Movie Success Prediction Final Report

Tsu-Hsin (Ian) Yeh
tyeh3@ncsu.edu

Guoyi Wang
gwang25@ncsu.edu

Xiaohan Liu
xliu74@ncsu.edu

Tanya (Wei Chen) Chu
wchu2@ncsu.edu

1 INTRODUCTION AND BACKGROUND

The movie industry has been expanding with the growth of technology and innovation. According to Statista [11], the number of movies produced each year in the US has been rising since 2000. The numbers saw a drop in 2019 which could be a result of the growth of on-demand, at home entertainment. It's becoming increasingly important for movie makers to draw viewers to the big screen. Fortunately, there are numerous datasets that can be analyzed to help producers forecast the success of their movie and make decisions about its release.

With data analytics, we can predict the success of movies based on several attributes. The analysis and result done by this project can be used by movie producers to determine elements of their product and anticipate the success of the movie. They can also use this information when presenting their ideas to stakeholders and potential investors. We will define the success of a movie by the return on investment: (revenue - budget)/budget.

This project will attempt to find out which features are the most important in predicting movie success and the best technique for selecting those features. Which specific directors, casts, and genres for a movie will result in the greatest success? Which model will be the best in predicting a movie's success? We will also be extracting features by combining the some of the given features. Will our extracted features produce a better performance compared to just using the original features?

1.1 Related Work

Numerous research has been done on predicting the success of movies using various features, sources, and techniques. Authors Meenakshi et al., 2018 [6] used the IMDb to find out factors that can determine the success of movies using K-means++ clustering and Decision Trees. They found that composers, directors, and genres play a large role in the success of a movie and budget is not a significant factor. Our project also attempts to find out which features in our dataset will be a significant factor in predicting the success of a movie.

Q. I. Mahmud et al., 2017 [5] used SVMs and sentiment analysis to analyze public comments of movie trailers on YouTube and IMDb and predicted movie success with an accuracy of 90.3%. A. A. Sinha et al., 2017 [10] considered similar factors, but created a system that allows users to enter actors and directors' names to forecast box office demand. They compared Random Forest Classification, KNN, Naive Bayes, and Support Vector Machine to classify movies based on their gross profit and found that Random Forest Classification had the highest classification accuracy. Similarly, our project will also be comparing three classification techniques to determine the success of a movie.

Researchers K. Lee et al., 2018 [2] extracted a special feature called "Transmedia Storytelling" which is when movies are based on stories from other media platforms. Their model has an accuracy of 58.5% for actual matches and 88.3% for one-away matches. Quader et al., 2017 [9] calculated the "Star power" of celebrities and directors and used those values along with other features to a movie's box office profit and found that budget, IMDb votes and no. of screens are the most important features. They used SVMs with an accuracy of 88.87% and Neural Network with an accuracy of 89.27%. These features are similar to the "Popularity Score," "Vote Score," and "ROI Score" that we extracted for our experiment.

In the following sections, we will describe our methods and experiment in more detail. Section two is about our approach to the problem and section three details our dataset, hypothesis and experimental design. Our results are recorded in section four and the project conclusion will be in section five. Due to the unique situation of the time period during which we conducted the project experiments, our team worked hard remotely and documented our meeting times in section six. Our references and appendix can be found in the final sections of our report and the link to our GitHub Repository is included in the appendix.

2 METHOD

We propose to explore many machine learning techniques to determine which technique is most suitable for movie success prediction by empirical experiments. Meanwhile, we decide to employ two feature selection techniques to identify the most important subset of features to predict the success of the movie. Lastly, we attempt a hybrid machine learning method that introduces clustering into the traditional supervised machine learning algorithm to figure out if it can improve the accuracy of the classification models.

2.1 Class Label Generation

The generation of the class label is based on the numerical distribution of the return on investment. We select this attribute because it best balances movies that may have very large or small budgets produced by major or indie production companies. It is also used by many investors to determine their profitability. We will use K-means++ clustering and percentiles to discretize return on investment into different levels. The clustering method generates class labels that is highly imbalanced while the percentile method produces balanced class labels.

For the K-means++ Clustering, we generate the elbow plot shown in figure 1 to select four as the optimal number of clusters based on the Within Cluster Sum of Squares. The interpretation of the four class labels are *Not Profitable*, *Slightly Profitable*, *Profitable*, and *Highly Profitable*. The numeric range of the class labels produced by two approaches is listed under Table 1. We will use balanced and unbalanced class label as input to our models to determine if one performs better than the other.

Figure 1: Elbow method for number of cluster

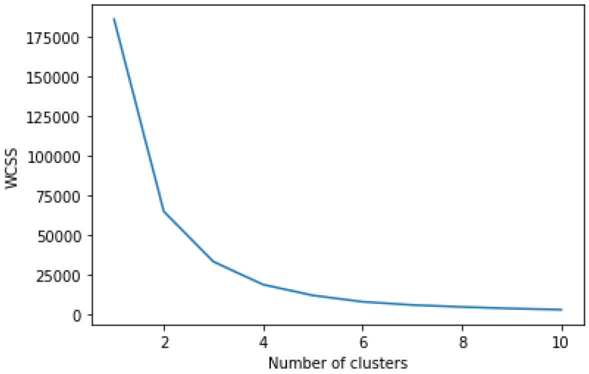


Table 1: ROI Label Ranges

Label	Clustering	Percentile
Not Profitable	< 4	< -0.25
Slightly Profitable	4-17	-0.25-0.89
Profitable	17-45	0.89-2.88
Highly Profitable	> 45	> 2.88

2.2 Feature Extraction

2.2.1 Popularity and Vote Scores. In the original dataset, the *genres*, *keywords*, *casts*, and *crews* for each movie are represented as an array of JSON objects with id and name. We decide to apply a feature extraction technique to create new columns that map these features to a numeric value to reflect the popularity and voting score associated with them. For instance, for a movie in which genres is originally stored as ['id': 16, 'name': 'Animation', 'id': 35, 'name': 'Comedy', 'id': 10751, 'name': 'Family'], we create two new columns called *genres_popularity_score* and *genres_vote_score*, which is derived correspondingly from *popularity* and *vote_average* in the original dataset. The following equations demonstrate the two steps to create *genres_vote_score* feature. The other new features are calculated in a similar manner. Noticed that not all the members of the crew are considered because we believe that directors are the most important factor to affect the success of the movie. Also, the *casts_popularity_score* and *casts_vote_score* are calculated with only the top three cast members in each movie. We decided to do this because most movies are recognized by their leading, most well-known cast and the average score could be influenced by less popular actors.

$$avg\_vote\_score(genre) = \frac{sum(movie\ voting\ score\ given\ genre)}{count(movie\ given\ genre)} \tag{1}$$

$$genres\_vote\_score = \frac{\sum_{g}^{all\ genres} avg\_vote\_score(g)}{count(genres\ of\ the\ movie)} \tag{2}$$

After applying feature extraction technique, we remove the original features including *popularity*, *vote_average*, *genres*, *keywords*, *casts*, and *crews*. Meanwhile, we introduce new features including *genres_popularity_score*, *genres_vote_score*, *keywords_popularity_score*, *keywords_vote_score*, *casts_popularity_score*, *casts_vote_score*, *directors_popularity_score*, and *directors_vote_score*.

2.2.2 ROI (Return on Investment) Scores. To find the individual feature values with the highest return on investment, we will calculate average return on investment of each value in *genres*, *keywords*, *directors*, and *casts*. This is done in a similar way to the popularity and vote scores as described above. For example, an actor's ROI score is the average of all of the ROIs of movies he/she has starred in. Then, for each sample movie, we calculate the sum of the ROI scores of the top three actors in the cast list and return it as a new attribute *casts_roi_score*. We do not simply use the average ROI score of the entire cast list because we believe only famous/popular actors have actual influence on the ROI. That being said, taking the average of all actors might cause loss of information. Such concerns are less for genres, keywords, and directors, so the average ROI score will be used for them. Notice that we will use ROI-related attributes for training only as they are generated from the dependent variable. Moreover, ROI scores can also reveal certain genres, keywords, directors, and casts that have the highest contribution to the success of movies, which is one of the major questions that our project explores. ROI-related attributes are used for Decision Tree and Random Forest model training only because having different number of attributes in train and test set would prevent the KNN model from working properly.

2.3 Feature Selection

Our original dataset has more than twenty features, but not all of them would have significant influence on the success of movies. We want to find the attributes that will most affect a movie's return on investment, so we will implement feature selection techniques Principal Component Analysis and chi-squared test. These approaches are different in that PCA finds the relationship between the features while chi-squared test shows that relationship between the features and the class. PCA feature selection first extracts new features from the dataset, and then uses those new components to select the most significant features. chi-squared test simply looks at the existing features to select the most influential. Both of these approaches are leading methods in dimension reduction, so we want to compare their performance to see which one will be most accurate in selecting the most significant features.

Principal Component Analysis is an unsupervised algorithm primarily used to reduce the dataset to lower-dimensional spaces.

We will do PCA on the twelve processed features and the ROI of each movie to generate fewer components while maintaining a high explained variance. The principal components selected usually have eigenvalues greater than one or if the researcher wants to achieve a certain cumulative percent variance. The features with the highest coefficients under the principal components are the most influential features.

The chi-squared test is used to test the relationship between the dependent and independent variables. High chi-squared value implies that there is a dependent relationship between this feature and predicted result, so it should be selected for model training. On the contrary, low chi-squared value indicates that the feature does not affect the response, so we can discard the feature. We can apply chi-squared test on the data set because the response is categorical and all the features have non-negative values.

2.4 Classification Models

To achieve the goal of predicting the level of return on investment of each movie, we will choose multiple classification models to handle this problem. We plan to train three classification models on our data set: *Decision Tree*, *Random Forest*, and *K-Nearest Neighbors*. All three models could handle the multi-class classification problem and work effectively on large training data.

K Nearest Neighbors(KNN) is very simple, easy to understand, versatile, and one of the top machine learning algorithms. At its most basic level, it is essentially classification by finding the most similar data points in the training data, and making an educated guess based on their classifications. The similarity of pairs samples is based on the distances and these distances are influenced by the measurement units. We tried values of k from 1 to 31. For each k, we applied 10-fold cross validation to get the average measurement of classifier performance which include accuracy and weighted kappa. In each cross validation, we normalized features first. We choose the k according to the highest average accuracy and weighted kappa. KNN is a lazy learner which means that there is no explicit training phase before classification. Lazy learners merely store the training dataset and wait until classification needs to be performed. For this reason, KNN tends to work best on smaller data-sets that do not have many features, which is why we believe it is a reasonable choice for our dataset.

Decision Tree is easy to understand and visualize, requires little work of data preparation, and can handle both numerical and categorical data. Moreover, it helps us to identify the most important features by simply observing the first few levels of the tree. We plan to use the Cart algorithm and C4.5 algorithm. We can leave the maximum number of leaf nodes as the default value because we do not have too many attributes. Then, we set the maximum depth as three and draw the tree. According to the preliminary fitting result from the Decision Tree, we could decide whether or not to increase its depth.

Random Forest has the benefit of being less likely to over-fit, which yields higher accuracy than of Decision Trees in most cases. We will create a parameter grid regarding the number of trees in the Random Forest, number of features to consider at each split, and minimum number of samples required to split a node and to be at each leaf node. An exhaustive grid search takes in as many hyperparameters as we would like, and tries every single possible combination of the hyperparameters as well as many cross-validations we would like it to perform. It could help us find the best hyperparameters and then build a more accurate model.

3 PLAN AND EXPERIMENT

3.1 Data Preprocessing

To predict whether or not the movie will be successful, we decide to use the movie dataset from Kaggle with more than 45,000 movies and 26 million ratings with a release date from 1874 to 2020. The original dataset contains 45,466 movies with 24 features.

The data preprocessing starts with excluding columns that are not useful from the dataset. The *adult* feature is not used because 100% of its value is false. The *belongs_to_collection* feature is not used because 90% of its value is null. The *overview* and *tagline* features are not used because the original dataset contains *keywords* feature that better summarizes the highlights of the movie. The *homepage*, *imdb_id*, *title*, *original_title*, *poster_path*, *production_companies*, *production_countries*, *video*, *original_languages*, *spoken_languages*, *status*, *vote_count* are excluded because they are not relevant to predicting the success of the movie.

Any movies with either runtime, revenue or budget equal to zero is removed from the dataset. As a result, the size of the dataset is reduced to 5,369 movies. Since the revenue and budget for each movie are most likely to be on a different scale, we decide to create a new feature called *return_on_investment* to better reflect the success of the movie. The *return_on_investment* is calculated by dividing the difference between revenue and budget by the budget of the movie.

After applying the feature extraction technique explained in the method section, we detect and remove the outliers whose absolute z-score value is greater than 3. As a result, the dataset used for the model contains 4,360 movies with 12 features. Table 2 shows a summary of the dataset.

3.2 Hypothesis

We are interested in finding the attributes that will impact the success of a movie the most, as well as exploring which directors, casts, and genres will produce the highest movie success. We will also see if including extracted features in our models will increase the model performance and finding out whether KNN, Decision Tree, or Random Forest will best predict movie success.

We hypothesize that the most important features will be cast, director and budget. Frequent movie-goers are often attracted to a film because of the cast and director, and marketing is often focused on those factors. Having a higher budget would mean there are more resources for marketing, screens, and compensation for more popular casts and director. K Meenakshi et al 2018 [6] found that cast, director, and genre are all important factors of predicting movie success, but budget was not significant. In contrast, Quader et al., 2017 [9] found that budget was the most crucial feature of their experiment and director was the least significant.

The next predictions we want to make are regarding the attribute values with the highest ROI. We predict the genres adventure, action, and drama will have the highest ROI because they are top three genres with the highest market share according to The Numbers

Table 2: SUMMARY OF THE DATASET

Feature	Type	Min	Max	Median	Mean	Std.Dev
budget	Float	1	112000000	15000000	23304691.792	23168677.380
genres_popularity_score	Float	5.975	10.713	8.959	8.902	0.814
genres_vote_score	Float	5.778	6.740	6.241	6.249	0.176
keywords_popularity_score	Float	0.341	19.750	9.503	9.484	1.947
keywords_vote_score	Float	3.000	7.725	6.391	6.382	0.334
casts_popularity_score	Float	0.039	19.354	9.398	9.131	2.811
casts_vote_score	Float	4.680	7.961	6.305	6.337	0.491
directors_popularity_score	Float	0.039	22.028	8.806	8.584	3.502
directors_vote_score	Float	4.300	8.200	6.300	6.291	0.650
runtime	Float	26.000000	338.000000	105.000000	109.225	20.732
release_year	Integer	1916	2017	2003	1998	16.757
release_month	Integer	1	12	7	7	3.436
return_on_investment	Float	-1	77.947	0.884	2.677	6.586

[7]. The actors with the highest ROI can be predicted using multiple sources. Yahoo Finance [3] lists Samuel L Jackson, Harrison Ford, Tom Hanks, Morgan Freeman, and Robert Downey Jr. as the top five highest grossing actors of all time, while The Numbers [8] lists Samuel L. Jackson, Robert Downey, Jr., Tom Hanks, Scarlett Johansson, and Bradley Cooper as the Top Domestic Leading Stars. Both of them have Samuel L Jackson, Tom Hanks, and Robert Downey Jr. in common, so we predict that they will have the highest ROI. For the movie directors with highest ROI, both Business Insider [4] and IndieWire [1] reported that Steven Spielberg, Michael Bay, and Peter Jackson are the highest earning movie directors of all time.

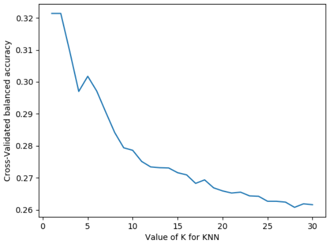
We believe the extracted features of the dataset (popularity score, vote score, and ROI score) will increase the performance of the model. Directors and casts bring in a large portion of the population to movie theaters, so their popularity and vote scores should affect a movie's success and therefore, increasing the accuracy of the models. It would also make sense for general movie popularity and vote rating scores to influence how successful a movie is.

We predict the Random Forest classifier will have the best accuracy in predicting the success of movies because it has better results for multi-class prediction. It has a wider range of hyperparameters than the other models that will enable us to test and find the best accuracy. Sinha et al., 2017 [10] found that the Random Forest classifier resulted in the highest classification accuracy compared to KNN, Naive Bayes, and SVM. K. Lee et al., 2018 [2] also found that Random Forest classifier has a higher accuracy over Neural Networks, Adaptive Tree Boosting, and Support Vector Classifier, although not as high as Gradient Tree Boosting.

3.3 Experimental Design

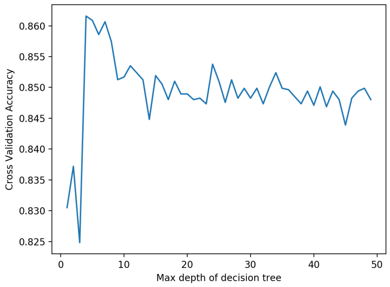
3.3.1 Training Model. **KNN hyperparameter tuning** For KNN, the k is set from 1 to 31 which will be chosen based on the best accuracy and weighted Kappa. These metrics are the average output during the 10-fold cross validation for each k. In training the KNN model, the weight of k neighbors is based on the distance, which means the closer neighbors will have a greater influence than neighbors which are further away. In figure 2, the KNN model is used for extracted features in the data set with cluster labels. When k is 3, the highest accuracy score is 0.33.

Figure 2: Value of k Tuning for KNN



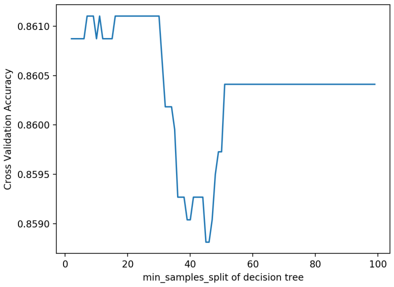
Decision Tree hyperparameter tuning We tested different values for *max_depth* (1-50) and *min_samples_split* (1-100) for our Decision Tree model. The accuracy reaches the highest with *max_depth* being five and gets lower, then flattens. For *min_samples_split*, a range of values from 30 to 40 can all give the best performance. Hence, we choose *max_depth*=5 and *min_samples_split*=35 for the Decision Tree training.

Figure 3: Max Depth Tuning for Decision Tree



Random Forest hyperparameter tuning We tried many values for *n_estimators* and *min_samples_leaf* in the experiments. In general, increasing the values of both these two parameters should enhance the model as long as runtime is not the primary

Figure 4: min_samples_split Tuning for Decision Tree



thing to consider. Yet, since our dataset is not sufficiently large, increasing them to however large we want will only result in performance being hurt. To figure out what specific values work better, we plot the number of *n_estimators* from 1 to 100 (Figure 5) and the value of *min_samples_leaf* from 1 to 30 (Figure 6) versus the accuracy score. The accuracy goes straight up as the number of *n_estimators* increases, but turn flattens after around n = 40. On the other hand, the accuracy only drops as the value of *min_samples_leaf* increases. Therefore, we believe the choice of *n_estimators* = 40 and *min_samples_leaf* =1 is reasonable for our Random Forest model.

Figure 5: Estimators Tuning for Random Forest

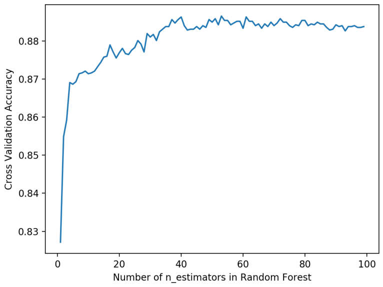
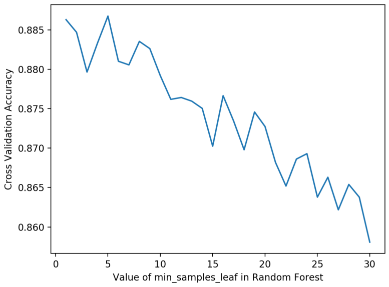


Figure 6: min_samples_leaf Tuning for Random Forest



Other key factors of the experiments We used "entropy" instead of the default "gini" as the criterion for both Decision Tree and Random Forest because we value the amount of information gain on each attribute more than the homogeneity of the labels. Empirically, using "entropy" as the criterion also gives better model than using "gini".

To ensure our results are more accurate, we used 10-fold cross validation technique in our model training and testing. We could not directly apply the cross_val_score function on the Decision Tree and Random Forest model because we need some further handling on the test set (see 2.2.2 ROI score). Instead, we randomly split the train and test set for 10 times and take the average accuracy of all the 10 rounds. We chose test size = 0.1, which is slightly less than the "rule-of-thumb" 80/20 split as our dataset is not that large.

3.3.2 Model Evaluation. We use both micro accuracy and weighted kappa to evaluate the performance of the model. The micro accuracy reflects how good the model is to make correct prediction. The weighted kappa is used because the response variance is ordinal, so we want to assess the similarity of the prediction by the model. For weighted kappa, we set the weight to be quadratic instead of linear because it is more devastating to classify a not profitable movie into a highly profitable movie than into a slightly profitable movie. The quadratic weighted kappa considers this situation and it has mild penalty initially but the penalty gets harsh quickly as the difference increases. By combining micro accuracy and weighted kappa, we will have comprehensive aspects to evaluate each model.

3.3.3 Principal Component Analysis. To identify the best subset of features that better predict the success of the movie, we will apply Principal Component Analysis and Chi-squared test to the dataset. We will be including twelve attributes as well as the class of each movie from both the K-means++ clustering and percentile techniques. After applying both techniques, we will select the most relevant features and feed them into the three classification models. The accuracy and square weighted kappa from inputting the selected features to the models will be analyzed. Before we apply PCA on the dataset, we randomly generate the training and testing datasets and standardize the values. We then apply PCA on the attributes in the training and testing dataset. To find the number of principal components to include in the models, we will first output the explained variance values of each principal component. As shown in figure 8, the first three principal components are on the steep slope. The three components account for about 60% of the explained variance of movies. Within those components, we analyze the weights of each of the features shown in figure 9. The *genres_popularity_score*, *keywords_popularity_score*, *keywords_vote_score*, *casts_popularity_score*, *casts_vote_score*, *directors_popularity_score*, *directors_vote_score*, and *release_year* have the highest weights in the first three components. These components will be used in each of the models to evaluate their performance and the outputs of PCA and Chi-squared test will be analyzed.

3.3.4 Original Features vs Extracted Features. To identify if the extracted features (popularity, vote and ROI scores of genres, keywords, casts, and crews) will increase the prediction, we perform two experiments, one with original features which include *budget*, *release_year*, *release_month*, *runtime*, *popularity*, *vote_average*, and the other one with extracted features which include *budget*, *release_year*, *release_month*, *runtime*, *genres_popularity_score*, *genres_vote_score*, *genres_roi_score*, *keywords_popularity_score*, *keywords_vote_score*, *keywords_roi_score*, *casts_popularity_score*, *casts_vote_score*, *casts_roi_score*, *directors_popularity_score*, *directors_vote_score*, *directors_roi_score*. The ROI-related attributes are extracted from the training data which won't be shown in test data, so they are only used for Decision Tree and Random Forest model training.

4 RESULTS

4.1 Most Important Features

After implementing PCA, we found that the most weighted features were *genres_popularity_score*, *keywords_popularity_score*, *keywords_vote_score*, *casts_popularity_score*, *casts_vote_score*, *directors_popularity_score*, *directors_vote_score*, and *release_year*. The top attributes using Chi-squared test were *budget*, *casts_popularity_score*, *directors_popularity_score*, *runtime*, *keywords_popularity_score*, and *release_year*. Combining the results, we found that *casts_popularity_score*, *casts_vote_score*, *directors_popularity_score*, *budget*, and *runtime* were the most important features for predicting the success of a movie. We used these common features as the input to the model and found that only using these features, the difference is within 0.06 for accuracy and within 0.07 for weighted Kappa score as seen in Table 3. This shows that our selected attributes are sufficient in predicting the movie success instead of using all the attributes. We conclude that *casts_popularity_score*, *casts_vote_score*, *directors_popularity_score*, *budget*, and *runtime* were important features to predict the success of the movie. This result proves that our original hypothesis with budget, casts, and directors as the most important features to predict movie success is correct.

This result matches previous related work that used similar attributes. Although some research had conflicting results on whether or not budget was an important attribute, the attribute had a significantly higher chi-squared value than other attributes, so we believe it is significant. We also had an unexpected significant attribute of runtime. This could be because of the recent increase in movie runtimes and if people are able to sit through a long movie.

Comparing the features selected and the techniques, we see that PCA used more attributes, but got accuracy and weighted kappa scores that were close to the values for Chi-squared. This is because even though we only chose three principal components, many attributes had similar, not very high weights in each principal component. More attributes needed to be selected to better represent the variations of movies. Meanwhile, Chi-squared directly describes the relationship between the features and class, so it was better able to select and reduce the features with greater influence on movie success.

4.2 Attribute Values With Highest ROI

The genres with the highest ROI scores are 'TV Movie', 'History', and 'Western'. The cast and directors with the highest ROI scores are 'Laurel Near', 'T. Max Graham', and 'Jean Lange' and 'Bruce Lee', 'Travis Cluff', and 'Chris Lofing' respectively. This result is interesting because Travis Cluff and Chris Lofing co-directed a movie and Laurel Near, T. Max Graham, and Jean Lange starred in the same movie. This is also very different from our hypothesis. That could be due to the fact that our dataset included movies from 1916 to 2017 and the articles researched did not include data from that long ago. It could also be because we only included 4360 movies in our analysis after data preprocessing despite there being over 45,000 movies to begin with. A huge success or a big flop can influence the ROI score. Some movies may have been produced with a small budget, but had unexpected success, resulting in a large return on investment.

4.3 Improving Model With Extracted Features

From the result shown in Table 4, we found that with the extracted features, the accuracy of Decision Tree and Random Forest greatly improved compared with the accuracy with the original features. The ROI-related attributes are vital in the Decision Tree and Random Forest. The *casts_roi_score* attribute is the root attribute in both the Decision Tree and Random Forest, which means it plays a critical role in classify instance. The extracted features like popularity-related and vote-related attributes could add more information about movie genres, keywords, casts, and directors. For example, information about the popularity of the casts and director in the movie could help in predicting the success of a movie. Extracted features improve the accuracy of model.

The accuracy for KNN model in the data with extracted feature does not improve as much compared with the original features. One of the reasons could be the KNN does not have good performance

Table 3: Model Performance with Selected Features

Accuracy / Weighted Kappa	Decision Tree		Random Forest		KNN	
	Clustering	Percentile	Clustering	Percentile	Clustering	Percentile
Original Features	0.81/0.16	0.39/0.19	0.812/0.29	0.37/0.22	0.31/0.20	0.40/0.24
with Selected Features	0.75/0.26	0.32/0.1	0.84/0.22	0.38/0.20	0.30/0.21	0.38/0.24

Table 4: Model Performance with Extracted Features

Accuracy / Weighted Kappa	Decision Tree		Random Forest		KNN	
	Clustering	Percentile	Clustering	Percentile	Clustering	Percentile
Original Features	0.81/0.16	0.39/0.19	0.812/0.29	0.37/0.22	0.31/0.20	0.40/0.24
with Extracted Features	0.89/0.55	0.59/0.37	0.902/0.77	0.6/0.47	0.32/0.23	0.41/0.25

in data with many attributes. Another possibility is that the ROI-related attributes were not applied to the modeling of KNN.

The best model for predicting the movie success is Random Forest with extracted features, and with the highest weighted kappa score is 0.77. Most of the previous related work also found Random Forest to have the highest accuracy, and it also makes sense for the weighted Kappa score for Random Forest to be the highest among our models.

5 CONCLUSION

Through our experiment, we found that the Random Forest model with all extracted features and class label generation using K-means++ clustering had the best accuracy and weighted kappa score. The Random Forest classifier was also found to be the method with highest accuracy in previous related works, but this is also true for the weighted kappa score in our experiment.

We learned to work with classification models with ordinal classes. Various considerations need to be taken into account and there are evaluation techniques that weigh the cost of the difference between classes like the weighted kappa score. We found that there is not much difference in weighted kappa score when balanced or unbalanced class labels are used by the model, but it is still an important measure for ordinal attributes.

More research could be done on methods for classifying and evaluating ordinal attributes. We could also try to use regression techniques for predicting movie success and compare the performance with our classifiers.

There are many factors to consider when producing or investing in movies and, although the audience's reactions to movies can sometimes be unexpected, focusing on the right parts can ensure that investments will be worth it.

6 SCHEDULE OF ONLINE MEETINGS

We met regularly on Zoom every Thursday. We also had many quick check-in meetings whenever we had updates.

(1) Mar 28 11-12pm
  - Attended by Ian, Guoyi, Xiaohan, Tanya
(2) Mar 31 2-3pm
  - Attended by Ian, Guoyi, Xiaohan, Tanya
(3) April 2 2-3pm
  - Attended by Ian, Guoyi, Xiaohan, Tanya
(4) April 9 2-3pm
  - Attended by Ian, Guoyi, Xiaohan, Tanya
(5) April 16 2-3pm
  - Attended by Ian, Guoyi, Xiaohan, Tanya
(6) April 23 2-3pm
  - Attended by Ian, Guoyi, Xiaohan, Tanya

REFERENCES

[1] IndieWire. 2018. *The 25 Highest-Grossing Directors in the World.* Retrieved April 16, 2020 from https://www.indiewire.com/gallery/highest-grossing-directors-worldwide/

[2] Kyuhan Lee, Jinsoo Park, Iljoo Kim, and Youngseok Choi. 2018. Predicting movie success with machine learning techniques: ways to improve accuracy. *Information Systems Frontiers* 20 (June 2018), 577–588. https://doi.org/10.1007/s10796-016-9689-z

[3] Andrew Lisa. 2020. *The Highest-Grossing Actors of All Time.* Retrieved April 16, 2020 from https://finance.yahoo.com/news/highest-grossing-actors-time-100057140.html

[4] John Lynch. 2017. *The 15 top-earning movie directors of all time at the US box office.* Retrieved April 16, 2020 from https://www.businessinsider.com/highest-earning-movie-directors-of-all-time-us-box-office-2017-8

[5] Q. I. Mahmud, A. Mohaimen, M. S. Islam, and Marium-E-Jannat. 2017. A support vector machine mixed with statistical reasoning approach to predict movie success by analyzing public sentiments. In *2017 20th International Conference of Computer and Information Technology (ICCIT).* Institute of Electrical and Electronics Engineers, Dhaka, Bangladesh, 1–6. https://doi.org/10.1109/ICCITECHN.2017.8281803

[6] K Meenakshi, G Maragatham, Neha Agarwal, and Ishitha Ghosh. 2018. A Data mining Technique for Analyzing and Predicting the success of Movie. *Journal of Physics: Conference Series* 1000, 1 (April 2018), 012100. https://doi.org/10.1088/1742-6596/1000/1/012100

[7] The Numbers. 2020. *Market Share for Each Genre 1995-2020.* Retrieved April 16, 2020 from https://www.the-numbers.com/market/genres

[8] The Numbers. 2020. *Top 100 Stars in Leading Roles at the Domestic Box Office.* Retrieved April 16, 2020 from https://www.the-numbers.com/box-office-star-records/domestic/lifetime-acting/top-grossing-leading-stars

[9] N. Quader, M. O. Gani, D. Chaki, and M. H. Ali. 2017. A machine learning approach to predict movie box-office success. In *2017 20th International Conference of Computer and Information Technology (ICCIT).* Institute of Electrical and Electronics Engineers, Dhaka, Bangladesh, 1–7. https://doi.org/10.1109/ICCITECHN.2017.8281839

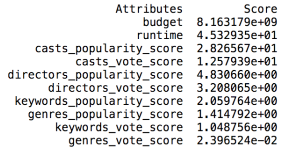
[10] A. A. Sinha, S. V. V. Krishna, R. Shedge, and A. Sinha. 2017. Movie production investment decision system. In *2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS).* Institute of Electrical and Electronics Engineers, Chennai, India, 494–498. https://doi.org/10.1109/ICECDS.2017.8390215

[11] Amy Watson. 2020. Number of movies released in the United States and Canada from 2000 to 2019. Retrieved April 2, 2020 from https://www.statista.com/statistics/187122/movie-releases-in-north-america-since-2001/

APPENDIX

GitHub Repository: https://github.ncsu.edu/tyeh3/CSC522_Project

Figure 7: Chi-Squared Test



```
                 Attributes         Score
                     budget  8.163179e+09
                    runtime  4.532935e+01
     casts_popularity_score  2.826567e+01
          casts_vote_score   1.257939e+01
 directors_popularity_score  4.830660e+00
      directors_vote_score   3.208065e+00
  keywords_popularity_score  2.059764e+00
    genres_popularity_score  1.414792e+00
       keywords_vote_score   1.048756e+00
         genres_vote_score   2.396524e-02
```

Figure 8: PCA Scree Plot

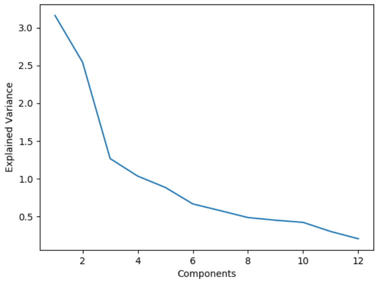
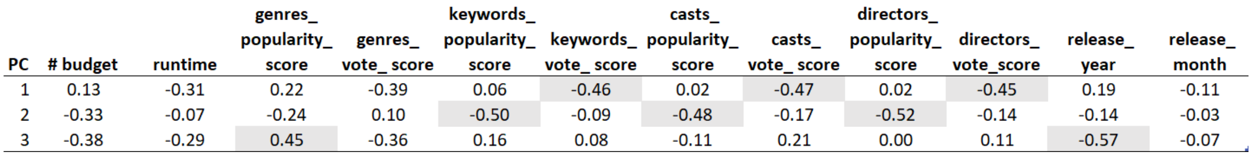


Figure 9: PCA Component Weights

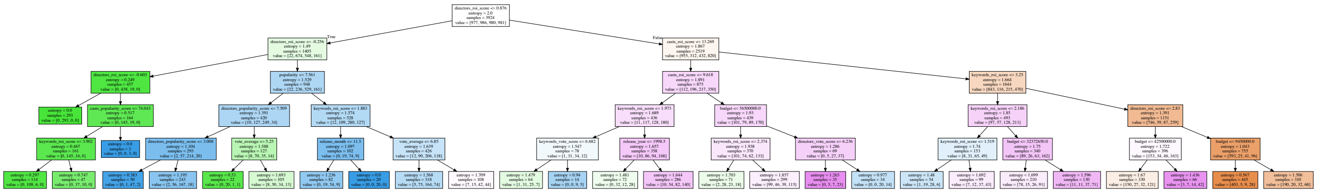


PC	# budget	runtime	genres_ popularity_ score	genres_ vote_ score	keywords_ popularity_ score	keywords_ vote_ score	casts_ popularity_ score	casts_ vote_ score	directors_ popularity_ score	directors_ vote_score	release_ year	release_ month
1	0.13	-0.31	0.22	-0.39	0.06	-0.46	0.02	-0.47	0.02	-0.45	0.19	-0.11
2	-0.33	-0.07	-0.24	0.10	-0.50	-0.09	-0.48	-0.17	-0.52	-0.14	-0.14	-0.03
3	-0.38	-0.29	0.45	-0.36	0.16	0.08	-0.11	0.21	0.00	0.11	-0.57	-0.07

Table 5: Model Performance with Feature Selection Methods

Accuracy / Weighted Kappa	Decision Tree		Random Forest		KNN	
	Clustering	Percentile	Clustering	Percentile	Clustering	Percentile
Features Selected by PCA	0.74/0.14	0.32/0.12	0.84/0.14	0.35/0.09	0.84/0.13	0.36/0.18
Features Selected by Chi-squared	0.76/0.24	0.32/0.18	0.83/0.24	0.36/0.18	0.83/0.24	0.35/0.23

Figure 10: Visualizing the Decision Tree


directors_roi_score <= 0.876 entropy = 2.0 samples = 3924 value = [977, 986, 980, 981]
True
directors_roi_score <= -0.256 entropy = 1.49 samples = 1405 value = [22, 674, 548, 161]
False
casts_roi_score <= 13.269 entropy = 1.867 samples = 2519 value = [955, 312, 432, 820]
directors_roi_score <= -0.603 entropy = 0.249 samples = 457 value = [0, 438, 19, 0]
popularity <= 7.561 entropy = 1.529 samples = 948 value = [22, 236, 529, 161]
casts_roi_score <= 9.618 entropy = 1.891 samples = 875 value = [112, 196, 217, 350]
keywords_roi_score <= 3.25 entropy = 1.664 samples = 1644 value = [843, 116, 215, 470]
entropy = 0.0 samples = 293 value = [0, 293, 0, 0]
casts_popularity_score <= 74.043 entropy = 0.517 samples = 164 value = [0, 145, 19, 0]
directors_popularity_score <= 7.509 entropy = 1.391 samples = 420 value = [10, 127, 249, 34]
keywords_roi_score <= 1.883 entropy = 1.574 samples = 528 value = [12, 109, 280, 127]
keywords_roi_score <= 1.973 entropy = 1.689 samples = 436 value = [11, 117, 128, 180]
budget <= 56500000.0 entropy = 1.93 samples = 439 value = [101, 79, 89, 170]
keywords_roi_score <= 2.186 entropy = 1.85 samples = 493 value = [97, 57, 128, 211]
directors_roi_score <= 2.83 entropy = 1.391 samples = 1151 value = [746, 59, 87, 259]
keywords_roi_score <= 3.902 entropy = 0.467 samples = 161 value = [0, 145, 16, 0]
entropy = 0.0 samples = 3 value = [0, 0, 3, 0]
directors_popularity_score <= 3.008 entropy = 1.104 samples = 293 value = [2, 57, 214, 20]
vote_average <= 5.25 entropy = 1.588 samples = 127 value = [8, 70, 35, 14]
release_month <= 11.5 entropy = 1.097 samples = 102 value = [0, 19, 74, 9]
vote_average <= 6.85 entropy = 1.639 samples = 426 value = [12, 90, 206, 118]
keywords_vote_score <= 6.682 entropy = 1.547 samples = 78 value = [1, 31, 34, 12]
release_year <= 1998.5 entropy = 1.657 samples = 358 value = [10, 86, 94, 168]
keywords_roi_score <= 2.374 entropy = 1.938 samples = 370 value = [101, 74, 62, 133]
directors_vote_score <= 6.236 entropy = 1.286 samples = 69 value = [0, 5, 27, 37]
keywords_roi_score <= 1.519 entropy = 1.74 samples = 153 value = [8, 31, 65, 49]
budget <= 32372650.0 entropy = 1.75 samples = 340 value = [89, 26, 63, 162]
budget <= 42500000.0 entropy = 1.722 samples = 396 value = [153, 34, 46, 163]
budget <= 9450000.0 entropy = 1.043 samples = 755 value = [593, 25, 41, 96]
entropy = 0.297 samples = 114 value = [0, 108, 6, 0]
entropy = 0.747 samples = 47 value = [0, 37, 10, 0]
entropy = 0.383 samples = 50 value = [0, 1, 47, 2]
entropy = 1.195 samples = 243 value = [2, 56, 167, 18]
entropy = 0.53 samples = 22 value = [0, 20, 1, 1]
entropy = 1.693 samples = 105 value = [8, 50, 34, 13]
entropy = 1.236 samples = 82 value = [0, 19, 54, 9]
entropy = 0.0 samples = 20 value = [0, 0, 20, 0]
entropy = 1.568 samples = 318 value = [5, 75, 164, 74]
entropy = 1.709 samples = 108 value = [7, 15, 42, 44]
entropy = 1.479 samples = 64 value = [1, 31, 25, 7]
entropy = 0.94 samples = 14 value = [0, 0, 9, 5]
entropy = 1.481 samples = 72 value = [0, 32, 12, 28]
entropy = 1.644 samples = 286 value = [10, 54, 82, 140]
entropy = 1.703 samples = 71 value = [2, 28, 23, 18]
entropy = 1.857 samples = 299 value = [99, 46, 39, 115]
entropy = 1.263 samples = 35 value = [0, 5, 7, 23]
entropy = 0.977 samples = 34 value = [0, 0, 20, 14]
entropy = 1.48 samples = 54 value = [1, 19, 28, 6]
entropy = 1.692 samples = 99 value = [7, 12, 37, 43]
entropy = 1.699 samples = 210 value = [78, 15, 26, 91]
entropy = 1.596 samples = 130 value = [11, 11, 37, 71]
entropy = 1.67 samples = 330 value = [150, 27, 32, 121]
entropy = 1.436 samples = 66 value = [3, 7, 14, 42]
entropy = 0.567 samples = 445 value = [403, 5, 9, 28]
entropy = 1.506 samples = 310 value = [190, 20, 32, 68]