Potential Movie Rating Prediction using Supervised Machine Learning Methods

Taslim Mahbub
Computer Science & Engineering Dept.
American University of Sharjah
Sharjah, United Arab Emirates
b00075270@aus.edu

Muhammed Yusuf Dada
Computer Science & Engineering Dept.
American University of Sharjah
Sharjah, United Arab Emirates
b00068047@aus.edu

Mohamed Shezan Rizny
Computer Science & Engineering Dept.
American University of Sharjah
Sharjah, United Arab Emirates
b00068372@aus.edu

Abstract **- Hollywood is a multi-billion-dollar industry which releases over a hundred films a year, with large variations in the budgets and box office grosses of the movies. Identifying which factors are crucial to a movie's profitability and subsequently predicting the success of a movie given its relevant parameters could save movie studios hundreds of millions of dollars a year. This paper analyses the efficiency of using supervised machine learning methods including decision tree, Naïve Bayes, KNN, non-linear SVM and linear SVM classifiers to predict the rating of movies, while analyzing the influence of variables like trailer budget, genre, popularity, run time and vote average. The results of this paper show that the KNN classifier was found to be the best with an F1 score 0.783097.**

Keywords - Data analytics, Prediction, Machine learning, Movie success, Box office

I. Introduction

In 2015, the global box office gross reached an all-time high of $38 billion, with 5 films grossing over 1 billion, which is the most in the history of Hollywood [1]. Most of the highest grossing films came from just 6 studios including 20th Century Fox, Marvel Studios, Columbia Pictures, Walt Disney Pictures, Paramount and Warner Bros. However, the total gross of a movie is not necessarily indicative of its profitability. Due to skyrocketing costs of marketing campaigns attached to blockbusters with budgets crossing $200 million, movies with higher initial budgets often need to gross a higher percentage of their budget to break even, and hence the success of a movie is critical to the movie industry.

A commercial success movie does not only entertain its audience, but also enables film production companies to receive tremendous profit and awards. With hundreds of movies being produced every year, investors are becoming more critical in their evaluation criteria to determine a worthy movie to invest in. Determining success factors based on statistical evaluation and machine learning models applied on historical data can allow us to predict a high rated movie. Often, popular crews and cast can generate revenue, but does not necessarily result in success in terms of rating. In our project, we will consider success based on the rating a movie has received.

With so many possible factors our objective is to predict the movie rating based on metrics like the budget, genre, runtime and popularity to determine which of these factors play the most critical roles. In this paper we considered 5 different classifiers that include the decision tree, Naïve Bayes, KNN, non-linear SVM and linear SVM classifiers and the training was done using 4 types of input data based on different feature selection and dimensionality reduction methods.

II. Literature Review

Haiyan et al. [2] tries to predict the film box office by analyzing the data from 147 films using certain indicators which include the premiere box office, days of film screening, ratings on movies and the 360 index. The methodology being used here is linear regression and factor analysis to predict and analyze the film box office. Factor analysis was mainly used to classify related variables into the same class whereas regression was used to obtain a linear equation. The results yielded a good prediction with an error rate ranging from 2% to 6%.

Ahmed et al. [3] attempts predictive analysis of movie popularity using machine learning algorithms on both conventional features and social media features. The results showcase that the sentiments harnessed from social media features can predict movie success with greater accuracy than conventional features. Results achieved include a classification of 77% and 61% using specific social media features for Rating and Income prediction respectively. Lastly, combining conventional and social media features was determined to be the ideal solution to predict movie success.

Subramaniyaswamy et al. [4] aimed to identify the factors that are vital to a movie's profitability and success so that movie studios can save hundreds of millions of dollars a year. The methodology used here involves multiple linear regression and Support Vector Machine (SVM) classification to predict the box-office success of a movie. The data set used populated with information from Wikipedia and BoxOfficeMojo for movies released in 2016. The features for the data set included the Opening Date, Movie Name, Budget, Domestic Gross, International Gross, Total Gross, Studio, Cast and Crew, Genre, Medium (Live action or Film), Trailer Views, Wikipedia Views, Rotten Tomatoes Score. This dataset uses several similar features as ours and hence will be a good reference. The results obtained showed that when SVM was used to train multiple variables, there was an accuracy rate of 56.52% which is a higher accuracy than previous attempts at SVM using only a single variable that were also mentioned in this paper.

Quader et al. [5] aims to help investors in the movie sector to avoid investment risk by predicting the approximate success rate of a movie based on its profitability by analyzing the historical data from sources such as IMDb and Rotten Tomatoes. The methodology proposed uses several machine learning techniques such as the Support Vector Machine (SVM), Neural Network and Natural Language Processing. These techniques consider pre-released features as well as post-released features. The results showed that the Neural Network gives an accuracy of 89.27% while the SVM yielded an accuracy rate of 88.87%. It was also determined that the budget, IMDb votes and number of screens were the most important features which play a vital role in the prediction of a movie's success. As you can see, two of these vital features are also part of our dataset and hence can be compared with other additional features that we have.

Bristi et al. [6] uses the Hollywood movie list from Wikipedia and their corresponding movie ratings from IMDb to create their data set to predict a movie's rating. It uses metrics such as the title of the movie, studio, cast and crew, genre, country, month and date of release, year. This is very similar to our data set and hence is highly relevant in our work. The data then undergoes data extraction and preprocessing and is then fed into several machine learning techniques using a tool called Weka 3.8.3. tool. The five machine learning algorithms include Bagging, Random Forest, J48, IBK and Naive Bayes. The results conclude that all five classifiers give an accuracy rate greater than 95% which is quite accurate, however the best outcomes in terms of accuracy was from the random forest classifier. This model can also be used to predict other ratings Rotten Tomato or Metacritic as well as other types of movies such as Bollywood.

Marović et al. [7] presents an overview of automatically predicting movie ratings based on various methods. The dataset used in this paper was obtained from the publicly available movie database from IMDb which is very similar to our own. Apart from the user ratings, other features that were taken into consideration including title, genres, year of release, directors, screenwriters and actors. This dataset consisted of 1059 users, 9428 movies, 1000 actors, 1066 directors, 1060 screenwriters, 28 genres and 65581 ratings in range from 1 to 10 was used for the purposes of this paper. The training set consisted of 64522 ratings while the test set consisted of 1059 ratings. The methodologies used were classified into three main categories which were the Content-based method (eg: Neural Network), Collaborative method (eg: k-NN algorithm), and the Hybrid method (eg: SVD-kNN). The results obtained showed that the probabilistic latent semantic analysis achieved the best predictions which comes under the Collaborative methods.

Kose et al. [8] compares the performance of three various algorithms in order to predict the ratings that will be given to the movies by potential users/audience. The methodologies adopted in this paper include the User-Based Collaborative Filtering, Iterative Matrix Factorization and Yehuda Koren's Integrated model using neighborhood and factorization where we use root mean square error (RMSE) as the performance evaluation metric. The data set used consisted of 1000 movies and 10000 users in which all users rated the movies. So, in total, there were 10000000 entries in the matrix of which 90% was given ratings in the matrix for training and the remaining 10% for validation. The results indicated that there were no significant differences between performances, especially if the complexity was considered. Overall, it was determined that the Iterative Matrix Factorization performs fairly well despite its simplicity.

Dhir et al. [8] uses conventional film features from an IMDB dataset to predict movie rating. The Machine learning techniques applied include Support Vector Machine, Random Forest, Ada Boost, Gradient Boost and K-Nearest Neighbors. A positive correlation was discovered for the following features: Number of Critic for reviews, Duration, Gross, Number of voted Users, Number of Facebook Likes and specific movie genres (Biography, Drama, Historical). It was found that Random Forest technique produces the best result with the highest precision and accuracy.

Bhave et al. [10] utilizes both classical features, such as cast, producer, director etc., and social features like anticipation and user feedback (as sourced from social media like YouTube and Twitter), to predict movie success and increase average returns. The machine learning technique applied is multivariate linear regression and the result was a multiple R-squared value of 0.7057 which indicates an interrelation between classical and social features.

Ericson et al. [11] attempts to address what makes a movie successful considering only input features which a producer can influence prior to the movie premier. Their dataset was sourced from multiple sources including IMDB, Rotten Tomatoes and Wikipedia. The Machine learning methods applied include Locally Weighted Linear Regression and Support Vector Machine. With SVM's, classification rates well over 0.5 were achieved (classifying movies as successful based on input features). In particular, meaningful results were obtained in linking specific keywords in the plot description to higher ratings and box-office gross.

III. Methods and Datasets

In this section, we shall look at the raw data available to us and select the attributes that are suitable to us. We will also examine how we selected the labels for the dataset. The dataset used for our project was taken from Kaggle – the movies dataset.

The data was obtained by using the TMDB API to retrieve attributes of over 45000 movies. The main csv folder included the following attributes on 45000 movies that were used for our study:

- *Budget – in US dollars*
- *Genres – multiple genres per movie if applicable*
- *ID- identification number as index*
- *Popularity – quantified popularity measure*
- *Runtime – length of movie in minutes*
- *Vote_average -the average of all the votes from users per movie – from 0 to 10 with 10 being the highest.*

The other attributes that could be potentially used were Revenue, number of Spoken Languages, and whether or not the movie belongs to a collection. To keep the experiment simple, we decided to only include the attributes as listed.

A. Data Preprocessing

At this stage, we shall describe the content that we extracted from our data to feed into the machine learning models. To preprocess and clean the dataset, we used Pandas library in python. Firstly, we dropped all the columns that were not meaningful to us. Then, we converted the numerical columns to numerical data type, this included budget, runtime and popularity.

We then removed all then null values, from both numerical and non-numerical data columns. However, this was not sufficient since a lot of movies had a 0 value for the numerical data. We spotted this by plotting the distribution of all the quantitative columns. All the 0 values were removed from the dataset. Next, the all the duplicates were removed.

The genres column had multiple values per movie. In our experiment, we wanted to focus on the primary genre of each movie. Hence, the list of json objects were processed to only keep the first (primary) genre per movie.

Another important aspect of preprocessing that remains is categorizing the ratings into discrete labels. Currently, we had a rating between 0 to 10 for every movie. By logically considering the movie ratings, we decided to simplify the labels to Good (7.5+), Average (5 to 7.5) and Bad (0 to 5). This resulted in an imbalanced dataset, as expected, since majority of the movies would fall within the Average class. There were only 479 Good movies and 884 Bad movies, while 5204 Average movies.

In an attempt to solve the imbalanced class problem, we attempted the following:

- Oversample the minority class labels (Bad, Good) to 2000 data points each.
- Under sample the majority class label (Average) to 2000 data points.

Over sampling the data can often lead to overfitting, while under sampling can result in valuable data features being lost in the process. To prevent a significant impact of either of these changes, we decided to compromise between the over and under sampling.

The remaining categorical column of genres were then one-hot encoded resulting in the following columns in our data frame: 'Action’, ‘Adventure', 'Comedy', 'Crime', 'Drama', 'Horror', 'Thriller'. The labels were saved for later analysis.

Finally, the numerical columns of 'budget', 'popularity', 'runtime', were normalized using the min-max normalization technique. It is imperative to normalize our data since some of the models are sensitive to the scaling issue. The data processing was complete and saved into arrays.

B. Feature Selection and Dimensionality Reduction

In addition to saving the raw preprocessed data, as mentioned above, we also applied feature selection and dimensionality reduction.

- First, the selectKbest method in SKLearn was used- which selects the k highest scoring features, where k was used as 5. The scoring function used was the chi-squared test, which measures the dependence between the random variables so using this function means that it would weed out the features that are likely to be independent of the class.
- Second, the selectPercentile method of SKLearn was used – it selects features according to a percentile of the highest scores. In our project, we selected to keep 50 percent of the data. For the scoring function, we used f_classif, which calculates the ANOVA F-value between feature for classification tasks. It measures if “the variance between the means of two populations significantly different?”. Variances measure the dispersal of the data points around the mean.
- Third, we use PCA to select an appropriate number of components. As seen on the figure below, the 6th component only explains 6.5% which is very small. The first 5 components explain a total of 92.3%. Hence, we decided to only take the first 5 components, since the remaining do not explain a lot of variance.

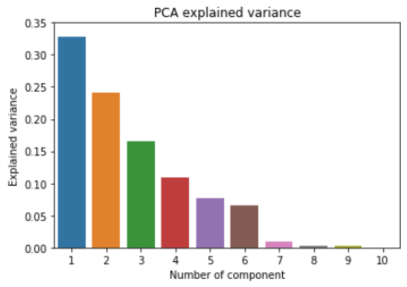


Fig 3.1 Number of PCA components and their explained variance.

Hence, we resulted in 4 different types of input data for each of the models, the raw data with all features, selectKbest data with 5 featues, the selectPercentile data with 50% of useful features, and the PCA data with 5 principal components.

C. Machine Learning Models

For each of the supervised machine learning models discussed below, we used a Grid Search to find the best hyper-parameters for each of the 4 input data formats.

1) Decision Tree classifier

The parameters tuned with their respective range:

- *Minimum samples required to split a node – range from 5 to 80 with increments of 5*
- *Maximum leaf nodes in the model – range from 5 to 80 with increments of 5*

2) Naïve Bayes classifier

The parameters tuned with their respective range:

- *Var_smoothing (portion of the largest variance added for calculation stability) – range from 1e-6 to 1e-15*

3) KNN classifier

The parameters tuned with their respective range:

- *Weight function used in prediction – either ‘uniform’ or ‘distance’*
- *Number of neighbors to use for classification - range from 5 to 80 with increments of 5*

4) SVM non-linear classifier

The parameters tuned with their respective range:

- *Kernel – either 'rbf' and 'sigmoid'*
- *C – a selection from the list: 0.1,1,10,100,1000*

- *Gamma – a selection from the list: 1e-4,1e-3,1e-2,0.1,1,5,10*
- *coef0 (Independent term in kernel function) – range from -200 to 200, increments of 50.*

5) SVM linear-kernel classifier

The parameters tuned with their respective range:

- *C – a selection from the list: 0.1,1,10,100,1000*

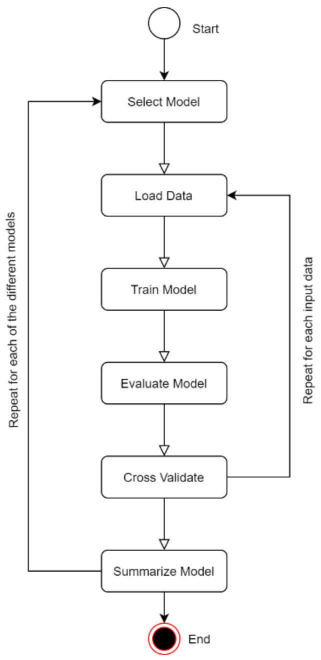
For each of the input data format to the models discussed, we used the train_test_split method from SKLearn with a fixed random state of 200 – to divide the data set into 80% training data and 20% testing. The random state ensures a consistent selection to increase fairness.

D. *Evaluation*

For each of the models discussed, the evaluation was repeated for the 4 different input data types to the model (Default data, K best Data, K select Percentile Data, PCA data.). The evaluation steps included:

- The best model from the Grid Search was chosen based on the F1-Score (macro).
- The predicted labels were generated from the trained model and compared with the actual to get the testing score. The training score was also saved for the best model.
- A classification report indicating the precision, recall and f1-score for every class (Average, Good, Bad) were generated using the predicted and actual test labels.
- The confusion matrix was plotted.
- Using the best hyper-parameter selected model, we performed cross-validation with 5 splits. The resulting F1-Scores for every model was printed out to show the variation due to different selection of training/testing data.

E. *Overall Workflow Diagram*



Note that there are 5 models and 4 input data.

IV. Experimental Results

A. *Summary of Results*

As mentioned earlier, we trained the 5 different supervised learning models (Naive Bayes, KNN, Decision Tree, Non-Linear SVM and Linear SVM) using 4 different types on input data (raw preprocessed, selectKBest, selectKPercentile and PCA) and randomly cross-validated all results (5-fold cross-validation). The standard for performance evaluation was the F1-Score. As expected, there was a variation in performance of a given model based on the type of input it received (4 different results for each model). The best result from each model is summarized in table 4.1 below

TABLE 4.1 SUMMARY OF EXPERIMENTAL RESULTS

#	*Model*	*Subhead*
1	Decision Tree Classifier	0.656956
2	Naive Bayes Classifier	0.547154
3	KNN Classifier	0.783097
4	SVM Non-Linear Kernel	0.651570
5	SVM Linear Kernel	0.579962

The KNN Classifier was found to be the best classifier with an F1 score of 0.783097. The KNN classifier produced this results when it considered the 20 nearest neighbor data points in its algorithm according to the distance similarity measure. The input data used to produce this result did not have any feature selection or dimensionality reduction applied.

A confusion matrix for this KNN classifier result is depicted below in Fig. 4.1. The confusion matrix shows the that the trained model classifies the test data into the correct rating class most of the time as indicated by the darker hues along the diagonal. The model works almost perfectly for the "Good" and "Bad" classes and fairly well for the "Average" class.

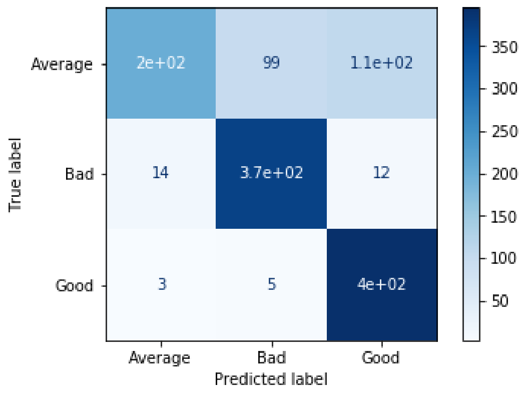


Fig. 4.1 Confusion matrix for overall best classifier – KNN

The cross validation also shows overall a very high score for each run using this particular classifier (range between 0.81 to 0.78), showing that this KNN model works well consistently.

The detailed classification report is given below in Table 4.2

TABLE 4.2 CLASSIFICATION REPORT FOR KNN

Class Label	*Precision*	*Recall*	*F1-Score*
Average	0.92	0.49	0.64
Bad	0.78	0.93	0.85
Good	0.77	0.98	0.86

B. Discussion of Results

Since we transformed our imbalanced dataset into a balanced one by applying a mix of over- and under- sampling, a lot of the data points from similar classes will be close together in a selective location. This helps explain why the KNN classifier performed so well in this experiment.

However, although we are able to detect the Good and Bad classes perfectly, we have to note that the Average class that was under-sampled has a considerably lower performance.

From table 4.2, we notice that for the “Average” data label we have the highest precision however the lowest recall. 92 percent of the points classified as “average”, actually were “average”. However, only 49% (less than half) of all real “average” movies were correctly classified by the model. For the remaining, classes of “Bad” and “Good” we have much better results as indicated by the higher recall and f1-score (harmonic mean of precision and recall).

The lower performance of “Average” class is likely due to overfitting which is indicated by the training score of this particular classifier being 1 - the KNN classifier can see a lot of the same points in a clustered region because they are replications of each other (Good and Bad classes were over-sampled). To overcome this issue, a future work could try to augment the data when over-sampling rather than simply replicating. This would add a bit of noise in the oversampled data, preventing the overfitting of the model.

V. CONCLUSION

Machine learning is a powerful tool with a wide array of applications. However, it is not all-powerful. The quality of the model trained, and results produced is highly dependent on the quality of the dataset and how well the data is prepared.

In our literature review, we mentioned several studies which surpassed our results, reaching above 90% accuracy in their classification. There could be several reasons for this. First, we could alter our approach in dealing with the imbalanced data problem as discussed above. In addition, we could categorize our voting average to a more appropriate range by first observing the data points, possibly using unsupervised learning methods. Moreover, concatenating data points from multiple sources can also increase samples for the categories which had small data. There exist multiple databases, such as IMDB and TMDB which provide API to query movies metadata, which can be utilized.

This experiment and paper showcase how machine learning can be applied on a practical real-world dataset and use case. There are a number of steps involved in simply preparing the data like data exploration, preprocessing, feature selection, etc. before even working with machine learning algorithms and techniques. This showcases the complexity of Machine Learning. Once that is done, the appropriate model or models need to be selected based on the form of the data and the research question. The hyper-parameters utilized by the complex algorithms need to be optimized to produce the best results possible given the dataset. In this experiment, we implemented all the above steps at a fundamental level to produce results and better understand the data and interrelation between attributes.

REFERENCES

[11] Forbes (2016) "Experts Predict a Drop in Box Office Revenue In 2016 After a Record Year for Hollywood", https://www.forbes.com/sites/simonthompson/2016/01/05/expertspredict-a-drop-in-box-office-revenue-in-2016-after-a-record-year-forhollywood/ #402059897195

[2] L. Haiyan, Z. Yang and Z. Hui, "Film Box Office Prediction Based On Factor Analysis," 2019 IEEE/ACIS 18th International Conference on Computer and Information Science (ICIS), Beijing, China, 2019, pp. 439-443.

[3] Ahmed, M., Jahangir, M., Afzal, H., Majeed, A., & Siddiqi, I. (2015, December). Using Crowd-source based features from social media and Conventional features to predict the movies popularity. In 2015 IEEE International Conference on Smart City/SocialCom/SustainCom (SmartCity) (pp. 273-278). IEEE.

[4] V. Subramaniyaswamy, M. V. Vaibhav, R. V. Prasad and R. Logesh, "Predicting movie box office success using multiple regression and SVM," 2017 International Conference on Intelligent Sustainable Systems (ICISS), Palladam, 2017, pp. 182-186.

[5] N. Quader, M. O. Gani, D. Chaki and M. H. Ali, "A machine learning approach to predict movie box-office success," 2017 20th International Conference of Computer and Information Technology (ICCIT), Dhaka, 2017, pp. 1-7.

[6] W. R. Bristi, Z. Zaman and N. Sultana, "Predicting IMDb Rating of Movies by Machine Learning Techniques," 2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Kanpur, India, 2019, pp. 1-5.

[7] M. Marović, M. Mihoković, M. Mikša, S. Pribil and A. Tus, "Automatic movie ratings prediction using machine learning," 2011 Proceedings of the 34th International Convention MIPRO, Opatija, 2011, pp. 1640-1645.

[8] A. Kose, C. Kanbak and N. Evirgen, "Performance Comparison of Algorithms for Movie Rating Estimation," 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA), Cancun, 2017, pp. 955-959.

[9] Dhir, R., & Raj, A. (2018, December). Movie Success Prediction using Machine Learning Algorithms and their Comparison. In 2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC) (pp. 385-390). IEEE.

[10] Bhave, A., Kulkarni, H., Biramane, V., & Kosamkar, P. (2015, January). Role of different factors in predicting movie success. In 2015 International Conference on Pervasive Computing (ICPC) (pp. 1-4). IEEE

[11] Ericson, J., & Grodman, J. (2013). A predictor for movie success. Stanford University.