Predicting Box Office Revenue of Movies Using YouTube Trailer Data

# Introduction

YouTube has become one of the most preferred platforms for movie studios to connect with potential viewers by posting trailers months in advance of a movie’s release. A large part of a movie’s box office success can be attributed to how effective it’s trailer is in generating interest.

Data collected from the YouTube movie trailer can provide useful insights into predicting a movie’s total box office revenue. By evaluating the performance (number of views, number of likes, dislikes, comments, sentiment of comments) of a movie trailer, studio’s and marketers can create more “efficient” trailers and theater owners can plan which movies to play, both of which can drive increased total box office revenue.

**Problem:**

The goal of the project is to predict the domestic (North America) total box office revenue of a movie after it’s opening weekend, based on the performance of it’s official YouTube movie trailer.

**Solution:**  
To solve this problem, I will be using the results of linear regression, support vector regression, and regression trees, and choosing the model that produces the most accurate results.

# Literature Review

There have been several studies performed by researchers in an effort to predict the financial success of a movie at the box office.

Rahim, Islam et al. [1] used linear regression, polynomial regression, gradient boosted tree and simple linear regression to predict the total gross income of a movie during it’s theatrical run. Data was collected from YouTube and Box Office Mojo. Using the values returned in R-squared, linear regression proved to be the best model with R-squared = 0.891.

In a study by Apala, Jose et al. [2] data was collected from social media and web sources including Twitter, YouTube and the Internet Movie Database. Prediction was based on several decision factors from IMDb (such as popularity of a genre), followers, count of actors and actresses from Twitter, and sentiment analysis of YouTube viewers comments. The authors labeled the prediction in three classes -- Hit, Neutral, and Flop – using Weka’s K-Means clustering tool. Data was collected for 35 movies in early 2013. The authors found the following patterns: 1) the popularity of leading actress is significant to a movies success, 2) the genre (in particular the genre action) and if the movie is a sequel determines the movies success, and 3) a movie that is not action or adventure and also has an actor with a low popularity could be patterns for a flop. Surprisingly, the authors found that sentiment score and view and comment counts did not have significance in their prediction.

Asur and Huberman [3] used Twitter sentiment analysis and a measure called tweet rate (average of tweets per hour one week prior to the movie’s release) to predict box office revenue for movies. 2.89 million tweets were extracted from Twitter for 24 movies over a period of three months using keywords from the movie’s title. Using linear regression, average tweet rate model predicted significant results with an R-squared of 0.80. They also performed a linear regression of the time series of the tweet-rate seven days before the release. This generated an R-squared of 0.93. When adding sentiment analysis as a variable to both equations, the authors found that R-squared climbed to 0.92 for the tweet rate average model, and to 0.93 to the time series tweet rate model.

Zhang and Skiena [4] used quantitative news data generated by Lydia (system of large scale news analysis) to predict movie grosses. They used two different models, regression and k-nearest neighbors for a dataset of 498 movies. The authors found the following: 1) movie news is highly correlated with movie grosses, 2) movie gross prediction using news data, when compared to prediction models using IMDB numeric data, produced similar performance, and 3) regression and k-nearest neighbor classifiers can be used for movie gross prediction (produced significant results). Regression models were found to have better performance with low-grossing movies, while K-NN models had better performance with high grossing movies.

Sentiment analysis from blogs was used in a study by Sadikov, Parameswaran et al. [5] for prediction of movie sales, user ratings and critics ratings using correlation, clustering and time-series analysis to study which features are the best predictors. Data was collected from spinn3r.com from November 2007 – November 2008 which contains practically all blog posts published on the internet during the aforementioned time frame (approximately 1.5TB of compressed XML). Features collected included movie references from blogs (using sentiment analysis), and distributor, genre, and budget from the-numbers.com. Movie sales were found to be more predictable than user and critics ratings (rest of the paper focused on predicting movie sales since predicting critic and user ratings had very low correlation to movie gross. Movie references in a blog’s post was found to be the most predictive feature, with correlation of 0.85 to a movie’s box office gross.

Ramesh and Sharda [6] trained a neural network to process pre-release data, such as quality and popularity variables (MPAA rating, competition, star value, genre, technical effects, sequel and number of screens). Movies were classified into nine categories, ranging from “flop” to “blockbuster.” The model was designed to predict the expected revenue range of a movie before its theatrical release. The neural network was able to only classify 36.9% of the movies correctly, while 75.2% of the movies at most one class away from being correct.

# Dataset

The dataset was self-curated since there was not a publicly available one to solve the problem. Two different data sources were used to collect the required information for the dataset -- Box Office Mojo and YouTube. Box Office Mojo was used to collect movie related information (total gross, production budget, number of theaters, genre etc.) while YouTube data was used to collect movie trailer information (number of views, likes, dislikes, comments, sentiment analysis etc).

Data was collected for movies that were released in the years 2012 to 2018 and that opened on greater than 600 screens in North America on it’s opening weekend. The 600 screen cut off was implemented as it is the generally accepted figure used to define whether a movie is a wide, or nationwide, release. Movies playing on less than 600 screens are defined as limited release, playing in only a few markets. They were excluded from the dataset since they can give inaccurate prediction results. An example of this is the movie American Sniper, released in 2014, on only 4 screens during it’s opening weekend. American Sniper’s opening weekend box office (a variable in the dataset) was under $1 million but it expanded the number of screens to well over 600 in subsequent weeks and became the biggest box office revenue earner that year. YouTube trailer views for American Sniper were very high in anticipation of the movie’s wide release, and, using the assumption that most trailer views occur before the end of a movies opening weekend, this would lead to inaccuracies in trying to predict the target variable, TotalGross. Movies playing on less than 600 screens were also excluded because data for the variable budget became seldom and not available.

Data for movies released before 2012 was used as a cutoff as the average number of YouTube videos watched per day has grown from 2 billion to 5 billion from 2011-2018[7][8]. Collecting data before 2012 could give inaccurate results, as movies released in recent years tend to have higher video views which can skew distributions.

The dataset contains 19 features and 869 observations (before removing non-impactful variables):

|  |  |  |
| --- | --- | --- |
| Variable Name | Type | Description |
| Trailer\_id | chr | YouTube movie trailer id |
| Title | chr | title of the movie |
| ReleaseYear | factor | release year of movie from 2012-2018 |
| Studio | chr | studio that released the movie |
| OpeningTheaters | int | number of screens the movie played on its opening weekend date |
| ViewCount | int | total number of comments for the YouTube movie trailer |
| LikeCount | int | total number of likes for the YouTube movie trailer |
| DislikeCount | int | total number of dis-likes for the YouTube movie trailer |
| LikeDislikeRatio | int | ratio of likes to dislikes (= LikeCount/DislikeCount) |
| CommentCount | int | total number of comments for the YouTube movie trailer |
| positiveWordsBL | int | total number of positive words in comments section of YouTube trailer video using Bing-Liu lexicon |
| negativeWordsBL | int | total number of negative words in comments section of YoutTube movie trailer using Bing-Liu lexicon |
| netSentimentBL | int | positive – negative words using Bing-Liu lexicon |
| scoreAbsoluteAFINN | int | total score using AFINN lexicon lexicon which assigns words with a score that runs between -5 and 5, with negative scores indicating negative sentiment and positive scores indicating positive sentiment. Total was calculated by summing the score of each word |
| budget | int | budget of the movie in $ |
| OpeningWeekendGross | int | box office revenue of the movie during its first weekend of release |
| genre | factor | genre of the movie |
| mpaaRating | factor | censor rating that the Motion Picture Association of America assigns to a movie |
| TotalGross | integer | target variable -- total box office revenue of a movie during its entire theatrical release. |

*Figure 1: Dataset*

Dataset can be found [here](https://github.com/Jeffrey-Russell/CKME136)

# Approach

## Step 1: Data Collection

Two data sources were used to collect the required information to build the dataset. The first one is Box Office Mojo and the second one is YouTube.

Box Office Mojo was used to collect data for movie related variables. The variables Title, ReleaseYear, Studio, TotalGross, OpeningWeekendGross, OpeningTheaters were collected by copying and pasting data into an Excel spreadsheet from the website’s tables for movie domestic gross box office for each year. To gather data for the variables budget, mpaaRating, and genre, a webscraper was coded in R using the RVest package.

YouTube was used to collect data for trailer statistics. Trailer\_id was collected by performing a search on the website using the movie title, release year and the expression “official trailer.” For example, the search query for the movie Avengers:Infinity War, was “Avengers Infinity War 2018 Official Trailer.” Search results turned up multiple trailers for each movie. The trailer with the most views was selected for it’s video id which was a necessary input to gather trailer statistics. This was done for each of the 869 observations in the dataset.

The R package “Tuber” was used to connect to the YouTube API to extract movie trailer data with Trailer\_id as an input. Data for the variables ViewCount, LikeCount, DislikeCount, LikeDislikeRatio (= LikeCount/DislikeCount), CommentCount were extracted using this method.

R code available [here](https://github.com/Jeffrey-Russell/CKME136/blob/master/Youtube%20Comments%20Extraction.R) and [here](https://github.com/Jeffrey-Russell/CKME136/blob/master/Scraping%20Box%20Office%20Mojo.R)

## Step 2: Sentiment Analysis

Four variables in the dataset were created using sentiment analysis of the Youtube movie trailer comments section.

The variables positiveWordsBL and negativeWordsBL is a sum of the positive and negative words using the Bing-Liu lexicon, which categorizes words in a binary fashion into positive and negative categories:

**get\_sentiments**("bing")

## # A tibble: 6,788 x 2

## word sentiment

## <chr> <chr>

## 1 2-faced negative

## 2 2-faces negative

## 3 a+ positive

## 4 abnormal negative

## 5 abolish negative

## 6 abominable negative

## 7 abominably negative

## 8 abominate negative

## 9 abomination negative

## 10 abort negative

## # ... with 6,778 more rows

The variable scoreAbsoluteAFINN was created using the AFIN lexicon, which assigns words with a score between -5 and 5, with negative scores indicating negative sentiment and positive scores indicating positive sentiment:

**get\_sentiments**("afinn")

## # A tibble: 2,476 x 2

## word score

## <chr> <int>

## 1 abandon -2

## 2 abandoned -2

## 3 abandons -2

## 4 abducted -2

## 5 abduction -2

## 6 abductions -2

## 7 abhor -3

## 8 abhorred -3

## 9 abhorrent -3

## 10 abhors -3

## # ... with 2,466 more rows

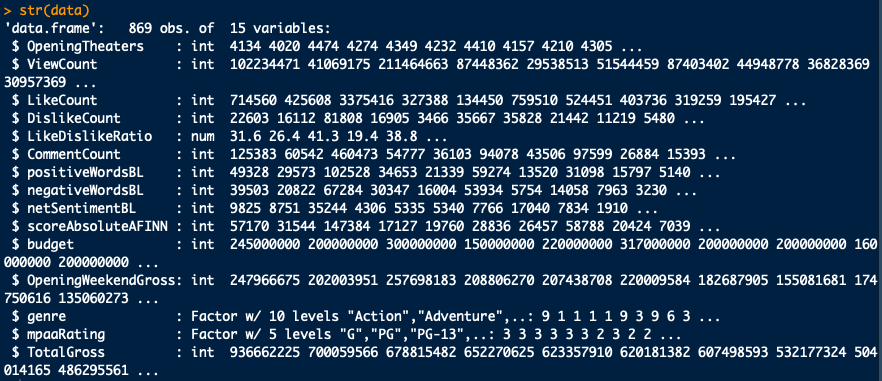
All of this information is tabulated in the sentiments dataset in R. The Tidytext package has a function, get\_sentiments(), that retrieves specific sentiment lexicons without the columns that are not used in that lexicon.

R code available [here](https://github.com/Jeffrey-Russell/CKME136/blob/master/Sentiment%20Scores.R).

## Step 3: Data Cleaning

Features were converted to the most appropriate format and non-impactful variables were removed (Trailer\_id, Title, ReleaseYear, Studio).

The final dataset, after cleaning, has 15 variables and 869 observations with the following structure:



*Figure 2: Dataset structure*

## R code available [here](https://github.com/Jeffrey-Russell/CKME136/blob/master/Data%20Cleaning%20%26%20EDA.R)

## Step 4: Descriptive Analysis

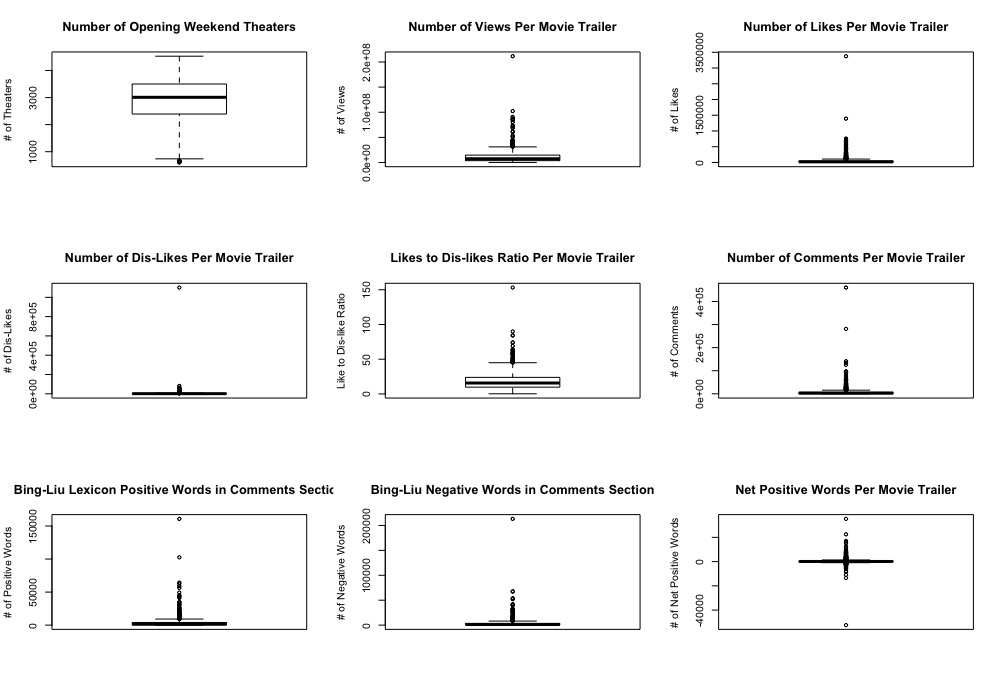
The following table shows the statistics for all of the numerical features:

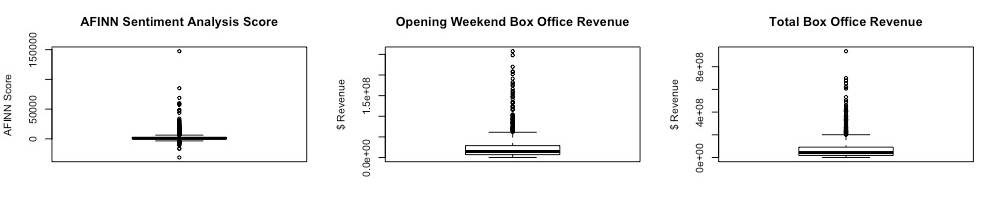
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Features | Min | Pctl(25) | Mean | Median | Std. Dev | Pctl(75) | Max |
| OpeningTheaters | 602 | 2,392 | 2,874 | 3,010 | 884 | 3,501 | 4,529 |
| ViewCount | 144,193 | 3,738,300 | 11,710,365 | 7,529,875 | 14,418,040 | 14,756,674 | 211,464,663 |
| LikeCount | 42 | 7,941 | 54,174 | 19,339 | 151,727 | 48,298 | 3,375,416 |
| DislikeCount | 4 | 555 | 4,717 | 1,268 | 37,912 | 3,176 | 1,101,616 |
| LikeDislikeRatio | 0 | 10 | 18 | 16 | 13 | 24 | 153 |
| CommentCount | 1 | 938 | 7,911 | 2,655 | 22,642 | 6,965 | 460,473 |
| positiveWordsBL | 1 | 518 | 4,120 | 1,423 | 9,483 | 4,034 | 160,790 |
| negativeWordsBL | 1 | 435 | 3,705 | 1,220 | 9,745 | 3,514 | 213,196 |
| netSentimentBL | -52,406 | -55 | 415 | 104 | 3,037 | 519 | 35,244 |
| scoreAbsoluteAFINN | -31,284 | 124 | 2,843 | 744 | 9,031 | 2,607 | 147,384 |
| budget | 100,000 | 15,000,000 | 55,940,967 | 35,000,000 | 58,554,925 | 75,000,000 | 317,000,000 |
| OpeningWeekendGross | 205,842 | 7,004,254 | 25,194,492 | 14,685,305 | 33,205,949 | 28,871,140 | 257,698,183 |
| TotalGross | 321,910 | 18,119,640 | 76,572,931 | 43,577,636 | 100,668,791 | 91,221,830 | 936,662,225 |

*Figure 3: Descriptive statistics of numerical variables*

For all features except OpeningTheaters, the mean is significantly greater than the median which implies that the distributions are skewed. Furthermore, the mean is highly sensitive to outliers and the discrepancy between the mean and median of most features in the dataset indicates the presence of them. The difference between the third quartile and the maximum values for most features is significantly greater than the difference between the minimum and first quartile values. This finding also indicates the presence of outliers in the third quartile to max values range, which will pull the mean higher than the median.

To see if there are potential problems in the dataset, it is best to visualize the numeric features with boxplots:





*Figure 4: Box plots of features*

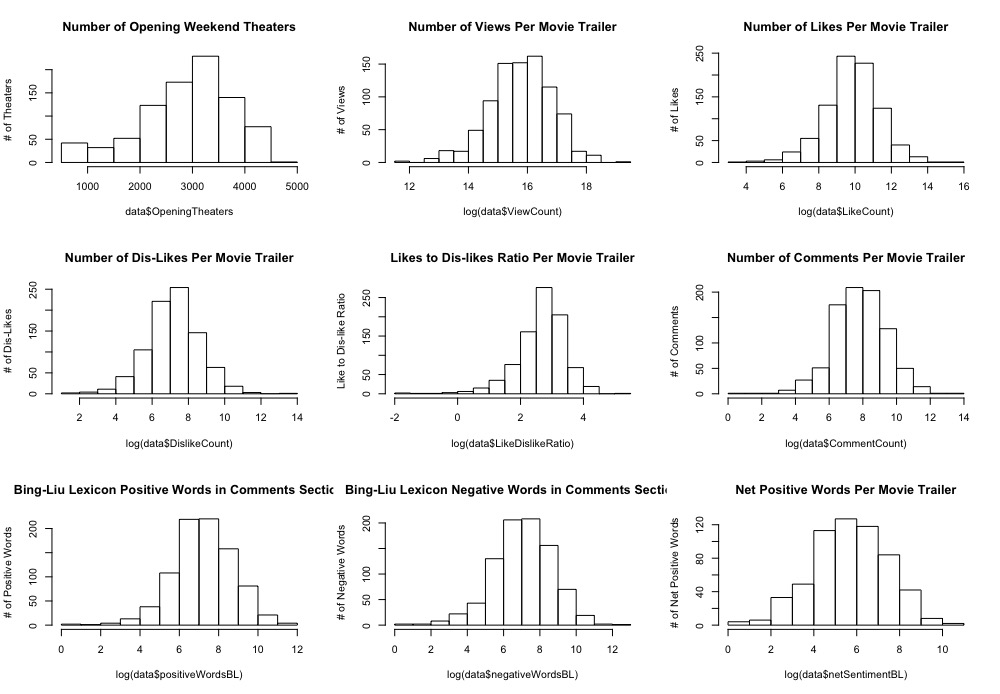
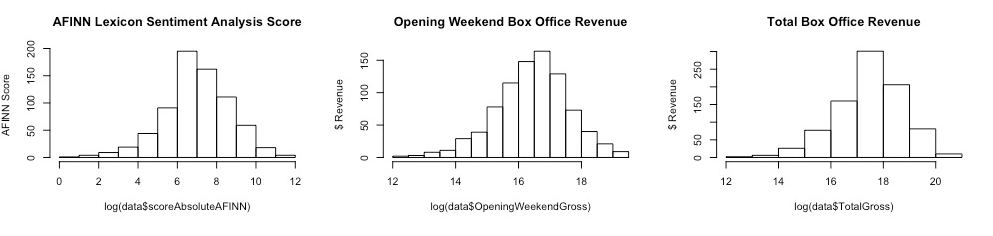
The boxplots reveal that all features except OpeningWeekendTheaters, have a significant number of outliers on the high end of the data. The outliers are responsible for the earlier finding that showed that the mean being significantly higher than the median for most features. The high number of outliers for each feature indicates a potential problem and will most likely need to be remedied to achieve accurate results in the linear regression prediction model.

The outliers are observations of the “mega-blockbuster” movies. These are movies that gross very high revenue and account for most of the overall market box office revenue, even though they represent a small portion of the total number of movies released. Thus, outliers will not be removed, since they are legitimate observations.

To visualize the spread of the numeric features, histograms were created:

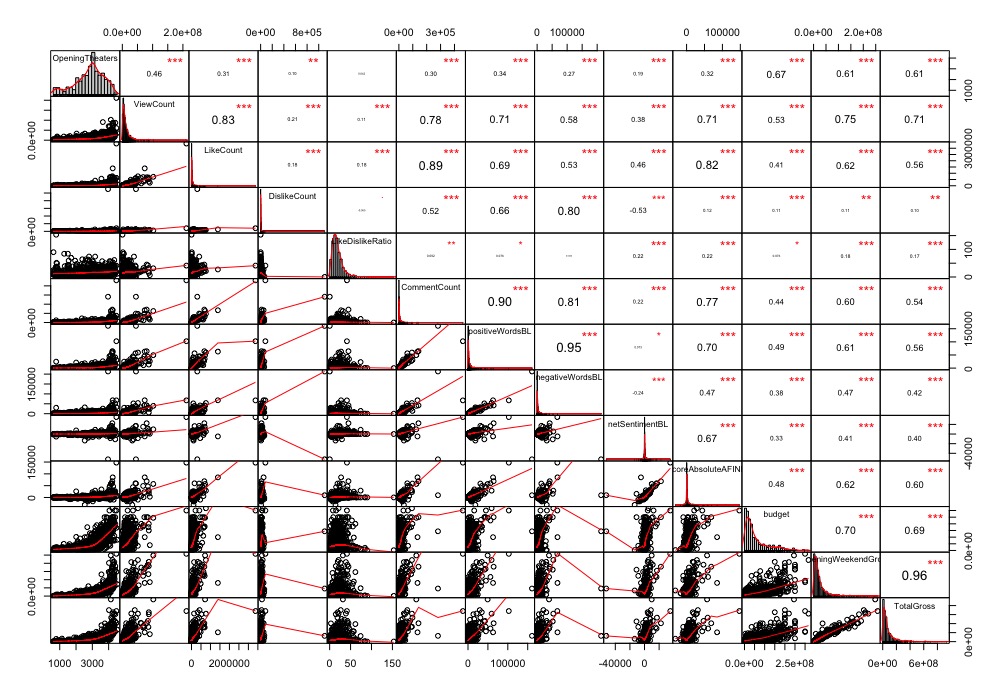
# *Figure 5: Histogram of features*

# The histogram plots clearly shows that all features except OpeningWeekendGross are heavily right skewed. A highly skewed dataset will affect prediction results for a linear regression model, which is one of the models to be tested in this report. To improve results for a linear regression model, a log transformation is applied to the data:

  
  
*Figure 6: Log transformation of features*

After transforming all features except OpeningWeekendTheaters, skewness has been greatly reduced. Most features have a normal or near normal distribution after log transformation was applied.

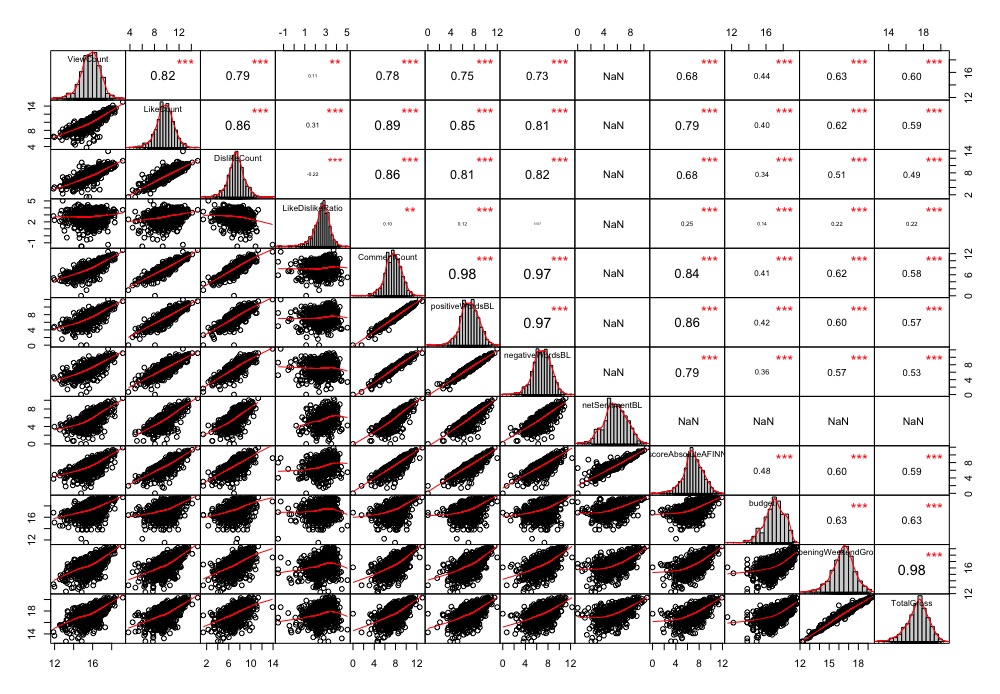
The following figure shows the Pearson correlation matrix of the dataset, which can help determine the relationships between features. It can help spot multi-collinearity between features and whether there is a linear relationship between the dependent and independent variables, both of which are assumptions that fall under a linear regression model.

  
*Figure 7: Correlation matrix of dataset*

The Pearson correlation matrix shows that several features are highly correlated with each other, which could indicate multicollinearity. The pairings ViewCount and LikeCount, ViewCount and CommentCount, LikeCount and CommentCount, LikeCount and scoreAbsoluteAFINN, DislikeCount and NegativeWords, CommentCount and positiveWordsBL, CommentCount and negativewordsBL, positiveWordsBL and negativeWordsBL all have correlation with each other that is approximately greater than 0.75. Assessment for multicollinearity, which can give inaccurate linear regression prediction results, will be discussed in the next section, using the Variance Inflation Factor test.

OpeningWeekendGross is highly correlated with the dependent variable, TotalGross, with 0.96, but this is expected since it is a subset of TotalGross. OpeningTheaters, ViewCount, LikeCount, CommentCount, positiveWordsBL, scoreAbsoluteAFINN, budget, all have greater than moderate positive (>0.5) correlation with the dependent variable TotalGross.

The plot shows evidence of non-linearity between the independent features and dependent feature, TotalGross, which was expected since most of the data is right skewed. When plotting the correlation matrix using log transformation, linearity becomes much more evident:



*Figure 8: Log Pearson correlation matrix*

## Step 5: Feature Selection

Feature selection was implemented in order to reduce error due to multicollinearity, which cannot be present in a linear regression model. To identify collinear predictors, the Variance Inflator Factor test in the VIF package in R was used to determine if the high correlations between several dependent features identified above are of any concern. The results are as follows:

|  |  |
| --- | --- |
| Feature | VIF |
| OpeningTheaters | 2.23572 |
| log(ViewCount) | 2.04286 |
| log(LikeCount) | 49.56032 |
| log(DislikeCount) | 48.21288 |
| log(LikeDislikeRatio) | 25.78419 |
| log(CommentCount) | 7.17641 |
| log(positiveWordsBL) | 6.65128 |
| log(negativeWordsBL) | 6.52727 |
| netSentimentBL | 1.43450 |
| scoreAbsoluteAFINN | 1.54788 |
| log(budget) | 1.67376 |
| log(OpeningWeekendGross) | 2.14425 |
| genre | 1.11507 |
| mpaaRating | 1.15057 |

*Figure 9: Variance Inflation Factor of features in dataset*

Several features have a VIF greater than 5, which is considered a threshold for moderate collinearity. VIF greater than 5 is considered to have high collinearity. Log(LikeCount), log(DislikeCount) and log(LikeDislikeRatio) are extremely high with VIF greater than 25. Log(CommentCount, log(positiveWordsBL) and log(negativeWordsBL) have a collinearity just above 5. These features were shown earlier to also have very high correlation between them.

To reduce multicollinearity, log(DislikeCount), log(LikeDislikeRatio), log(negativeWordsBL), and Log(CommentCount) were removed. Running the VIF function again produces the following results:

|  |  |
| --- | --- |
| Feature | VIF |
| OpeningTheaters | 2.22214 |
| log(ViewCount) | 1.95711 |
| log(LikeCount) | 2.37841 |
| log(positiveWordsBL) | 2.14931 |
| netSentimentBL | 1.39176 |
| scoreAbsoluteAFINN | 1.53303 |
| log(budget) | 1.66693 |
| log(OpeningWeekendGross) | 2.07889 |
| genre | 1.10419 |
| mpaaRating | 1.14189 |

*Figure 10: Reduced feature set using Variance Inflation Factor*

Multicollinearity has been greatly reduced. All features are well below 5, the threshold for moderate collinearity. The dataset at this stage is ready for linear regression.

R code available [here](https://github.com/Jeffrey-Russell/CKME136/blob/master/Feature%20Selection.R).

## Step 6: Model Selection

The goal of the problem is to predict the total box office revenue of a movie after it’s opening weekend using YouTube movie trailer statistics and Box Office Mojo movie related data. Since this is a numerical problem, the following models were chosen to be tested:

1. Linear Regression
2. Support Vector Regression
3. Regression Tree

Support vector regression and regression trees algorithms were chosen in addition to the linear regression algorithm since it fits non-parametric data, which, unless log transformed, the dataset to be tested displays properties of. Unlike Gauss-Markov assumptions for linear regression, SVR and regression trees do not depend on normal distributions of the independent and dependent variables.

The SVR algorithm depends on kernel functions (linear, polynomial, sigmoid, radial) and it also allows a non-linear model to be built without transforming the variables. The kernels transforms the dataset from non-linear space to linear space and allows the SVR model to find a fit and then map back to the original non-linear space. SVR does not care about the prediction as long as the error is less than a certain value, which is known as the principle of maximal margin. [9]

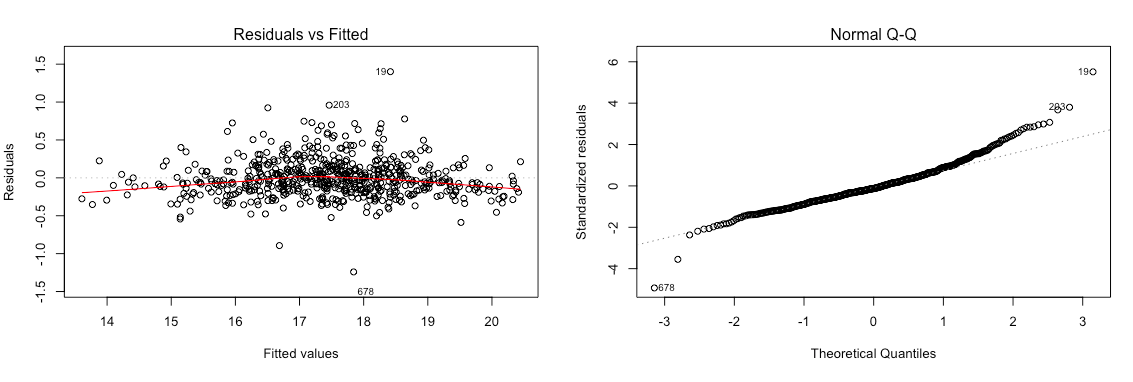
For all regression models, a 70/30 training/test split was used since that is a generally accepted figure to perform accurate tests.

SVR turned out to be the highest performing model with an RMSE of 23,964,321, which was much more accurate than the linear regression model, which produced an RMSE of 53,981,526. The regression tree model, with random forest implemented, was much improved over the linear regression model as well (RMSE 24,414,600), and just slightly less accurate than SVR.

# Results

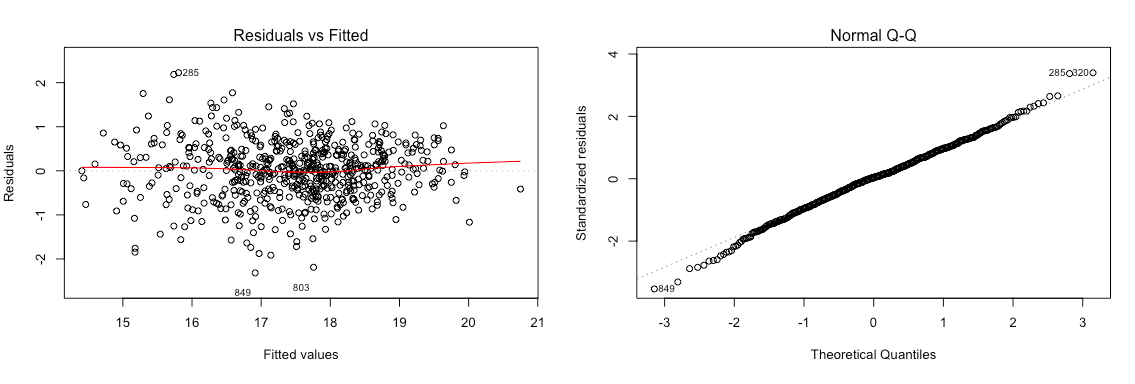
**Linear Regression**

A first pass using the extracted features from the results of the VIF test produced a very high adjusted R-squared (0.9598). This is oddly very high, and could be an indication of potential errors in the model. Running QQ plot and residual plot produced the following results:



*Figure 11: Residual and QQ plot*

The results show that the residuals are clustered around the center, when they should be more spread out. The QQ plot results show that there is evidence of non-normal distribution of the residuals on the high end of the theoretical quantile. Normality can be assumed when the residuals fall largely along the line. To rectify this problem, OpeningWeekendGross was removed from the dataset since it is a subset of the TotalGross. Running QQ plot and residual plot again produced the following results:



*Figure 12: Residuals and QQ plot with OpeningWeekendGross removed*

The results show that the residuals are more spread out, and the residuals in the QQ plot more closely follow the line. Therefore normality can be assumed.

Running another pass of the model, excluding OpeningWeekendGross, produced a more realistic adjusted R-squared (0.7229) and RMSE and MAE of 53,154,569 and 29,783,942 respectively. This became the starting point for building the most accurate model.

Forward and backward stepwise selection was applied next. This resulted in a slightly higher adjusted R-squared (0.7238), but also slightly higher RMSE (53,981,526) and MAE (30,035,497). Forward and back step reduction were applied next separately, but they did not improve the model.

The categorical/factor variable, “genre,” was removed next from the dataset since several of the categories did not have a significant effect on TotalGross in the backward and forward stepwise regression model. Adjusted R-squared was slightly lower (0.7146) and RMSE (55,099,417) and MAE (30,251,839) were higher, showing no improvement in the model. Removal of the categorical/factor variable “mpaaRating” was applied, producing even less improvement over the previous model.

Ultimately, the forward and backward stepwise selection model proved to be the most accurate model, when using adjusted R-squared as the key performance measure. OpeningTheaters, log(positiveWordsBL), log(budget), genreComedy, genreDrama has the most impact on the target variable, log(TotalGross) with p-value less than 0.001.

The following table contains the summary of the linear regression models with the features extracted from the VIF test and with OpeningWeekendGross removed:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model Description | R- Squared | Adjusted  R-Square | RMSE | MAE |
| Linear regression with all variables extracted from VIF test and excluding OpeningWeekendGross | 0.7312 | 0.7229 | 53,154,569 | 29,783942 |
| Linear regression using forward and backward step | **0.732** | **0.7238** | **53,981,526** | **30,035,497** |
| Linear regression using backward step | 0.732 | 0.7238 | 53,981,526 | 30,035,497 |
| Linear regression using forward step | 0.732 | 0.7229 | 53,154,569 | 29,783,942 |
| Linear regression removing “genre” variable | 0.717 | 0.7146 | 55,099,417 | 30,251,839 |
| Linear regression removing “mpaaRating variable | 0.7174 | 0.7141 | 55,068,904 | 30,077,898 |

*Figure 13: Linear regression models*

R code available [here](https://github.com/Jeffrey-Russell/CKME136/blob/master/Linear%20Regression%20Model.R).

**Support Vector Regression**

The following table shows the RMSE and MAE scores for different test error rates (cost) for SVR without any feature reductions. All models were ran setting epsilon equal to 0.1 (the default in the R function “svm”). The goal is to find the cost function that minimizes error RMSE.

Applying SVR to the initial set of features in the dataset results in the linear model with cost set to 100 and the radial model with cost set to 1 having nearly identical RMSE -- 25,676,408 for linear model vs 25,675,306 for radial model, which is the lowest numbers of all models tested. The linear model has less error when observing MAE. Therefore, without performing feature reduction, the linear model with cost set to 100 is the most accurate model with RMSE at 25,676,408 and MAE at 14,462,509.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Kernel | Epsilon | Cost | RMSE | MAE |
| Linear | 0.1 | 1 | 25,843,860 | 14,482,534 |
| Linear | 0.1 | 10 | 25,8181,76 | 144,83,724 |
| Linear | **0.1** | **100** | **25,676,408** | **14,462,509** |
| Linear | 0.1 | 1000 | 29,787,582 | 148,83,420 |
| Radial | **0.1** | **1** | **25,675,306** | **14,790,392** |
| Radial | 0.1 | 10 | 28,550,871 | 159,93,188 |
| Radial | 0.1 | 100 | 3134,86,05 | 18,547,448 |
| Radial | 0.1 | 1000 | 366,343,37 | 22,034,730 |
| Polynomial | 0.1 | 1 | 64699168212 | 4039222837 |
| Polynomial | 0.1 | 10 | 143385586919 | 8902382328 |
| Polynomial | 0.1 | 100 | 1.94175e+11 | 12045852820 |
| Polynomial | 0.1 | 1000 | 2.24043e+11 | 13894989563 |
| Sigmoid | 0.1 | 1 | 65091647 | 166989900 |
| Sigmoid | 0.1 | 10 | 6582580944 | 1696672204 |
| Sigmoid | 0.1 | 100 | 65524517198 | 17017972453 |
| Sigmoid | 0.1 | 1000 | 654934162039 | 170209303582 |

*Figure 14: SVR with initial features*

The following table shows RSME and MAE scores and different test error rates (cost) with the features that were extracted from the VIF test for detecting multicollinearity. Sigmoid and polynomial models were excluded from this round of testing since they produced significant errors in the previous model. (see figure 14).

The results show an overall improvement in the SVR model, with linear kernel set and cost to 200 having the lowest RMSE (23,964,321). Thus, with feature reduction, the linear model with cost equal to 200 has the lowest error rate of all SVM models tested (using RMSE as the key performance measure).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Kernel | Epsilon | Cost | RMSE | MAE |
| Linear | 0.1 | 1 | 23,919,228 | 14,144,839 |
| Linear | 0.1 | 10 | 23,976,432 | 14,157,638 |
| Linear | 0.1 | 100 | 23,972,757 | 14,165,560 |
| Linear | **0.1** | **200** | **23,964,321** | **14,164,409** |
| Linear | 0.1 | 250 | 23,966,377 | 14,162,629 |
| Linear | 0.1 | 300 | 23,970,212 | 14,161,348 |
| Linear | 0.1 | 1000 | 24,985,612 | 14,584,813 |
| Radial | 0.1 | 1 | 26,732,553 | 14,941,701 |
| Radial | 0.1 | 10 | 26,326,294 | 14,756,409 |
| Radial | 0.1 | 100 | 27,766,890 | 16,961,866 |
| Radial | 0.1 | 1000 | 40,184,958 | 22,8324,34 |

*Figure 15: SVR with feature extraction*

A process called tuning, which finds the best SVR model by training lots of models with different allowable errors and cost parameters was intended as a next to achieve more accurate results, but due to computer resource limitations, this proved to be a difficult task and was left out of the SVM modeling.

R code available [here](https://github.com/Jeffrey-Russell/CKME136/blob/master/Support%20Vector%20Regression%20Model.R).

**Regression Tree**

The following table shows the RMSE and MAE scores for different models of regression trees. All models were ran using the “rpart” package in R with default settings. The goal is to find the model that minimizes error RMSE.

Applying the regression tree algorithm to the initial set of features/variables in the dataset results in an RMSE of 35,207,020. Next, running the regression on the feature set extracted from the VIF test produced the exact same results as the full feature set. This implies that both models are using the same number of nodes, which are automatically pruned in the rpart package. For both models, OpeningWeekendGross was the most important variable. Given that OpeningWeekendGross is a subset of the target variable, TotalGross, OpeningWeekendGross was removed from the feature set and the regression ran for a third time. The removal of OpeningWeekendGross decreased performance of the model, with RMSE increasing substantially to 69,609,712.

|  |  |  |
| --- | --- | --- |
| Model Description | RMSE | MAE |
| All variables | 35207020 | 21890060 |
| Variables extracted from VIF test | 35207020 | 21890060 |
| All variables extracted from VIF test with OpeningWeekendGross removed | 69609712 | 38470326 |

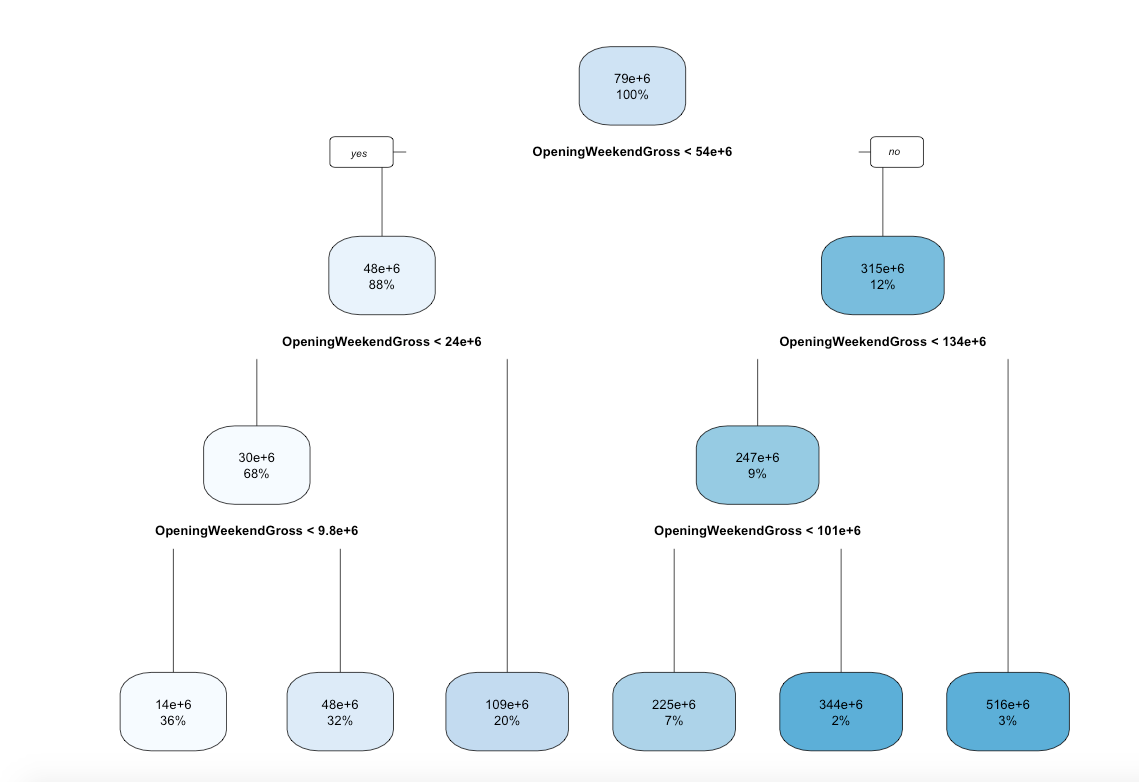
*Figure 16: Regression Tree results*

The following table shows the cross validation (xerror) of the model for all variables (which is also the same as the model with variables extracted from the VIF test):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | | | | |
|  | CP | nsplit | rel error | xerror | xstd |
|  | | | | | |
| 1 | 0.63133 | 0 | 1 | 1.00389 | 0.15877 |
| 2 | 0.13601 | 1 | 0.36867 | 0.38573 | 0.06948 |
| 3 | 0.08330 | 2 | 0.23266 | 0.26191 | 0.03787 |
| 4 | 0.01715 | 3 | 0.14936 | 0.17126 | 0.03358 |
| 5 | 0.01625 | 4 | 0.13221 | 0.15126 | 0.03281 |
| 6 | 0.01000 | 5 | 0.11596 | 0.13870 | 0.03271 |
|  | | | | | |

*Figure 17: Cross Product table*

The table indicates that the optimal cross-validated pruned tree model is a tree with six nodes. Adding more nodes will not substantially decrease the error rate of 0.13870. The optimal regression tree is as follows:



The model that the regression tree algorithm chose as the best, only uses one predictor, OpeningWeekendGross.

To see if more accurate results could be achieved with regression trees, random forest was applied next. Results are show in in figure 18.

Applying random forest significantly improved the performance of the model over the regression tree model, reducing RMSE to 29,453,319. Tuning the model further increased performance, reducing RMSE to 24,414,600.

For the tuned model, 10 fold cross validation was implemented. A tuning grid was set up to find the optimal parameter, mtry, which defines how many features are randomly selected at each split. By default, random forest will use mtry = sqrt(16), or four features per tree. Mtry = 16 proved to be the optimal tuned model

|  |  |  |
| --- | --- | --- |
| Model Description | RMSE | MAE |
| Random Forest default | 29145319 | 16820431 |
| Random Forest Tuned | 24414600 | 13596727 |

*Figure 19: Random forest results.*

R code available [here](https://github.com/Jeffrey-Russell/CKME136/blob/master/Regression%20Tree%20Model.R).

# Conclusions

The goal of the problem at hand was to create an effective predictive model for predicting the total box office gross of a movie after it’s opening weekend.

After building several different versions of linear regression,support vector regression, and regression tree models, the SVM model with linear kernel and cost set to 200 proved to be the most accurate model, producing the least error when using RMSE as a key performance measurement.

Studio’s and marketers now have a fairly accurate model that can predict total box office revenue, which can help them increase sales by creating more “efficient” trailers to generate interest and drive sales.

However, movie trailers are not the only factors that drive box office revenue. Other factors such as actors, movie reviews, word of mouth, competition from other movies, seasonality can have an effect on box office sales. Further studies could conducted with those variables included.

# References

[1] Md Shamsur Rahim, Asiful Islam, AZM Ehtesham Chowdhury, Mir Riyanul Islam (2017). Mining Trailers Data from YouTube for Predicting Gross Income of Movies. *Conference: IEEE Region 10 (Asia Pacific) Humanitarian Technology.* Retrieved from <https://www.researchgate.net/publication/322083844_Mining_Trailers_Data_from_YouTube_for_Predicting_Gross_Income_of_Movies>

[2] Krushikanth R. Apala, Merin Jose, Supreme Motnam, Chien-Chung Chan (2013). Prediction of Movies Box Office Performances Using Social Media. *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. Retrieved from <https://www.researchgate.net/publication/262249589_Prediction_of_movies_box_office_performance_using_social_media>.

[3] Sitaram Asur, Bernardo Huberman (2010). Predicting the Future with Social Media. *CoRR*. Retrieved from https://www.researchgate.net/publication/45909086\_Predicting\_the\_Future\_with\_Social\_Media

[4] Wenbin Zhang, Steven Skiena (2009). Improving Movie Gross Prediction Through News Analysis. Retrieved from <https://www.cs.cmu.edu/~nasmith/TDF/ZhangWenbinISF2009Paper.pdf>

[5] Eldar Sadikov, Aditya Parameswaran, Petros Venetis (2009). Blogs as Predictors of Movie Success. Retrieved from <https://pdfs.semanticscholar.org/21c8/1e867eeeca27ca91b129c24a3dd430dabeb2.pdf>

[6] Ramesh Sharda, Dursun Delen (2006). Predicting Box Office Success of Motion Pictures with Neural Networks. Retrieved from <http://www.academia.edu/17161024/Predicting_box-office_success_of_motion_pictures_with_neural_networks>

[7] Robertson, Mark R. (2011). 25 Jawdropping YouTube Video Facts, Figures, & Statistics. Retrieved from <https://tubularinsights.com/youtube-statistics/>

[8] Aslam, Salman (2018). YouTube by the Numbers: Stats, Demographics & Fun Facts. Retrieved from https://www.omnicoreagency.com/youtube-statistics/

[9] Building Regression Models in R using Support Vector Regression. https://www.kdnuggets.com/2017/03/building-regression-models-support-vector-regression.html