Analysis on Features that Contribute to the Success of a Movie

CS506: Computational Tools for Data Science - Spring 2018
Final Report

https://github.com/czheng78/CS506_FinalProject

Zexu Chai
chaizexu@bu.edu

Chenyu Wang
wangcy@bu.edu

Chuan Xing Zheng
czheng78@bu.edu

May 3, 2018

1. Introduction

The success of a movie in recent years are influenced by many factors that makes the accurate prediction of a success for the new movies being released a difficult task. There also have been several semantic analysis techniques to analyze user reviews which were applied to analyze several movie data-sets. None of the studies has succeeded in suggesting a model good enough to be used in the film industry. In this paper, we examined the movie data-set we found online and analyzed features that may lead a movie to be successful. We came up with several comprehensive and detailed questions to address, which is listed below. Data analysis techniques we used were Random Forest, Linear Regression, K-Nearest Neighbor, and Collaborative Filtering.

Questions Investigated:

- Which important features (e.g. genres, age of users, sex of users, occupation of users, etc.) influence the IMDB score the most?
- Are there differences in movies' ratings by occupations?
- Do the type of movies the users watched predict the users' gender?
- Can we predict this year's biggest box office hits in the theaters with mostly "inherent" movie characteristics?
- How well can we predict if a person might enjoy a certain type of movie based on the movies he has already watched and the ratings of such movies?

2. Data Sets Used for Evaluation

2.1. MovieLens 100K

2.1.1 Description of the Data Set

One of the data-sets we downloaded and analyzed was the MovieLens 100K data-set and it was collected by the GroupLens Research Project at the Univeristy of Minnesota.

2.1.2 Access to the Data Set

Download it at https://grouplens.org/datasets/movielens/

2.1.3 Attributes of the Data Set

This dataset was useful because it contains: 100,000 ratings (1-5) from 945 users on 1682 movies. In addition, each user has rated at least 20 movies. Importantly, this dataset has users demographic information, such as age, sex, occupation, and zip. Users who did not rated more than 20 movies or do not have complete demographic information was removed from the dataset. The movies in the data-set are also categorized into the 19 genres they provided.

2.2. IMDB Movies Dataset

2.2.1 Description of the Data Set

This dataset contains information about 14,762 movies. The data was preprocessed and cleaned to be ready for machine learning applications.

2.2.2 Access to the Data Set

The information of these movies was downloaded from kaggle for the purpose of analysis. Download it at https://www.kaggle.com/orgesleka/imdbmovies

2.2.3 Attributes of the Data Set

The information we extracted were the movie's title, imdbRating, ratingCount, duration, year, type, nrOfWins, nrOfUserReviews, nrOFNominations, genres (in 0/1 indiciating if the movie has the given genre), etc. This dataset was useful to us because it provided us the duration of the movie and the number of nominations for Oscars and wins.

2.3. MovieLens 1M Dataset

2.3.1 Description of the Data Set

These files contain 1,000,209 anonymous ratings of approximately 3,900 movies made by 6,040 MovieLens users who joined MovieLens in 2000.

2.3.2 Access to the Data Set

Download it at https://grouplens.org/datasets/movielens/

2.3.3 Attributes of the Data Set

All ratings are contained in only one file ”ratings.dat” and are in the following format:UserID::MovieID::Rating::Timestamp. Also we could get movie information from the data set about genres and movie ID.

- UserIDs range between 1 and 6040
- MovieIDs range between 1 and 3952
- Ratings are made on a 5-star scale (whole-star ratings only)
- Each user has at least 20 ratings

2.4. 350000+ movies from themoviedb.org

2.4.1 Description of the Data Set

We used this dataset because it contained extra information about the crew and casting in the movies. This dataset is very similar to the Box Office Mojo site, which supplemented with Oscars information.

2.4.2 Access to the Data Set

The primary source of this dataset is from *themoviedb.org*. We extracted the movie details from *www.themoviedb.org* API: movie/details. The movies crew & casting were extracted from the *themoviedb.org* API: movies/credits. There are at least 350,000 movies, from the 19th century to 2017. We used this dataset because it contained extra information about the crew and casting in the movies. This dataset is very similar to the Box Office Mojo site, which supplemented with Oscars information.

3. Methodologies

We employed several data analysis techniques, such as Random Forest, Linear Regression, K-nearest neighbors, and Collaborative Filtering algorithm to answer our questions and hit upon any interesting patterns. Different techniques were used for exploring different questions:

- For the first question, we wanted to predict the rating of a movie based on the different important features that could influence the IMDB score. We first dealt with the dataset by extracting important features like, the title year, IMDB score, age of users, sex of users, occupation of users and the movie genres. We employed the Random Forest ensemble approach in this section because Random Forest classification works well with features consist of a mix of categorical and numerical features. We used the scikit-learn’s Random Forest Regressor to do the classification after data extraction. We used the IMDB score as the dependent variable and all other features as the independent variables. We used scikit-learn’s cross-validation to test which *n_estimators* affect the validation score the most (*n_estimators* is the number of trees in the forest). Finally, *feature_importances_* attribute of *sklearn.ensemble.RandomForestRegressor* were used to calculate each features’ weights. Then, we were able to see which features are more influencing to the IMDB score.

- The MovieLens 100K data-set with 11 occupations, e.g. programmer, student, artist, etc., were used for the second question. The Linear Regression algorithm (least squares fitting) was used to see which occupation have the same rating trend compared to the overall rating. We wanted to see which occupations’ average ratings are similar with the overall rating. Hence, we could notice which occupation have the most influence in rating a movie.

  Linear Regression was used here because this algorithm is useful in studying relationship between two variables namely dependent and independent, e.g. occupation and movie rating.

  We grouped the data by occupations and took the average of the ratings for each of them. The major problem with the data-set was that some movies only have a few ratings. We consider those movies as ”meaningless movies” so we discard them when pre-processed the data. The other problem was that some occupations did not watch enough movies, which could impact our result when we do the comparison. In order to have enough data, we decided to curtail the data down and picked out the top 25 and top 100 movies with the highest average rating and used them for analysis.

  We derived the calculation of coefficients of the linear regression and implemented it with Numpy.

Then using those coefficients we can obtain the prediction for each x. We calculated the euclidean distance between the prediction and actual result and summed them up, and then we can fetch the overall error of the model.

Ultimately, we outputted the value of every error function for every occupation provided in the data, which was the maximum likelihood of the prediction model. After computing the minimum value of error, we were able to measure the similarity of movie rating between each occupation and the overall data.

- For question 3. We were curious if we could predict the gender of the users based on the genres they watched before. As follows, we analyzed on the data-set that contained 100,000 ratings (from 1 to 5) from 945 users on 1682 movies. There were 18 types of movies (i.e. Action, Adventure, etc.) in the data-set. We decided to implement a model using K-Nearest Neighbors algorithm (KNN). The advantages of KNN are: (1) Robust to noisy training data (2) Flexible to feature or distance choices (3) Do well with outliers (4) Time complexity is quite low to run large data-set.

  In this section, we considered those genres of the movies that the users watched as features. Male users were labeled as 0 and female users were labeled as 1. Training data was transformed into a dictionary data structure in Python, which the keys were the labels and the values were the features. Labels of the testing data were hidden. Every features of the testing data was compared with their corresponding features and we calculated the Euclidean distance to do the comparisons. K most similar training samples were found and their labels were also extracted. Finally, the highest frequency label were assigned to the test sample as its label. After we obtained the result based on KNN, we also did the prediction using the Random Forests classification. It was used because it is an useful ensemble learning method for classification, regression and other tasks. And operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. In the end, we also did a comparison of the two methods, which is detailed in the result.

- For question 4, the features we considered to predict Box Office were the movie rating, budget, action, comedy, animated, horror, HSX, and Fandango. These features were extracted from the *themoviedb.org* dataset. We picked out these features because we thought these were the major influences to the Box Office of movies. Namely, we employed the Linear Regression model for predicting the Box Office. The dependent variable was the column, *Box*, and the remaining columns were the independent variables (i.e. rating, budget, action, etc.). We used *sklearn.linear_model.LinearRegression* because we wanted to explore which among the independent variables are related to the dependent variable. The accuracy of the outcome was analyzed by inspecting the result parameters (i.e. R-square, coefficients, RMSE, etc.).

- For our recommendation system, we employed the User-Based collaborative filtering (CF). The reason to use User-based CF was that we only have 2000+ movies and the matrix is accordingly a dense matrix. It works by looking for users who share the same rating patterns with the target user. We used the ratings from those like-minded users to make the predictions. First, we preprocessed the data by re-scaling the ratings from a range of 0 to 1. Next, we made 2 dictionaries. The first one was a movie dictionary which contains information of every movie along with the users who rated the movie (in the format of {'MovieID:ID of the movie', [UserID:ID of users]}. The second one was a user dictionary which contains the information of every user and all movies he rated (in the format of {'UserID:ID of the user':[(MovieID:ID of the movie, Rating:the rating the user gave)]}). For the first step of collaborative filtering system, we needed to decide what type of users we will use to make prediction and so we used KNN algorithm to do this. The input is a user corresponding to the movies he watched. We then calculated the cosine distance for pairwise comparison between the target person and all other users. For this purpose, we calculated the distance by summing up the cosine distances of each movie. If the movie is watched by only one person of the two, then the distance should be 0. After that we sorted the neighbors by similarity and choose the top k of them for afterwards process. For this part, we rescaled the ratings so that all ratings are between 0 and 1. Cosine distance in this situation would decrease the influence of some big data as lesser of two evils.

4. Results and Visualizations

4.1. Features that Affect IMDB Score

4.1.1 Result of Random Forest

Initially, we computed a pairwise correlation matrix between all the features with *.corr()*, which is shown below as a heatmap.

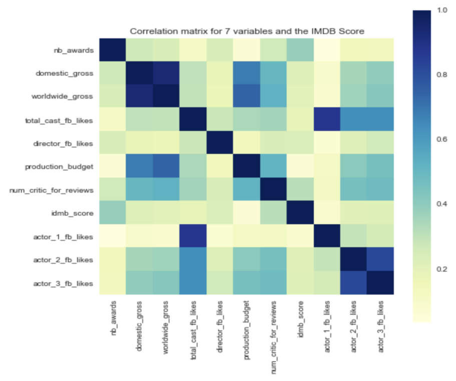


Figure 1: A heatmap of that displays the pairwise correlation between the features in the IMDB *Kaggle* data-set.

From the heatmap shown above, the IMDB score is most correlated to the number of awards the movies won and is least correlated with the production budget. However, it is not informative enough to see the correlations between IMDB score and other features in the data-set. Hence, we used the Random Forests classification to do feature ranking and prediction of the IMDB score. We split the data into 2 arrays, one with the IMDB scores and the other contains other movie features. Then, we divided them into 80% and 20% for training and testing, respectively. Once split, we applied the *sklearn.ensemble.RandomForestRegressor* with *n_estimators* = 500, *oob_score* = True, and 10-fold cross-validation (*sklearn.cross_validation*). *oob_score* defines as the score of the training dataset obtained using an out-of-bag estimate and so using the *oob* error rate that was created with the model gave us an unbiased error rate. *n_estimators* is the number of trees in the forest. We got a training score of 0.934 and *oob_score* of 0.514, which gave us an unknown fit. In essence, we did a follow-up test with several *n_estimators* lower than 500 [Fig. 2]. We found that the more trees in Random Forest, we got higher validation score or less accurate on prediction the IMDB score. Thus, *n_estimators* = 500 is'nt optimal.

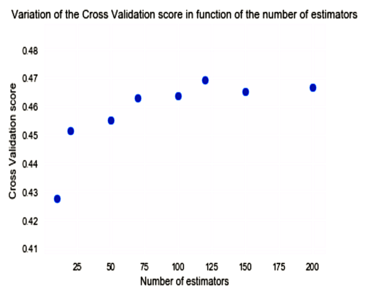


Figure 2: Number of estimators vs Cross Validation Score

4.1.2 Evaluation of the result

Additionally, we looked for the most important features that affect the Random Forest classification or more importantly which feature have bigger impact on the IMDB score. We used the attribute *feature_importances_* (calculation of features' weights) of *sklearn.ensemble.RandomForestRegressor* to plot out the features from more important to less important [Fig. 3]. As shown below, the number of rewards a movie received and the number of Facebook likes are the top 2 features that impact IMDB score the most, which is what we expected from the heatmap in Figure 1.

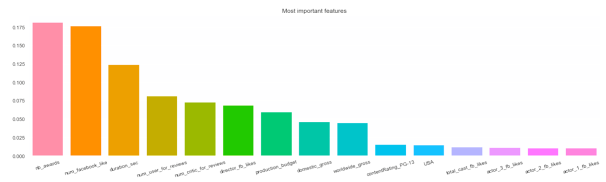


Figure 3: Most important features that influence the IMDB score. The y-axis is the *feature_importances_* values and x-axis is the features.

4.2. Analysis of Each Occupations' Average Ratings

4.2.1 Result of analysis

After calculating the value of error function, we found that administrator and educator owns the minimum error value, as shown in Figure 4.

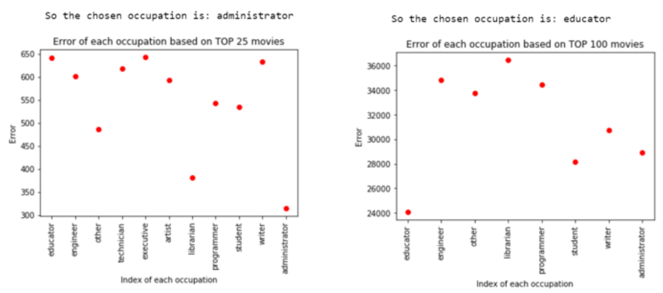


Figure 4: Error of Each Occupation's average rating compared to the overall average rating.

4.2.2 Result of prediction

The results are different because the features we utilized to train our model are different. We think that means when considering TOP25 movies, administrator share the same rating trending with the overall society. When consider the TOP100 movie, educators occupy the main occupation. So we decided to use these two occupations to predict the movie ratings in our prediction model separately.

Using the administrator and educators occupation, we utilized the coefficient we obtained from our model to predict the rank of the top 25 movies, as shown in Figure 5.

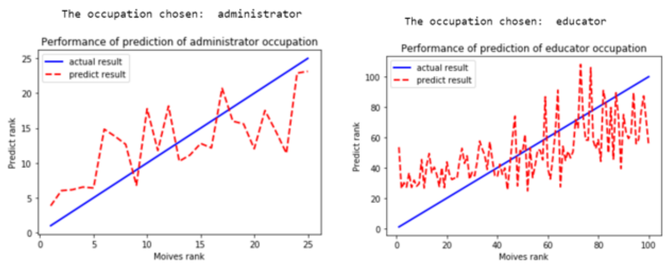


Figure 5: Performance of two predictions using predicted occupation based on the top 25 movies and top 100 movies.

Even though the predicted result is not the same as the actual result, they shared the same trend. Linear regression models were not precise as what we expected. We thought there are a few possible reasons why our result was not accurate:

- The size of the data may not be large enough, many occupations don't have enough reviews, which may create errors.
- There doesn't exist an occupation which can represent the masses perfectly.
- We did not re-scale the values while implementing the linear regression and so one order could have impacted the whole result.

4.2.3 Evaluation of prediction

After the coefficients we are looking for had been calculated, we could use coefficients to predict corresponding target and calculate the distance between the actual result and target using the euclidean distance. After summing up those distances, we were able to obtain the error of the whole model.

4.3. Predict the Gender of the Users

4.3.1 Result of prediction

The k in the KNN Algorithm affects the accuracy of the prediction. In essence, we varied k from 1 to 8; all using the same training and testing data. We tested accurateness with *sklearn.model_selection's* cross-validation for each k, which is shown below:

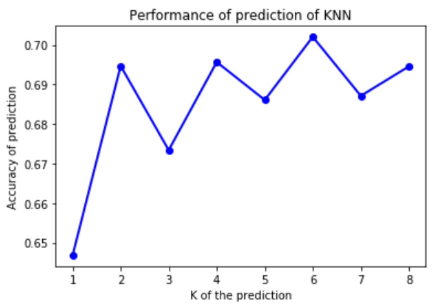
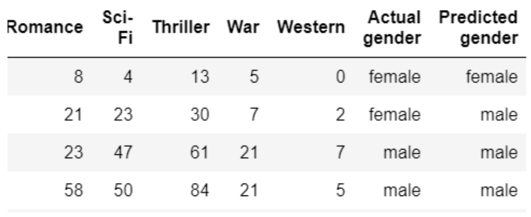


Figure 6: Accuracy of each prediction's average according to different k.

We concluded from the figure above that when k is best at 6. We used this k to predict the gender of the user based on the genres of the movies they watched [Fig. 6]. In Figure 7, it shows a snapshot of our prediction result. The first five columns of the table shows part of the attributes of the sample. The last two columns represent the actual gender of users and the predicted gender from our prediction model.



Romance	Sci-Fi	Thriller	War	Western	Actual gender	Predicted gender
8	4	13	5	0	female	female
21	23	30	7	2	female	male
23	47	61	21	7	male	male
58	50	84	21	5	male	male

Figure 7: A snapshot of the prediction result with k as 6.

4.3.2 Evaluation of the result

We used cross validation to verify our results with 5-Fold. 80% of data were used for training set and the remaining data were utilized as the testing data. This process was done five times and we chose the training set and testing set randomly. The model gave us a high accuracy, which was 70.20%. This high accuracy told us that we are able to predict the gender if we are given the types of movies that person watched. Moreover, we calculated the running time five times to evaluate our efficiency. It costed us about 1.45 seconds to perform the prediction. The result of our evaluation is shown as follow:

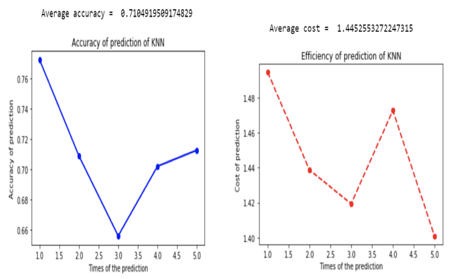


Figure 8: Left plot shows the accuracy for each k. Right plot shows the efficiency for each k.

4.3.3 Comparison with Random Forest Method

The accuracy of Random Forest was around 70.70% and calculated with 5-fold cross-validation (*sklearn.cross_validation*). And it took about 2 seconds to finish the whole process, which was slower than KNN method. Thus, we would like to conclude that both methods have the similar accuracy in predicting the gender, but the KNN method computes with higher efficiency.

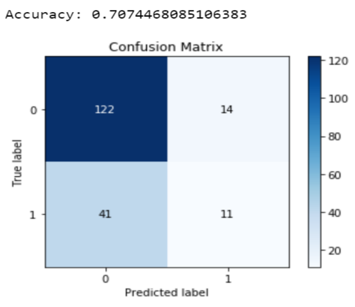
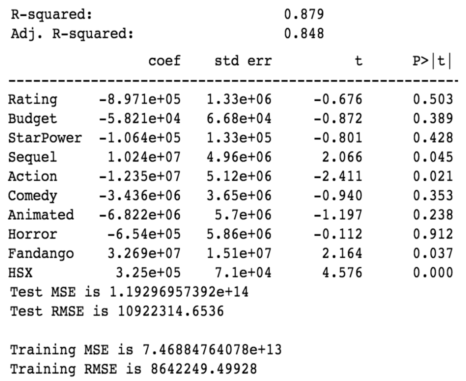


Figure 9: The confusion matrix of the prediction result based on Random forest.

4.4. Prediction of Box Office

4.4.1 Result

The final prediction model for Box Office whose accuracy parameters are visualized are shown in Figure 8. Originally, we divided the 350,000 movies dataset into 80% and 20% for training and testing, respectively. Once split, a LR model was fitted with the training data, and an outcome was predicted from the testing data. We used the *statsmodels.api.OLS* to build the prediction model. It is interesting to note that we got a R-squared of 0.879, which is not too bad for our model. A high R-square means that the model explains 88% of the variance in our dependent variable ("Box Office"). The coefficients of the dependent variables in Figure 10 shows that as the independent variable increases by 1, the predicted value of dependent variable increases by the coefficient shown in the figure. From the coefficients in Figure 10, interestingly, the Fandango variable has the highest positive coefficient, which as the Fandango variable increases by 1, the predicted variable of Box Office increases by 3.269e07. The Fandango variable represents the Fandango score for the movies.



```
R-squared:                         0.879
Adj. R-squared:                    0.848
                 coef    std err          t      P>|t|
------------------------------------------------------
Rating     -8.971e+05   1.33e+06     -0.676      0.503
Budget     -5.821e+04   6.68e+04     -0.872      0.389
StarPower  -1.064e+05   1.33e+05     -0.801      0.428
Sequel      1.024e+07   4.96e+06      2.066      0.045
Action     -1.235e+07   5.12e+06     -2.411      0.021
Comedy     -3.436e+06   3.65e+06     -0.940      0.353
Animated   -6.822e+06    5.7e+06     -1.197      0.238
Horror      -6.54e+05   5.86e+06     -0.112      0.912
Fandango    3.269e+07   1.51e+07      2.164      0.037
HSX          3.25e+05    7.1e+04      4.576      0.000
Test MSE is 1.19296957392e+14
Test RMSE is 10922314.6536

Training MSE is 7.46884764078e+13
Training RMSE is 8642249.49928
```

Figure 10: Accuracy Parameters of the Linear Regression Model for Box Office.

4.4.2 Evaluation and Discussion

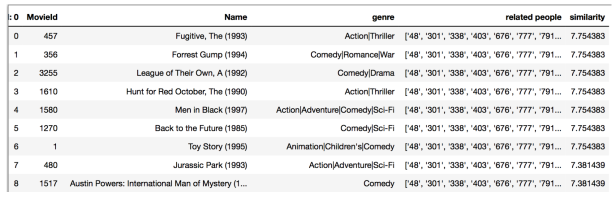
The RSME for the training and test set were also outputted to see how an absolute fit was our model to the data. RSME is interpreted as the standard deviation of the unexplained variance. Lower values of RMSE indicates better fit and it is a good measure of how accurately the model predicts the response. Although our RMSE is high, this makes sense because we are talking about Box Office, which values can be in millions. Box Office could be predicted with attributes like if the movie have sequels and the type of the movies.

4.5. Recommendation System

4.5.1 Result

In order to test the precision of our recommendation system. We separated our data into two part: training set and testing set with the ratio 8:2. We selected one user of the testing set, then let his User ID as input and ran our system to get the recommendation list. The personal accuracy was defined as the percentage of the same movies existed in both recommendation list and the list of movies he watched. Users in the testing set were considered as input and we obtained every personal accuracy of them. Finally, we calculated the average of personal accuracy to get our whole accuracy with the 5-Folds cross-validator (*sklearn.model_selection.KFold*). It randomly generates 5 times training and tesing sets which are both considered as the measurement of the accuracy. The accuracy of our recommendation system was approximately 57%-65%, which was not high.

In order to run our system, you first need to decide who you want to recommend by selecting a specific User Id. Then, the function already well-defined will search for users who gave similar ratings to movies similar as the target person. Finally, a table would show up and every movie is ordered by the similarity score [Fig. 11].



: 0	MovieId	Name	genre	related people	similarity
0	457	Fugitive, The (1993)	Action|Thriller	['48', '301', '338', '403', '676', '777', '791...	7.754383
1	356	Forrest Gump (1994)	Comedy|Romance|War	['48', '301', '338', '403', '676', '777', '791...	7.754383
2	3255	League of Their Own, A (1992)	Comedy|Drama	['48', '301', '338', '403', '676', '777', '791...	7.754383
3	1610	Hunt for Red October, The (1990)	Action|Thriller	['48', '301', '338', '403', '676', '777', '791...	7.754383
4	1580	Men in Black (1997)	Action|Adventure|Comedy|Sci-Fi	['48', '301', '338', '403', '676', '777', '791...	7.754383
5	1270	Back to the Future (1985)	Comedy|Sci-Fi	['48', '301', '338', '403', '676', '777', '791...	7.754383
6	1	Toy Story (1995)	Animation|Children's|Comedy	['48', '301', '338', '403', '676', '777', '791...	7.754383
7	480	Jurassic Park (1993)	Action|Adventure|Sci-Fi	['48', '301', '338', '403', '676', '777', '791...	7.381439
8	1517	Austin Powers: International Man of Mystery (1...	Comedy	['48', '301', '338', '403', '676', '777', '791...	7.381439

Figure 11: A snapshot of the result for our recommendation system for User ID: 237.

4.5.2 Comparison with Convolution Neural Network

A more precise algorithm in designing recommendation system is Convolution Neural Network.We didn't have enough time implementing such algorithm but we came up with the idea on how to improve our system using convolution neural network. First, we could focus on more features of the movies rather than just the ratings of the movies. Name of the movie, genre of the movie and Id of the movie can be thought to be the features of the movies. Accordingly, features of users can also be transferred into feature vectors before the convolution. We could use three layers. Such algorithm will train a more precise model for prediction.

4.5.3 Evaluation and Discussion

The recommendation list shows a list of movies that a target user would like to watch. In the list, movies with high similarity scores were shown on the top and that meant the target user would like those movies more. In this case, we have succeeded in making movie recommendation for people. However, the precision we calculated was low. There are several reasons may account for this. First, we only focused on ratings of movies and didn't try to include other features for prediction. In order to solve this, we can implement Convolution Neural Network by creating feature vectors of both movies and users. Second, the way we calculated the precision was not the appropriate one. Since the method of splitting training and testing set will cause different result, we should compare use a different method such as using bagging or boosting algorithm. Also, a new question came out while working on this system, which is given a new movie that haven't came out yet, how well can we predict if the target user will like to watch it or not.

5. Conclusion

We accomplished several tasks and learned a lot from this project. We were able to use algorithms we learned from class and used in this project. Based on Random Forest, we found interestingly that the number of rewards a movie received and the number of Facebook likes are the top 2 features that impact the IMDB score the most. From the top 25 most rated movies, we concluded that librarians can embodies the public's preference for movies. Given the genre of a movie, we were able to predict which gender would rate movies higher and our prediction model gave us a high accuracy of 70.2%. We also used a Random Forest to compare the accuracy and we got a similar validation score, but the efficiency of Random Forest was not better than KNN. The prediction model for Box Office is also not bad. We got a R-Squared value of 0.879, which is high. Yet, we would want to test this hypothesis with other regression models and see if our model would be more concrete. In addition, the test RMSE was just slightly higher than the training RMSE, which means that we did not badly over fit the data; our model tests was well in sample and has good predictive value for when tested out of sample. If we have time, we would like to run a goodness-of-fit test on the model to see if the model is adequate or inadequate of representing the data. At last, we developed a recommendation system to recommend movies and the accuracy was 57%-65%. The accuracy was not high enough and so we hoped to improve this system if we have more time.