Timothy Nguyen

CPSC 483

Th 7:00-9:45 PM

Classifying Movie Success

Table of Contents

[Table of Contents 1](#_Toc531796110)

[1. Introduction 1](#_Toc531796111)

[2. Data Collection 1](#_Toc531796112)

[3. Preprocessing 2](#_Toc531796113)

[a. Data Cleaning 2](#_Toc531796114)

[b. Feature Extraction 2](#_Toc531796115)

[c. Feature Selection 3](#_Toc531796116)

[d. Final Dataset 4](#_Toc531796117)

[4. Data Analysis 5](#_Toc531796118)

[5. Classification Models 8](#_Toc531796119)

[a. Overview 8](#_Toc531796120)

[b. Logistic Regression 8](#_Toc531796121)

[c. K-nearest Neighbors 9](#_Toc531796122)

[d. Decision Tree 10](#_Toc531796123)

[e. Random Forest 11](#_Toc531796124)

[6. Conclusion 11](#_Toc531796125)

[References 12](#_Toc531796126)

# 1. Introduction

In this project, I will be attempting to predict if a movie is successful. A movie’s success can be interpreted in various ways: number of awards, total revenue, and box office. But a basic guideline is that a movie is considered profitable if the revenue is twice the budget [1]. So, success will be a boolean feature that is true if the movie revenue exceeds twice the budget. I will be attempting to predict this value using several classification algorithms: logistic regression, K-nearest neighbors, decision tree, and random forest.

# 2. Data Collection

While there are many movie datasets available, I wanted to start from scratch. So, I collected my own data from The Movie Database API (TMDB) [2]. The API was accessed using the python library tmdbsimple. The movies were collected by iterating by year using the discover method from the API.

The dataset consists of 375,377 movies dating from 1884 to 2018. It consists of the following features:

* id – unique movie ID for TMDB
* title
* original title – alternate title for foreign countries
* release date
* budget – rounded to nearest dollar
* revenue – rounded to nearest dollar
* popularity – value updated daily by TMDB. Considers page views, votes, activity, etc.
* runtime – rounded to nearest minute
* vote\_average – average user ratings on a scale of 1-10
* vote\_count – number of voters
* adult – boolean value. True if movie is a pornographic film
* status – state of movie production. i.e. released, in production, canceled
* genres
* production\_companies
* production\_countries
* certification\_US – movie rating that determines suitability by viewer age [3]

# 3. Preprocessing

Before this raw dataset is used for classification, it must be modified so it can be suitable for analysis. Many features are missing from many movies and it must be handled. New features will be created to help analysis. And analyzing the data will determine which features are useful for classification.

## Data Cleaning

In the raw dataset, over 75% of movies have a missing budget and revenue. And about 100,000 movies have a null runtime. All these features are necessary for analysis, so the movie is dropped if any of these features are missing.

For this classification to be useful in determining the success for new movies, movies released before 1970 are removed. Also, movies with a vote\_count less than 5 are removed to avoid the bias of only a few voters. For example, if a movie has a vote\_average of 2 but only has 1 voter, it may not accurately represent the score.

A movie is also dropped if it is a pornographic film. This is because nature of these movies is extremely different than theatric movies.

## Feature Extraction

Some features are created using existing features. The target variable, ‘success’, is created. This is a boolean value that is true if the movie revenue exceeds twice the budget and false otherwise. The feature ‘year’ is created from release\_date because a single year is much easier to process compared to an entire date. Also, month and day are not relevant to a movie’s success.

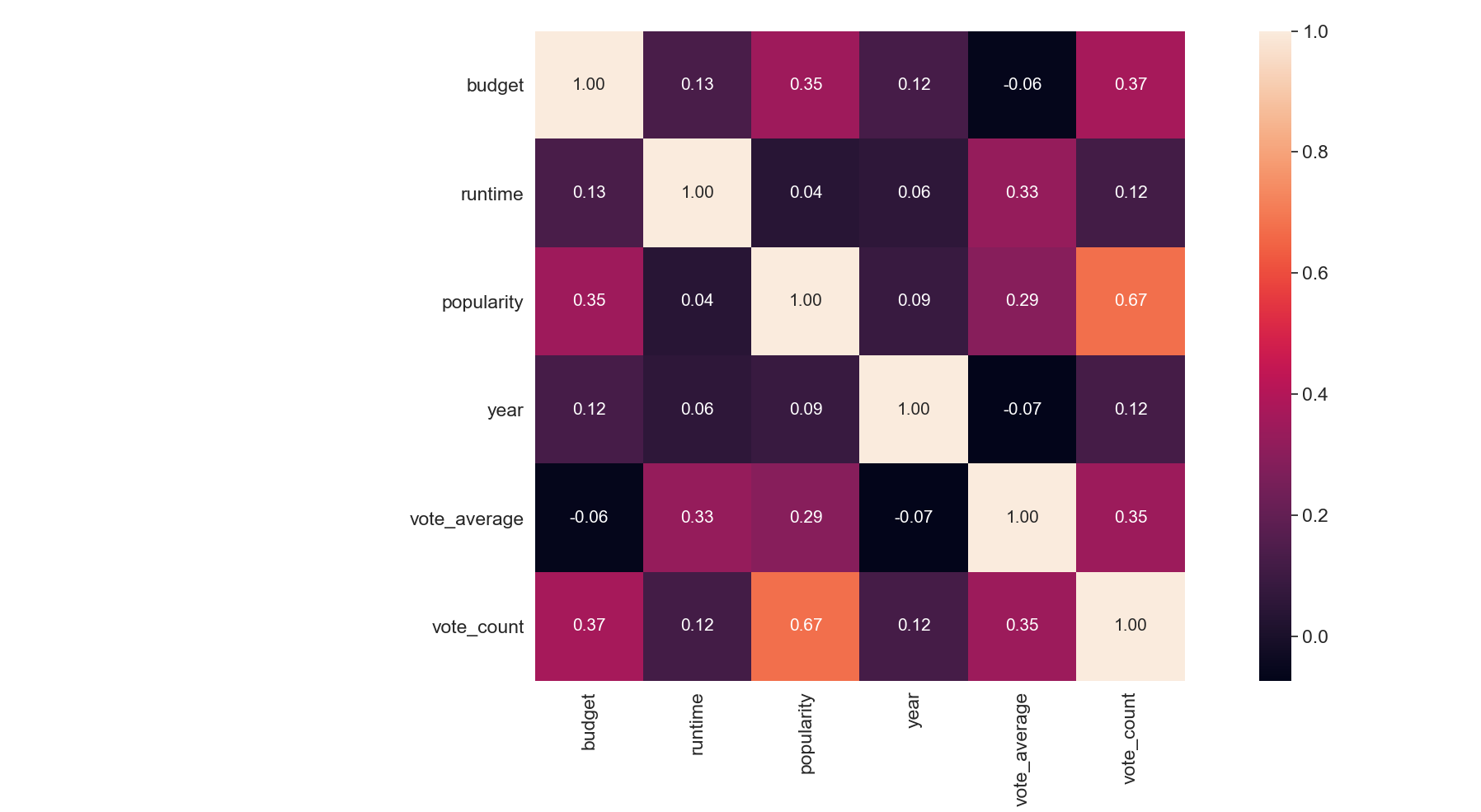
The most significant genre, country, and production company are extracted from their respective lists. This is to improve processing time and because a feature that is a list of categorical labels requires a more complex analysis.

Obscure foreign films are not wanted because it will lower the accuracy of the model. So, movies with countries that appear in the dataset less than 5 times are removed. Doing this decreased the number of unique countries in the dataset from 65 to 37. This was not a huge loss because only 60 movies were removed.

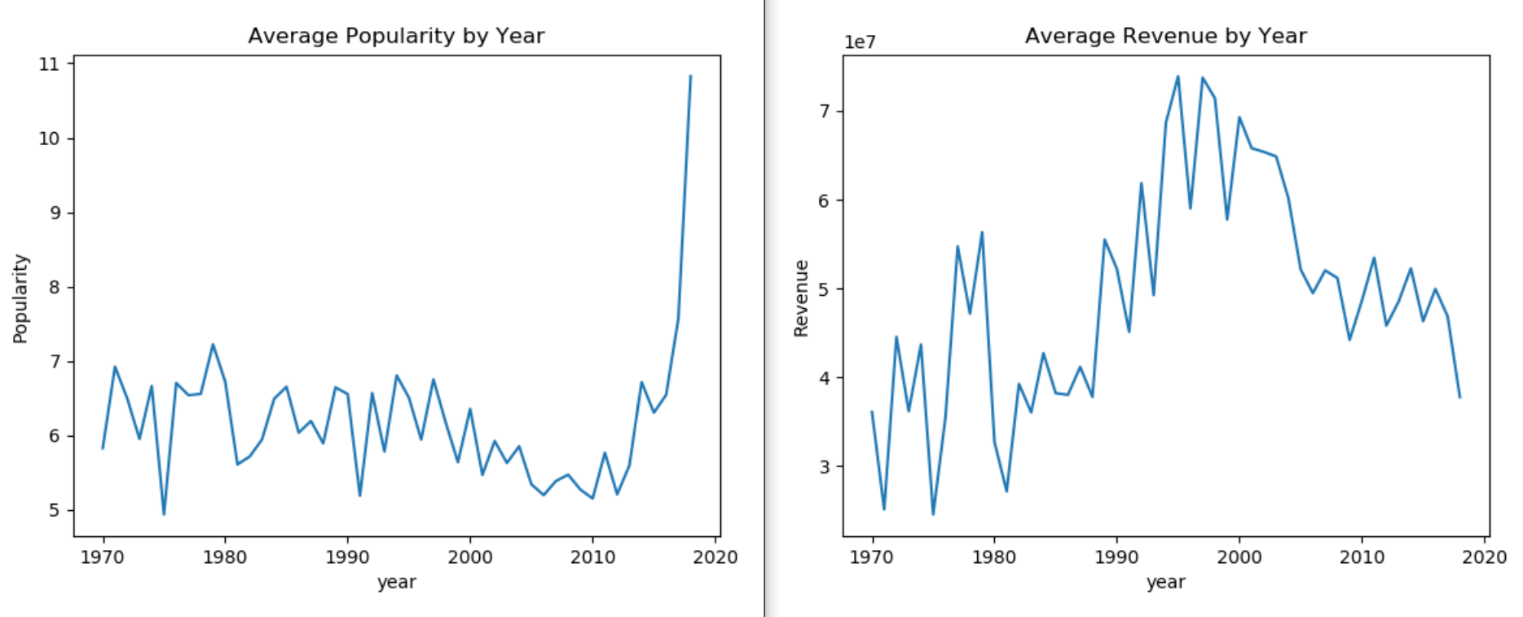
There are 2266 unique production companies in the current dataset. Nearly of third of the dataset consists of movies that have a unique production company. This is not desired because there will be too much variance in the production company feature. Removing movies with a unique production company is not ideal because it removes a third of the entire dataset. Instead, the production company feature is removed from the dataset.

## Feature Selection

The following heatmap shows the correlation coefficient between continuous variables. If a value is too high, the corresponding features are redundant and unnecessary. This can handled by removing one of the features or combining both of the features



All the values seem to be relatively low except for vote\_count and popularity with a correlation coefficient of 0.67. Looking at the trend in popularity will show how to handle this.



The graph on the left shows popularity, which seems to be heavily skewed toward the most recent movies. The graph on the right shows that the revenue does not share any pattern with popularity. So, popularity is not useful for classification and is dropped from the dataset.

In summary, the following features are dropped for not being useful in predicting success:

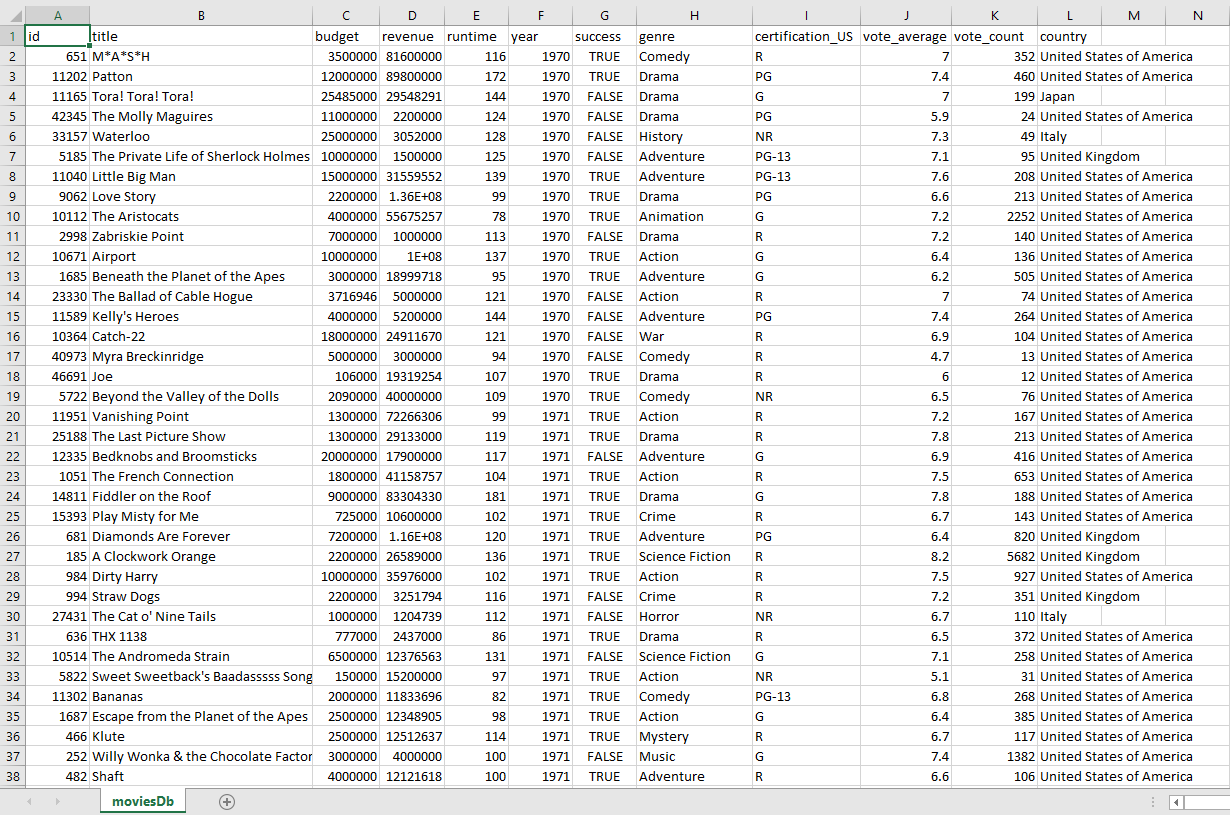
* original\_title – alternate title is not relevant
* release\_date – replaced by year
* popularity – value is skewed toward most recent film
* status – dataset assumes all movies are released
* genres – replaced by single genre
* production\_countries – replaced by single country
* production\_companies – too much variance in companies
* adult – pornographic movies are removed from the dataset

## Final Dataset

After all this preprocessing, the final dataset consists of the following features:

* Identifiers
  + id – unique numeric id given by TMDB
  + title – movie title
* Predictors
  + budget – total money used to produce movie
  + runtime – total movie time in minutes
  + year – release year
  + vote\_average – mean community score
  + vote\_count – number of votes attributing to vote\_average
  + genre – most significant genre
  + country – most significant production country
  + certification\_US – movie rating that determines suitability by viewer age [3]
* Target Variable
  + success – true if (budget \* 2 >= revenue)

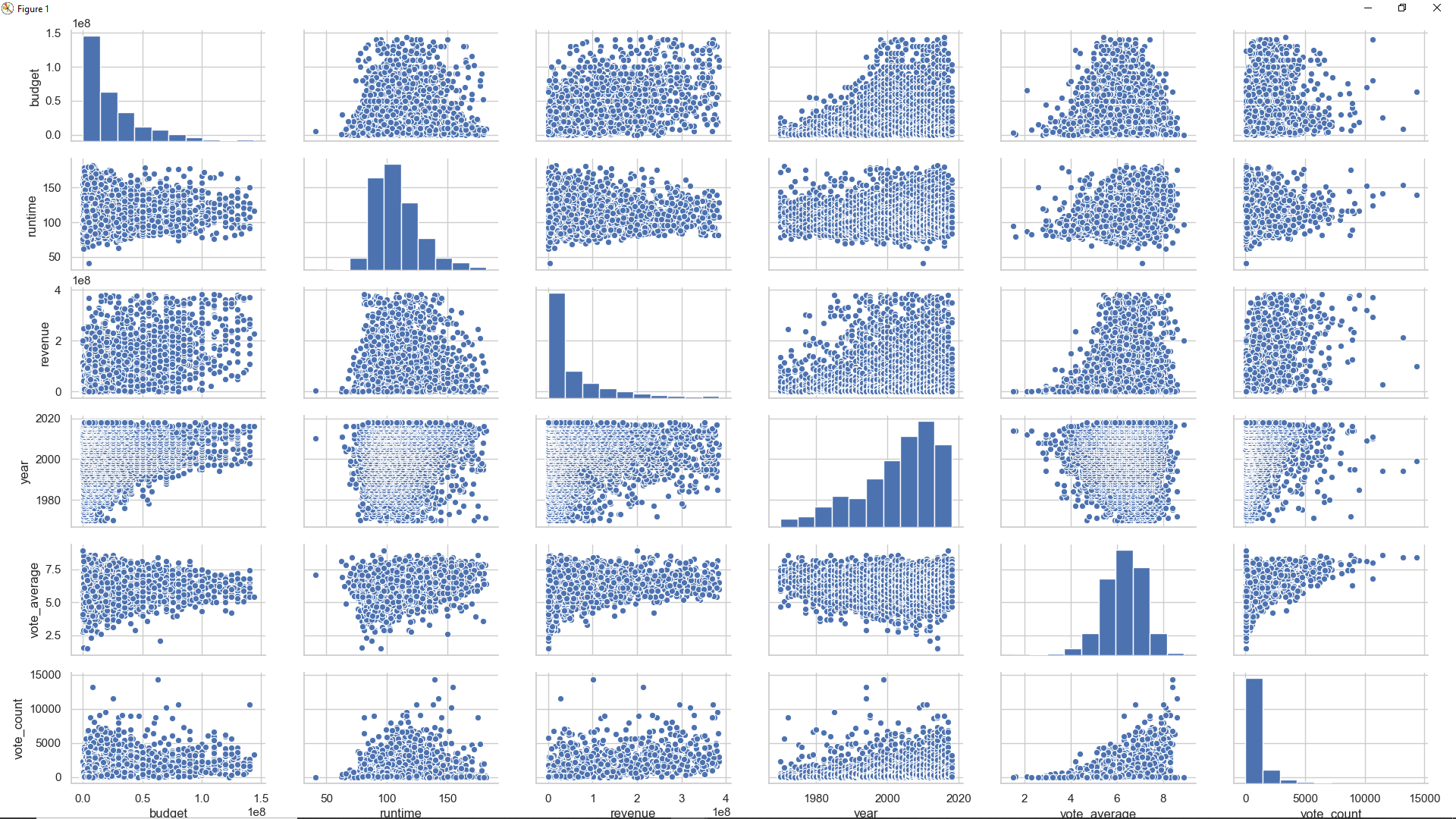
The following is a sample of the final dataset:



The final dataset consists of about 5100 movies, which is 0.13% (5000 / 375,000) of the original dataset. A huge majority of this loss is due to movies with missing budget and revenue. The rest of the preprocessing decisions only attributed to a small fraction of this loss. The dataset is now ready for analysis.

# 4. Data Analysis

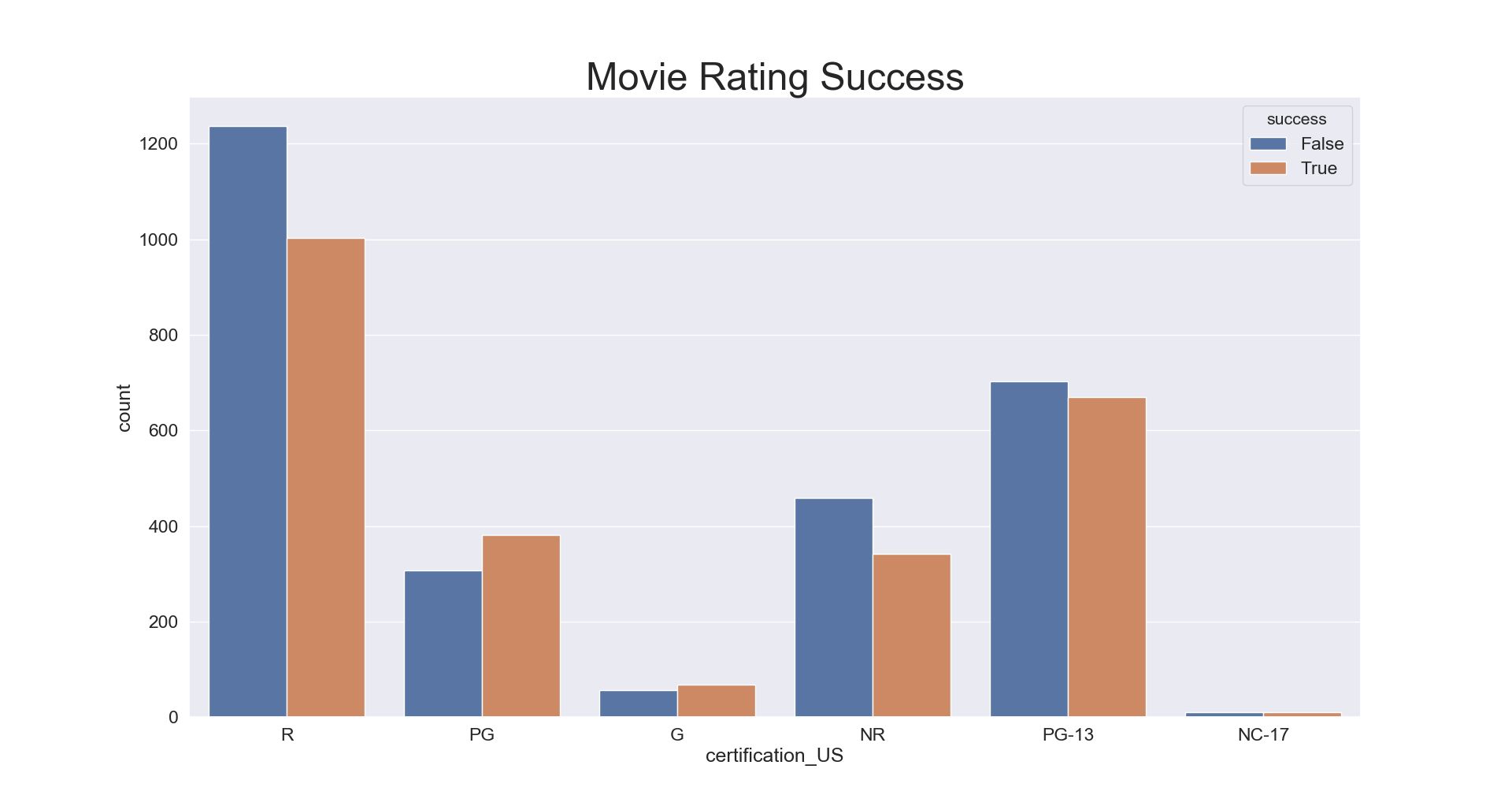
Before classification begins, the data must be analyzed to understand the decisions made by the algorithms. Below is a scatterplot matrix showing the relationships between the continuous features.



The revenue and budget barplots are skewed left, which means that the dataset is mostly made of movies with low budget and revenue. The year barplot is skewed right, which shows that most of the movies in the dataset are more modern.

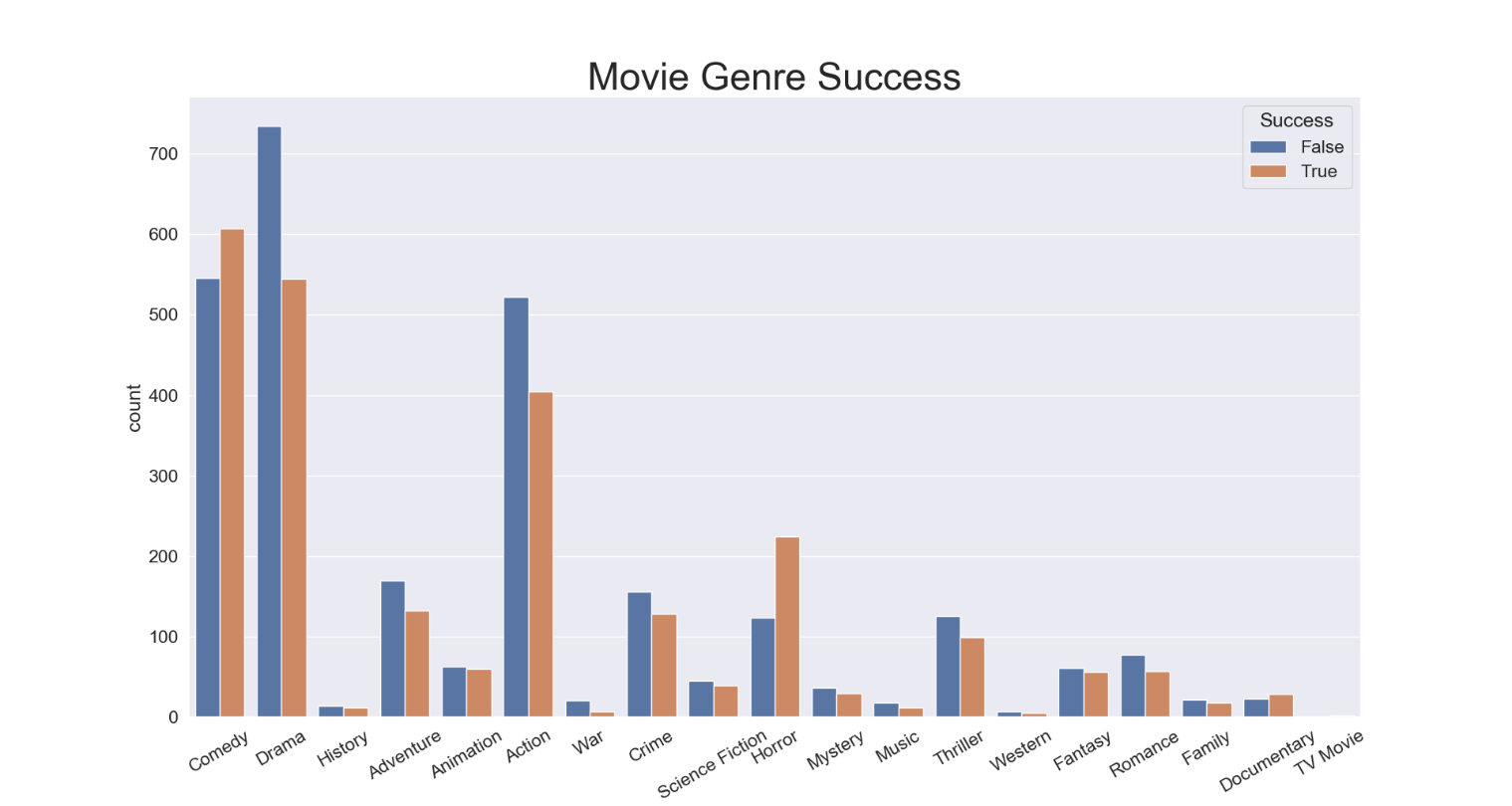
The scatterplots of runtime seem to have a loose linear relationship that is not increasing nor decreasing. And the scatterplots of (budget \* year) and (revenue \* year) have an increasing relationship. This means that if a movie is more recent, it is more likely to spend and make more money, which makes sense due to inflation.

Below is a double bar plot that shows the frequency of movie certifications and their success:



It seems that R-rated movies make up most of the dataset. But the only certification types where success is greater than failure is PG and G-rated movies, which are made for children.

Below is a double bar plot that shows the frequency of movie genres and their success:



It seems that comedy, drama, and action movies make up most of the dataset. The genres with a positive success ratio are comedy, horror, and documentary.

# 5. Classification Models

## a. Overview

To predict a movie’s success, four classification algorithms are used: logistic regression, K-nearest neighbors, decision tree, and random forest. Each classification algorithm will be using the same target value and predictors. The target value is success, a boolean value that is true if the revenue exceeds twice the budget. The predictors are listed below:

* budget – total money required to produce movie
* runtime – total movie time in minutes
* year – release year
* vote\_average – mean community score
* vote\_count – number of votes attributing to vote\_average
* genre – most significant genre
* country – most significant production country
* certification\_US – movie rating that determines suitability by viewer age [3]

For the categorical predictors (genre, country, certification\_US), dummy variables are created so the algorithm can process them.

For each algorithm, sklearn’s grid search is used to find the hyper-parameters with the best accuracy. The grid search also uses a k-fold cross-validation where k=10 to ensure that the entirety of the dataset is tested. The model accuracy is measured using the mean of the test scores from the cross-validation. The mean training score will also be shown to determine if the model is overfitted or underfitted

## b. Logistic Regression

Logistic regression is promising because it works best when the target variable is a boolean value, and our target variable, success, is boolean. The hyper-parameter C is the inverse of regularization strength, which decides the number of elements to penalize for misclassification. The solver selects the variation of the logistic regression algorithm.

The following hyper-parameter values are tested:

* C – [104, 10-3, 10-2,, … 104]
* solver – [newton-cg, lbfgs, sag, saga]

Below is a table of the results of the grid search. There are too many hyper-parameter combinations to show, so only the best results of each solver are shown.

|  |  |  |  |
| --- | --- | --- | --- |
| Solver | Mean Test Score | Mean Train Score | C |
| sag | 0.5268 | 0.5266 | 1 |
| lbfgs | 0.5266 | 0.5266 | 1 |
| saga | 0.5270 | 0.5268 | 10 |
| newton-cg | 0.7047 | 0.7055 | 1000 |

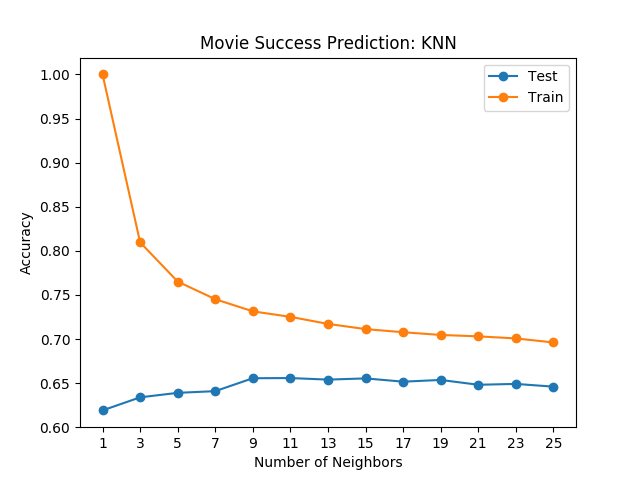
The best logistic regression model has a mean test score of 0.7047 and a mean train score of 0.7055. The difference between these values is very small, so the model is neither overfitted nor underfitted. The hyper-parameters of this model are {C: 1000, solver: ‘newton-cg’}.

## c. K-nearest Neighbors

The K-nearest neighbors algorithm works by classifying a datapoint using the majority class of nearest points around it. The Euclidian distance metric is sufficient for this dataset. Since the target variable is boolean, the number of neighbors must odd to avoid ties in classification. So, the hyper-parameter tested is listed below:

* n\_neighbors – [1, 3, 5, … 25]

Below is a visualization of hyper-parameter tuning KNN:



The graph shows that the difference in test score and training score is larger when the number of neighbors is low. This shows that the model is more overfitted when the number of neighbors is lower. But the test score and training score seem to converge as the number of neighbors increase.

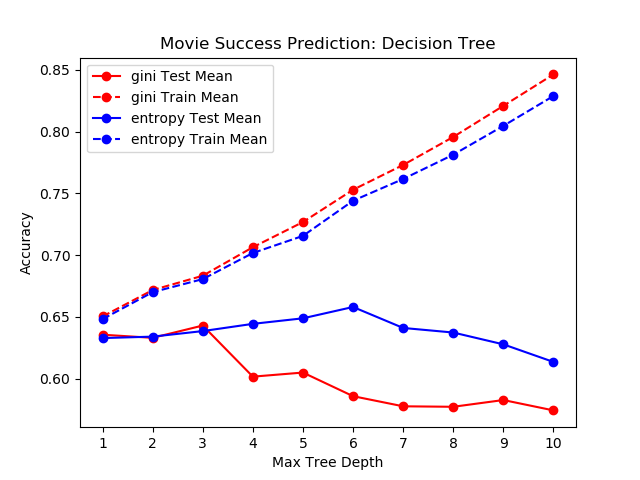
The best KNN model has a mean test score of 0.6557 and a mean train score of 0.7251. The hyper-parameter of this model is {n\_neighbors: 11}.

## d. Decision Tree

The decision tree works by setting a division threshold for each feature to predict the target variable. There are two main criteria for determining when to split a tree node, gini and entropy. A decision tree tends to get overfitted when there are no boundaries for tree size. So, the max\_depth of the tree must be used to restrict the size and avoid overfitting. The following is a list of the hyper-parameters tested:

* criterion – [gini, entropy]
* max\_depth – [1, 2, 3, … 10]

Below is a visualization hyper-parameter tuning the decision tree:



The graph shows that the test score decreases and training score increases as the max tree depth increases. This shows that the tree more overfitted as it grows larger.

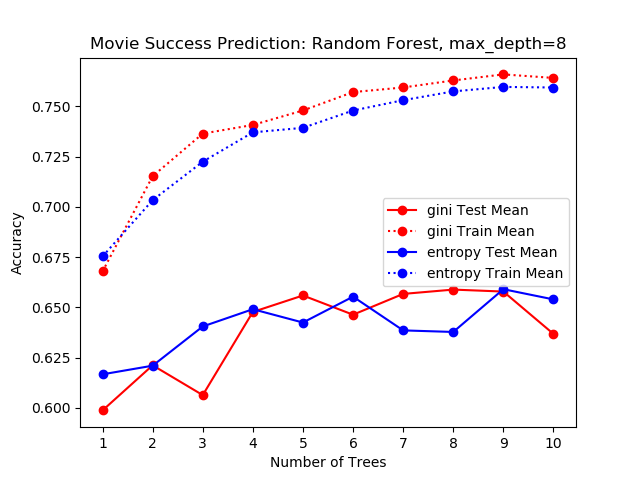
The best decision tree model has a mean test score of 0.6581 and mean training score of 0.7441. The hyper-parameters of this model are{criterion: entropy, max\_depth: 6}

## e. Random Forest

The random forest algorithm is an ensemble version of the decision tree algorithm, which means that the model uses multiple decision trees. Because of this, the random forest accuracy is expected to be better than the previous decision tree. This algorithm uses the same hyper-parameters as the decision tree except for number of trees (n\_estimators). The hyper-parameters tested are as follows:

* criterion – [gini, entropy]
* max\_depth – [1, 2, 3, … 10]
* n\_estimators – [1, 2, 3, … 10]

Below is a visualization hyper-parameter tuning the random forest. This visualization is only limited to the max\_depth that had the highest accuracy because there would be too many variations shown otherwise.



As expected, the random forest accuracy is greater than the decision tree accuracy. The best random forest model has a mean test score of 0.6735 and mean training score of 0.7558. The hyper-parameters of this model are {criterion: gini, max\_depth: 8, n\_estimators: 9}

# 6. Conclusion

Logistic regression is the best algorithm for classifying movie success because it yields the highest mean test score (0.7047). It also has the lowest distance between mean test score and mean training score (0.0008), which means that it is fit just right.

# References

[1] <https://io9.gizmodo.com/5747305/how-much-money-does-a-movie-need-to-make-to-be-profitable>

[2] <https://developers.themoviedb.org/3/getting-started/introduction>

[3] <https://www.mpaa.org/film-ratings/>

[4] [Source Code](https://github.com/timothyng-164/Movie-Success-Predictor)

[5] [Presentation Slides](https://docs.google.com/presentation/d/1vmgTafgC7aaFDvgAOm0iZpUN5gIFbun6x-Wh_JhSIdo/edit#slide=id.g47f1dcdda0_0_831)