Predicting Blockbuster Success

Chris Cook Ben Cunningham Emma Reading Mat Sedgewick Kira Tilcock

***Abstract*—We have created an application that predicts domestic box office success. We define box office success as a movie that generates 110% of its budget as gross revenue [1]. The attributes we are using to determine success are: actors, director, writers, gross, distributor, widest release, genre, release month, running time, MPAA rating, and budget. Our algorithm looks to solve the problem of studios losing money on making movies. Making use of our application, studios will be able to predict the success of a movie before production starts by looking at the controllable attributes that influence the success. Our application is able to predict the success or failure of a movie with 65% accuracy before production begins, using a random forest classifier.**

This report was written as part of the requirements for SENG 474 / CSC 578D: Data Mining, offered at the University of Victoria, taught by Dr. Alona Fyshe.

Problem Description

The film industry is one of the most popular industries in North America, grossing over $10.4 billion from over 700 movies released in 2014 [2]. More than two thirds of the Canadian and American population, over 225 million people, attended a movie theatre at least once in 2012, and this number is growing each year [3]. However, it turns out that a large number of Hollywood movies do not end up making a profit. Instead, studios often survive on the few blockbuster hits that gross a large percentage of their budget each year, such as "Star Wars: The Force Awakens" that grossed 157% of its budget in 2015 [4, 5]. Essentially, it is a gamble whether a movie will be a success or failure. Our algorithm looks to solve this problem by reducing the number of movies that lose money domestically. Additionally, our application hopes to uncover new and interesting pairs of attributes that will fare well at the box office, such as unique actor and director combinations.

Our initial motivation for this project was to prevent studios from making unprofitable movies. Making use of our application, studios will be able to look into what attributes make a profitable movie before production begins. Another possible application is to help investors make decisions on whether a given studio will be profitable. For instance, if an investor knew Disney was going to release several successful movies in the upcoming year, they may be more inclined to invest before the stock of the company rose.

Related Work

There are multiple works related to this project, but none of which predict movie success before production begins. In 2002, a number of researchers evaluated whether or not a film's box office performance could be foreseen by estimating the probability of a film's revenue passing a certain threshold [6]. Through analysis of activity on Wikipedia pages, another algorithm was created to predict whether a film flops or becomes a blockbuster [7]. There are also multiple resources estimating lifetime gross of a film based on their success during opening weekend [8]. Google; however, has created an application that predicts box office revenue previous to opening weekend based on the search volume of the movie's trailer [9].

The main difference between these related works and ours, is that none of them predict movie success before the production begins. For example, many of these works incorporate movie reviews, from sources such as Rotten Tomatoes, as well as trailer views and movie hype - the extent to which the movie is publicized, promoted and discussed in the public. Although some of these attributes have been proven to influence a movie's success [10], research also indicates that movie reviews do not have a significant relationship with box office success [11]. Due to this research, we chose to develop our application without the use of reviews and were therefore able to focus on predicting a movie's success before it is even created. Additionally, many of these aforementioned works did not discuss how actors influence a movies success; therefore we decided to include actors in our project.

Approach

Data

For our project there was no public dataset available which had all of the attributes that we deemed necessary for predicting successes. Therefore, we considered different options for acquiring our data from various APIs. Our original plan was to scrape BoxOfficeMojo's (BOM) list of movies for relevant information and then use an API to fill details which BOM did not provide. However, we were rejected access to the Rotten Tomatoes API, and the IMDB API is undocumented. Therefore, those options were then thrown out. Finally, we created a new dataset by scraping information relevant to us from BoxOfficeMojo.com [12].

This was done using Python and the BeautifulSoup library [13].

Research from the past 10 years indicates that differences within attributes, such as genre and MPAA rating, may greatly affect a movie's success. For example, comedy movies have grossed more over the last 10 years than any other genre [14] and PG-13 movies generally gross more than R or G-rated movies [15]. It is from this information that we decided what attributes to consider when creating our application. The attributes we used to predict success were actors, director, writer, gross, distributor, month released, genre, runtime, MPAA rating, budget, and widest release. In order to maintain a reasonable data set, we limited our scope to domestic gross. We also originally planned to limit our widest release to movies that were shown in over 3000 theaters, because box office movies are usually associated with heavily marketed and mass released movies. However, since limited releases can also boast large box office gains [11] and our definition of success is a percentage opposed to a monetary value, we decided against limiting our dataset in this way.

We originally stored our data into two .csv files, one with only actors and the other with all the remaining attributes. We separated the actors from the rest of our data set due to the number of actors being variable and thus causing discrepancies when indexing through our data. We then created a Python program to clean these .csv files. This program cleaned our dataset by taking numerical values out of strings and removing rows with "N/A" values. This program also combined our actor file with our remaining attributes. In addition, we looked only at movies that had a budget of over $1,000,000. Before cleaning our dataset we had approximately 1800 movies and 742 actors stored in two separate files. Now, we have one .csv file containing information from 1261 movies to be used as input to our algorithm.

Algorithm

After scraping and cleaning the dataset, we ran six classifiers on our data. We used the classifiers from the Scikit-Learn Python library to determine whether each movie would be a success or a failure. As previously discussed, we define success as a movie making 110% of its budget. Our six classifiers were: Gaussian Naive Bayes, Multinomial Naive Bayes, Bernoulli Naive Bayes, Decision Trees, Random Forests, and SVM. We ran each classifier 30 times, changing actor frequency each time. An actor's frequency is the number of times an actor appears in our dataset. For example, if actor frequency was set to five, only actors appearing in five or more movies within our dataset would be considered. We divided our dataset into a training set and validation set, using a 60/40 split respectively. We experimented with different splits, such as 70/30, however these different splits did not improve our accuracy. Therefore, we chose to continue with a 60/40 split throughout our project. The results obtained from these classifiers represent the probability of successfully predicting if a film would be a success or failure. Based on our initial results from each classifier, we continued to make further adjustments to those that gave us the highest accuracy. This is discussed in the results and discussion sections of this report.

Results

We first ran each of our six classifiers across all 30 actor frequencies in order to compare predictions across each classifier. The comparisons between the six classifiers across each actor frequency are shown in Figure 1. Shown in the SVM graph of Figure 1, the accuracy we obtained using SVM showed a clear pattern as number of actors increased. With SVM, as we increased our actor frequency, our accuracy increased until reaching a peak accuracy of 58.6% at an actor frequency of 17. Also shown in Figure 1, decision trees and random forests produced inconsistent results. This lead us to believe that actor frequency did not play as large of a role in predicting a movie's success when using those classifiers.

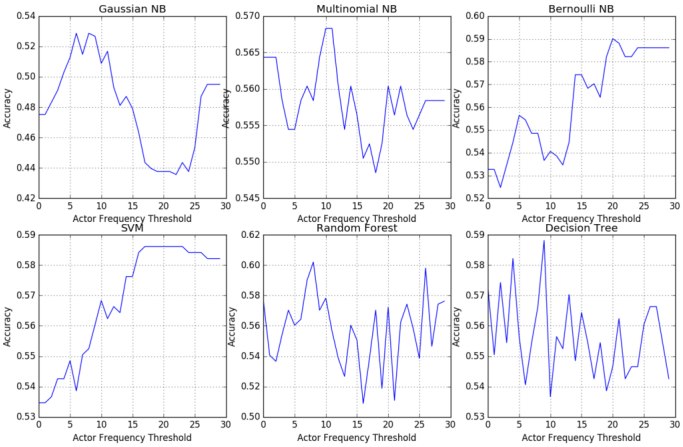

Figure 1. Initial results across all six classifiers: Gaussian Naive Bayes, Multinomial Naive Bayes, Bernoulli Naive Bayes, SVM, Random Forest, and Decision Tree.

Table 1 shows the highest accuracy rates we obtained from each classifier during the initial run of our data. As shown in Table 1, random forests, with an actor frequency of 8, produced our most accurate results at 60.1%. Because, random forests produced the most accurate predictions, we continued to explore random forests.

Classifier	Accuracy	Actor Frequency Threshold
Gaussian NB	0.528712871287	6
Multinomial NB	0.568316831683	10
Bernoulli NB	0.590099009901	20
SVM	0.586138613861	17
Random Forest	0.60198019802	8
Decision Tree	0.588118811881	9

Table 1. Initial highest accuracy rates and associated actor frequency across all six classifiers.

We started exploring random forests by increasing the number of trees; we started with ten and added more from there, testing values such as 100, 500, and 1000. The best results were achieved using 1000 trees, which produced an accuracy of 62.97%. While experimenting with the number of trees in random forests, the ideal actor frequency threshold was inconsistent, but highest accuracies were generally achieved with an actor frequency threshold of between one and nine. This implies that including more actors of lower frequency, as opposed to fewer actors with greater frequency, in our dataset is more useful in predicting box-office success.

The random forest classifier in scikit learn has a number of parameters that we adjusted to increase our accuracy [17]. The n_jobs variable was used to parallelize across multiple threads and this made the program run marginally faster. When max_depth was set to low, the best performance was seen when all actors were used, but all accuracies were relatively poor. We anticipated that setting max_depth would build simpler trees and therefore make our model less prone to overfitting; however, we found that not restricting the classifier from creating bigger trees likely allowed for useful relationships between the attributes to be found that ended up increasing the model's accuracy.

The max_features variable originally defaulted to ten, but when the variable was set to None, the accuracy with 500 trees increased to 64.76%. This is shown in Figure 2. For other classification algorithms that can not handle continuous unbinned values, the number of theatres variable had to be taken out; however, since random forests can handle such values, the variable was added back into the dataset and the accuracy was further increased to 65.15% - the best accuracy achieved thus far.

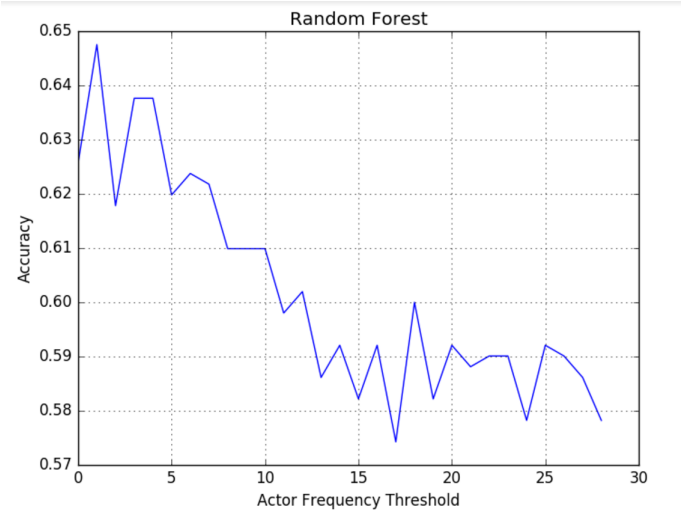

Figure 2. Accuracy across all actor frequencies using random forests with 500 trees and "max_features" set to "None". The ideal actor frequency threshold was found to be 2.

When choosing each classifier we initially expected our most accurate results to come from SVM. We expected this because given a set of training examples SVM is able to assign new information into one of two categories. As discussed earlier, we divided our dataset into a training and test set, 60/40 respectively, and we expected our SVM classifier to learn from this split and to give us fairly accurate results. We also expected to get accurate results using decision trees and random forest. Specifically, we expected to get a higher accuracy with random forests over decision trees, as random forests iterate over all simpler decision trees in the forest and return the majority prediction from all of those trees. We also expected random forests to give a higher accuracy since they are more robust to overfitting than other classifiers.

We did not expect any of the Naïve Bayes classifiers to give us an accurate prediction as Naïve Bayes assumes the independence of features and many of our features are interconnected and dependent on each other. Specifically, we did not expect Bernoulli Naïve Bayes to fit our data, as Bernoulli assumes each value to be binary values. However, it was interesting to see how Bernoulli gave us the highest accuracy of the Naïve Bayes classifiers, even though we figured the model was a poor match for our data.

Discussion

The dataset we created and cleaned turned out to be useful for our purposes. Currently, the information available is disorganized and requires access to multiple API's as it is not together in one place. Because our dataset conveniently provides all the information together in one place, we believe other people working on similar projects may find our dataset useful. Although our dataset provides a lot of information, movies are very complex and there are a number of factors that may have been useful in predicting

box office success. For example, it may have been useful to look at plot keywords and whether the movie is part of an established franchise such as Harry Potter and Star Wars.

However, looking into other attributes, such as franchise, would likely lead to having to consider a number of other factors. For example, whether all movies within the franchise were written and directed by the same people. There are a large number of factors that may affect the profitability of a movie, but since the complexity of looking into every variable would be endless, we decided to draw the line at which attributes to include in our application based on research discussion previously.

One of the interesting things that we discovered through running our algorithms is the effect of actors on a films success. Although, intuitively, actors should play a role in the success of a film, we learned it is more difficult than expected to understand their influence. As discussed earlier, the related works did not train their classification models on actors. We understand now that this may be due to the variability and uncertainty of how an actor will influence a movie's success. We also learned that overfitting is easier to accomplish than we thought, especially when taking a large numbers of variables into account. We saw overfitting in the graphs plotted for some of our models, and it may have been worth the time to try taking out some of the variables we trained on to see if the model would become less prone to overfitting.

Conclusion and Future Work

It is clear that a movie's success is determined by much more than obvious attributes such as actors, directors, and genre. Although, we did determine that some of these attributes do influence the success of a movie, in hindsight we realize that the driving factor behind a movie's success is more than just the sum of it's parts. It may be actually impossible to accurately determine a movie's success without knowing the buzz/hype and reviews surrounding a release. Additionally, after becoming aware of the fact that the movie industry survives on few big box office hits to make up the differences that are lost on other movies [4], our definition of success may not have been the best choice. In the future, we may do additional research and reconsider defining success as a movie reaching 110% of its budget.

If time had permitted we would have liked to fully implement linear regression to determine a monetary value for which each movie's profit would be. We began to implement this, but too many changes to our dataset needed to be made and unfortunately the few necessary values we began to create did not make sense. If time had permitted we would have taken the time to further fine tune our dataset to work well with linear regression. Additionally, we would have liked to further refine our algorithms to achieve an accuracy of 70%. We would also like to further understand which of our training variables had little to no effect on the classification of movies with the hope that if they were removed, our model would be less likely to overfit. We might also try finding a way to determine whether or not a movie is part of a franchise, and possibly try looking at plot keywords in hopes that they might be a good depiction as to whether or not a movie would be a box office success.

Task Distribution

We assigned "team leads" for each of our main project tasks. The lead of each task was responsible for directing and delegating smaller tasks to each member of the team. The lead was also be responsible for editing and putting each members work together into the final product. Ben and Chris were team leads for dataset creation and algorithms, Mathew was team lead for cleaning and filtering the data, and Emma and Kira were team leads for the reports and presentation. It is important to note that each member of our team contributed to each section of our project. Our github can be found at https://github.com/ben-cunningham/SENG474.

References

[1] Iowa Now, "Predicting box office boffo or bomb", 2016. [Online] Available: http://now.uiowa.edu/2016/02/boffo-or-bom. [Accessed: 26- Feb-2016].

[2] "2014 Yearly Box Office Results - Box Office Mojo", Boxofficemojo.com, 2016. [Online]. Available: http://www.boxofficemojo.com/yearly/chart/?yr=2014. [Accessed: 01- Apr- 2016].

[3] "Theatrical Market Statistics Report", Motion Picture Association of Amerca, Inc., 2012. [Online]. Available: http://www.mpaa.org/wp-content/uploads/2014/03/2012-Theatrical-Market-Statistics-Report.pdf. [Access: 01-Apr-2016].

[4] "Star Wars: The Force Awakens (2015)" - Box Office Mojo, Boxofficemojo.com, 2016. [Online]. Available: http://www.boxofficemojo.com/movies/?id=starwars7.htm. [Accessed: 01- Apr- 2016].

[5] Priceonomics.com, 2016. [Online] Available: http://priceonomics.com/why-do-all-hollywood-movies-lose-money/. [Accessed: 26- Feb- 2016].

[6] A. Colins, et al., "What makes a blockbuster? Economic analysis of film success in the United Kingdom", Managerial and Decision Economics, vol. 23, John Wiley Sons Ltd, 2002, pp. 343-354.

[7] M. Mestyan, et al. (2013). "Early Prediction of Movie Box Office Success Based on Wikipedia Activity Big Data". [Online]. Available FTP: http://journals.plos.org/plosone/article?id=10.1371/journal.pone.007122 6

[8] P. Perner. "Machine Learning and Data Mining in Pattern Recognition," New York, NY, Final Rep., July 2013.

[9] K. Acuna, "Google Says It Can Predict Which Films Will Be Huge Box-Office Hits",Business Insider, 2016. [Online]. Available: http://www.businessinsider.com/google-study-can-predict-success-of-movies-2013-6. [Accessed: 01- Apr- 2016].

[10] M. Woolfe. "Movie Review Aggregator Ratings Have No Relationship With Box Office Success", 2016. [On-

line]. Available: http://minimaxir.com/2016/01/movie-revenue-ratings/

[11] A. Kennedy. “Predicting Box Office Success: Do Critical Reviews Really Matter?”, 2016. [online]. Available: https://www.stat.berkeley.edu/ aldous/157/Old_Projects/kennedy.pdf. [Accessed: 01- Apr- 2016].

[12] “Box Office Mojo”, Boxofficemojo.com, 2016. [Online]. Available: http://www.boxofficemojo.com/. [Accessed: 24- Jan- 2016].

[13] L. Richardson, “Beautiful Soup: We called him Tortoise because he taught us.”,Crummy.com, 2016. [Online]. Available: https://www.crummy.com/software/BeautifulSoup/. [Accessed: 24- Jan- 2016].

[14] “The Numbers - Leading Genres 1995-2016”, The-numbers.com, 2016. [Online]. Available: http://www.the-numbers.com/market/genres. [Accessed: 01- Apr- 2016].

[15] “The Numbers - Leading MPAA Ratings 1995-2016”, The-numbers.com, 2016. [Online]. Available: http://www.the-numbers.com/market/mpaa-ratings. [Accessed: 01- Apr- 2016].

[16] “3.2.4.3.1. sklearn.ensemble.RandomForestClassifier — scikit-learn 0.17.1 documentation”, Scikit-learn.org, 2016. [Online]. Available: http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html. [Accessed: 17- Mar- 2016].