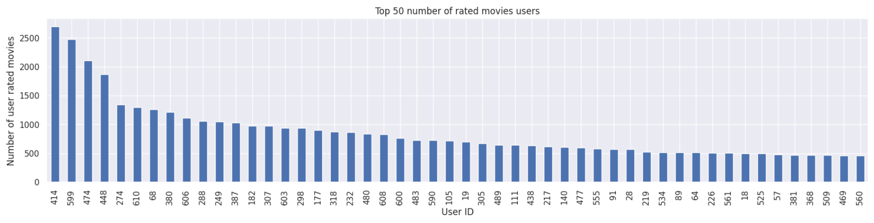
We created two plots to help us understand the distribution of the number of ratings received among movies for this question:

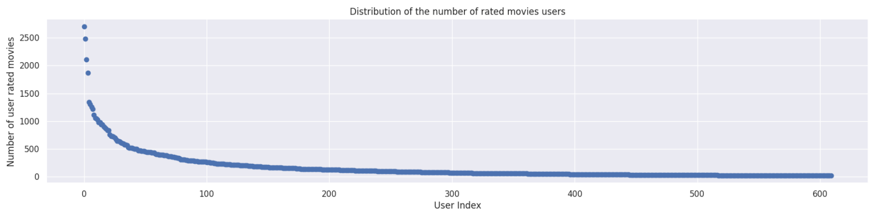




D. Plot the distribution of ratings among users.

We created two plots to help us understand the distribution of ratings among users for this question:

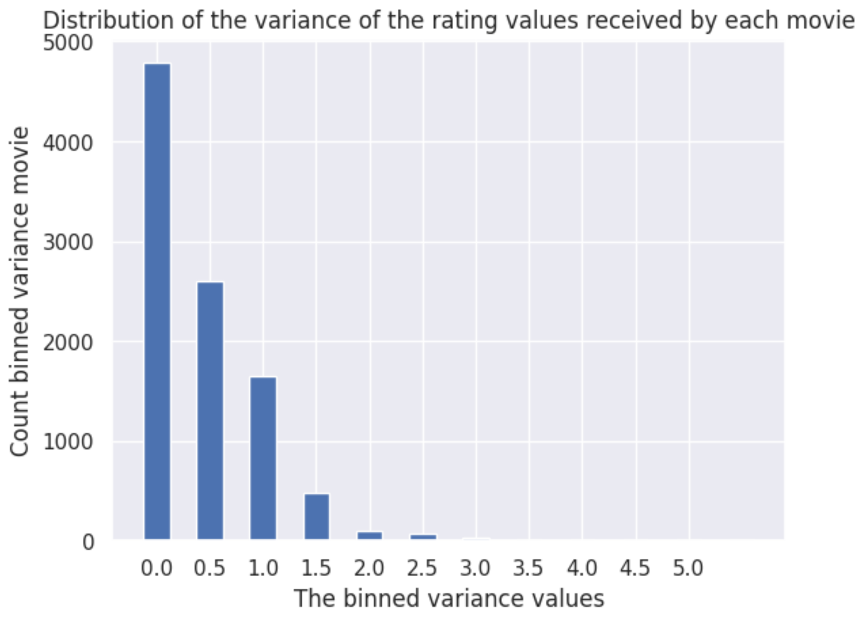




E. Discuss the salient features of the distributions.

According to questions C and D, nearly 80% of movies do not have a rating, and nearly 50% of users do not provide a rating. The rating matrix R then is sparse, as shown by question A. As a result, if we use current data to predict the recommendations, the results should be more accurate for popular movies and activated users compared to for others.

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



According to the plot above, the majority of movie rating values have a variance of less than 1.5, indicating that the majority of movies have very similar ratings among users.

5. Neighborhood-based collaborative filtering

Question 2: Understanding the Pearson Correlation Coefficient

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

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

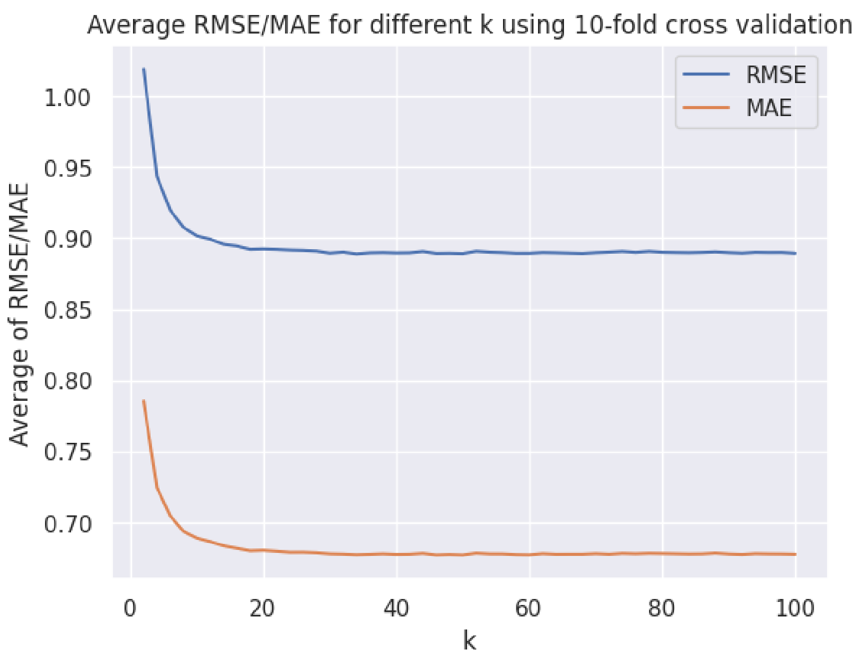
B. In plain words, explain the meaning of $I_u \bigcap I_v$. Can $I_u \bigcap I_v = \phi$? (Hint: Rating matrix R is sparse).

A set of distinct movies that both users u and v have given a rating is denoted by $I_u \bigcap I_v$. We know from the previous question that the rating matrix R is sparse. And, it also make sense for the users to have distinct movie preferences. As a result, $I_u \bigcap I_v$ can be ϕ.

Question 3: Understanding the Prediction function

In the prediction function, mean-centering the raw ratings lowers the bias between users who evaluate all items highly and users who rate all items poorly. The model can focus on relative variations in ratings among users rather than absolute rating values by subtracting each user's average rating μ_v from their individual ratings r_{vj}. It ensures accurate prediction by taking into consideration differences from each user's average rating, as well as individual preferences and changes in rating behavior.

Question 4: Design a k-NN collaborative filter



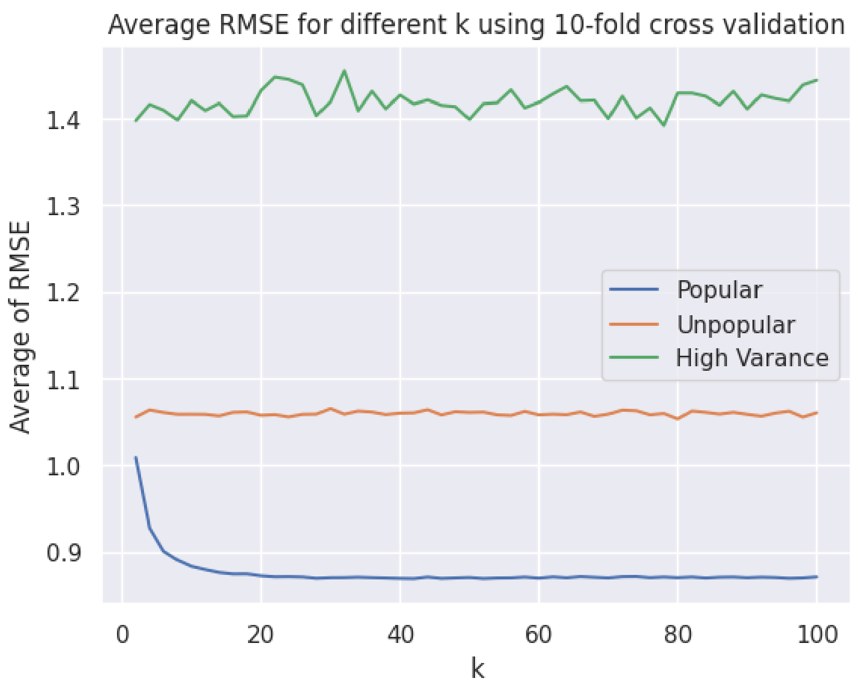
Question 5: Use the plot from question 4, to find a minimum k

From the question 4, we can find that the minimum k = 20, the corresponding steady-state average RMSE = 0.892496065425736, and the corresponding steady-state average MAE = 0.6803162530772907.

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

6. Model-based collaborative filtering

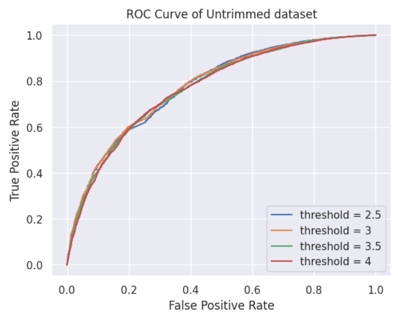
Compute and plot the average RMSE of the 3 trimmed subsets in the dataset:

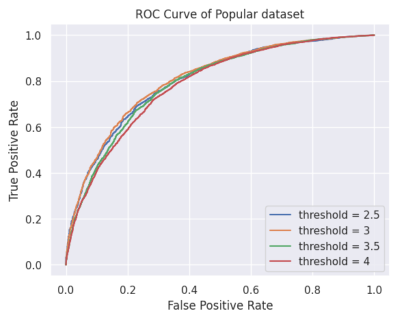


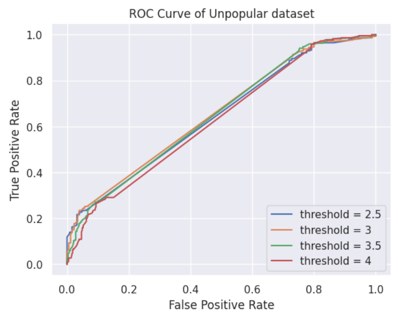
Trimmed subsets	Minimum average RMSE
Popular	0.8688034890626988
Unpopular	1.053226227471778
High variance	1.3918942225463664

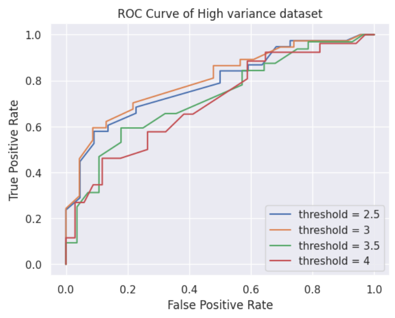
Table 1: The minimum average RMSEs of 3 trimmed subsets

Plot the ROC curves for the k-NN collaborative filters for threshold values [2.5, 3, 3.5, 4]:









Subsets	Threshold 2.5	Threshold 3	Threshold 3.5	Threshold 4
No trimming	0.7774	0.7794	0.7717	0.7702
Popular	0.8021	0.8080	0.7923	0.7844
Unpopular	0.6459	0.6573	0.6505	0.6291
High varance	0.7865	0.8038	0.7238	0.7048

Table 2: The AUC of 4 subset (1 no trimming and 3 trimmed)

6. Model-based collaborative filtering

Question 7

The NMF cost function is the following:

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

In order to determine whether a function is convex or concave, we can examine its Hessian matrix, which involves taking the second derivative of the function. If the Hessian matrix is not positive semi-definite, it indicates that the function is non-convex.

The Hessian Matrix of L is given by the following:

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

The determinant of the above is given by the following: $|\nabla^2 L(U,V)| = -(R - UV)(R - 3UV)$

The determinant mentioned above is unequivocally not positive semi-definite. Consequently, for any general scenario involving variables m and n, the objective function is deemed non-convex.

Furthermore, within the gradient plane of the objective function, numerous local minima can be observed. As a result, the optimization problem is not concurrently convex for both the user latent space (U) and the item embedding space (V). This arises from the fact that the matrix factorization model predicts ratings by employing a combination of U and V, which fails to meet the convexity requirement due to the permutation and rotation invariance of the objective function.

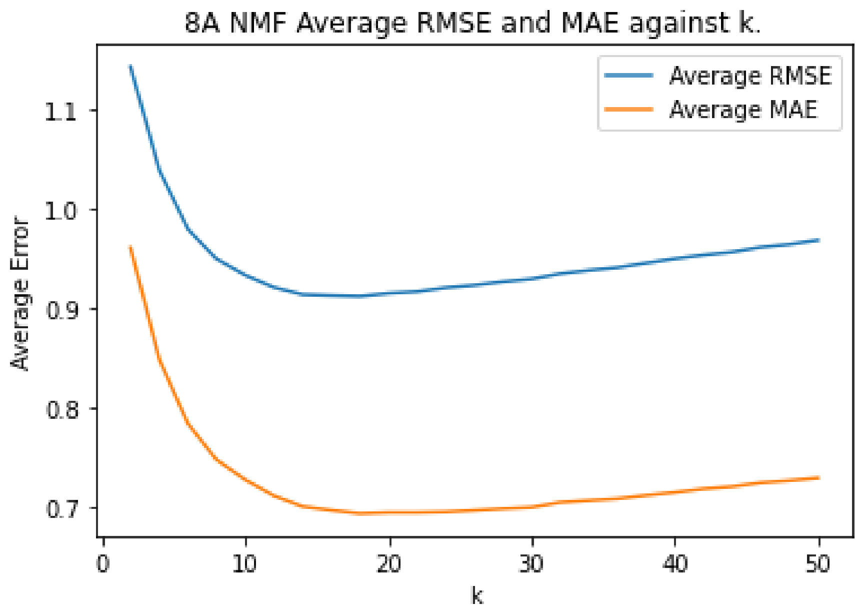
The least-squares optimization problem can be approached by either keeping U constant and solving for V, or vice versa. When U is fixed (without any regularization), the objective function takes the following form:

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

And $V = (UU^T)^{-1}UR$ where R is the ratings matrix.

Therefore, at each iteration of the ALS algorithm, we are effectively addressing a least-squares problem. This characteristic lends stability to the algorithm and contributes to its rapid convergence rate.

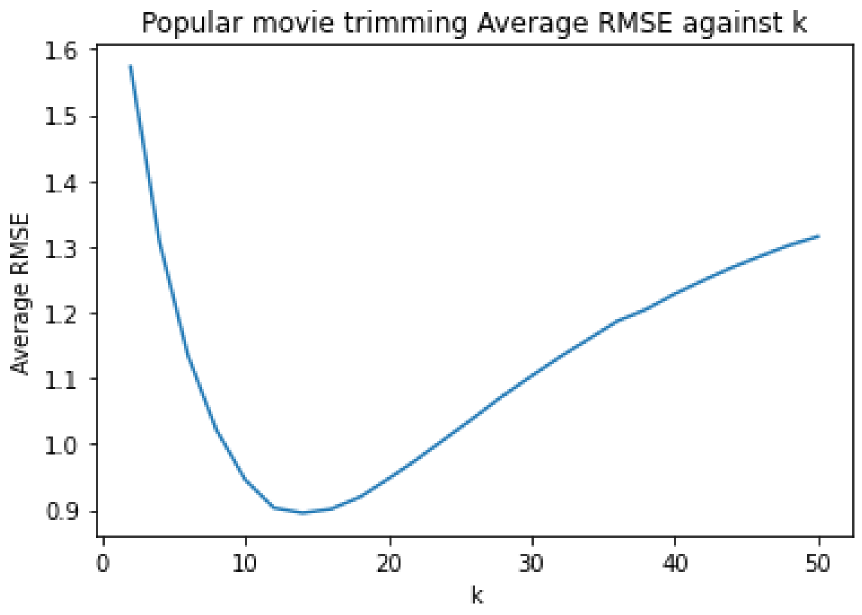
Question 8



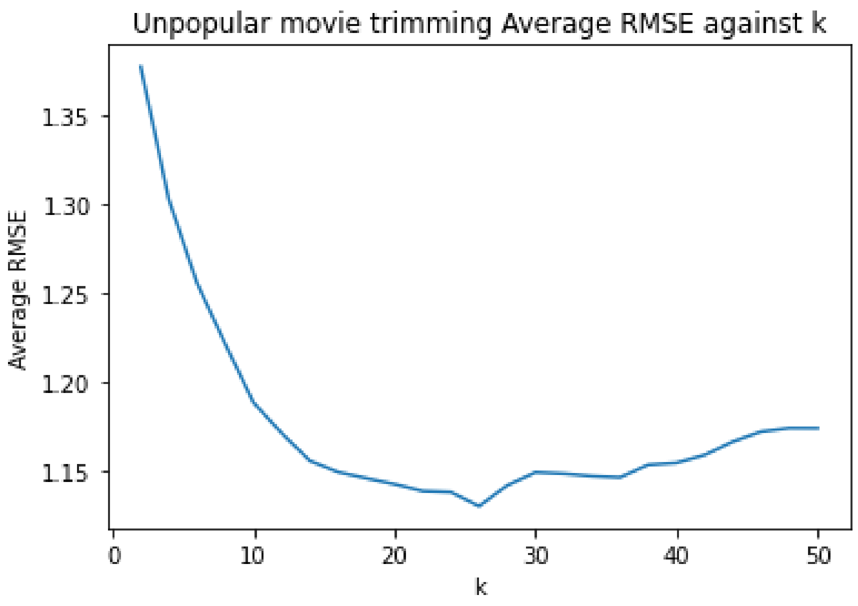
A A NMF-based collaborative filter (CF) was developed for predicting movie ratings in the MovieLens dataset. The performance of the filter was evaluated using 10-fold cross-validation. The Pearson-correlation function was employed as the similarity metric, with the parameter k ranging from 2 to 100 of size 2. For each value of k, the average RMSE (Root Mean Square Error) and average MAE (Mean Absolute Error) were computed across all 10 folds. The implementation of this NMF-based collaborative filter utilized the matrix factorization algorithm.

B Minimum Average RMSE (NMF) = 0.967, k = 50 Minimum Average MAE (NMF) = 0.693, k = 22 The MovieLens Dataset consists of 19 movie genres, which is in line with the average k value obtained by our analysis, calculated as (22 + 50) / 2 = 36. In NMF, we aim to discover latent factors that can be derived from users and are associated with the movies. These latent factors typically correspond to one or a few genres of the movies. Hence, it is logical that the number of latent factors tends to be similar to the number of genres present in the dataset.

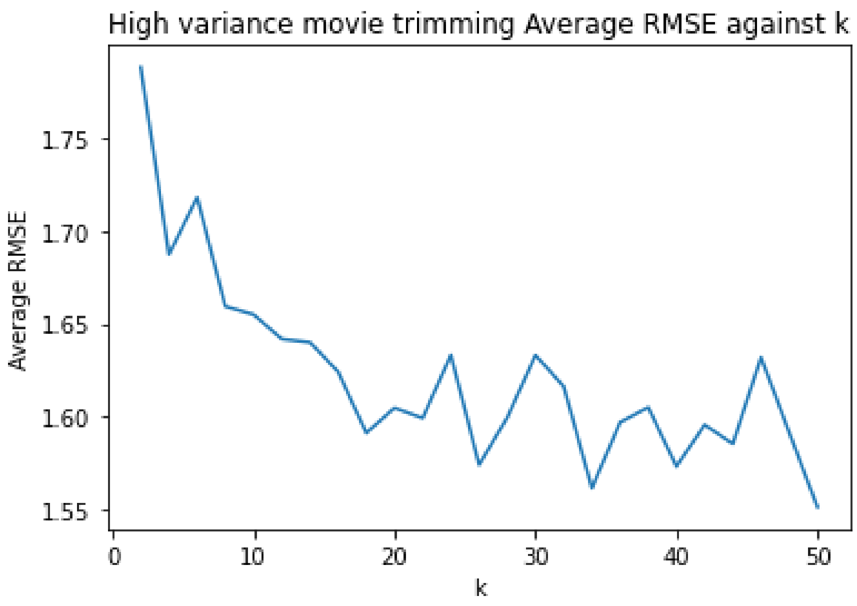
C • Popular movie trimming: In this approach, the test set is trimmed to include only movies that have received more than 2 ratings. If a movie in the test set has received 2 or fewer ratings in the entire dataset, it is removed from the test set, and its rating is not predicted using the trained filter. To evaluate the performance of the popular movie trimmed test set, we conducted 10-fold cross-validation. The Pearson-correlation function was used as the similarity metric, and the parameter k was varied from 2 to 100 in increments of 2. For each value of k, the average RMSE (Root Mean Square Error) was computed across all 10 folds. The minimum average RMSE obtained for the popular movie trimmed test set was 0.754 with best k=14. Upon observing the resulting plot, we noticed a similar curve to the one observed in 8a. As anticipated, the prediction error was significantly reduced compared to the values in 8a. This can be attributed to the removal of movies with fewer ratings, which helps eliminate certain outliers. The pattern of the plot after popular movie trimming closely resembles that of the plot without the trimming operation. This indicates that removing unpopular movies from the testing dataset stabilizes the testing performance. Unpopular movies with limited ratings are challenging to predict accurately for most users, and their exclusion contributes to improved stability in the predictions.



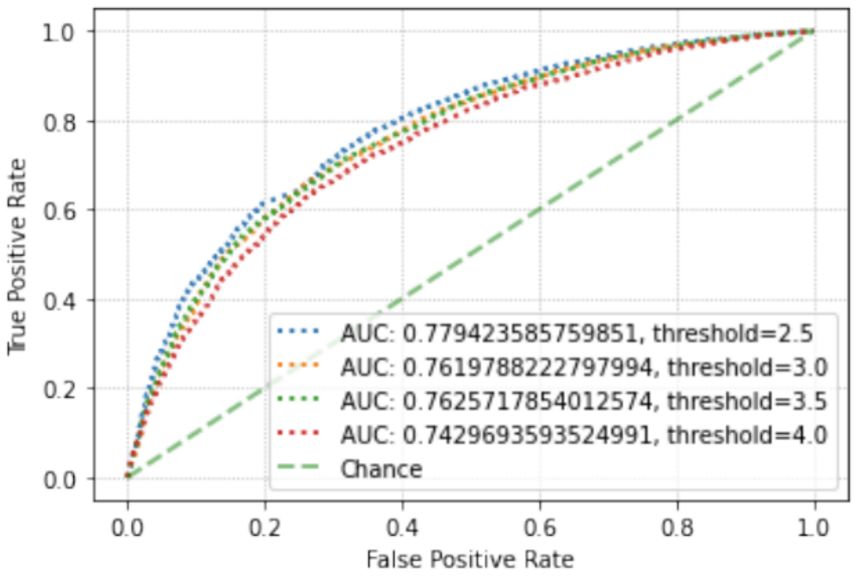
• Unpopular movie trimming: The minimum average RMSE for the unpopular movie trimmed test set is 1.139. The minimum k for RMSE is 22 . Comparing these RMSE values with those obtained for the popular movie trimmed set and the original test set, we can conclude that the NMF Collaborative Filter model is not suitable for the unpopular movie trimmed test set. This is because the remaining movies in the test dataset after trimming tend to receive low user ratings. As the predictor was trained on the full dataset, which includes popular movies, it struggles to accurately predict the ratings of ”unusual” movies due to a lack of sufficient user ratings for those movies. We observe a decreasing pattern in the RMSE values instead of the erratic pattern observed previously. Unpopular movie trimming Minimum k for RMSE is 22, average RMSE is 1.139



- High variance movie trimming: The minimum average RMSE for the high variance movie trimming test set is 1.524 with the best k = 50. Similar to the scenario with the unpopular movie trimmed test set, the same situation applies here. Predicting ratings on the test dataset after high variance movie trimming encounters the same challenge: there is a scarcity of high variance data points in the training dataset. Therefore, the model struggles to provide accurate results for these outlier points.The movies remaining in the testing dataset after high variance movie trimming are predominantly contentious. Here, contentiousness implies that the movies are somewhat popular, with a significant number of ratings, but the ratings assigned by different individuals exhibit substantial disagreement, resulting in large variance. Predicting ratings for such movies becomes difficult. This explains the erratic patterns observed in the plot.



We generated an ROC curve for the NMF Collaborative filters using threshold values of [2.5, 3, 3.5, 4]. The dataset was divided into a 0.9 training set and a 0.1 test set. The chosen value of k from the previous question, k = 18, was used. In this analysis, we assumed that if a user's rating for a movie is equal to or higher than the threshold, the label is set to 1 to indicate that the user likes the movie. The ROC curve depicted above illustrates the relationship between the true positive rate and the false positive rate. It serves as a measure of the effectiveness of the recommended films for the user, indicating the relevance of the recommendations.



Question 9

Latent Factor 0

Genre: Action—Adventure—Sci-Fi Value: 2.0018
Genre: Action—Adventure—Comedy Value: 1.9872
Genre: Drama—Musical Value: 1.8357
Genre: Comedy Value: 1.8237
Genre: Comedy—Fantasy Value: 1.7826
Genre: Comedy—Romance Value: 1.7189
Genre: Action—Comedy—Crime—Drama Value: 1.6778
Genre: Adventure—Children Value: 1.6582
Genre: Children—Comedy Value: 1.5932
Genre: Comedy—Crime—Romance Value: 1.5444

Latent Factor 1

Genre: Action—Adventure—Fantasy—Mystery Value: 1.9228
Genre: Thriller Value: 1.8736
Genre: Action—Sci-Fi Value: 1.8701
Genre: Action Value: 1.8459
Genre: Horror—Sci-Fi Value: 1.8285
Genre: Action—Adventure—Children—Comedy—Crime Value: 1.8020
Genre: Action—Adventure—Fantasy—Sci-Fi Value: 1.7744
Genre: Action—Adventure—Comedy—Romance Value: 1.6233
Genre: Children—Comedy Value: 1.6198
Genre: Action—Comedy—Crime Value: 1.6154

Latent Factor 2

Genre: Action—Adventure—Fantasy—Thriller Value: 1.9371
Genre: Drama—War Value: 1.8634
Genre: Animation—Drama—Romance Value: 1.7843
Genre: Animation—Comedy—Drama—Romance—Sci-Fi Value: 1.7075
Genre: Adventure Value: 1.6176
Genre: Action—Adventure—Drama Value: 1.5969
Genre: Horror—Romance Value: 1.5603
Genre: Horror—Thriller Value: 1.5484
Genre: Drama Value: 1.5107
Genre: Comedy—Crime—Drama—War Value: 1.4954

Latent Factor 3

Genre: Comedy—Musical—Romance Value: 3.2933
Genre: Documentary Value: 2.5280
Genre: Horror—Mystery—Thriller Value: 1.9742
Genre: Comedy—Romance Value: 1.8011
Genre: Comedy—Fantasy Value: 1.7765
Genre: Comedy—Drama Value: 1.7528
Genre: Action—Drama—War Value: 1.7422
Genre: Comedy—Drama Value: 1.7403
Genre: Sci-Fi—IMAX Value: 1.7035
Genre: Comedy—Horror—Sci-Fi Value: 1.6899

Latent Factor 4

Genre: Crime—Thriller Value: 2.2998
Genre: Comedy Value: 2.0570
Genre: Horror Value: 1.9620
Genre: Documentary Value: 1.9288
Genre: Comedy Value: 1.8853
Genre: Comedy—Crime—Drama—Mystery Value: 1.8332
Genre: Documentary Value: 1.8276
Genre: Adventure—Animation—Fantasy—IMAX Value: 1.7967
Genre: Drama—Fantasy Value: 1.7746
Genre: Drama Value: 1.7393

Latent Factor 5

Genre: Action—Drama—War Value: 2.8381
Genre: Drama—Romance Value: 2.1068
Genre: Drama—Horror—Mystery—Thriller Value: 2.0511
Genre: Action—Drama Value: 1.9869
Genre: Comedy—Drama Value: 1.8472
Genre: Comedy—Musical—Romance Value: 1.8127
Genre: Adventure—Drama—Sci-Fi—Thriller Value: 1.7993
Genre: Drama—War Value: 1.7713
Genre: Adventure—Drama—Horror—Thriller Value: 1.6999
Genre: Comedy Value: 1.6873

Latent Factor 6

Genre: Action—Drama—Thriller Value: 2.5284
Genre: Comedy Value: 1.8688
Genre: Horror—Thriller Value: 1.8627
Genre: Action—Adventure Value: 1.8457
Genre: Drama Value: 1.7572
Genre: Drama—War Value: 1.7235
Genre: Comedy—Thriller Value: 1.6878
Genre: Action—Adventure—Animation—Children—Fantasy—Sci-Fi Value: 1.6807
Genre: Crime—Thriller Value: 1.6468
Genre: Adventure—Animation—Children—Fantasy—IMAX Value: 1.6431

Latent Factor 7

Genre: Horror Value: 2.0799
Genre: Horror—Mystery—Thriller Value: 2.0440
Genre: Documentary Value: 2.0286
Genre: Children—Comedy—Fantasy Value: 1.9636
Genre: Comedy—Drama Value: 1.8393
Genre: Romance Value: 1.7490
Genre: Action—Sci-Fi—Thriller Value: 1.7289
Genre: Comedy—Drama Value: 1.7164
Genre: Documentary Value: 1.7065
Genre: Drama—War Value: 1.6963

Latent Factor 8

Genre: Drama—Romance Value: 2.2493
Genre: Adventure—Children—Comedy—Mystery Value: 2.1117
Genre: Drama Value: 1.9078
Genre: Action—Adventure—Comedy—Romance Value: 1.7621
Genre: Drama Value: 1.7521
Genre: Children—Comedy Value: 1.7238
Genre: Action—Crime—Thriller Value: 1.7158
Genre: Action—Adventure—Western Value: 1.6991
Genre: Drama Value: 1.6701
Genre: Comedy—Drama Value: 1.6619

Latent Factor 9

Genre: Crime—Drama—Thriller Value: 2.1923
Genre: Animation—Children Value: 1.9936
Genre: Action—Drama—Mystery Value: 1.9843
Genre: Comedy Value: 1.9614
Genre: Comedy—Horror Value: 1.8632
Genre: Comedy—Horror—Sci-Fi Value: 1.8374
Genre: Documentary Value: 1.7656
Genre: Action—Adventure—Comedy Value: 1.7393
Genre: Drama Value: 1.7310
Genre: Action—Drama—Mystery—Thriller Value: 1.6578

Latent Factor 10

Genre: Comedy Value: 2.0003
Genre: Comedy—Drama Value: 1.8061
Genre: Crime—Romance—Thriller Value: 1.7542
Genre: Crime—Drama—Romance—Thriller Value: 1.7384
Genre: Adventure—Drama—Fantasy—Romance Value: 1.5307
Genre: Comedy—Drama Value: 1.5298
Genre: Drama Value: 1.5289
Genre: Comedy Value: 1.5143
Genre: Comedy—Drama Value: 1.4731
Genre: Comedy—Drama Value: 1.4238

Latent Factor 11

Genre: Action—Comedy—Crime—Romance Value: 2.3027
Genre: Crime—Drama—Thriller Value: 2.1116
Genre: Drama—War Value: 1.8482
Genre: Comedy—Romance Value: 1.7970
Genre: Horror Value: 1.7756
Genre: Adventure—Animation—Comedy Value: 1.7361
Genre: Comedy Value: 1.7357
Genre: Drama—Mystery—Thriller Value: 1.7141
Genre: Documentary—Drama Value: 1.6103
Genre: Comedy—Drama—Romance Value: 1.5996

Latent Factor 12

Genre: Action—Crime Value: 2.4327
Genre: Action—Adventure—Fantasy—Thriller Value: 2.3699
Genre: Drama—Romance Value: 1.9844
Genre: Children—Comedy—Drama Value: 1.8249
Genre: Drama Value: 1.7771
Genre: Comedy—Drama—Romance Value: 1.6915
Genre: Comedy Value: 1.6326
Genre: Action—Adventure—Sci-Fi—Thriller Value: 1.6322
Genre: Documentary Value: 1.6272
Genre: Drama Value: 1.5874

Latent Factor 13

Genre: Horror Value: 2.7033
Genre: Action—Drama—War Value: 2.5029
Genre: Action—Horror—Sci-Fi Value: 2.1431
Genre: Comedy Value: 2.1063
Genre: Comedy—Drama—Romance Value: 2.0154
Genre: Action—Comedy—Crime—Romance Value: 1.8878
Genre: Action—Adventure—Mystery—Thriller Value: 1.8439
Genre: Drama Value: 1.7634
Genre: Action—Crime—Drama—Thriller Value: 1.6709
Genre: Crime—Drama—Mystery Value: 1.6521

Latent Factor 14

Genre: Action—Adventure—Animation—Comedy—Fantasy—Sci-Fi Value: 2.3395
Genre: Action—Drama Value: 1.9889
Genre: Drama—Thriller Value: 1.9486
Genre: Comedy Value: 1.8717
Genre: Drama—War Value: 1.8387
Genre: Adventure—Animation—Comedy Value: 1.7458
Genre: Comedy Value: 1.7211
Genre: Horror—Sci-Fi Value: 1.6644
Genre: Drama—Thriller Value: 1.6047
Genre: Comedy—Sci-Fi Value: 1.5615

Latent Factor 15

Genre: Comedy Value: 2.2919
Genre: Horror Value: 1.9645
Genre: Comedy—Drama—Romance Value: 1.9042
Genre: Drama Value: 1.7042
Genre: Drama—Romance—Sci-Fi Value: 1.6879
Genre: Comedy Value: 1.6846
Genre: Action—Crime—Drama—Thriller Value: 1.6642
Genre: Comedy Value: 1.6492
Genre: Documentary Value: 1.6476
Genre: Action—Adventure—Sci-Fi—IMAX Value: 1.6424

Latent Factor 16

Genre: Drama Value: 2.2757
Genre: Action—Drama Value: 2.0960
Genre: Drama—Horror—Thriller Value: 1.9394
Genre: Action—Adventure—Fantasy Value: 1.9072
Genre: Adventure—Drama—Fantasy—Romance Value: 1.8742
Genre: Drama—Romance Value: 1.8234
Genre: Drama—Horror—Mystery—Thriller Value: 1.7857
Genre: Action Value: 1.7785
Genre: Crime—Drama—Thriller Value: 1.7515
Genre: Drama—Fantasy Value: 1.7474

Latent Factor 17

Genre: Comedy Value: 2.5311
Genre: Crime—Drama—Thriller Value: 2.4155
Genre: Adventure—Drama—Fantasy—Romance Value: 2.1033
Genre: Drama Value: 1.8202
Genre: Adventure—Drama Value: 1.7604
Genre: Comedy—Drama Value: 1.6972
Genre: Drama—Romance Value: 1.6508
Genre: Drama Value: 1.6298
Genre: Drama—Western Value: 1.6246
Genre: Action—Crime—Drama—Thriller Value: 1.6076

Latent Factor 18

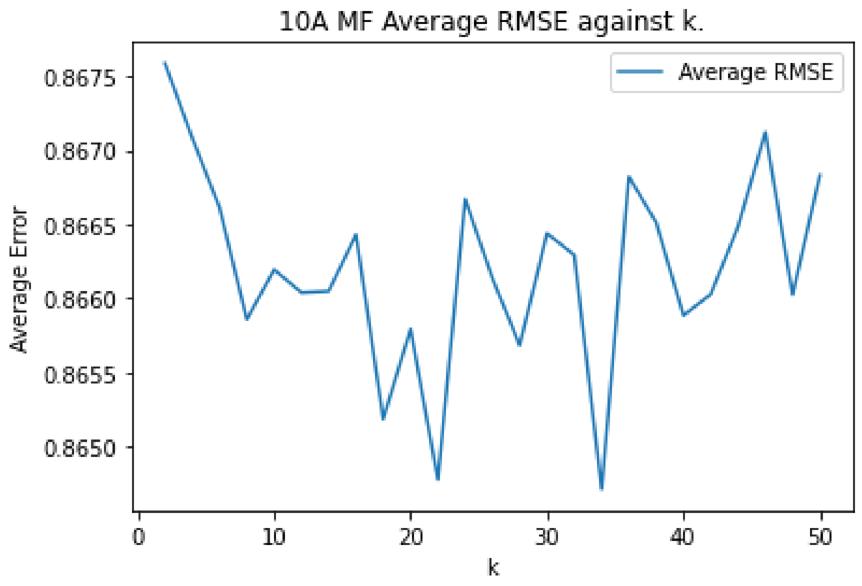
Genre: Drama—Romance—War Value: 2.3591
Genre: Crime—Drama—Mystery—Thriller Value: 2.1408
Genre: Drama—Romance Value: 2.0681
Genre: Comedy—Crime—Drama Value: 1.9558
Genre: Drama—Sci-Fi Value: 1.9398
Genre: Comedy—Horror—Sci-Fi Value: 1.9061
Genre: Horror—Thriller Value: 1.8722
Genre: Crime—Drama—Mystery—Thriller Value: 1.7561
Genre: Comedy—Romance Value: 1.7542
Genre: Crime—Drama Value: 1.7174

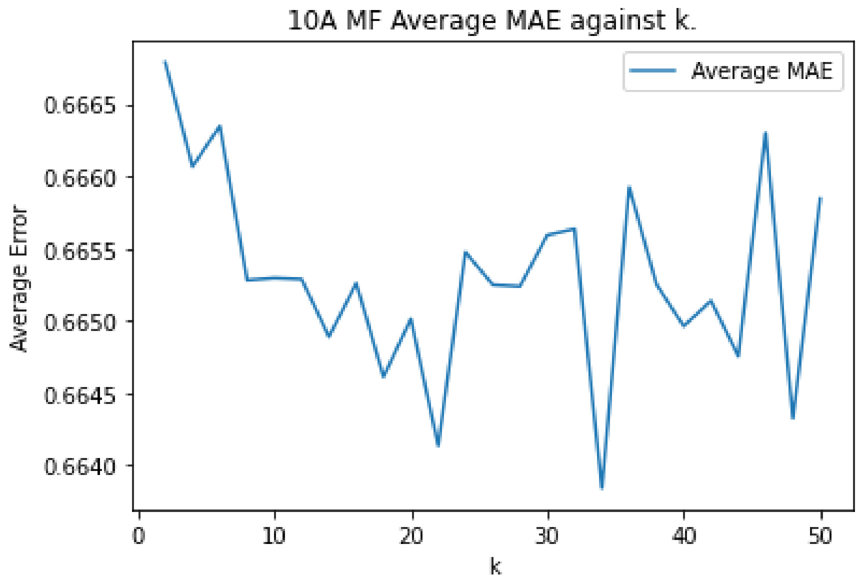
Latent Factor 19

Genre: Action—Sci-Fi Value: 1.9617
Genre: Action—Adventure—Sci-Fi—Thriller—IMAX Value: 1.9350
Genre: Animation—Comedy—Drama—Fantasy Value: 1.9201
Genre: Adventure—Animation—Fantasy—IMAX Value: 1.8516
Genre: Horror—Thriller Value: 1.8428
Genre: Horror—Thriller Value: 1.8075
Genre: Comedy—Drama Value: 1.7709
Genre: Comedy Value: 1.6834
Genre: Action—Adventure—Fantasy—Thriller Value: 1.6237
Genre: Comedy—Crime—Drama—Musical Value: 1.6079

Through the application of NMF, it becomes evident that each latent factor corresponds to a different movie genre. The clustering performed by NMF effectively groups movies based on their latent factors, with genre being a significant contributing factor. As the number of latent factors increases, the distinct movie genres become more apparent. This implies that a higher number of latent factors leads to improved clustering and representation of movie genres.

Question 10

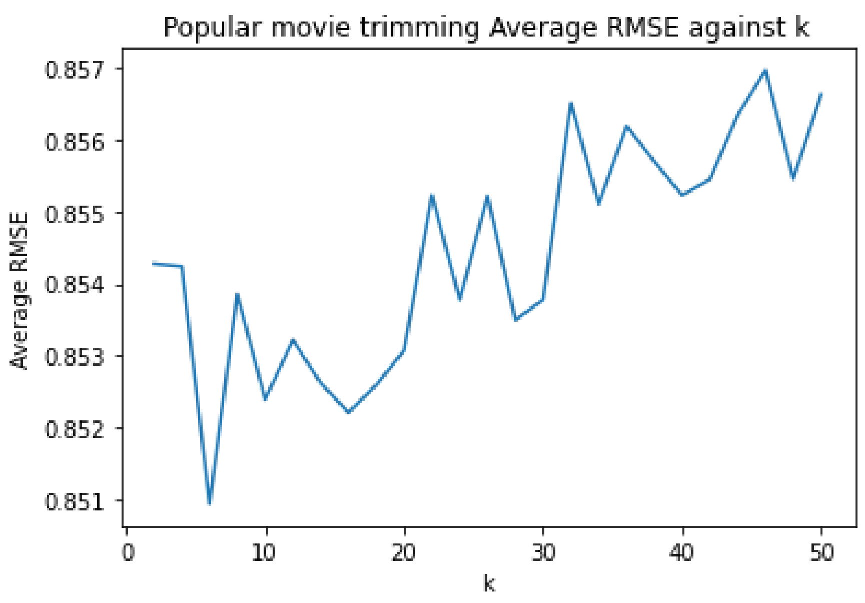




A The user bias term captures how a user's characteristics influence their ratings, causing them to deviate from the average ratings. On the other hand, the item bias term accounts for how a specific movie or item is rated compared to the average ratings. By incorporating bias information, the resulting embeddings or representations become more effective in capturing user-specific or movie-specific noise.

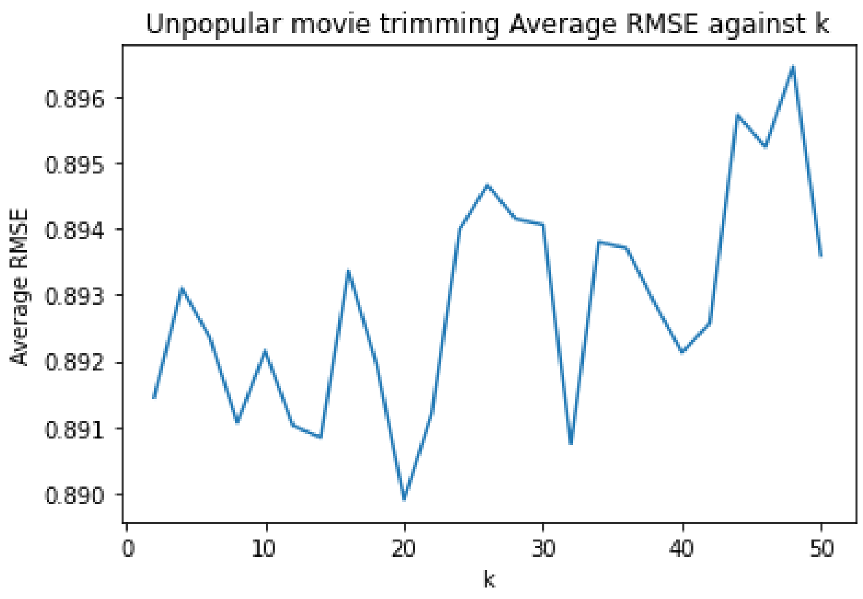
B Minimum Average RMSE (SVD) = 0.865, k = 34 Minimum Average MAE (SVD) = 0.664, k = 34 The optimal number of latent factors is not exactly the same as the total number of movie genres. The latent components in this model are difficult to interpret.

C Popular movie trimming Minimum k for RMSE is 2, average RMSE is 0.854. Contrary to earlier models, the plot does not exhibit a consistently decreasing curve. As anticipated, the prediction error is significantly reduced compared to the values in10A . This improvement is attributed to the removal of movies with fewer ratings, effectively eliminating some outliers. The plot displays a jumpy pattern. This pattern is not indicative of a relationship between RMSE and different values of k, but rather a result of different data splitting schemes generated by different random states in the cross-validation process. Additionally, the difference in RMSE values is so small that, if the dataset is partitioned in the same manner each time, the corresponding curve would appear as a horizontal line.

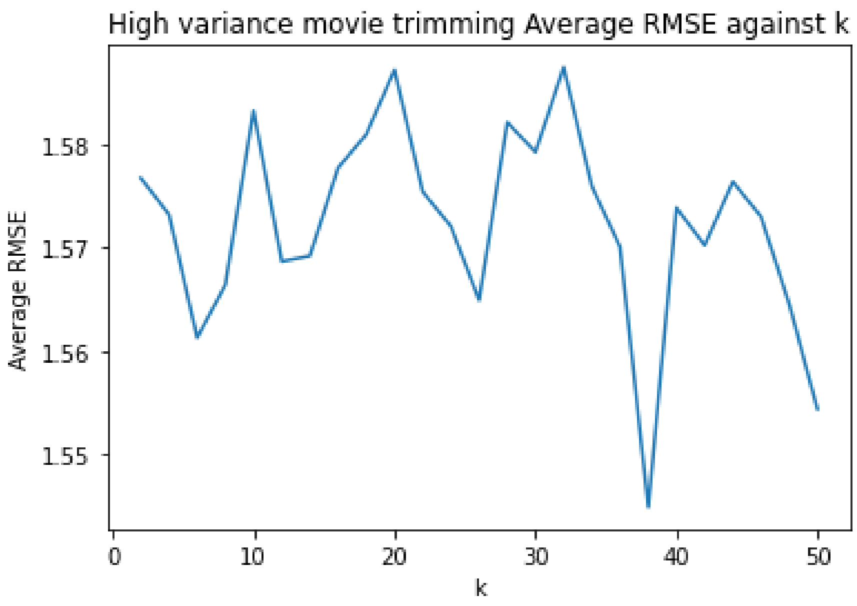


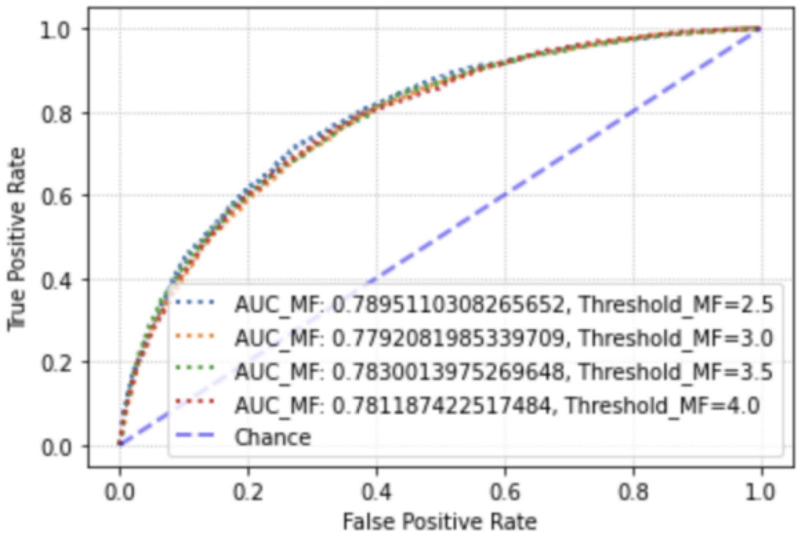
Unpopular movie trimming Minimum k for RMSE is 4, average RMSE is 0.893 The obtained RMSE value for the unpopular movie trimmed test set is higher compared to that of the popular movie trimmed set and the original test set. This suggests that the model is not suitable for accurately predicting ratings for movies with low user ratings. The model was trained on the full dataset, which primarily includes popular movies, resulting in difficulties when forecasting ratings for less popular or ”unusual” movies due to limited user ratings available for those movies.

The observed the pattern in the figure is not indicative of a direct relationship between RMSE and different choices of k. Instead, this pattern arises from the varying data splitting schemes generated by different random states during cross-validation. Additionally, the small difference in RMSE values indicates that if the entire dataset is consistently partitioned in the same manner, the relevant curve would essentially be a horizontal line.



High variance movie trimming Minimum k for RMSE is 12, average RMSE is 1.569 After applying high variance movie trimming, the testing dataset primarily consists of movies that are contentious in nature. In this context, contentiousness indicates that these movies are somewhat popular, with a substantial number of ratings, but the ratings provided by different individuals exhibit significant disagreement, resulting in a large variance. Consequently, predicting ratings for these films becomes challenging. This difficulty in prediction is reflected in the erratic patterns observed in the plot.


High variance movie trimming Average RMSE against k
Average RMSE
1.55
1.56
1.57
1.58
k
0
10
20
30
40
50


True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
AUC_MF: 0.7895110308265652, Threshold_MF=2.5
AUC_MF: 0.7792081985339709, Threshold_MF=3.0
AUC_MF: 0.783013975269648, Threshold_MF=3.5
AUC_MF: 0.781187422517484, Threshold_MF=4.0
Chance

7. Naive collaborative filtering

In this segment of the project, we aim to develop a basic collaborative filter in order to forecast the ratings of movies in the original dataset. This filter's predictive mechanism is structured to return the user's average rating as the predicted rating for any given item.

With the predictive function of the basic collaborative filter established, our next step is to create such a filter and gauge its performance through a method known as 10-fold cross-validation. It's crucial to understand that with this basic collaborative filter, there's no concept of model training. In order to train the model, the dataset is divided into 10 combinations of training sets and validation sets. For each combination, the movie ratings in the validation set are predicted using the aforementioned prediction function, with no requirement for model fitting. Subsequently, the Root Mean Square Error (RMSE) is computed for this fold and the process is repeated for each of the 10 folds. Finally, the average RMSE is determined by calculating the mean of the RMSE values obtained from all 10 folds.

Question 11

A To design a naive collaborative filter for predicting movie ratings from the original dataset, we started by defining a simple prediction function. This function is based on returning the average rating of a user as the predicted rating for any given item. This approach is deemed 'naive' as it doesn't require a training process or a model fitting procedure.

In order to evaluate the performance of our naive collaborative filter, we utilized a 10-fold cross-validation approach. This involved splitting our dataset into 10 distinct pairs of training and validation sets. For each pair, we applied our prediction function to estimate the ratings of the movies in the validation set.

Once we had the predicted ratings, we calculated the Root Mean Square Error (RMSE) for each fold. The RMSE is a widely accepted measure of prediction error, providing a sense of how much our predictions deviate, on average, from the actual ratings.

After computing the RMSE for each fold, we then computed the average RMSE across all 10 folds to get a single, comprehensive measure of our filter's predictive accuracy. This average RMSE is **0.935**.

The lower the average RMSE, the better the performance of our collaborative filter. However, it's important to note that this is a naive approach, and more sophisticated models could potentially improve upon this result. Nonetheless, our naive collaborative filter serves as a good starting point for understanding and developing more complex recommendation systems.

B In order to evaluate the performance of our naive collaborative filter on different subsets of the dataset - namely, Popular, Unpopular, and High-Variance subsets - we conducted additional testing using 10-fold cross-validation for each subset.

For the Popular subset:

We designed a naive collaborative filter specifically for the Popular subset, and tested its performance using the same 10-fold cross-validation approach as before. After applying our prediction function to each validation set, we calculated the RMSE for each fold, and then averaged these values across all 10 folds. The average RMSE for the Popular subset is **0.932**.

For the Unpopular subset:

We took a similar approach with the Unpopular subset. After designing a naive collaborative filter and implementing 10-fold cross-validation, we calculated and averaged the RMSE for each fold. The average RMSE for the Unpopular subset is **0.970**.

For the High-Variance subset:

Finally, we repeated the process with the High-Variance subset, designing a naive collaborative filter, implementing 10-fold cross-validation, and calculating the average RMSE across all 10 folds. The average RMSE for the High-Variance subset is **1.383**.

These results demonstrate the performance of our naive collaborative filter on different subsets of our data. By comparing these average RMSE values, we can understand how our model's predictive accuracy varies across different types of movie rating patterns - popular, unpopular, and high-variance. However, as mentioned before, this is a simple approach and more sophisticated models could potentially deliver better results.

8. Performance comparison

In this segment, we aim to evaluate and contrast the effectiveness of the different collaborative filters, previously developed, in their ability to forecast the ratings of the movies in the initial dataset.

Question 12

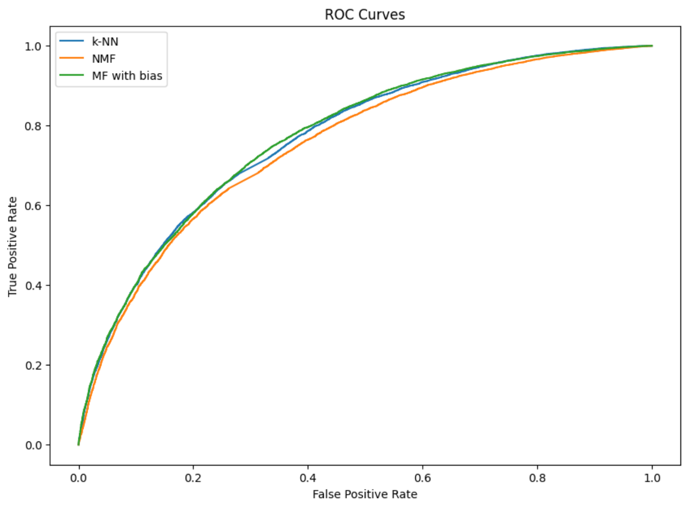
To compare the most effective models across different architectures, we've plotted the best ROC curves (with a threshold of 3) for the k-NN, NMF, and MF with bias-based collaborative filters in a single figure.

In the attached figure, you can clearly see the ROC curves for each of the aforementioned filters. Each ROC curve represents the trade-off between the True Positive Rate (TPR) and False Positive Rate (FPR) for the respective collaborative filter.

Looking at the figure, it can be seen that the MF with basis model seems to perform best, marginally, as evidenced by its ROC curve having the highest area under the curve (AUC). This suggests that this model has the best overall performance in terms of sensitivity and specificity.

However, it's crucial to note that all models show a commendable performance, with each having its strengths and potential areas for improvement. It's also important to remember that the optimal model may vary depending on the specific trade-off between false positives and false negatives that is acceptable for a particular use case.

In conclusion, our visual comparison using ROC curves reveals that while all three models - k-NN, NMF, and MF with bias - are effective, the MF with basis model appears to offer the best performance in predicting movie ratings.



9. Ranking

The recommendation problem can be approached in two primary ways:

1. Prediction: This involves forecasting a rating for a specific user-item pair.
2. Ranking: This strategy recommends the top 'k' items for a specific user.

Up until now, our focus has been on the prediction aspect of the problem. However, we now aim to explore methodologies for addressing the ranking version of this issue. There exist two methods to tackle the ranking problem:

1. Developing algorithms specifically designed to directly handle the ranking problem.
2. Resolving the prediction problem first and subsequently ranking the predictions.

Our chosen strategy is the latter one - solving the ranking problem via first addressing the prediction problem.

By leveraging the predicted ratings, it is feasible to rank all items. The ranking process can be implemented as follows:

1. For each user, calculate its predicted ratings for all items using one of the collaborative filtering techniques. Retain the predicted ratings as a list, denoted as 'L'.

2. Arrange the list 'L' in descending order, such that the item with the highest predicted rating appears first and the item with the lowest predicted rating appears last.
3. From the sorted list, select the top 't' items to recommend to the user.

To evaluate the relevance of the ranked list, a Precision-Recall curve can be employed. We use the following notation: S(t): Represents the set of 't' items recommended to the user. In this recommended set, items without a ground truth rating are excluded (dropped). G: Represents the set of items that the user likes (ground-truth positives).

With this information, the precision and recall can be computed as follows:

$$Precision(t) = |S(t) \cap G|/|S(t)| \tag{1}$$

$$Recall(t) = |S(t) \cap G|/|G| \tag{2}$$

Question 13

Precision and Recall are fundamental measures in information retrieval and they provide insights into the performance of a recommendation system (or any classifier, in general).

- Precision: It measures how many of the items that we recommended were actually liked by the user. In other words, it's the ratio of correctly recommended items to the total items recommended. A high precision means that an algorithm returned substantially more relevant results than irrelevant ones, i.e., most of the items we recommend are liked by the user.

- Recall: It measures how many of the liked items were actually recommended. In other words, it's the ratio of correctly recommended items to all items that should have been recommended. A high recall means that an algorithm returned most of the relevant results, i.e., out of all the items the user likes, we managed to recommend a good proportion of them.

To summarize:

- If your recommendation system has a high precision but low recall, that means it recommends a few items and most of them are liked by the user, but it fails to recommend a lot of other items the user would have liked. - If your recommendation system has high recall but low precision, it means it recommends many items and the user likes many of them, but it also recommends many items that the user doesn't like.

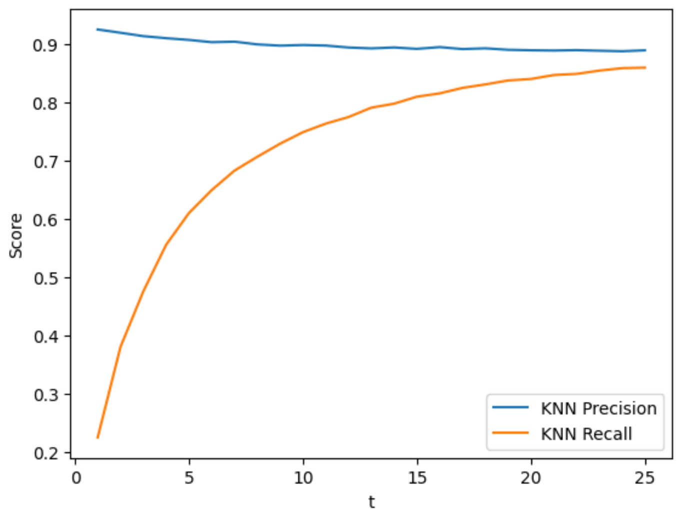
In a practical scenario, we need a balance between precision and recall. This balance can be obtained by using measures such as the F1 score, which is the harmonic mean of precision and recall, or by using precision at k (P@k) and average precision, which focus on the quality of the top-k recommendations, which is often what matters in a recommendation system.

Question 14

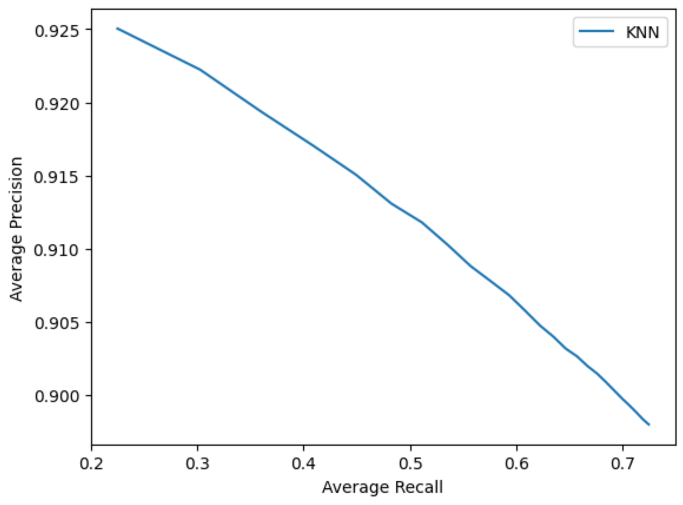
A To compare the precision-recall metrics for the different models, we performed an analysis for each of the three architectures, namely k-NN, NMF and SVD, using the best 'k' value determined from the previous sections. For this, we swept 't' from 1 to 25 in increments of 1.

For k-NN:

The first set of plots illustrate the average precision (Y-axis) against 't' (X-axis), and the average recall (Y-axis) against 't' (X-axis) using predictions from the k-NN model. From observing the shape of these plots, it is apparent that precision quantifies the proportion of accurate recommendations (items enjoyed by the user) out of all the suggestions made for a user. As 't' increases, precision is observed to decrease. This is intuitively understandable because if the recommendation system is tasked to recommend a single movie to a viewer, it is highly likely to choose a movie with the top predicted rating, thus intersecting with the user's preferred (or ground-truth positive) films. Recall quantifies the proportion of accurately recommended items from the collection of items favored by the user. Consequently, as 't' expands and more items are suggested, the likelihood of more items from the user's preference list being recommended also escalates. This is why an increase in 't' corresponds to an increase in recall. This can be further clarified by examining the mathematical expression for recall: since the denominator, —G—, represents the count of movies the user enjoys, which is a fixed value, an increase in 't' causes the intersection of S(t) and G to enlarge, leading to a consistent rise in recall. Moreover, the early stages of recall demonstrate a significant enhancement. For instance, if —G— were to be 10, then for t less than 10, recall would indeed be low, as less than 10 items are being recommended. Hence, for smaller 't' values, we observe a steep ascent in recall. Once 't' surpasses —G—, we start noticing a decrease in the rate of recall increase.

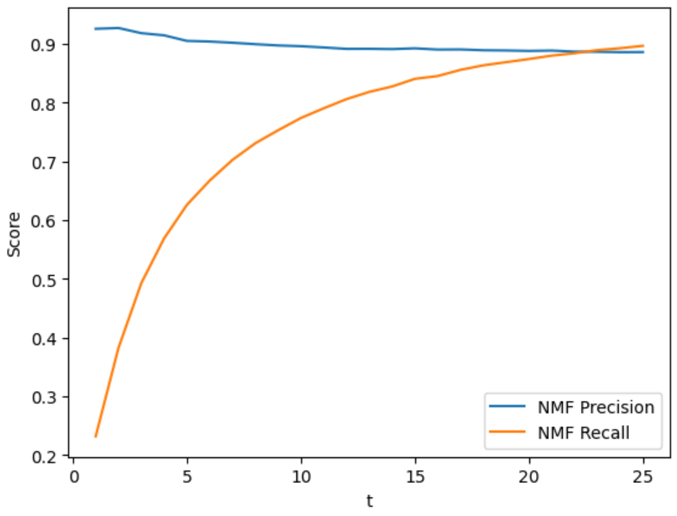


The second plot provides a comparison of the average precision (Y-axis) against the average recall (X-axis) for this model. Observing the shape of this plot, we can conclude that precision and recall share an inverse correlation, demonstrating a push-pull dynamic. The reasoning behind this is fairly straightforward: precision exhibits a negative relationship with 't', while recall presents a positive one. Consequently, 't' serves as a mediator between precision and recall, reflecting the curve in the final subplot as an underlying variable. Thus, a balance must be struck, as it is not feasible to optimize both precision and recall concurrently..

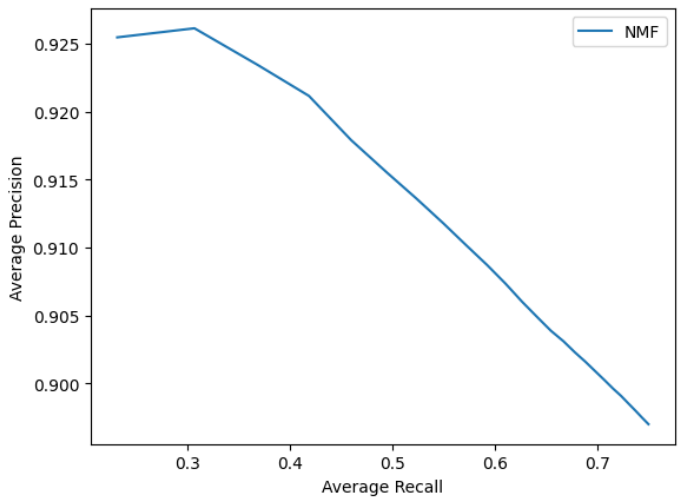


For NMF:

Similarly, for the [second architecture], the first set of plots illustrate the average precision and recall against 't', using predictions from this model. The shape of these plots suggests that precision evaluates the ratio of accurate recommendations (items favored by the user) out of the total number of recommendations presented to the user. As 't' increases, a downward trend in precision is observed. This is intuitively comprehensible as if the recommendation system is tasked to recommend a single movie to the viewer, it's most likely to choose the one with the highest predicted rating, aligning with the user's preferences or the ground-truth positives. However, as the quantity of recommended movies expands, for instance, to 30, the 30th predicted movie may not necessarily be one that the user enjoys, despite its predicted rating being relatively close to the threshold of 3. Consequently, as 't' enlarges, it becomes increasingly challenging for all 't' movies to coincide with the user's preferences. Recall assesses the proportion of accurately recommended items from the user's preference list. As we expand 't' and propose more items, the possibility of recommending a greater number of items liked by the user also amplifies. This is why we see recall augmenting as 't' grows. The mathematical formula for recall can provide a simple explanation: as the denominator, —G—, indicates the count of movies the user enjoys - a constant, the intersection of S(t) and G enlarges with increasing 't', which results in a steady growth in recall. There's also a significant rise in recall during its initial phase.

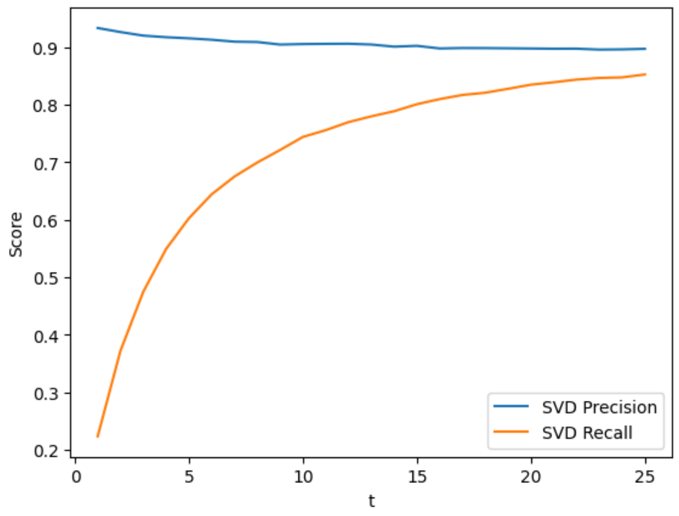


The comparison of the average precision against average recall for the NMF model presents a shape that precision and recall exhibit a reverse correlation, indicating an inverse association. The explanation for this is straightforward: precision and 't' have a negative correlation, while recall and 't' share a positive one. Hence, 't' acts as a connector between precision and recall and represents the curve in the final subplot as an underlying intermediate variable. This implies a trade-off; it is impracticable to maximize both precision and recall at the same time..

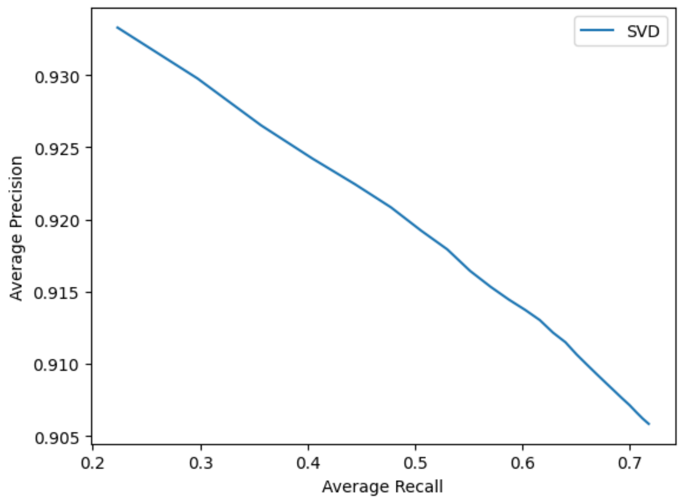


For SVD:

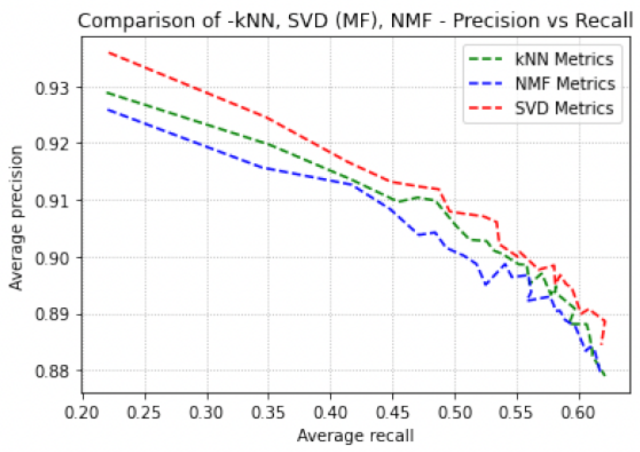
Lastly, for the [third architecture], the initial set of plots depicts the average precision and recall against 't', using predictions from this model. The shape of these plots leads us to observe the same observations as seen above however the increase in recall isnt as sharp as it is for the NMF model.



The comparison plot of the average precision against average recall for the SVD model shows that precision and recall are inversely related and have a negative correlation. This can be explained by the fact that precision and the parameter t have an inverse relationship, while recall and t have a positive relationship. The parameter t acts as a bridge between precision and recall, appearing as a latent variable in the last subplot. Consequently, there is a trade-off between precision and recall, and it is not possible to maximize both metrics simultaneously.



B The precision-recall plots for all three models reveal an inverse interplay between average precision and recall. Nevertheless, the MF with bias outperforms the others, followed by kNN and then NMF. This suggests that a slight decline in average precision allows for an increase in average recall for MF with bias. Given that precision and recall are distinct measures of predictive accuracy, the goal is to get them as close to one as possible. To evaluate the relevance of the recommendation lists generated by varying collaborative filters, one could fix either precision or recall, and then compare the other measure across different models under the same fixed value. Just as with the AUC for the ROC curve, the area under the precision-recall curve can serve as a singular measure to compare the performances of different recommendation systems. The aforementioned precision-recall curves are depicted above. The precision-recall curves in the Figure indicate that the MF with bias-based collaborative filter delivers the most accurate recommendation list among all three systems, with the NMF-based collaborative filter ranking last. This observation aligns with the conclusions derived from the ROC curves. Similar logic can be applied here as well for this outcome.


Comparison of -kNN, SVD (MF), NMF - Precision vs Recall
kNN Metrics
NMF Metrics
SVD Metrics
Average precision
0.93
0.92
0.91
0.90
0.89
0.88
Average recall
0.20
0.25
0.30
0.35
0.40
0.45
0.50
0.55
0.60