EE219: Large-Scale Data Mining: Models and Algorithms
Winter 2018

Project 3: Collaborative Filtering

Akshya ARUNACHALAM (UID: 904-943-191)
Zhi Ming CHUA (UID: 805-068-401)
Ashwin Kumar KANNAN (UID: 605-035-204)
Vijay RAVI (UID: 805-033-666)

February 23, 2018

Contents

1 Introduction

Recommender systems have become pivotal in electronic and business transactions as they allow potential customers' interests to be inferred using feedback from current users. There are generally two types of models for recommender systems:

1. Collaborative Filtering: Utilizes user-item interactions, e.g. ratings
2. Content-Based: Uses attribute information about the users and items, e.g. textual profiles, relevant keywords

In this project, ratings of movies by various users are predicted using **collaborative filtering** recommender systems.

2 Collaborative Filtering Models

There is a huge challenge in working with ratings matrices due to their sparsity as only a small portion of the large collection of available movies have been viewed by each user, resulting in most of the ratings being unspecified.

Collaborative filtering methods allow mutual information between users and items to be exploited. The inter-user correlations and the inter-item correlations would allow unspecified ratings to be inferred.

In this project, two types of collaborative filtering methods are investigated:

1. Neighborhood-Based
2. Model-Based

3 MovieLens Dataset

The MovieLens dataset is used in this project, where the recommender systems predict the ratings of movies by each user.

The ratings matrix is denoted by R, which is an $m \times n$ matrix. The rows correspond to users and columns correspond to movies, hence there are m users and n movies. The (i, j) entry of R is the rating of user i for movie j and is denoted by r_{ij}.

Before building the recommender systems, the dataset is first analyzed in this section.

3.1 Sparsity

The *sparsity* of the movie rating dataset is defined as follows.

$$\text{Sparsity} = \frac{\text{Total number of available ratings}}{\text{Total number of possible ratings}} \tag{1}$$

There are 671 users and 9066 movies, giving a total of $671 \times 9066 = 6{,}083{,}286$ possible ratings. Since the number of ratings given is 100,004, the sparsity is $100{,}004 \div 6{,}083{,}286 = 0.01644$.

3.2 Frequency of Rating Values

The frequency of rating values is visualized by binning them into intervals of width 0.5.

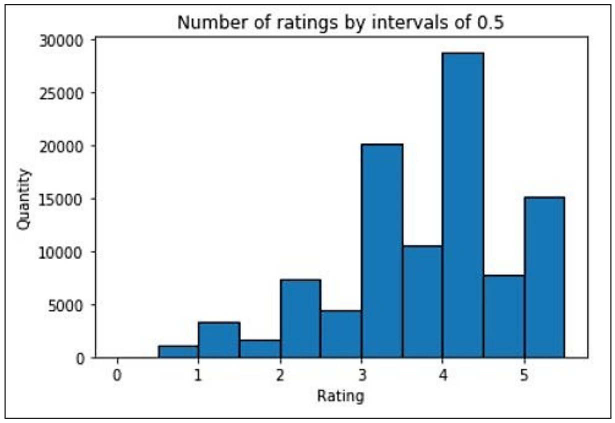

Figure 1: Frequency of Rating Values

The distribution is left-skewed and users tend to give rating values that are of whole numbers.

3.3 Distribution of Ratings Among Movies

The distribution of ratings among movies is plotted in Figure 2.

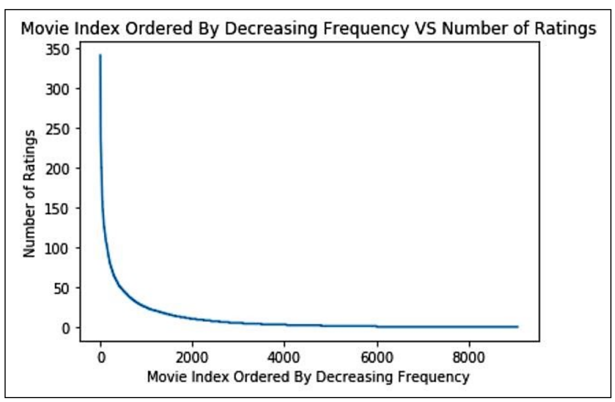

Figure 2: Ratings Distribution

It can be observed that only a small number of movies received a large number of ratings. Conversely, many movies received only few ratings. This may be undesirable for the recommendation process as there are few data points for a large amount of movies.

3.4 Distribution of Ratings Among Users

The distribution of ratings among users is plotted in Figure 3.

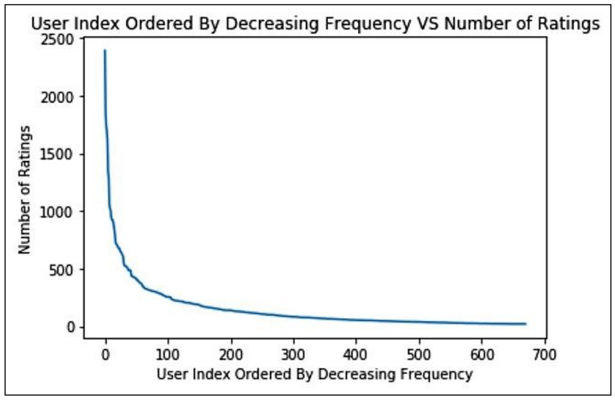

Figure 3: Ratings Distribution

3.5 Variance of Rating Values By Movie

The variance of the rating values received by each movie is computed and the distribution is shown in Figure 4.

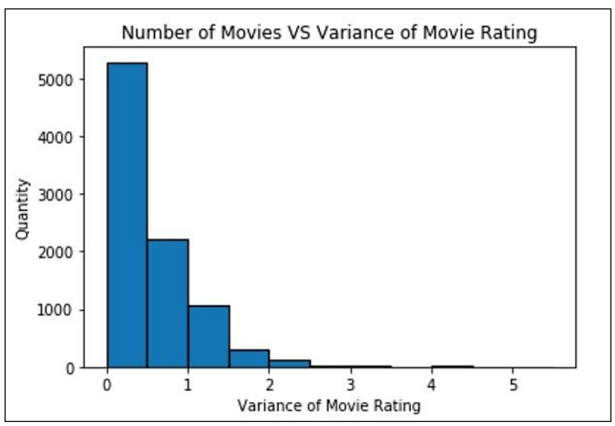

Figure 4: Ratings Distribution

The distribution shows that the number of movies decreases with variance of rating values. This suggests that there are many movies that receive few ratings of similar values.

4 Neighborhood-Based Collaborative Filtering: User-Based

Neighborhood-based methods use either inter-user correlations or inter-item correlations for the prediction process. In this project, user-based collaborative filtering is implemented, with the basic idea that similar users have similar ratings on the same item.

In this approach, the *neighborhood* of a user needs to first be defined. This can be determined by computing the user's similarity to all the other users. Hence, a similarity function needs to be defined between the ratings specified by user. The Pearson-correlation coefficient is used in this project to measure similarity between users.

4.1 Pearson-Correlation Coefficient

The Pearson-correlation coefficient between users u and v is denoted by $\text{Pearson}(u, v)$, which is defined by Equation (2).

$$\text{Pearson}(u, v) = \frac{\sum_{k \in \mathcal{I}_u \cap \mathcal{I}_v} (r_{uk} - \mu_u)(r_{vk} - \mu_v)}{\sqrt{\sum_{k \in \mathcal{I}_u \cap \mathcal{I}_v} (r_{uk} - \mu_u)^2} \sqrt{\sum_{k \in \mathcal{I}_u \cap \mathcal{I}_v} (r_{vk} - \mu_v)^2}} \tag{2}$$

where $\mathcal{I}_i$ is the set of item indices for which ratings have been specified by user i for $i = u, v$, μ_u is the mean rating for user u computed using the specified ratings, and r_{ik} is the rating of user i for item k for $i = u, v$.

Hence, μ_u can be formulated as follows.

$$\mu_u = \frac{\sum_{k \in \mathcal{I}_u} r_{uk}}{|\mathcal{I}_u|} \tag{3}$$

$\mathcal{I}_u \cap \mathcal{I}_v$ denotes the set of item indices for which ratings have been specified by **both** users u and v. Since R is a sparse matrix, $\mathcal{I}_u \cap \mathcal{I}_v$ can be null, i.e. it is possible that there are no items that have ratings specified by both users.

4.2 k-Nearest Neighbors (k-NN) Collaborative Filter

With the similarity between users defined, the neighborhood of a user can now be defined as well. The k-nearest neighbors (k-NN) of a user u is the set of k users with the highest similarity, in this case, the Pearson-correlation coefficient. The k-NN of user u is denoted by $\mathcal{P}_u$.

4.2.1 Prediction Function

The prediction function for the user-based neighborhood model is defined by Equation (4).

$$\hat{r}_{uj} = \mu_u + \frac{\sum_{v \in \mathcal{P}_u} \text{Pearson}(u, v))(r_{vj} - \mu_v)}{\sum_{v \in \mathcal{P}_u} |\text{Pearson}(u, v)|} \tag{4}$$

where $\hat{r}_{uj}$ is the predicted rating of user u for item j.

The mean-centering of the raw ratings $(r_{vj} - \mu_v)$ helps to reduce the effect of a bias by only taking into account the deviations from the mean. This is because different users have different baselines and hence may consistently rate items highly or poorly. In Equation (4), the first term on the right-hand side is the "center" rating for user u and the second term gives the predicted deviation from this "center".

4.2.2 Performance - Whole Test Set

The performance of the k-NN collaborative filter is tested via 10-fold cross validation. The number of neighbors k is swept from 2 to 100 with a step size of 2, and for each k, the average root-mean-square error (RMSE) and average mean absolute error (MAE) are computed. The average RMSE and average MAE against k are plotted in Figures 5 and 6, respectively.

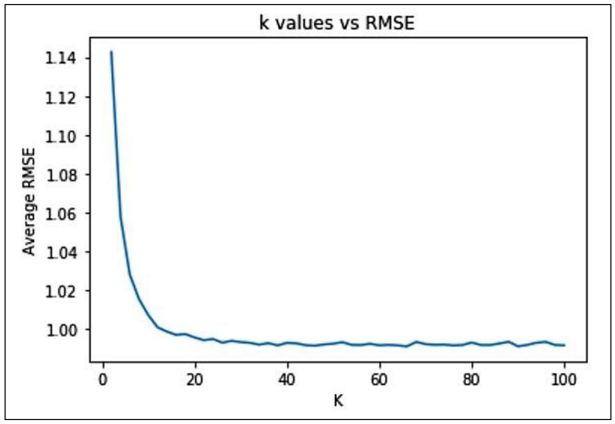

Figure 5: Average RMSE

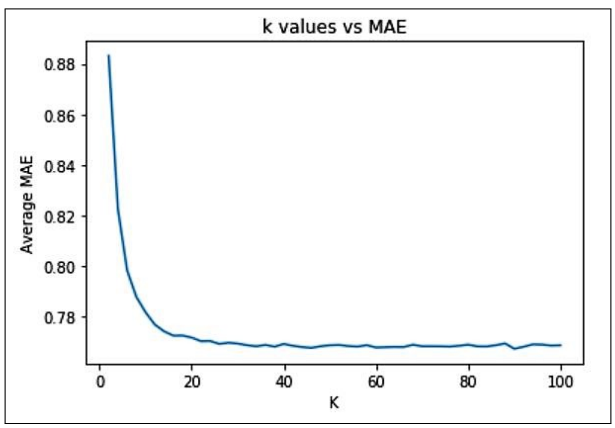

Figure 6: Average MAE

The average RMSE and average MAE converge to a steady-state value at around $k = 30$. At this k value, the steady-state values of the average RMSE and average MAE are 1.1374 and 0.8822, respectively.

In the subsequent sections, the performance of the k-NN collaborative filter is analyzed using trimmed test sets.

4.2.3 Performance - Trimmed Test Set: Popular Movies

The test set is trimmed to contain only movies that received more than 2 ratings.

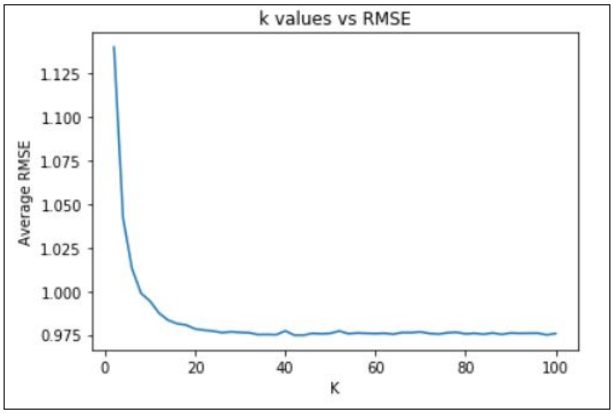

Figure 7: Average RMSE

As seen from the plot, the average RMSE converges to a steady-state value for a large k. This is probably because the test set consists of movies that are rated by relatively more users, hence the accuracy improves significantly with large neighborhoods.

The minimum average RMSE is 0.9749.

4.2.4 Trimmed Test Set: Unpopular Movies

The test set is trimmed to contain only movies that received less than or equal to 2 ratings.

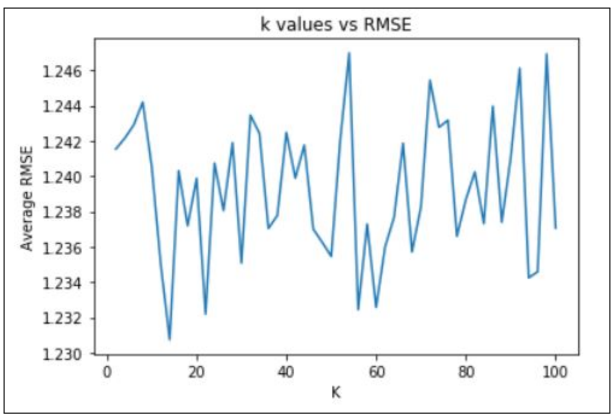

Figure 8: Average RMSE

Regardless of the size of the neighborhood, there are only at most 2 users that have rated any of the movies. Hence, the predicted rating would be decided by the mean rating of the neighbors, leading to a naïve collaborative filter-like prediction. The large fluctuation in average RMSE is due to a small test set, which means that the average RMSE is more likely to be dominated by a large RMSE.

The minimum average RMSE is 1.231.

4.2.5 Trimmed Test Set: High Variance Movies

The test set is trimmed to contain only movies that have variance (of the rating values received) of at least 2, and have received at least 5 ratings.

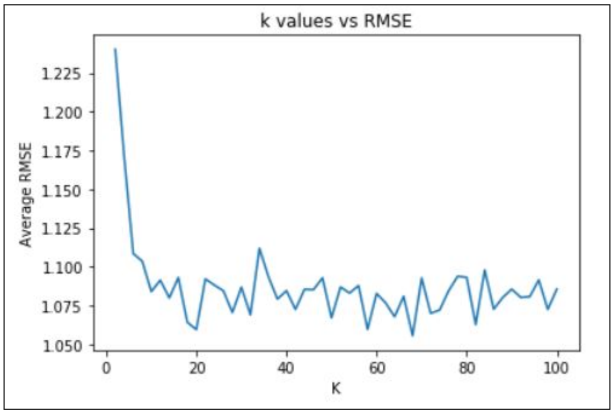

Figure 9: Average RMSE

The average RMSE generally decreases with increasing k. However, it has fluctuations just like for the unpopular movies test set. High variances would correspond to higher average RMSE and due to the small test set, the error is not "distributed" over a large number of

The minimum average RMSE is 1.056.

4.2.6 Performance Evaluation

The performance of the k-NN collaborative filter is evaluated using the receiver operating characteristic (ROC) curve., which plots the true positive rate (TPR) against the false positive rate (FPR).

In the context of recommender systems, it is a measure of the relevance of the items recommended to the user. Since the ratings are in a continuous scale (0 to 5), they need to first be converted to a binary scale, which can be done by applying a threshold to the ratings. If the rating is greater than the threshold value, set it to 1. If the rating is less than the threshold value, set it to 0. They can be interpreted as the user liking or disliking the item, respectively.

The ROC curve is plotted for threshold values $[2.5, 3, 3.5, 4]$ and $k = 30$.

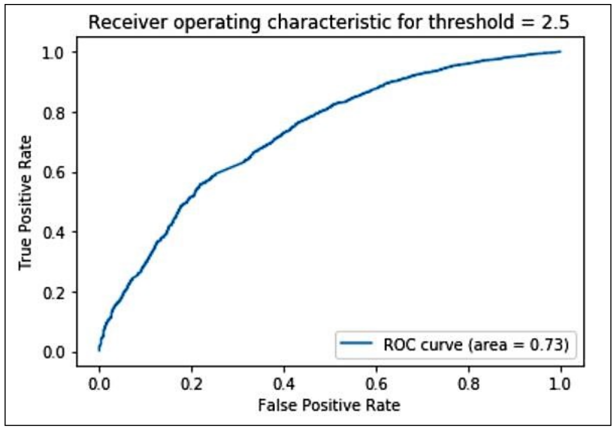

Figure 10: ROC Curve: Threshold = 2.5

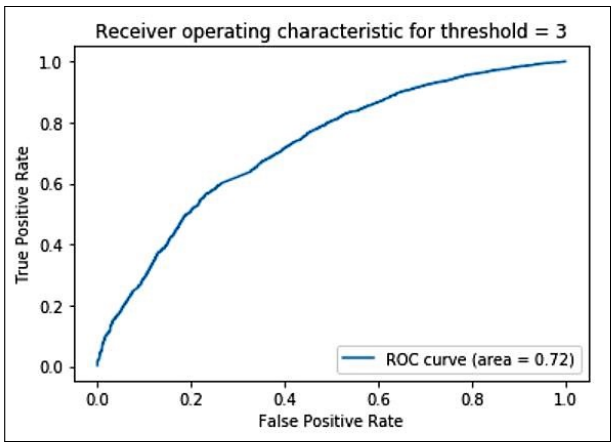

Figure 11: ROC Curve: Threshold = 3

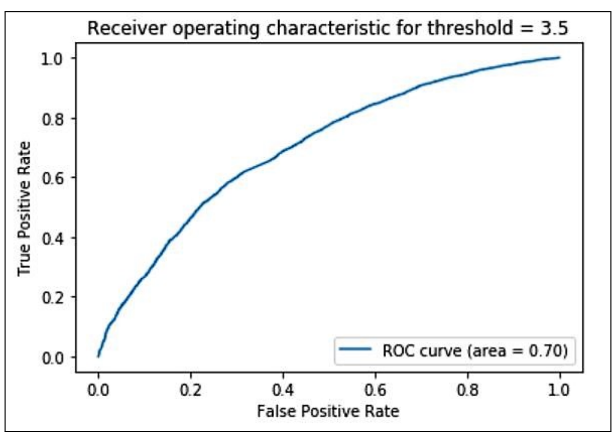

Figure 12: ROC Curve: Threshold = 3.5

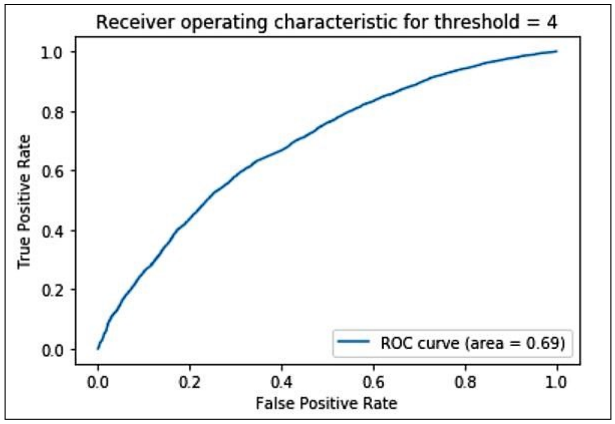

Figure 13: ROC Curve: Threshold = 4

The area under the ROC curve (AUC) for varying threshold value is tabulated below.

Threshold Value	**2.5**	**3**	**3.5**	**4**
AUC	0.7279	0.7224	0.7000	0.6882

Table 1: AUC for Varying Threshold Value

The AUC is the largest when the threshold value is set at 2.5. This is a reasonable result as it is the middle of the rating value range (0 to 5).

5 Model-Based Collaborative Filtering: Latent Factor-Based

Model-based methods rely on machine learning algorithms to develop models that predict users' rating of unrated items. In this project, latent factor-based models are investigated.

Latent factor-based models can be considered as a direct method for matrix completion. Due to the high inter-user and inter-item correlations, significant portions of the rows and columns of the ratings matrix R are correlated. Therefore, the data has built-in redundancies and R can be approximated by a low-rank matrix.

The method of approximating a matrix by a low-rank matrix is called *matrix factorization* (MF). The matrix factorization problem in latent factor-based models can be formulated as an optimization problem given by Equation (5).

$$\min_{U,V} \sum_{i=1}^{m} \sum_{j=1}^{n} (r_{ij} - (UV_{ij}^T))^2 \tag{5}$$

where U and V are matrices of dimension $m \times k$ and $n \times k$, respectively, and k is the number of latent factors.

For sparse matrices, the optimization problem is modified to taken into account only ratings that are known. It is reformulated as follows.

$$\min_{U,V} \sum_{i=1}^{m} \sum_{j=1}^{n} W_{ij}(r_{ij} - (UV_{ij}^T))^2 \tag{6}$$

where

$$W_{ij} = \begin{cases} 1 & \text{if } r_{ij} \text{ is known} \\ 0 & \text{if } r_{ij} \text{ is unknown} \end{cases}$$

To prevent over-fitting, regularization is used and the optimization problem is further modified to as given by Equation (7).

$$\min_{U,V} \sum_{i=1}^{m} \sum_{j=1}^{n} W_{ij}(r_{ij} - (UV_{ij}^T))^2 + \lambda \|U\|_F^2 + \lambda \|V\|_F^2 \tag{7}$$

where $\lambda > 0$ is the regularization parameter, which controls the weight of the regularization term.

There are many variations to the optimization problem formulated in Equation (7) with different objective functions and constraint sets. In this project, two such variations are explored:

- Non-Negative Matrix Factorization (NNMF)
- Matrix Factorization (MF) with Bias

5.1 Non-Negative Matrix Factorization (NNMF)

NNMF may be used for ratings matrices that are non-negative and it has a major advantage in its high level of interpretability. The NNMF can be formulated as an optimization problem by applying non-negative constraints on U and V in Equation (7). This optimization problem is then as follows.

$$\begin{aligned} \min_{U,V} \quad & \sum_{i=1}^{m} \sum_{j=1}^{n} W_{ij}(r_{ij} - (UV_{ij}^T))^2 + \lambda \|U\|_F^2 + \lambda \|V\|_F^2 \\ \text{s.t.} \quad & U \geq 0 \\ & V \geq 0 \end{aligned} \tag{8}$$

The optimization problem in Equation (6) is not convex because U and V are variables and the the optimization problem involves the term $u_{i*}v_{*j}^T$. In fact, the optimization problem is NP-hard and approximate approaches such as stochastic gradient descent (SGD) are often used to solve it.

SGD works by going along the objective function in the opposite direct of the gradient and finally stabilizes at a local minimum. Hence, it is very sensitive to initialization (initial point) and step size. The alternative least-squares (ALS) methods is less sensitive to these parameters and is a more stable algorithm with faster convergence rate. It works by first keeping U fixed then solve for V, then keep V fixed then solve for U. In both stages of the algorithm, a least-squares problem is being solved.

For U fixed, the optimization problem becomes a minimization over only V.

$$\min_{V} \sum_{i=1}^{m} \sum_{j=1}^{n} W_{ij}(r_{ij} - (UV_{ij}^T))^2 \tag{9}$$

5.1.1 Prediction Function

After the optimal U and V are solved for Equation (8), they are used to predict unspecified ratings, as shown in Equation (10).

$$\hat{r}_{ij} = \sum_{s=1}^{k} u_{is} v_{js} \tag{10}$$

where $\hat{r}_{ij}$ denotes the predicted rating of user i for item j.

5.1.2 Whole Test Set

The performance of the NNMF-based collaborative filter is tested via 10-fold cross validation. Again, the number of neighbors k is swept from 2 to 50 with a step size of 2, and for each k, the average RMSE and average MAE are computed. The average RMSE and average MAE against k are plotted in Figures 14 and 15, respectively.

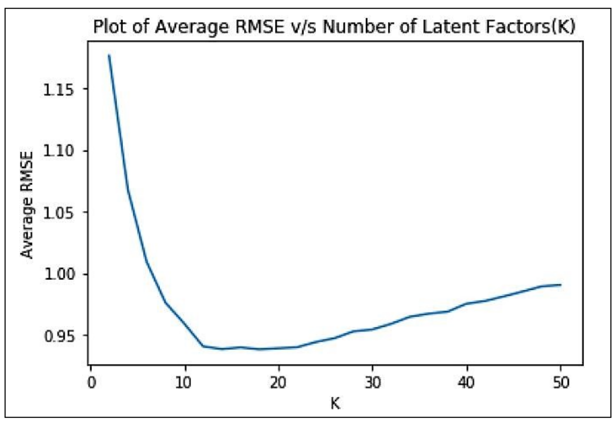

Figure 14: Average RMSE

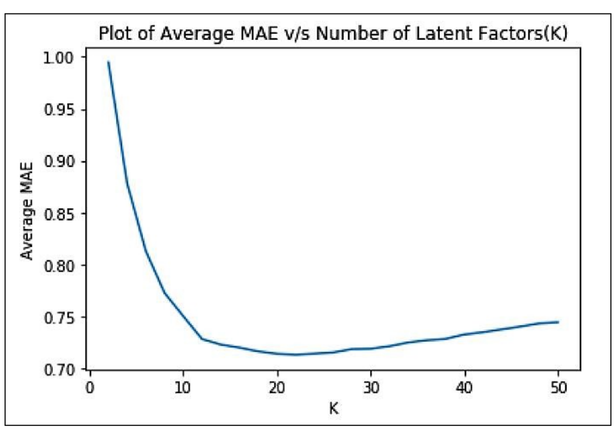

Figure 15: Average MAE

Using the plots above, it can be determined that the optimal number of latent factors is 18, which corresponds to the number of movie genres. For $k = 18$, the average RMSE and average MAE are 0.9383 and 0.7131, respectively.

The performance of the collaborative filter on trimmed test sets is investigated in the subsequent sections.

5.1.3 Trimmed Test Set: Popular Movies

The average RMSE against k is plotted in Figure 16.

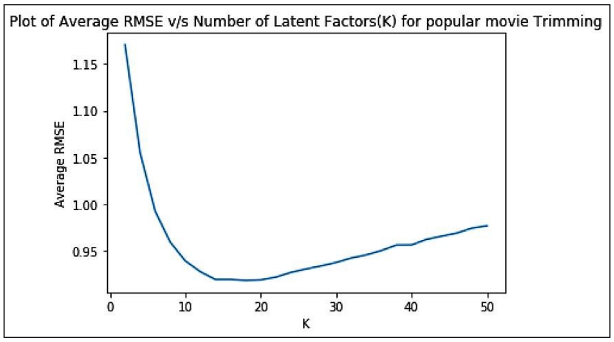

Figure 16: Average RMSE

The minimum average RMSE is 0.9183.

5.1.4 Trimmed Test Set: Unpopular Movies

The average RMSE against k is plotted in Figure 17.

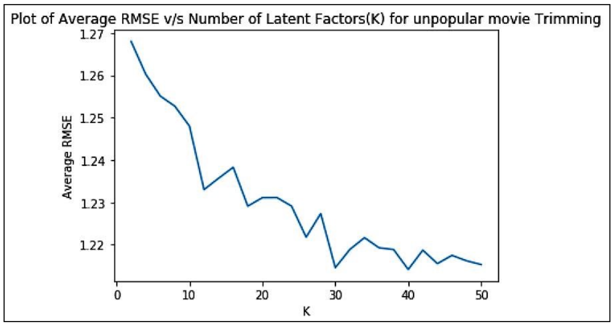

Figure 17: Average RMSE

The minimum average RMSE is 1.214.

5.1.5 Trimmed Test Set: High Variance Movies

The average RMSE against k is plotted in Figure 18.

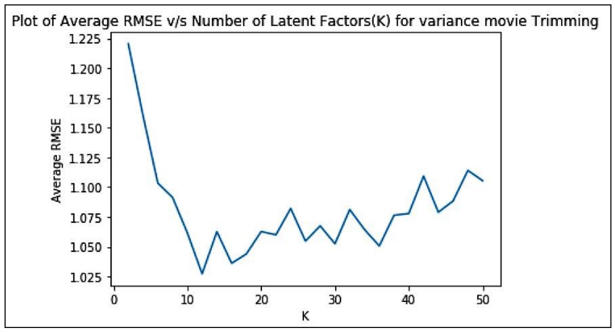

Figure 18: Average RMSE

The minimum average RMSE is 1.027.

5.1.6 Performance Evaluation

The ROC curve is plotted for threshold values $[2.5, 3, 3.5, 4]$ and $k = 18$.

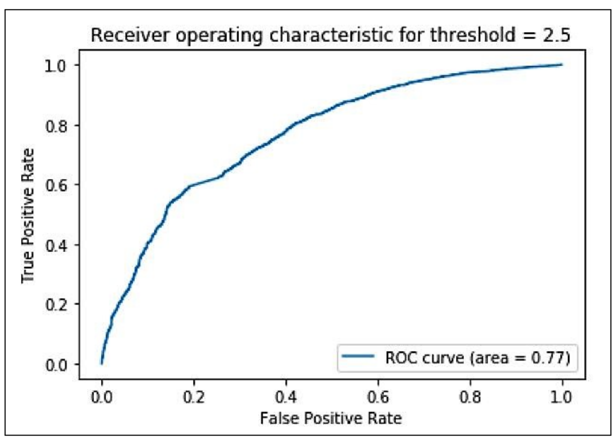

Figure 19: ROC Curve: Threshold = 2.5

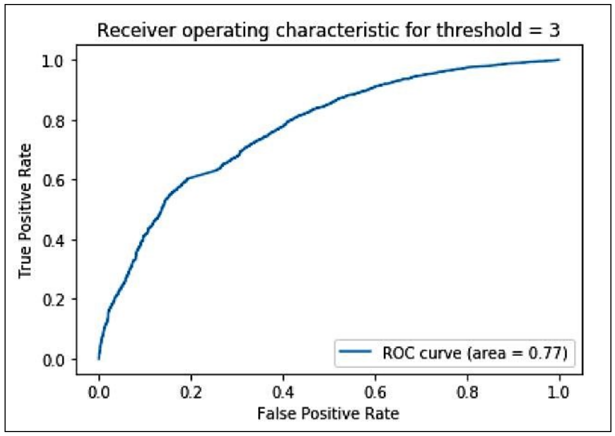

Figure 20: ROC Curve: Threshold = 3

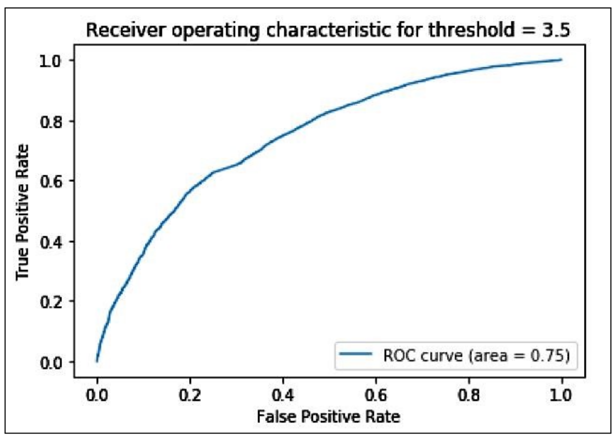

Figure 21: ROC Curve: Threshold = 3.5

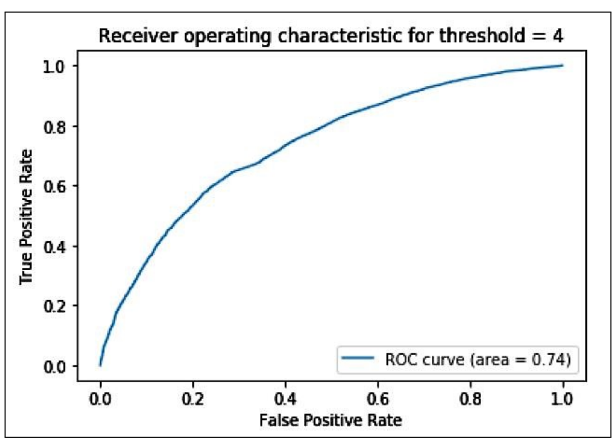

Figure 22: ROC Curve: Threshold = 4

The area under the ROC curve (AUC) for varying threshold value is tabulated below.

Threshold Value	**2.5**	**3**	**3.5**	**4**
AUC	0.7668	0.7698	0.7463	0.7354

Table 2: AUC for Varying Threshold Value

The largest AUC value is obtained when the threshold value is set at 3.

5.1.7 Interpretability of NNMF

NNMF provides a high level of interpretability in understanding user-item interactions. In this section, the connection between the latent factors and the movie genres is explored.

When NNMF is performed, the ratings matrix R is factorized into two factor matrices U and V. U represents the user-latent factors interaction, while V represents the movie-latent factors interaction. The number of latent factors is set to 20 and for each column of V, the genres of the top 10 movies are extracted. A few examples are displayed below.

Action, Adventure, Comedy	Drama, War	Comedy, Drama
Drama	Action, Animation, Children, Comedy	Drama, Mystery, Thriller
Action, Sci-Fi, Thriller	Action, Crime, Drama	Drama, Romance
Comedy, Crime	Action, Crime, Drama	Drama, Mystery, Sci-Fi
Horror	Action, Sci-Fi, Thriller	Comedy, Drama
Crime, Drama	Drama, Thriller	Comedy, Drama
Drama	Crime, Drama	Comedy, Drama
Actiona, Drama, Thriller, War	Drama, Romance	Drama, Mystery, Thriller
Action, Adventure, Comedy, Sci-Fi	Comedy, Romance	Crime, Drama
Action, Mystery, Sci-Fi, Thriller	Drama	Mystery, Thriller

Table 3: Genres of Top 10 Movies

It can be seen that almost all of the top 10 movies in each column belong to a small group of genres. The latent factors are in fact grouping genres that are similar as a single feature. This is an interesting result as genres were not used as part of the training process.

5.2 Matrix Factorization (MF) with Bias

In MF with bias, the objective function in Equation (7) is modified by adding bias terms for each user and item. The optimization formulation then becomes as follows.

$$\min_{U,V,b,c} \sum_{i=1}^{m} \sum_{j=1}^{n} W_{ij}(r_{ij} - (UV_{ij}^T))^2 + \lambda \|U\|_F^2 + \lambda \|V\|_F^2 + \lambda \sum_{u=1}^{m} b_u^2 + \lambda \sum_{i=1}^{n} c_i^2 \tag{11}$$

where b_u is the bias of user u and c_i is the bias of item i.

5.2.1 Prediction Function

After the optimal U, V, b and c are solved for Equation (11), they are used to predict unspecified ratings, as shown in Equation (12).

$$\hat{r}_{ij} = \mu + b_i + c_j + \sum_{s=1}^{k} u_{is} v_{js} \tag{12}$$

where μ is the mean of all ratings.

5.2.2 Whole Test Set

The performance of the MF with bias collaborative filter is tested via 10-fold cross validation. Again, the number of neighbors k is swept from 2 to 50 with a step size of 2, and for each k, the average RMSE and average MAE are computed. The average RMSE and average MAE against k are plotted in Figures 23 and 24, respectively.

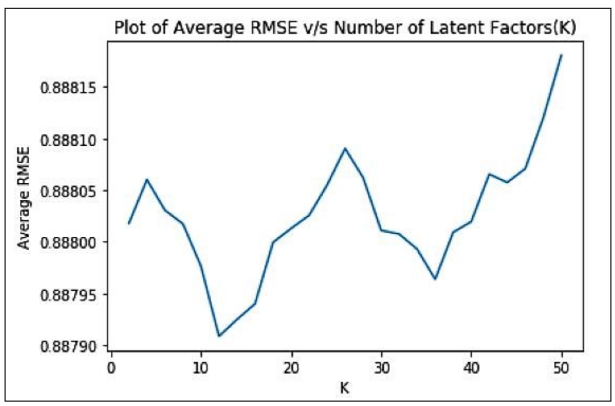

Figure 23: Average RMSE

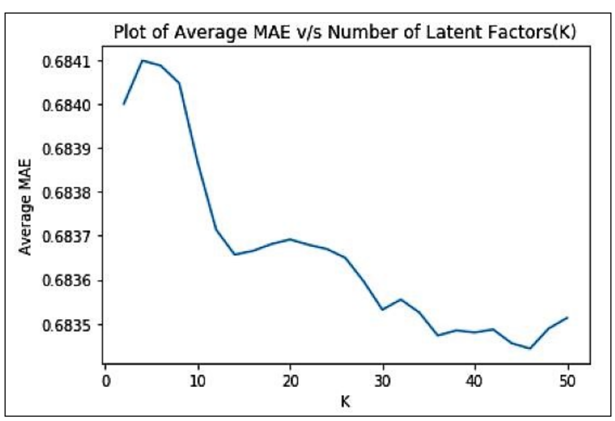

Figure 24: Average MAE

From the plot of the average RMSE above, the optimal number of latent factors is 12. The average RMSE and average MAE are 0.8879 and 0.6834, respectively.

The performance of the collaborative filter on trimmed test sets is investigated in the subsequent sections.

5.2.3 Trimmed Test Set: Popular Movies

The average RMSE against k is plotted in Figure 25.

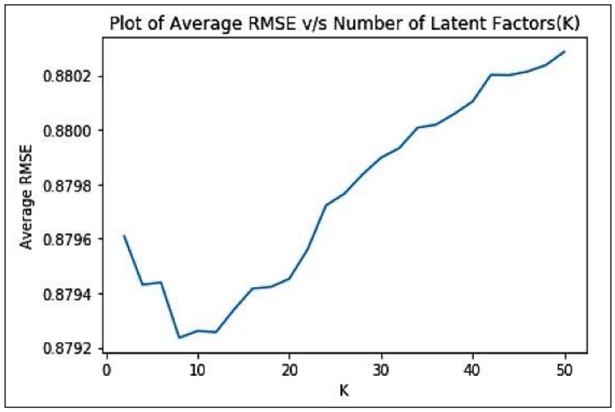

Figure 25: Average RMSE

The minimum average RMSE is 0.8792.

5.2.4 Trimmed Test Set: Unpopular Movies

The average RMSE against k is plotted in Figure 26.

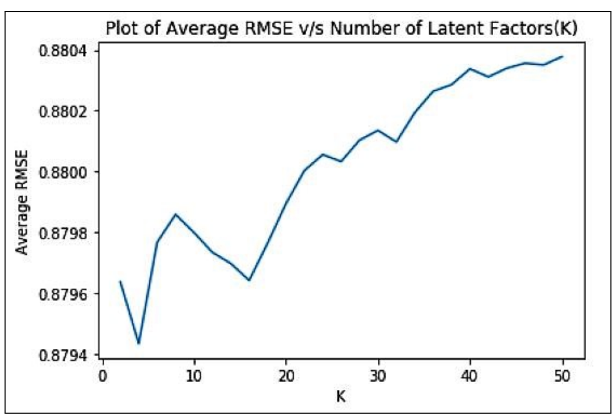

Figure 26: Average RMSE

The minimum average RMSE is 0.8794.

5.2.5 Trimmed Test Set: High Variance Movies

The average RMSE against k is plotted in Figure 27.

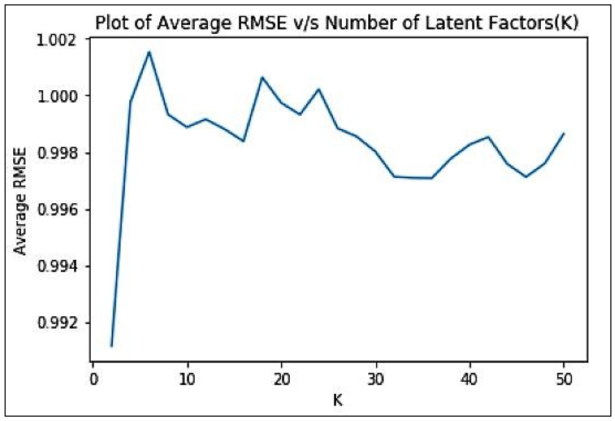

Figure 27: Average RMSE

The minimum average RMSE is 0.9911.

5.2.6 Performance Evaluation

The ROC curve is plotted for threshold values $[2.5, 3, 3.5, 4]$ and $k = 12$.

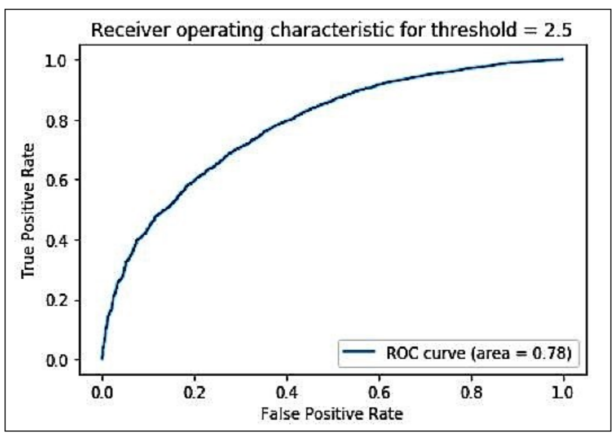

Figure 28: ROC Curve: Threshold = 2.5

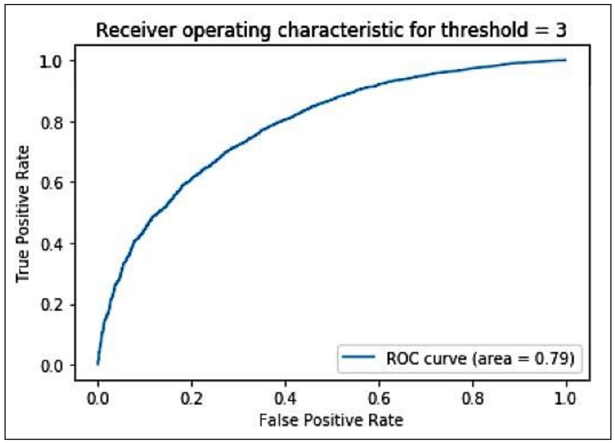

Figure 29: ROC Curve: Threshold = 3

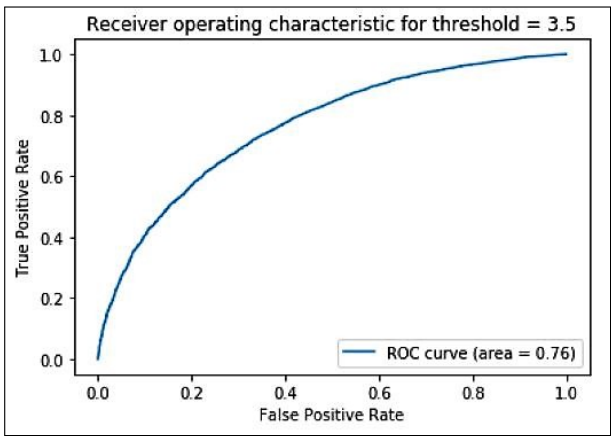

Figure 30: ROC Curve: Threshold = 3.5

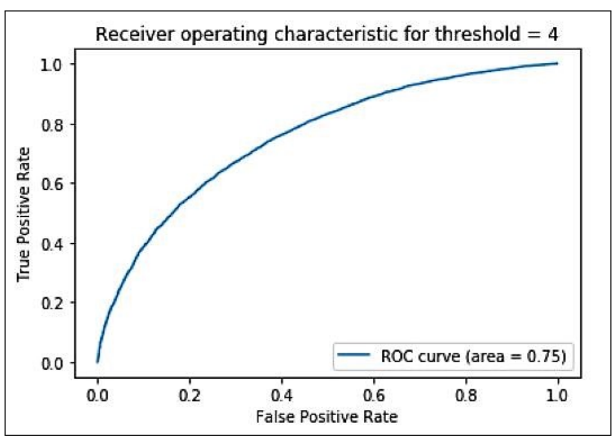

Figure 31: ROC Curve: Threshold = 4

The area under the ROC curve (AUC) for varying threshold value is tabulated below.

Threshold Value	2.5	3	3.5	4
AUC	0.7805	0.7862	0.7637	0.7530

Table 4: AUC for Varying Threshold Value

The largest AUC value is obtained when the threshold value is set at 3.

6 Naïve Collaborative Filtering

A naïve collaborative filter simply returns the mean rating of the user as its predicted rating for an item.

6.1 Prediction Function

The predicted rating of user i for item j, which is denoted by $\hat{r}_{ij}$, is computed as in Equation (13).

$$\hat{r}_{ij} = \mu_i \tag{13}$$

where μ_i is the mean rating of user i.

6.2 Whole Test Set

The performance of the naïve collaborative filter is evaluated using the whole test set. The average RMSE is 0.9554.

6.3 Trimmed Test Set: Popular Movies

When the test set contains only popular movies, the average RMSE is 0.9538.

6.4 Trimmed Test Set: Unpopular Movies

For the unpopular movies test set, the average RMSE is 0.9724.

6.5 Trimmed Test Set: High Variance Movies

When the test set is trimmed to contain only high variance movies, the average RMSE is 1.013.

7 Performance Comparison

In this section, the performance of the k-NN, NNMF and MF with bias collaborative filters in predicting ratings of movies is compared. For the ROC curves plotted, the threshold value is set to 3.

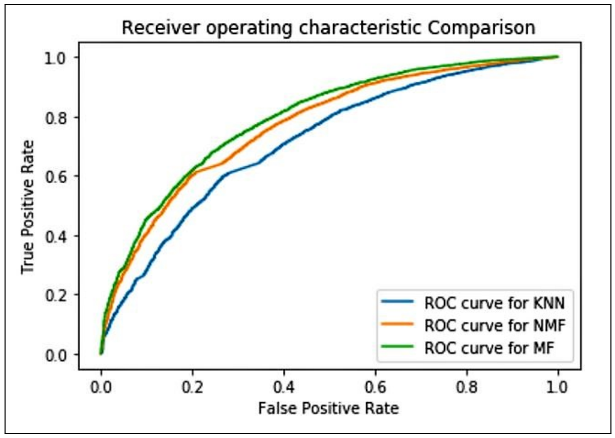

Figure 32: ROC Curve: Threshold = 3

The ROC curve for MF with bias lies above the other two. Hence, MF with bias yields the collaborative filter with the best performance.

8 Ranking

There are two main ways a recommendation problem can be formulated:

1. Prediction Version: Predict the rating value for a user-item pair
2. Ranking Version: Recommend the top k items for a particular user

The prediction version of the problem has been explored in the previous sections using collaborative filtering techniques. In this section, methods to solve the ranking version of the problem will be investigated. There are two approaches to this:

1. Design algorithms for solving the ranking problem directly
2. Solve the prediction version and then rank the predicted rating values

Since the prediction version of the problem has already been solved in the previously, the second approach will be taken to solve the ranking problem.

8.1 Predictions

For each user, the predicted ratings for all the items are predicted using one of the collaborative filtering techniques. The first t items with the highest predicted ratings will then be recommended to the user.

8.2 Performance Evaluation

The precision-recall curve is used to evaluate the relevance of the ranked list. The expressions for precision and recall are by Equations (14) and (15), respectively.

$$\text{Precision}(t) = \frac{|\mathcal{S}(t) \cap \mathcal{G}|}{|\mathcal{S}(t)|} \tag{14}$$

$$\text{Recall}(t) = \frac{|\mathcal{S}(t) \cap \mathcal{G}|}{|\mathcal{G}|} \tag{15}$$

where $\mathcal{S}(t)$ is the recommended set of size t and $\mathcal{G}$ is the set of items liked by the user. Items that do not have a ground truth rating are dropped from the recommended set.

In words, **precision** is the fraction of movies recommended that are liked by the user; **recall** is the fraction of movies liked by the user that are recommended to them.

8.2.1 k-NN

The average precision and average precision for varying t are plotted below.

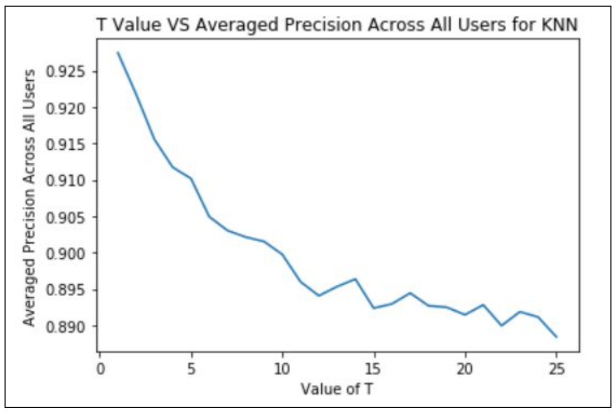

Figure 33: Average Precision Against t

There is general decreasing trend which is convex. This can be interpreted as the recommender system's ability to retrieve a relevant movie decreases with the size of the recommended set, which is expected.

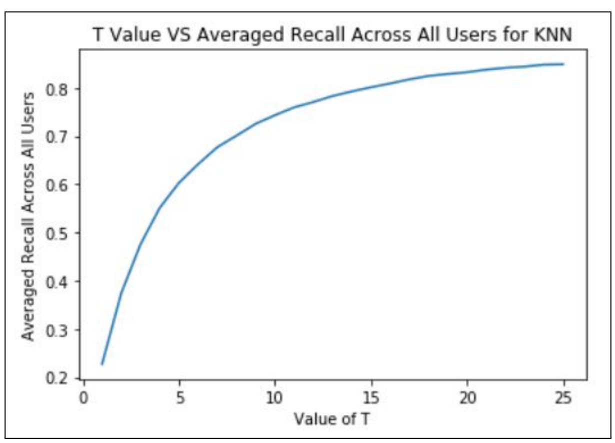

Figure 34: Average Recall Against t

The average recall curve against increasing t is strictly increasing and concave. It has an "inverse" relationship with average precision.

The average precision is plotted against the average recall in Figure 35.

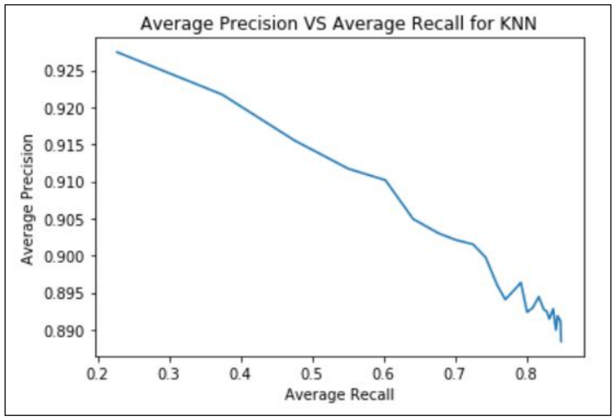

Figure 35: Average Precision Against Average Recall

Average precision generally decreases with average recall. As the average precision against t plot is not monotonic, it is expected that the precision-recall curve is also not monotonic.

8.2.2 NNMF

The average precision and average precision for varying t are plotted below.

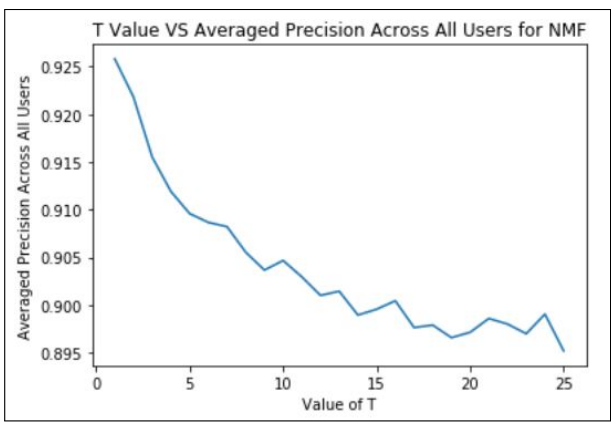

Figure 36: Average Precision Against t

Similar to the results obtained for the k-NN collaborative filter, average precision generally decreases with t.

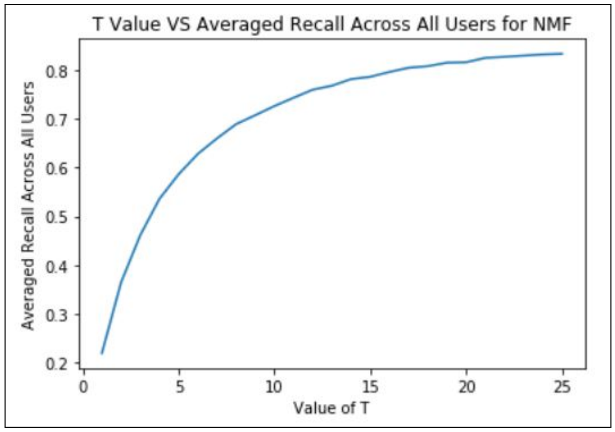

Figure 37: Average Recall Against t

Average recall strictly increases with t and it is a concave function of t.

The average precision-average recall curve is plotted below.

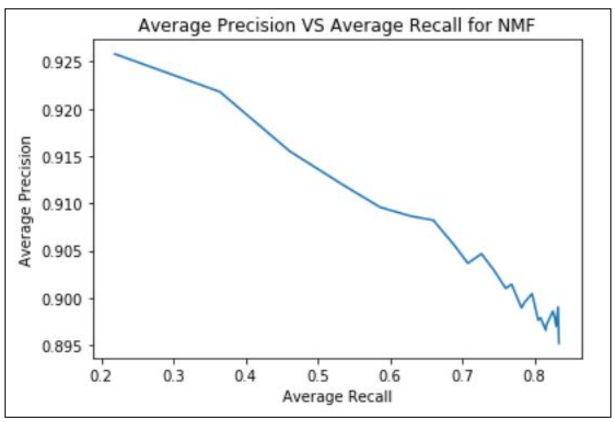

Figure 38: Average Precision Against Average Recall

Precision and recall have a generally inverse relationship.

8.2.3 MF with Bias

The average precision and average precision for varying t are plotted in Figures 39 and 40, respectively.

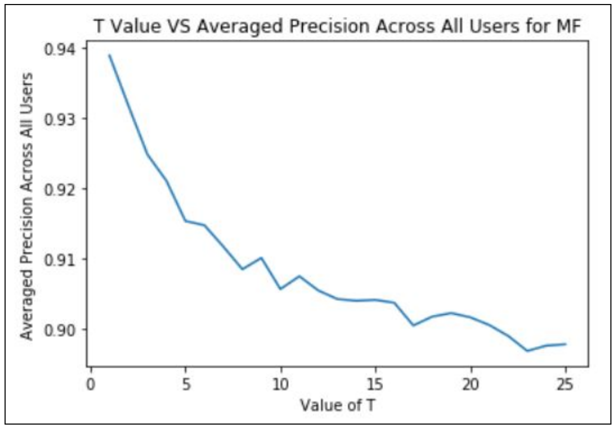

Figure 39: Average Precision Against t

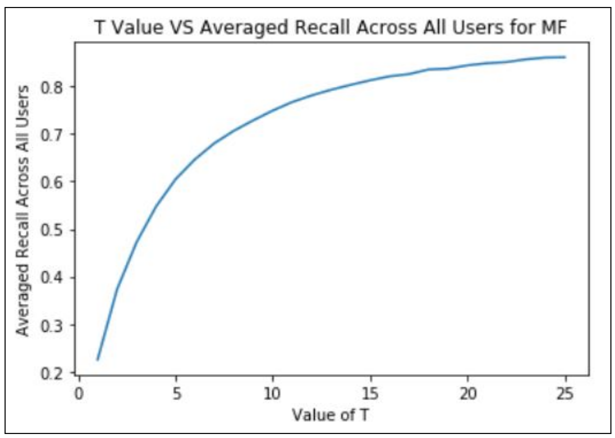

Figure 40: Average Recall Against t

Average precision generally decreases with t whereas average recall strictly increases with t.

The average precision-average recall curve is plotted below.

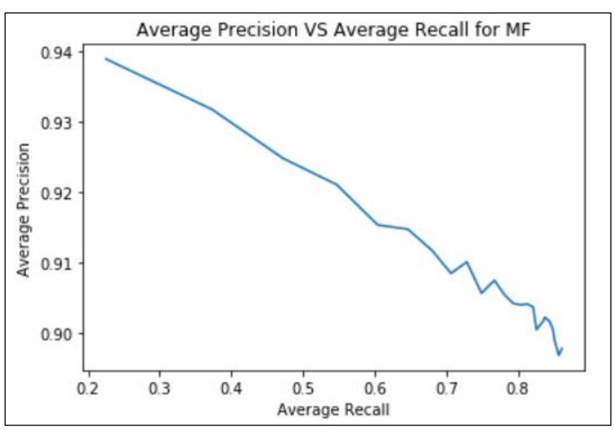

Figure 41: Average Precision Against Average Recall

There is an approximate inverse relationship between average precision and average recall.

8.2.4 Performance Comparison

To compare the performance of the three collaborative filters, the average precision-average recall curves for each of them are shown in the same plot below.

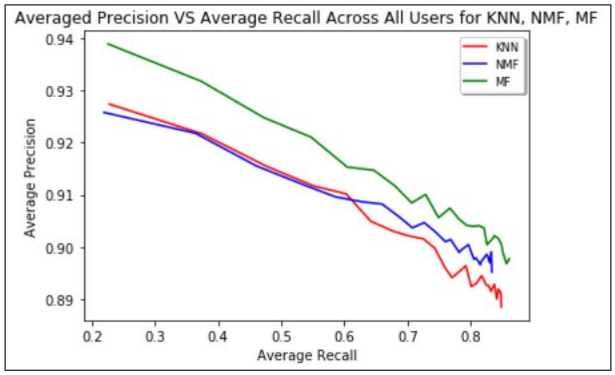

Figure 42: Average Precision Against Average Recall

From the plot, it can be observed that curve for MF with bias lies above the other two. This means that for the same recall value, MF with bias yields the highest precision, i.e. it yields the recommended set of movies that is the most relevant.