Jaideep Adusumelli Kaushik Kompella Sushaanth Srirangapathi

Data Mining Assinment 5

## Question 1

### (a) Explore the data to obtain an understanding of users, movies and how users have rated movies. - what is the overall distribution of ratings?

### - on average, how do users rate movies; what ratings do movies have on average? (you may want to plot the distribution of average ratings for users, movie. Can you show this on a single plot?)

### - how many movies do users rate, and how many ratings do movies get? (consider the distribution of rating counts)

### - how are rating levels distributed, do many people have high/low ratings?

## Solution

In this assignment, we are focused at rating the movies and recommending movies to the users based on their individual preferences, their background and the information of the movies. The data we would be working on contains attributes like – User ID, Movie ID and ratings. The data we have on users is mainly based on the following attributes – age, gender, occupation and zip; and that of movies contains – date of release and movie genre attributes such as Action, Adventure etc.

The rating data has 90,570 user ratings for 1,682 movies as provided by 943 users. Data is already cleaned and has no missing values. Users are represented by a unique number ranging from 1 to 943. Similarly, each movie is represented by a number from 1 to 1,682. The movie ratings range from 1-5 with 5 representing the most liked.

User-rating-averages (average ratings grouped by user) range between 1.489 and 4.929 indicating that user rating levels are subjective. Some users tend to be easy with their ratings while others may be quite the opposite. Most people (702 of 943) have an average rating between 3 and 4.

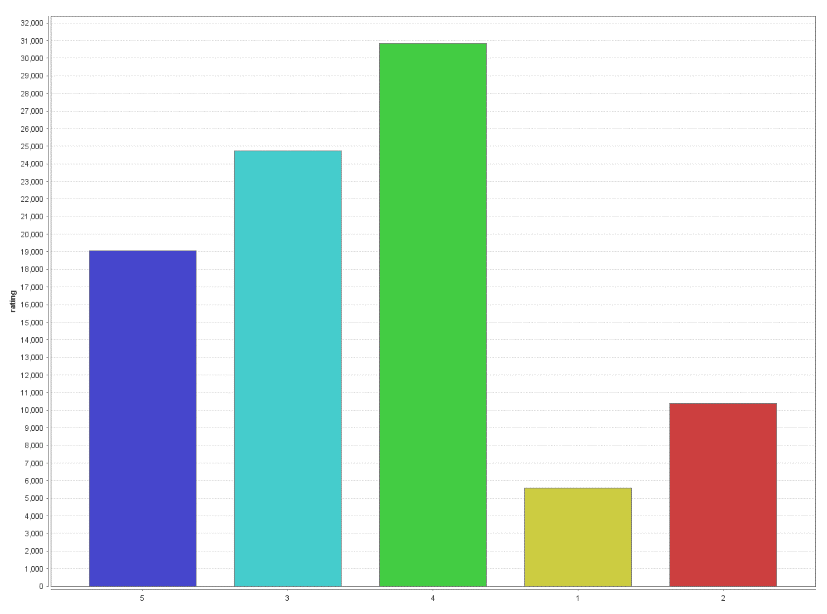


Fig 1. Frequency of each rating as given by all the users.

Here we can see that most of the users have rated “4” followed by “5” then “3” then “2” and then “1”. The average rating given by all the users across all the movies was found to be 3.524. Movie-rating-averages (average ratings grouped by movie) range between 1 and 5 with most movies (1176 of 1682) getting an average rating between 2.5 and 4.1.

From the figure shown below, one can observe that the distribution of Movie-rating-averages has a higher spread compared to that of User-rating-averages which could be explained by the varying preferences of the users. This provide us with an opportunity to predict the kind of movies that a user may like.

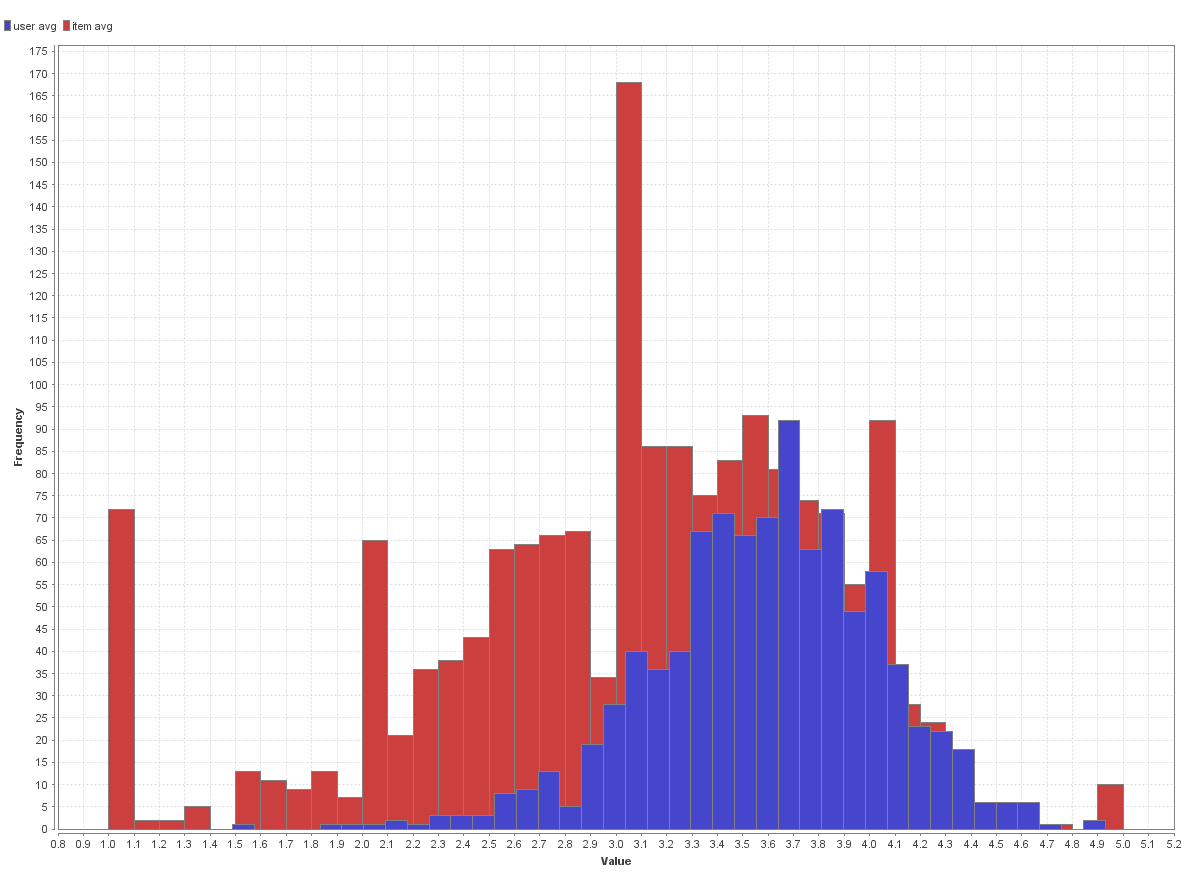


Fig 2. Distribution of Average rating given by users (in red) and Average rating obtained by each movie (in blue).

This shows us that the distribution of a given user’s average rating seems to resemble a normal distribution around the average of 3.6. About 80% of the users have an average rating between 3 and 4. There not many users that always have high or low ratings, which means their ratings will have value in differentiating movies a user tends to like.

The number of movies rated by individual users range between 10 and 727 with most users rating 10 movies. The average number of ratings per user is 96 as shown in Fig 3. Similarly, the number of ratings per movie ranges between 1 and 495, an average of 53.9 ratings per movie and 1 user rating is the most prevalent case. Stats are depicted in Fig 4.

We tried to check for correlation between the number of movie ratings and the rating themselves. We found a medium level correlation value of 0.437. This means that if a movie is perceived to be good quality), then that movie tends to attract more users to rate it (quantity). This correlation must does not imply causation in either direction. Refer to F.1.7 in the appendix.

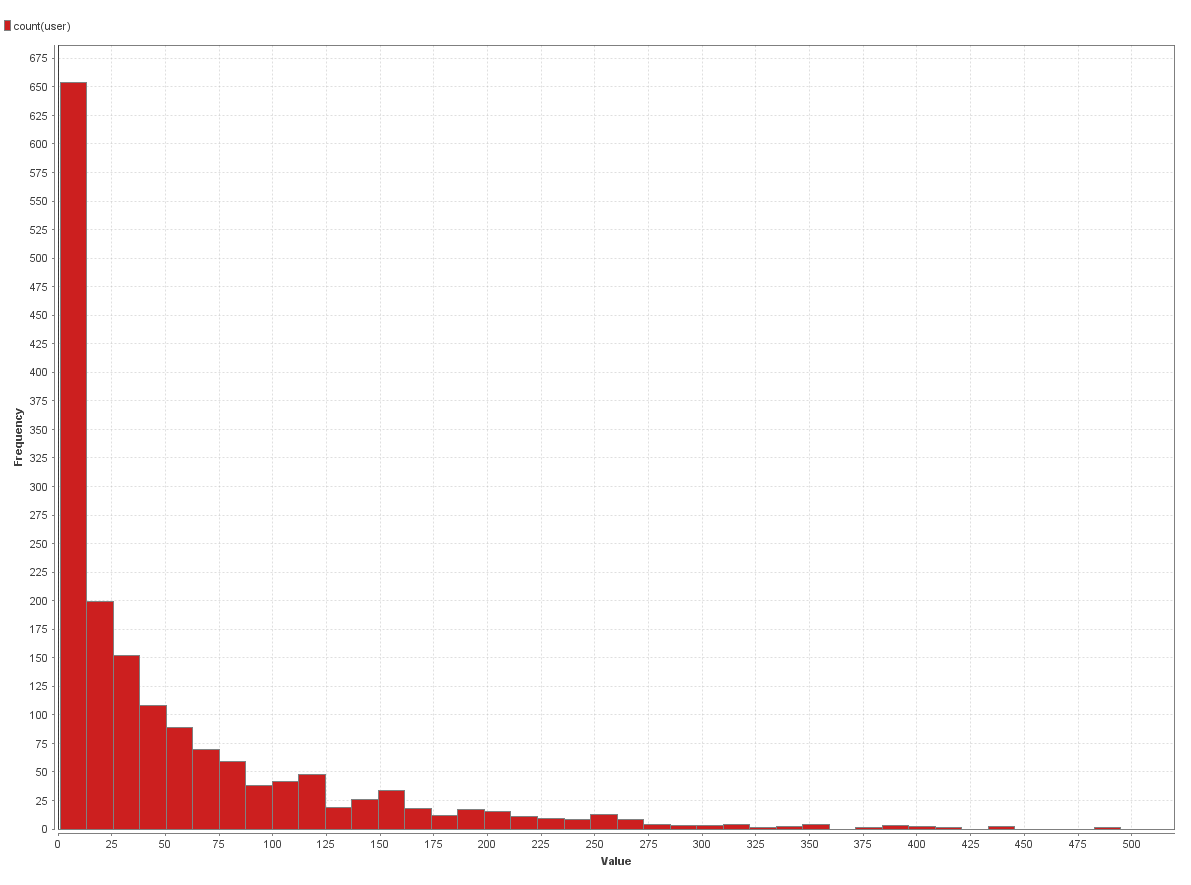


Fig 3. Number of movies rated per user.

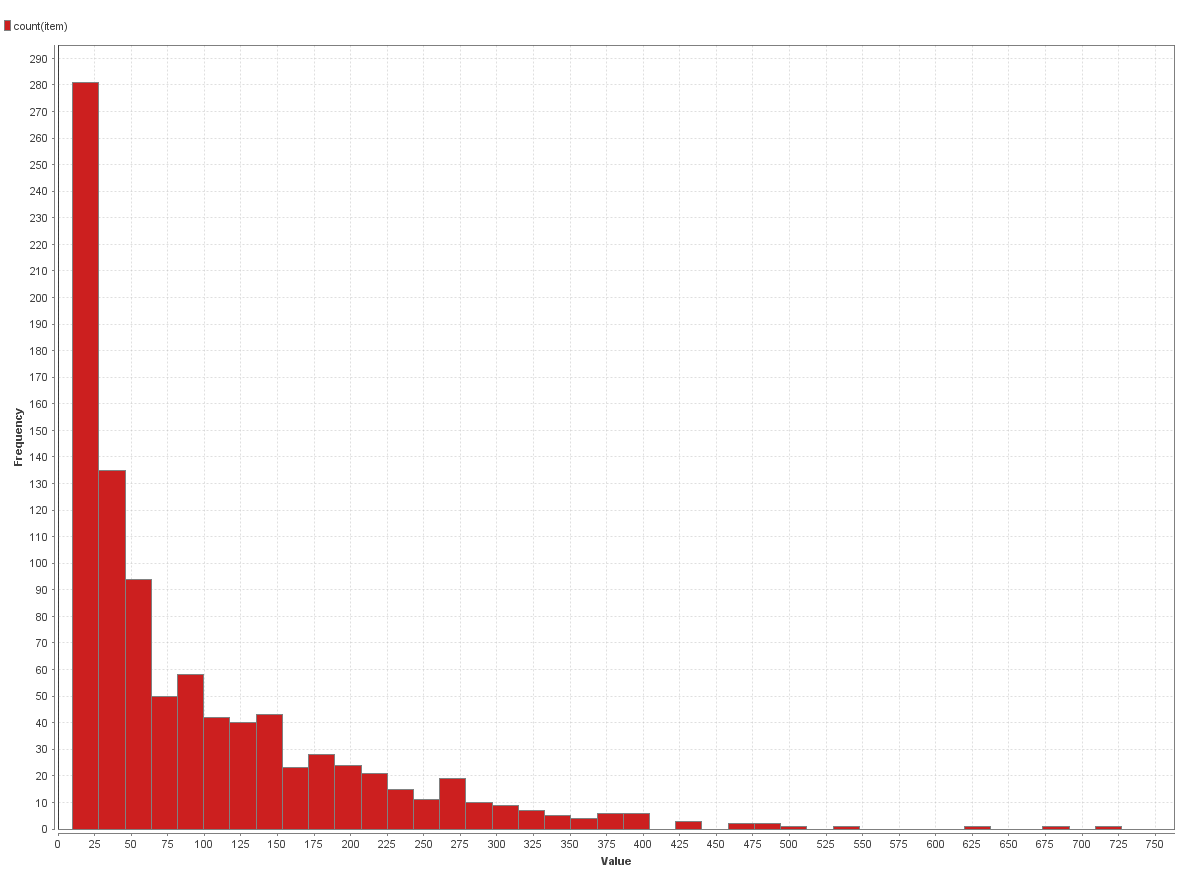


Fig 4. Number of users rating each movie.

### (b) Consider the movie attributes in the file u\_item.csv and the user attributes in the file\_user.csv.

### How do ratings differ by genre, by user age (group), gender and occupation? You can analyze his in various ways – please describe what you do and any interesting findings.

## Solution

We observed that the ratings had some indirect relationships with the background of the users and the genre of the movie as well.

Relationship with user data:

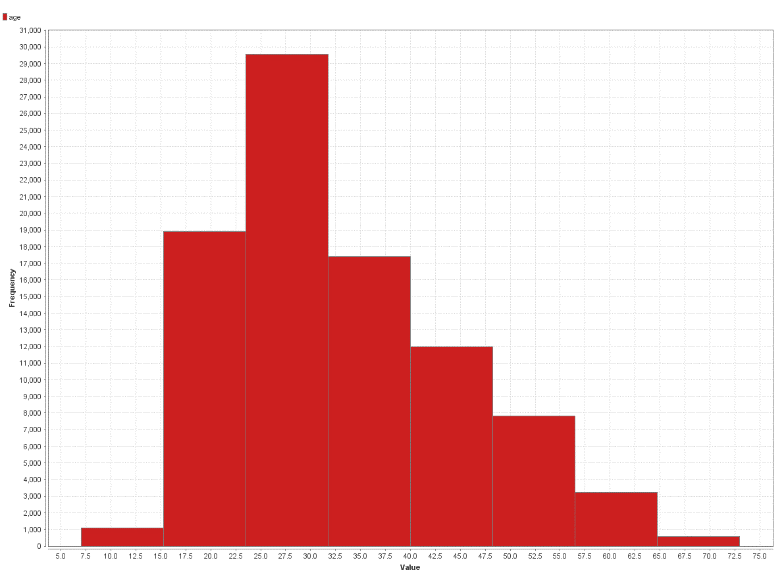


Fig 5. Count of number of ratings per age group.

From the above we can see that a lot of ratings are coming from the people who belong to the age group of 15-24 years. Hence these are the most frequent movie watchers.

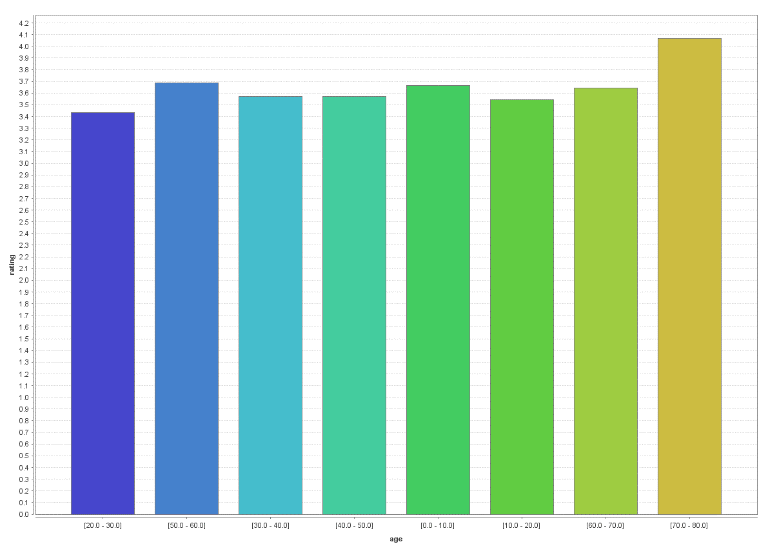


Fig 6. Average rating for different age groups.

We could see that the average rating was highest from the people of age group 70-80 years.

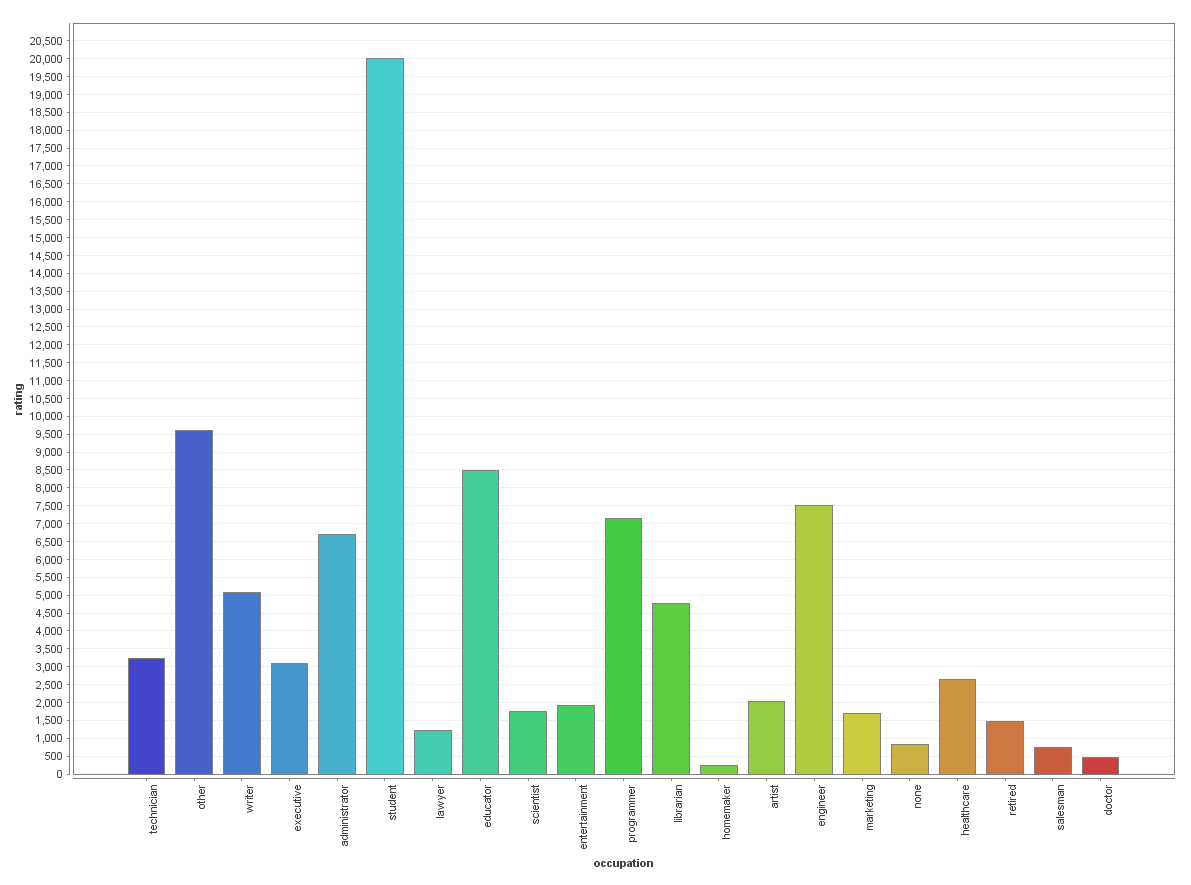


Fig 7. Distribution of number of ratings from people from different occupations.

From the above it is clear that students are the most frequent voters.

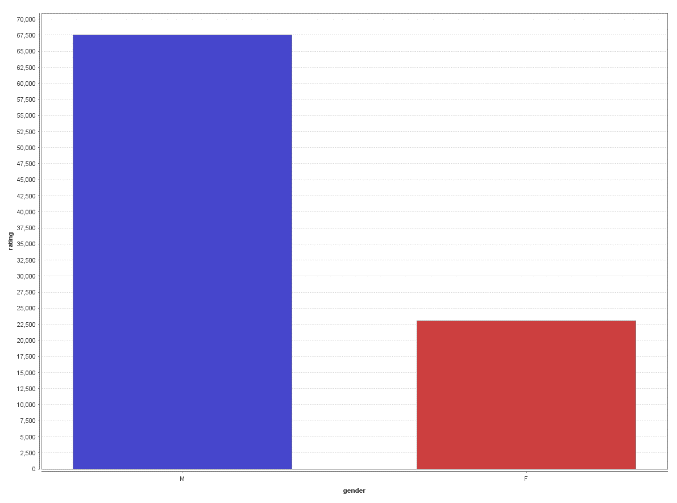
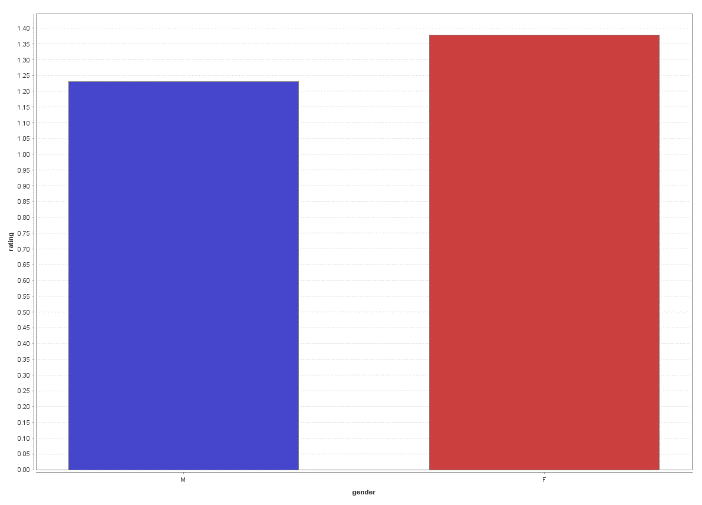


Fig 8. Distribution of number of ratings by different genders.

Fig 9. Distribution of variance in ratings by different genders.

From the above figures, there are higher number of male raters than females, but the variance in female ratings given by females is higher. This means that females don’t prefer watching all kinds of movies.

Relationship with Movie data:

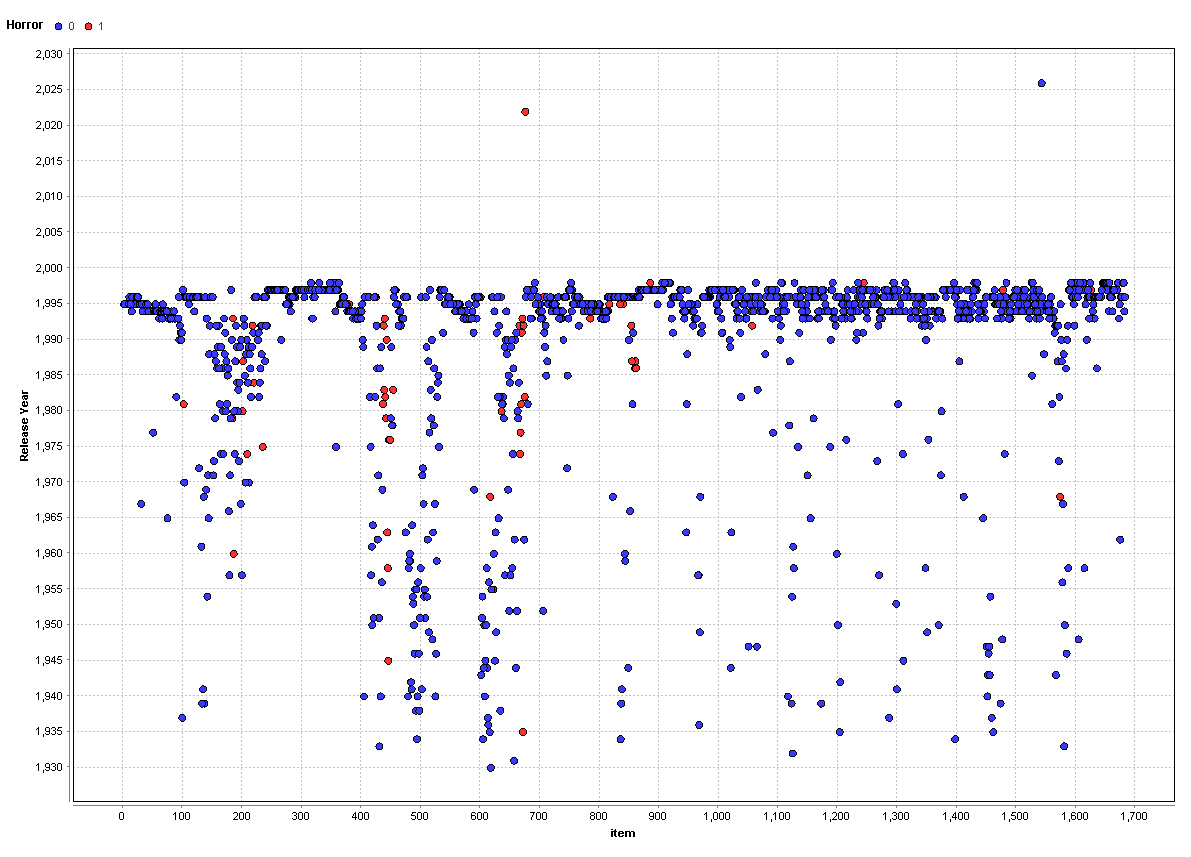
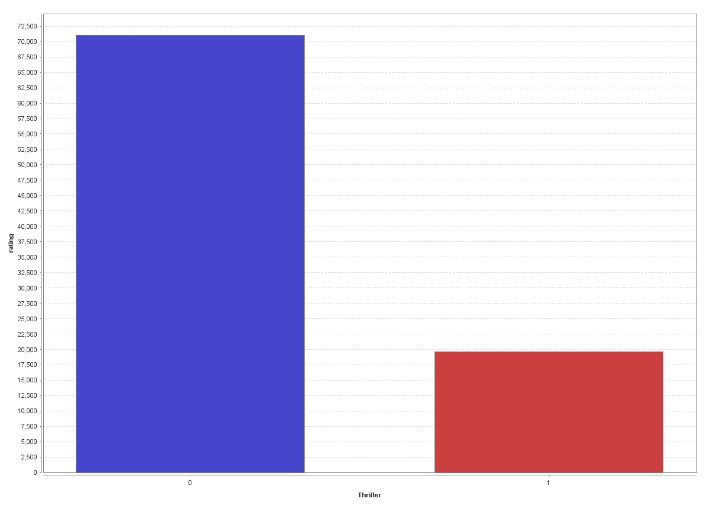
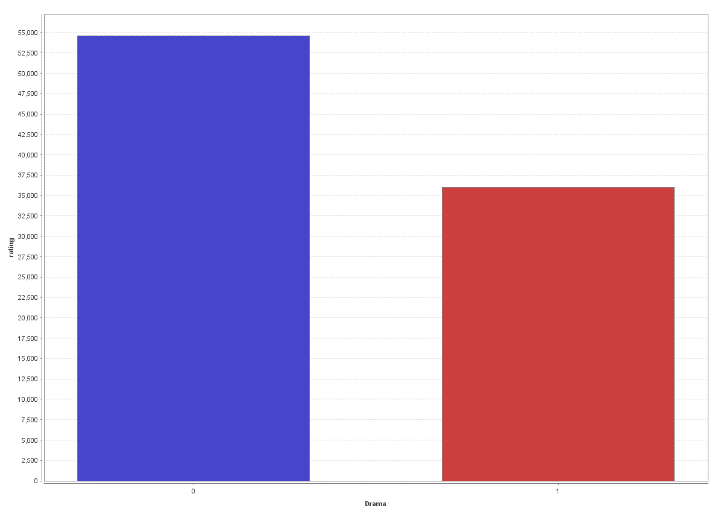
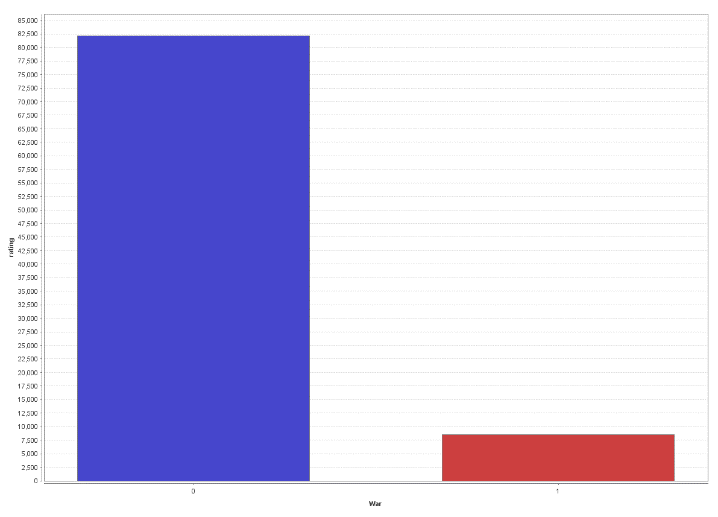


Fig 10. Distribution of movies on a timeline based on their release date with horror(in red) and non-horror(in blue) movies.

This shows that the data that we have mostly contains movies released around the year 1995 or in 1995. We have very few movies which were released near or before 1950. Hence inferring anything about them is difficult, and there might be a lot of errors in prediction of ratings to such movies.

The figure also depicts that some genres have very few movies which were released before 1950, hence it is mostly about the movies which released around 1995.

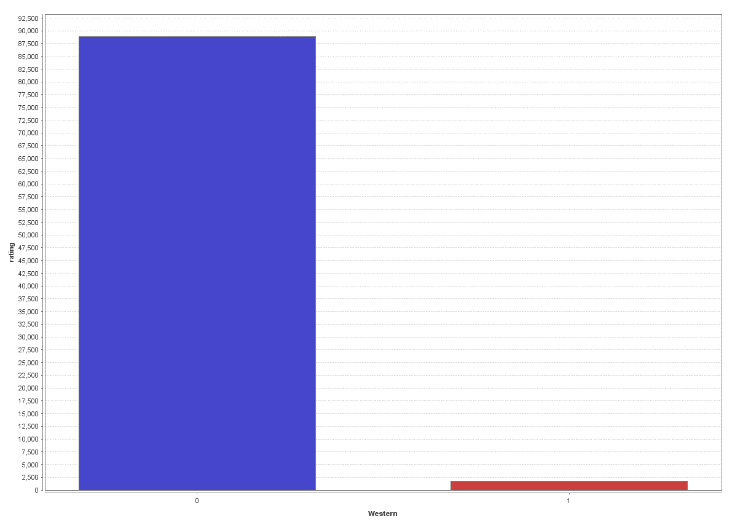


Fig 11. Distribution of number of ratings per movie based on genres. Western(Top-Left), War(Bottom-Left), Thrillers(Top-Right) and Drama(Bottom-Right).

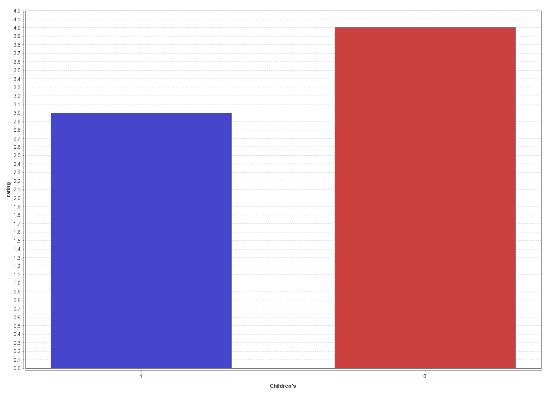


Fig 12. Median ratings of Children movies(in blue) and non-children movies(in red).

## Question 2

### Consider collaborative filtering based rating prediction.

### We will evaluate performance of different approaches for predicting ratings. What measures will you use for assessing performance (why)? And what relationships will you examine -- for example, error (or accuracy) at different levels of ratings; are errors distributed equally across movies, users? etc.

### [Remember - in the different operators that you will experiment with below, the regularization parameters can help reduce overfit]

### The Optimize Parameters (Grid) operator in Rapidminer can be useful for evaluating different parameter values and determining which value or parameter(s) give best performance. Try using this operator to find optimal parameter settings. (The sample rmp process includes a (de-selected) Optimize Parameters operator, showing how it may be used. Note – optimizing multiple parameters together can take a long time, and one may not need to optimize all parameters.)

### 

### (a) Use the Global Average method and User-Item Baseline methods. Do you find any performance differences? Do parameter changes for the user-item baseline operator make any difference?

## Solution

In this section, we were required to build a model using Recommender Systems to predict ratings. We built multiple models as mentioned in sections a, b and c of this question such as Global Average, User-Item Baseline, Matrix Factorization, User-knn and Item-knn.

We have built multiple models using different approaches like User-Item Baseline, Matrix Factorization, User-knn, Item-knn to predict user ratings between 1 and 5 for movies. We have compared this against actual ratings we already have in test data. We were then required to choose a best model based on its ability to correctly predict the user-ratings. Performance of these models in predicting the ratings can be assessed by measuring distance between the prediction and the actual rating. For doing this, we used error rate as a performance evaluator as opposed to accuracy/precision/recall.

There are three measures of error: RMSE, MAE, NMAE. RMSE (Root Mean Square Error), is data-sensitive and the value is unduly affected by outliers. So, we chose not to use RMSE as our measure. MAE (Mean Absolute Error), gives the average of the distances between predicted rating and actual rating for each observation. NMAE (Normalized Mean Absolute Error) is MAE whose values are normalized between 0 and 1 for easy comparison. In our case, we do not need to compare the distances/errors between different users or items, hence there is no need to use normalized value. Based on this reasoning, we chose to use MAE as a measure to performance of the prediction models. Therefore, we will select our best model based on the MSE.

In part (a) We were required to compare between the outputs obtained by us for two different models namely Global Average Method and User Item Baseline method.

The global averages method had the lowest performance. The main problem with this method was it predicted all the movies to be rated at the mean value of all the ratings.

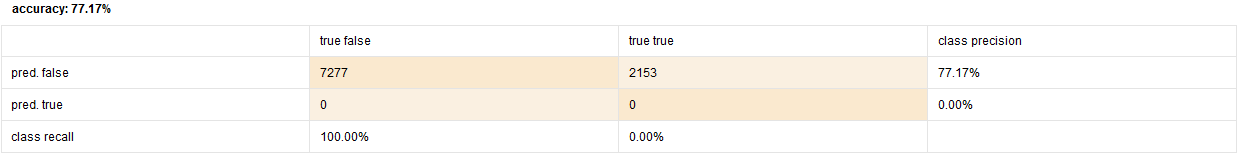


Fig 13. Confusion matrix of test data for Global averages method.

Performance vector of global average method on test data:

RMSE: 1.122

MAE: 0.945

NMAE: 0.236

Hence this method is obviously not a good model for our data with a very high MAE.

For User Item Baseline method:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Trail Number** | **Model: User Item baseline** | | | **Training Data** | | | **Test Data** | | |
| **Iteration** | **Reg u** | **Reg i** | **RMSE** | **MAE** | **NMAE** | **RMSE** | **MAE** | **NMAE** |
| 1 | 10 | 15 | 10 | 0.92 | 0.729 | 0.182 | 0.963 | 0.765 | 0.191 |
| 2 | 25 | 15 | 10 | 0.92 | 0.729 | 0.182 | 0.963 | 0.765 | 0.191 |
| 3 | 50 | 15 | 10 | 0.92 | 0.729 | 0.182 | 0.963 | 0.765 | 0.191 |
| 4 | 100 | 15 | 10 | 0.92 | 0.729 | 0.182 | 0.963 | 0.765 | 0.191 |
| 5 | 10 | 1 | 10 | 0.917 | 0.725 | 0.181 | 0.96 | 0.758 | 0.189 |
| 6 | 10 | 5 | 10 | 0.918 | 0.726 | 0.182 | 0.959 | 0.759 | 0.19 |
| 7 | 10 | 10 | 10 | 0.919 | 0.728 | 0.182 | 0.961 | 0.762 | 0.19 |
| 8 | 10 | 1 | 1 | 0.91 | 0.717 | 0.179 | 0.963 | 0.757 | 0.189 |
| 9 | 10 | 1 | 5 | 0.913 | 0.721 | 0.18 | 0.96 | 0.757 | 0.189 |
| 10 | 10 | 1 | 15 | 0.921 | 0.729 | 0.182 | 0.962 | 0.759 | 0.19 |

Table 1. Trail values for all the iterations tried on User Item Baseline Method.

We used it over a range of values for the three parameters. The number of iterations did not make much of a difference, whereas the values of regularization constants had a noticeable impact on the error rate. We chose trail number 9 to be our best model with test data MAE to be 0.757.

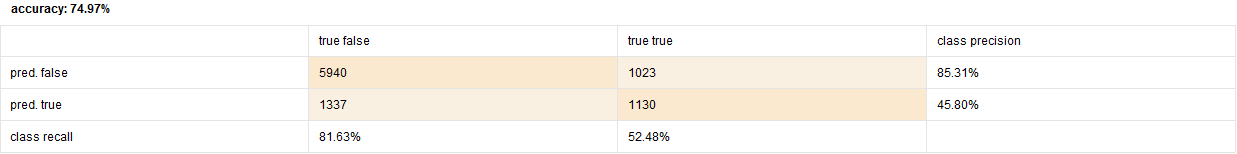


Fig 14. Confusion matrix of test data for User Item Baseline for the best model.

User Item Baseline method was much better than the global averages method in every aspect. It had an accuracy of 74.97% with 45.80% and 52.48% of precision and recall respectively. It also had a lower error rate than the global averages method with MAE to be 0.757.

### (b) Use the Matrix factorization operator. Explore performance with varying number of factors. Does learning rate make a difference to performance?

## Solution

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Trail number** | **Model: Matrix Factorisation** | | | | **Training data** | | | **Test Data** | | |
| **Num Factors** | **Learning Rate** | **Iteration Number** | **Regularization** | **RMSE** | **MAE** | **NMAE** | **RMSE** | **MAE** | **NMAE** |
| 1 | 5 | 0.01 | 30 | 0.015 | 0.82 | 0.644 | 0.161 | 0.958 | 0.755 | 189 |
| 2 | 10 | 0.01 | 30 | 0.015 | 0.745 | 0.584 | 0.146 | 0.979 | 0.768 | 0.192 |
| 3 | 50 | 0.01 | 30 | 0.015 | 0.439 | 0.34 | 0.085 | 1.037 | 0.814 | 0.203 |
| 4 | 3 | 0.01 | 30 | 0.015 | 0.849 | 0.667 | 0.167 | 0.949 | 0.75 | 0.187 |
| 5 | 3 | 0.05 | 30 | 0.015 | 0.854 | 0.669 | 0.167 | 0.973 | 0.761 | 0.19 |
| 6 | 3 | 0.1 | 30 | 0.015 | 0.883 | 0.692 | 0.173 | 0.99 | 0.778 | 0.194 |
| 7 | 3 | 0.15 | 30 | 0.015 | 0.923 | 0.726 | 0.182 | 1.015 | 0.799 | 0.2 |
| 8 | 3 | 0.01 | 15 | 0.015 | 0.886 | 0.7 | 0.175 | 0.976 | 0.78 | 0.195 |
| 9 | 3 | 0.01 | 50 | 0.015 | 0.841 | 0.659 | 0.165 | 0.946 | 0.744 | 0.186 |
| 10 | 3 | 0.01 | 100 | 0.015 | 0.838 | 0.656 | 0.164 | 0.956 | 0.748 | 0.187 |
| 11 | 3 | 0.01 | 50 | 0.05 | 0.856 | 0.675 | 0.169 | 0.955 | 0.754 | 0.189 |
| 12 | 3 | 0.01 | 50 | 0.1 | 0.866 | 0.688 | 0.172 | 0.947 | 0.753 | 0.188 |
| 13 | 3 | 0.01 | 50 | 0.15 | 0.89 | 0.713 | 0.178 | 0.959 | 0.769 | 0.192 |

Table 2. Trail values for all the iterations tried on User Item Baseline Method.

As we can see from the table as we increased the number of factors in matrix factorization model the error rate for the training data decreased, whereas the error rate constantly increased for the test data set. This clearly indicates that there is an overfit of the model.

On the other hand, as we increased the learning rate the error for both the training and test data kept increasing. Which was also the same for the regularization parameter.

For the number of iterations parameter, the error rate decreased till 50 and then started increasing. Hence the parameters set in trail number 9 were chosen to get our best model for matrix factorization model.

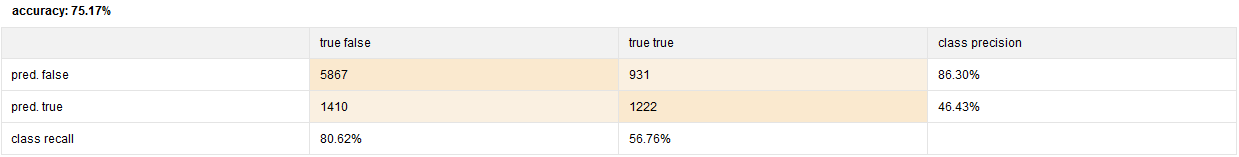


Fig 15. Confusion matrix of test data for Matrix Factorization for the best model.

Matrix factorization model also had a better accuracy, precision and recall as compared to global averages method and User Item Baseline method.

### (c) Use the User-knn and Item-knn operators. Explore performance with varying the number of nearest neighbors k? Also, do you notice any differences between using the cosine similarity measure and the Pearson measure? Are the neighborhood sizes, k, that give good performance, comparable across the two operators (why?)?

## Solution

For user K-NN Model:

Using the "USER K-NN" operator with "cosine similarity" as the correlation mode, we ran the model on the training as well as test data sets for various values of k ranging from as low as 25 to as high as 100 and tabulated their RMSE, MAE, NMAE values respectively. As the value of k increases from 25 to 100, the RMSE follows a decreasing trend till k=40 and then, increased by a not so significant value of 0.001 till k=100. MAE and NMAE plots also shows the same trend.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Trail number** | **Model: User K-NN** | | | **Training data** | | | **Test data** | | |
| **k** | **reg u** | **reg i** | **RMSE** | **MAE** | **NMAE** | **RMSE** | **MAE** | **NMAE** |
| 1 | 25 | 1 | 1 | 0.921 | 0.721 | 0.18 | 0.959 | 0.751 | 0.188 |
| 2 | 40 | 1 | 1 | 0.92 | 0.72 | 0.18 | 0.957 | 0.75 | 0.187 |
| 3 | 80 | 1 | 1 | 0.921 | 0.722 | 0.18 | 0.958 | 0.75 | 0.188 |
| 4 | 100 | 1 | 1 | 0.921 | 0.723 | 0.181 | 0.958 | 0.751 | 0.188 |
| 5 | 40 | 5 | 1 | 0.92 | 0.721 | 0.18 | 0.956 | 0.751 | 0.188 |
| 6 | 40 | 10 | 1 | 0.921 | 0.723 | 0.181 | 0.957 | 0.756 | 0.189 |
| 7 | 40 | 15 | 1 | 0.922 | 0.724 | 0.181 | 0.959 | 0.758 | 0.19 |
| 8 | 40 | 5 | 5 | 0.921 | 0.722 | 0.181 | 0.955 | 0.751 | 0.188 |
| 9 | 40 | 5 | 10 | 0.921 | 0.722 | 0.181 | 0.955 | 0.751 | 0.188 |
| 10 | 40 | 5 | 15 | 0.922 | 0.723 | 0.181 | 0.955 | 0.751 | 0.188 |

Table 3. Trail values for all the iterations tried on User K-NN model with Cosine similarity.

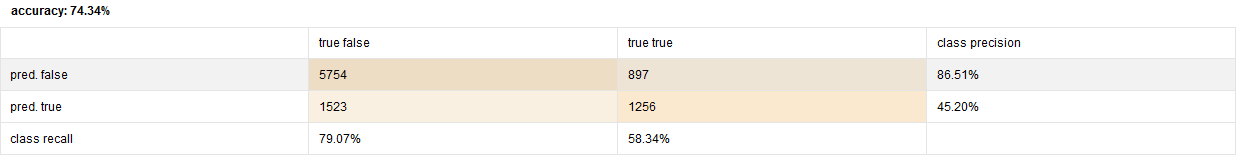


Fig 16. Confusion matrix of test data for User K-NN using Cosine similarity for the best model.

To minimize the error, we consider the model with least MAE. We observed least MAE of 0.751 when k = 40 using Cosine similarity and 0.742 when k = 40 using Pearson measure. So, the best model confined to "USER k-NN" model is achieved when k = 40 with correlation mode being “Pearson” because it gives slightly higher precision. The accuracy, recall and precision values of both the models (cosine & Pearson) are almost the same with a minute difference of 0.01%.

As we can see by comparing the below and above tables there is not much difference in trend while using Cosine similarity of Pearson similarity, but the model using Pearson similarity anyways had a slightly lower error rate when compared to model using the Cosine similarity.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Trail number** | **Model: User K-NN** | | | **Training data** | | | **Test data** | | |
| **k** | **reg u** | **reg i** | **RMSE** | **MAE** | **NMAE** | **RMSE** | **MAE** | **NMAE** |
| 1 | 20 | 1 | 1 | 0.762 | 0.59 | 0.147 | 0.953 | 0.744 | 0.186 |
| 2 | 40 | 1 | 1 | 0.781 | 0.605 | 0.151 | 0.95 | 0.742 | 0.185 |
| 3 | 80 | 1 | 1 | 0.801 | 0.623 | 0.156 | 0.95 | 0.742 | 0.185 |
| 4 | 100 | 1 | 1 | 0.807 | 0.628 | 0.157 | 0.951 | 0.743 | 0.186 |
| 5 | 40 | 5 | 1 | 0.781 | 0.606 | 0.152 | 0.948 | 0.743 | 0.186 |
| 6 | 40 | 10 | 1 | 0.782 | 0.608 | 0.152 | 0.95 | 0.747 | 0.187 |
| 7 | 40 | 15 | 1 | 0.783 | 0.609 | 0.152 | 0.952 | 0.75 | 0.188 |
| 8 | 40 | 5 | 5 | 0.782 | 0.607 | 0.152 | 0.947 | 0.742 | 0.186 |
| 9 | 40 | 5 | 10 | 0.782 | 0.607 | 0.152 | 0.947 | 0.742 | 0.185 |
| 10 | 40 | 5 | 15 | 0.783 | 0.607 | 0.152 | 0.947 | 0.742 | 0.186 |

Table 4. Trail values for all the iterations tried on User K-NN model with Pearson similarity.

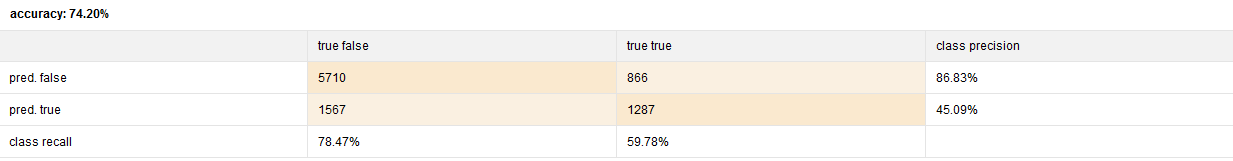


Fig 17. Confusion matrix of test data for User K-NN using Pearson similarity for the best model.

For Item K-NN model:

Using the "ITEM k-NN" operator with "cosine similarity" as the correlation mode, we ran the model on the training as well as test data sets for various values of k ranging from as low as 20 to as high as 100 and tabulated their RMSE, MAE, NMAE values respectively. As the value of k increases from 20 to 100, the RMS follows a decreasing trend till k=20 and then, increased by a value of 0.002 till k=100. MAE and NMAE plots also increased by 0.001 gradually.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Trail number** | **Model: User K-NN** | | | **Training data** | | | **Test data** | | |
| **k** | **reg u** | **reg i** | **RMSE** | **MAE** | **NMAE** | **RMSE** | **MAE** | **NMAE** |
| 1 | 20 | 1 | 1 | 0.893 | 0.698 | 0.174 | 0.942 | 0.737 | 0.184 |
| 2 | 40 | 1 | 1 | 0.894 | 0.701 | 0.175 | 0.945 | 0.741 | 0.185 |
| 3 | 80 | 1 | 1 | 0.9 | 0.707 | 0.177 | 0.948 | 0.744 | 0.186 |
| 4 | 100 | 1 | 1 | 0.902 | 0.708 | 0.177 | 0.949 | 0.746 | 0.186 |
| 5 | 20 | 5 | 1 | 0.893 | 0.698 | 0.174 | 0.942 | 0.737 | 0.184 |
| 6 | 20 | 10 | 1 | 0.893 | 0.698 | 0.174 | 0.942 | 0.737 | 0.184 |
| 7 | 20 | 15 | 1 | 0.893 | 0.698 | 0.174 | 0.942 | 0.737 | 0.184 |
| 8 | 20 | 5 | 5 | 0.894 | 0.699 | 0.175 | 0.939 | 0.736 | 0.184 |
| 9 | 20 | 5 | 10 | 0.897 | 0.701 | 0.175 | 0.939 | 0.737 | 0.184 |
| 10 | 20 | 5 | 15 | 0.899 | 0.703 | 0.176 | 0.94 | 0.737 | 0.184 |

Table 5. Trail values for all the iterations tried on Item K-NN model with Cosine similarity.

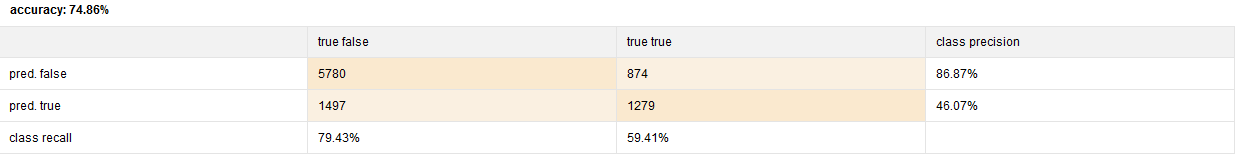


Fig 18. Confusion matrix of test data for Item K-NN using Cosine similarity for the best model.

When we used "Pearson" as the correlation mode, we observed that the change in correlation mode slightly deviated from the above trend. The error values decreased till k = 20, and then started an increasing trend. The least value of MAE of 0.735 is observed across k = 20.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Trail number** | **Model: User K-NN** | | | **Training data** | | | **Test data** | | |
| **k** | **reg u** | **reg i** | **RMSE** | **MAE** | **NMAE** | **RMSE** | **MAE** | **NMAE** |
| 1 | 20 | 1 | 1 | 0.683 | 0.529 | 0.132 | 0.94 | 0.735 | 0.184 |
| 2 | 30 | 1 | 1 | 0.698 | 0.542 | 0.135 | 0.94 | 0.736 | 0.184 |
| 3 | 40 | 1 | 1 | 0.71 | 0.551 | 0.138 | 0.94 | 0.736 | 0.184 |
| 4 | 80 | 1 | 1 | 0.741 | 0.577 | 0.144 | 0.941 | 0.737 | 0.184 |
| 5 | 20 | 5 | 1 | 0.683 | 0.529 | 0.132 | 0.94 | 0.736 | 0.184 |
| 6 | 20 | 10 | 1 | 0.683 | 0.529 | 0.132 | 0.94 | 0.736 | 0.184 |
| 7 | 20 | 15 | 1 | 0.683 | 0.529 | 0.132 | 0.94 | 0.736 | 0.184 |
| 8 | 20 | 5 | 5 | 0.688 | 0.535 | 0.134 | 0.937 | 0.735 | 0.184 |
| 9 | 20 | 5 | 10 | 0.692 | 0.54 | 0.135 | 0.938 | 0.736 | 0.184 |
| 10 | 20 | 5 | 15 | 0.696 | 0.543 | 0.136 | 0.939 | 0.737 | 0.184 |

Table 6. Trail values for all the iterations tried on Item K-NN model with Pearson similarity.

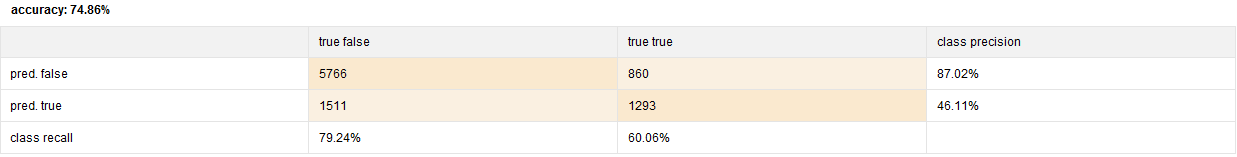


Fig 19. Confusion matrix of test data for Item K-NN using Pearson similarity for the best model.

To minimize the error, we consider the model with least MAE. We observed least MAE of 0.736 when k = 20 using Cosine similarity and 0.735 when k = 20 (least K) using Pearson measure. So, the best model confined to "ITEM k-NN" model is achieved when k = 20 with correlation mode being “Pearson” because it gives slightly higher precision. The accuracy, recall and precision values of both the models (Cosine & Pearson) are almost the same with a minute difference of 0.01%.

When we consider both the User K-NN model and the Item K-NN model, our best model in terms of least MAE was Item K-NN model using Pearson similarity as correlation mode with K=20, reg u=5 and reg I =5, giving an MAE of 0.735.

### Comparing performance across the different operators, which would you prefer to use (why)?

### (d) Use the Model Combiner operator to predict ratings using a combination of models. Do you notice any significant performance improvement?

## Solution

We have considered Global Average, User Item Baseline, Matrix Factorization, User K-NN and Item K-NN operators.

Comparing the operators to global average operator doesn’t provide any improvement to the recommender system and its’s performance is the lowest among all operators.

User item baseline as show below with no of iteration=10 provides MAE =0.72 which is higher than that of Matrix factorization and item KNN.

In KNN we have considered both ITEM KNN and USER KNN , as seen in 2(C) **Item KNN with K=20 provides MAE = 0.735 with Pearson correlation the best model and the preferred model.**

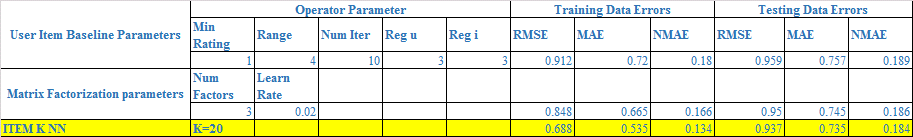


Table 7. Comparison of model performances.

Hence we chose Item K-NN as the best model of all because of a lower MAE value and by observing the confusion matrices of all the models we could infer that Item K-NN had the highest accuracy of all the models without compromising on the precision and the recall.

Model Combiner Operator:

In this project, we observed that combining different models gave us a better model performance with lower MAE value and higher accuracy, precision and recall.

We got our best model by combining Item K-NN with Matrix Factorization models.

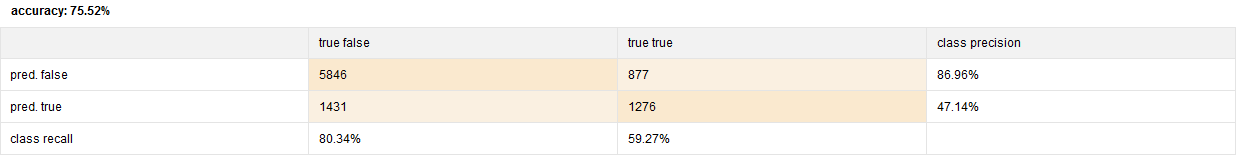


Fig 20. combined model performance of the above model.

Performance matrix of the above model on test data:

RMSE: 0.927

MAE: 0.729

NMAE: 0.182

## Question 3

### Consider the decision support objective of recommending movies to users. Movies predicted to receive high ratings will be recommended for a user. We then need to determine a cutoff rating for 'high' (for example, any rating >=4 is 'high'). To access performance for this, we can consider a confusion matrix and related measures like precision, sensitivity etc. (or, how many predicted highs correspond to actual high, etc.). Using the predicted ratings for the test data, determine such decision support performance using the operators in Question 2.

### Comparing performance across the different operators, which would you prefer to use (why)? What value of 'cutoff' will you use? Are errors distributed equally across movies and across users?

## Solution

Now we will recommend to users based on what we think they will “like” based on our model’s predicted user-ratings of movies. Movies predicted to receive high ratings will be recommended for a user, but we must determine the appropriate cutoff rating for what value is high enough to warrant a recommendation. We need to balance the goals of wanting to prevent recommendations of a movies that a user is not likely to watch or like with the desire not to miss out on the opportunity of recommending a movie he would watch or like, as we want to have a large enough number of recommendations to give users an appropriate level of choice. For this purpose, we consider a confusion matrix to assess the appropriateness of the cut-off value. From the confusion matrix, we look at recall rate of class “yes” and precision rate of class “yes” collectively to zero in on the cut-off rating. The high precision value means that we are correctly making recommendations, as we do not want to recommend a movie the user will not like, but if that was coupled with a low recall, we are not capturing enough movies in the recommendations to appropriately reflect the user’s preferences. Recall gives us an estimate of “out of all the movies the user would have watched, how many were we able to recommend to him” while precision is a measure of “out of the movies we recommend to him, how many did we correctly recommend per his taste”. For our project, we disregard Accuracy as basis of selection, as it is an overall indicator of performance, which is not enough to compare between the models.

We further observed the 5 models from question 2 – Global Average, User-Item Baseline, User k-NN, Item k-NN, Matrix Factorization – with their optimal parameter values (so chosen by end of Q2). We then measured the performance of each of these models against different values for ‘high’ starting from 3.0 to 4.0 with increments of 0.2. We chose to start with 3.0 as the lower bound for the cutoff, as the average rating per movie is close to 3.0 and by common sense, a rating of 3 on a scale of 1-5 implies a “not good, not bad” rating for that movie. We chose the upper limit as 4.0, as choosing above 4.0 would narrow down the pool of movies to recommend significantly.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Cut-off for rating = 3.5 | User-Item Baseline | User k-NN | Item k-NN | Matrix Factorization |
| Recall | 73.74 | 74.86 | 76.1 | 75.19 |
| Precision | 73.62 | 73.45 | 74.61 | 73.9 |
| Accuracy | 69.45 | 70.11 | 71.12 | 70.59 |

Table 8. Model performances comparison for threshold=3.5 in terms of accuracy, precision and recall.

We found that the optimal cut-off value is 3.5 for the rating. We then analyzed the performance of various models at rating cut-off 3.5.

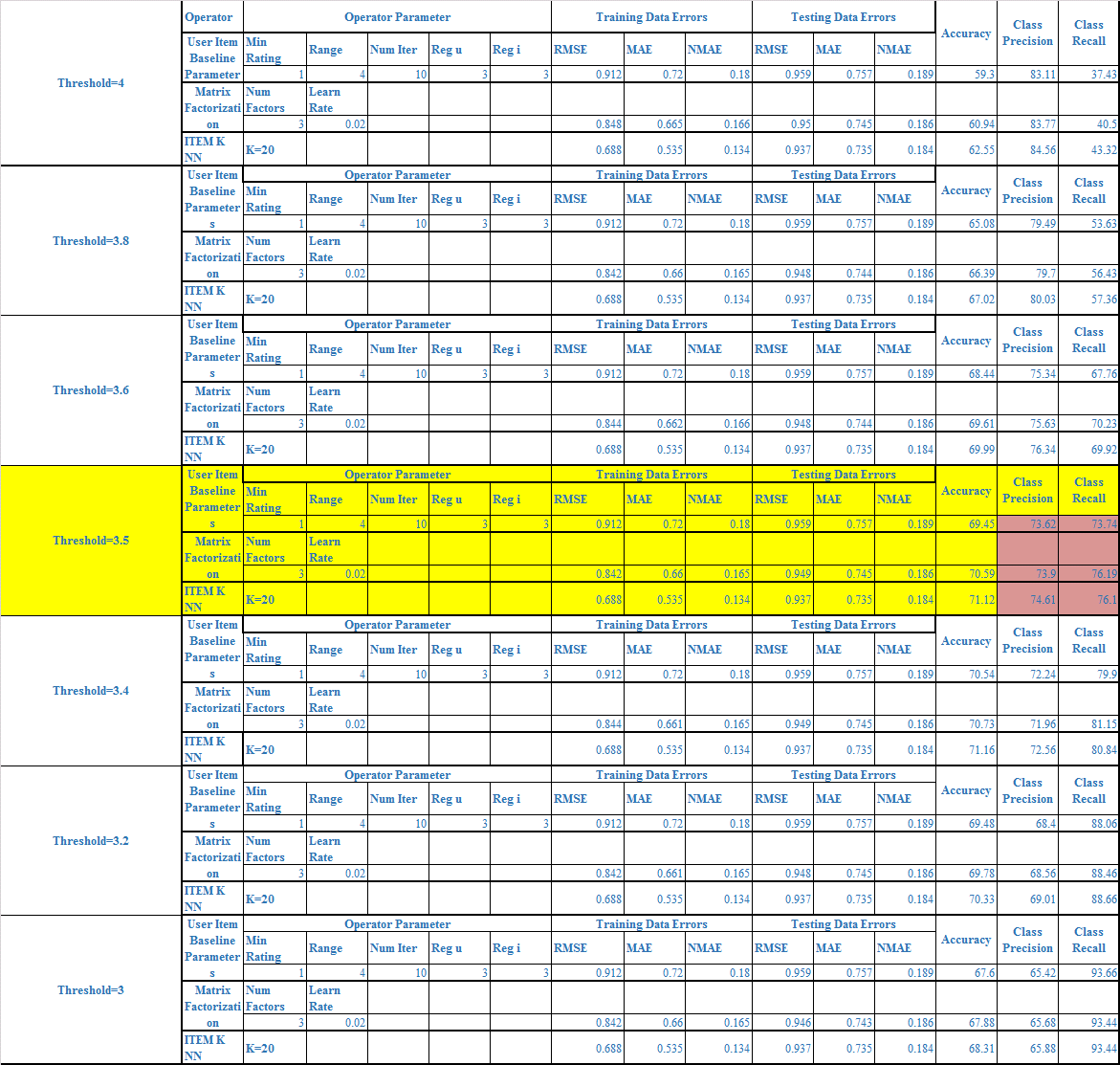


Table 9. Model performances for different thresholds.

In attrition to selecting out best model based on error, precision and recall, we also want to review the errors before we implement the recommendations. Ideally there would not be a bias in the errors, they should be concentrated close to 0 and be approximately normal. Below are histograms for the knn model.

Global Average has very large errors, user baseline and matrix factorization both have a lower error across movies as well. On our best selected case above is the error and its spread evenly across zero so we recommend implementing our Item-knn as the best choice for the recommender system.

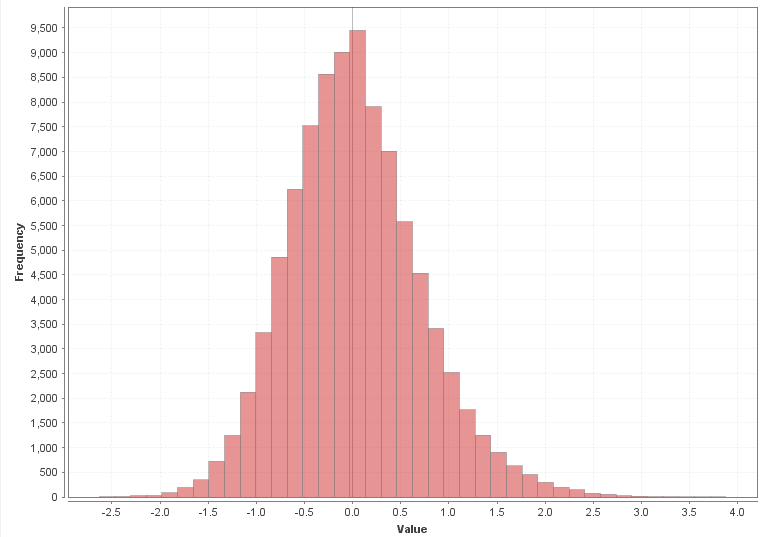


Fig 21. Distribution of errors across users.

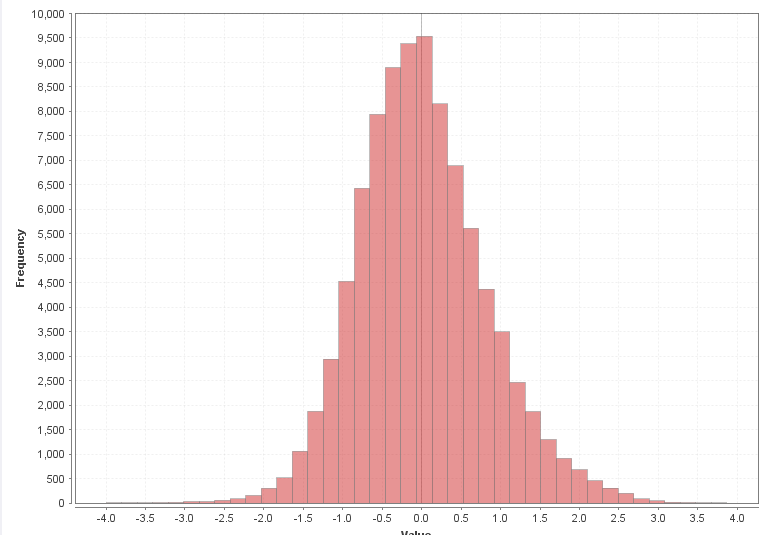


Fig 22. Distribution of errors across items.