# Determining Box Office Success

A Closer Look at the TMBD Dataset

**Project Abstract**

In looking to present Netflix executives with information of what movies they should add to their streaming service, our group *Fusion, Inc.* assessed the TMDB 5000 Movie Dataset. This dataset takes information extracted from IMDb (the Internet Movie Database) to assess movie data. *Fusion, Inc.* used this data to answer our main question: is there a way to predict a movie’s success based on critical review and public review? To examine this question, we used financial analytics, descriptive analytics, linear regression models to test multicollinearity based on the variance inflation factor (VIF), and a KSVM model. For financial analytics, we used linear regression models and a multicollinearity to better understand the impact finances have on the success of a movie. For the descriptive analytics we used data visualizations tools such as scatter plots and bar graphs to track a historical record of how movies have done in the past to aid in our other tests of predicted popularity and success. For our KSVM model we used a machine learning algorithm to predict which movies would be a bomb or a success based on the information provided in the dataset. All of these calculations were done in RStudio to help us come to a conclusion what factors play into making a movie a success.

**Contribution Statement**

TIM’S CONTRIBUTION

* Attended bi-weekly meetings.
* Found and presented dataset that was used for the analysis
* Wrote code for:
  + Read the data from excel into R
  + Munge the data to delete unwanted columns, pull actor and director names from JSON format, create new variables for ROI, month, year and box-office success.
  + Determine descriptive success for overall dataset as well as for each genre, month, year, and production company
  + Pull out actors and directors that were involved in multiple movies.
  + Create and test the KSVM model
  + Create a score based on the scale function for each genre, month, the top 20 actors and top 20 directors.
  + Create plots to visualize the scores.
* Created slides and presented on the KSVM model, the characteristic scores and our conclusions.

CHARMAINE’S CONTRIBUTION:

* Scheduled recurring bi-weekly Zoom meetings and generated meeting invitations.
* Attended bi-weekly meetings.
* Created initial draft for Final presentation deck updating applicable slides.
* Wrote code to support:
  + Financial Analytics
  + Regression Models
  + Plot Regression and model fit
  + Multicollinearity test
* Conducted and documented Regression Analysis
* Presented on Financial Analytics; Regression Model and Analysis and Multicollinearity test and results
* Wrote the Data Dictionary

CAITLIN’S CONTRIBUTION:

● Devised code for regressions and associated visualizations

● Attended bi-weekly Zoom meetings to discuss progress and direction of project

● Co-authored Project Update Submissions

● Co-authored manuscript

● Collaborated on PowerPoint presentation and proposed visualizations

● Presented overview, Introduction and overall scope of project

PAIGE’S CONTRIBUTION:

* Compiled the team’s presentation work into a written document for secondhand consumption.
* Attended bi-weekly meetings to discuss progress and direction of project
* Created and managed Trello board to visualize task management and completion. (<https://trello.com/b/uRyAwYTm/imdb-dataset>)
* Wrote code to support:
  + Descriptive analytics
  + Data analysis
  + Data visualizations
* Was assigned to present explaining what data tools we used to examine the data, as well as help teammates explain what data visualization tools were chosen and why.

**Final Report**

1. Introduction

The TMDB 5000 Movie Dataset presented itself as a compelling find for the four of us. When we first looked at the data, we thought it was extremely interesting to have all of this data about movies, and immediately we were itching to figure out how we wanted to digest this information. After much discussion, we narrowed down our main research question: is there a way to predict a movie’s success based on critical review and public review? With this in mind we began to think about how a production company, or a streaming service like Netflix may want that question answered. This data set has 3,183 observations in it, and 19 variables. This is a *lot* of data that we wanted to narrow down and make more readable for our potential business partners, so we decided to try and trim the dataset down to make it more manageable. To start this process, we wrote a data dictionary to explain what each part of the variables represented in their industry. By using our several data analytic and machine learning tools we found:

1. The more money spent on a project, the better that project is going to do.
2. Predicting a movie’s ROI is difficult, but the model created within this project can predict fairly accurately whether or not a movie will be a bomb.
3. We are able to predict which movies would do best in a streaming environment, as well as in an awards season.

Based on these findings we recommend to streaming services and production companies to discover what their need from the movie is. Are they looking to make a movie that’s going to get a high amount of views? Are they looking to add new movies to their streaming service? What about Oscar season, are they looking for a sweep? With our machine learning algorithm, for streaming services we were able to find that adventure, action, animated and science fiction genres generate the most viewers. Dwayne “The Rock” Johnson, Ian McKellan, Chris Evans, Anne Hathaway, Matthew McConaughey and Chris Hemsworth are the actors that will most well received in new movie additions on a streaming platform. Movies directed by Sam Mendes or Bobby Farrelly did best in this category, and the movie should have been released in June. For Oscar season, the algorithm is slightly different. Western, historical and war genres, a star studded cast with Leonardo DiCaprio, Jennifer Connelly, and Ben Kingsly, directed by Bobby Farrelly and released during the holidays (specifically December) is sure to get any movie a sweep.

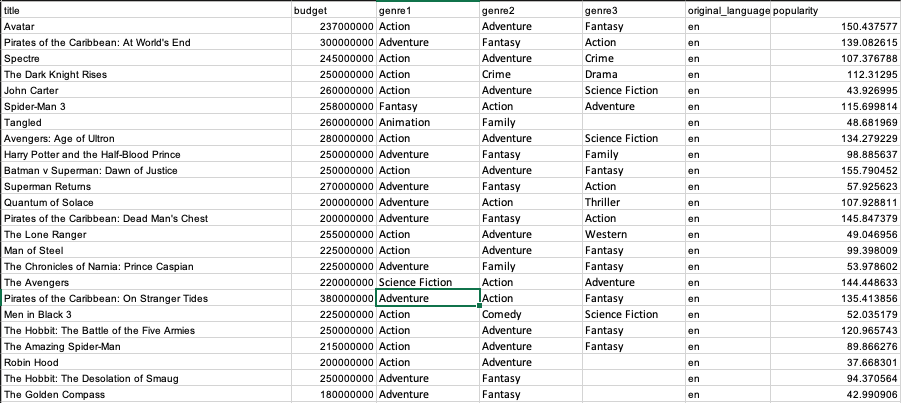
1. The Dataset

The dataset consisted of two excel spreadsheets, one containing the cast and crew for each movie, the other containing other information about the movies. The biggest issue with the spreadsheets were that any variable that contained text strings was in a JSON format. For example, the genre variable consisted of each genre for the movie in a JSON text format. Using excel and R, we were able to separate these JSON text strings and pull out the first three genres, production companies and actors for each movie. We then pulled the director out of the crew variable.



*Screenshot of first two entries from original JSON file found at* [*https://www.kaggle.com/tmdb/tmdb-movie-metadata*](https://www.kaggle.com/tmdb/tmdb-movie-metadata)

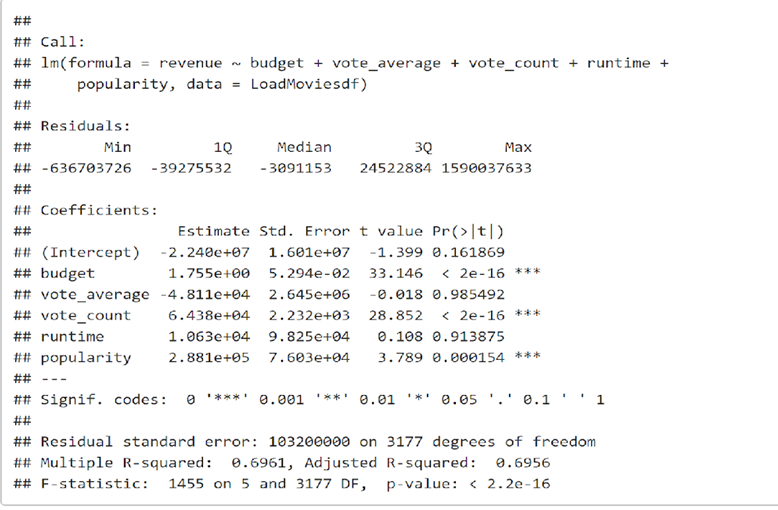
Once the file was more readable for us, we were able to see exactly what we were dealing with. The next step was to determine which variables were useful for us and remove the ones that were not. The variables we ended up with in the dataset include: title, budget, genre1, genre2, genre3, original\_language, popularity, production\_company1, production\_company 2, production\_company3, production\_country, release\_date, runtime, vote\_average, vote\_count, actor1, actor2, actor3 and director. Vote\_average and vote\_count refer to IMDB’s user interaction with the movie, which is where this data was pulled from.



*The first 24 entries in the dataset.*

1. Data Analysis Methods

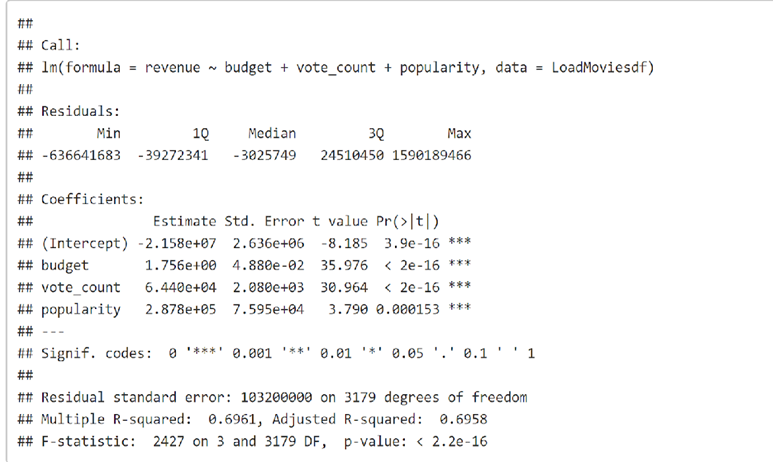
Initial Linear Regression Model and Analysis



Ran an initial linear regression model and concluded the following:

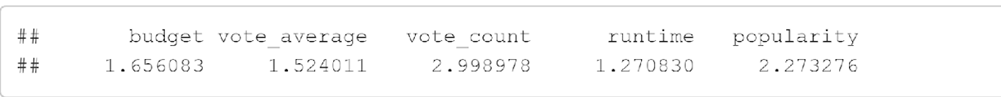
* Has an Adjusted R-Squared of 69.56%, which indicates that 69.56% of a change in revenue (dependent variable) can be explained by changes in the independent variables, which indicates that the model only explains 69.56 % of the variability of the response data around its mean and therefore is not a perfect fit.
* The model shows that independent variables budget, vote count and popularity are strong predictors of revenue and are considered statistically significant given their low p-values very close to zero, which indicates that we can reject the null hypothesis.
* However note that vote average and runtime are not strong predictors of revenue given their very high p-values of 98.5% and 91.4%, respectively. As a result we excluded these two independent variables and ran the model again.

Reran regression model removing independent variables that aren't statistically significant (vote\_average and runtime):



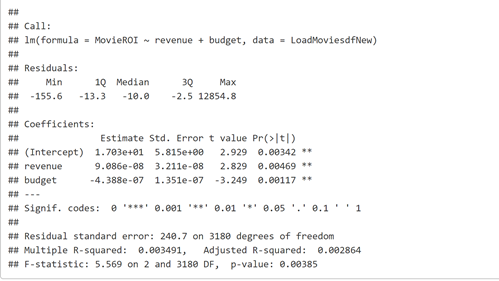
* Budget, vote count and popularity are strong predictors of revenue given low p-values of very close to zero and confidence level of close to 100%. As budget, vote count and popularity increases revenue increases.
* Adjusted R-squared only changed marginally and indicates that only 69..6% of a change in revenue can be explained by changes in the budget, vote count and popularity. Indicating that the model explains only 69.6 % of the variability of the response data around its mean. Though not a perfect fit, this model represents the best fit.

Multicollinearity Test - Variance Inflation Factor (VIF) - Low



Multicollinearity occurs when predictors in the model are correlated with other predictors. We used the variance inflation factor (VIF) to test for multicollinearity. A VIF of 10 or more indicates high multicollinearity in the model. Multicollinearity increases the standard errors of the co-efficient. Therefore by overinflating the standard errors multicollinearity makes some variables statistically insignificant when they should be significant. From our results, we can see that all independent variables in the model have a VIF of less than 3, which indicates that multicollinearity does not exist in the model.

Ran a Regression Model to understand drivers of ROI and Measurement of Box Office Success:



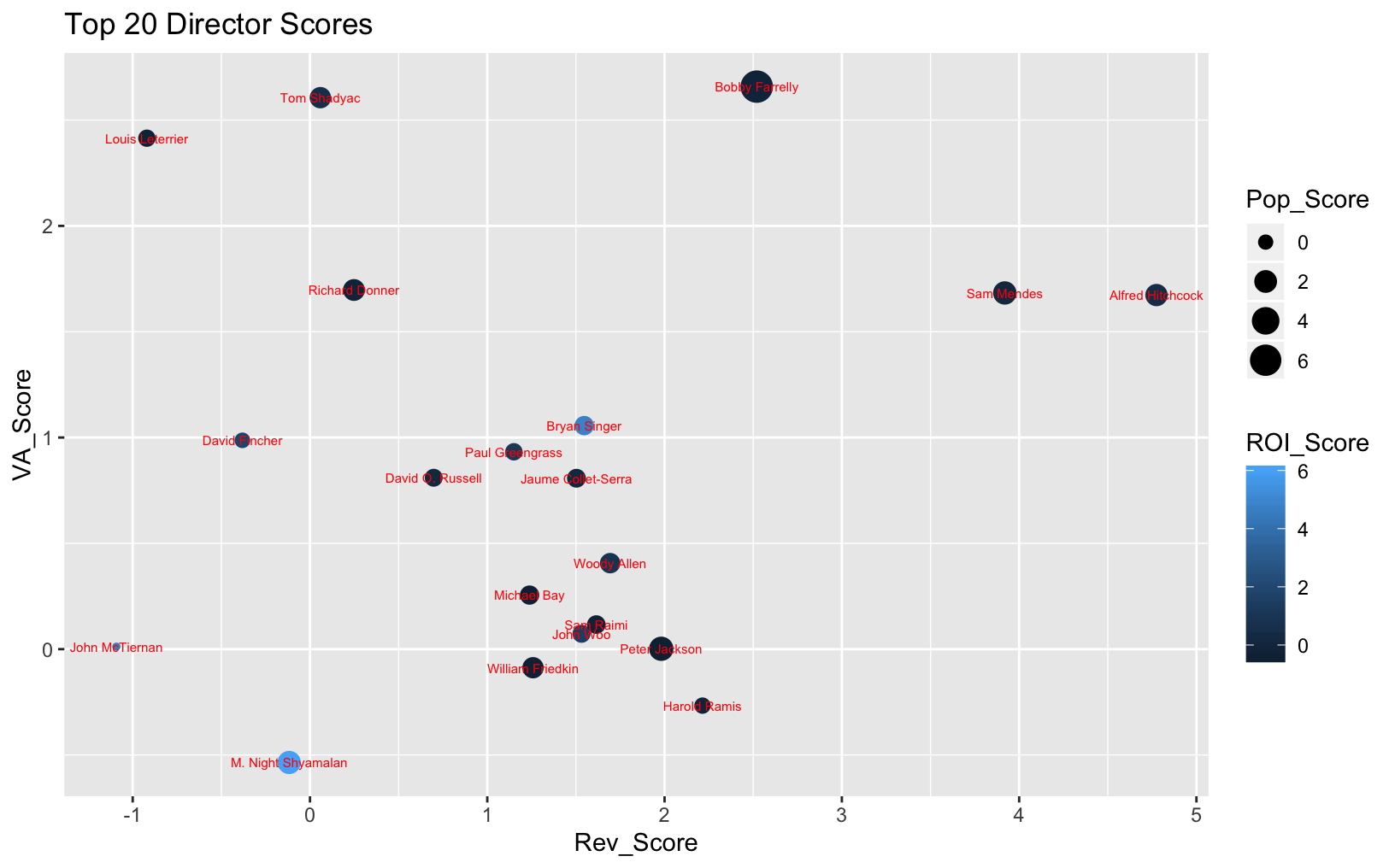
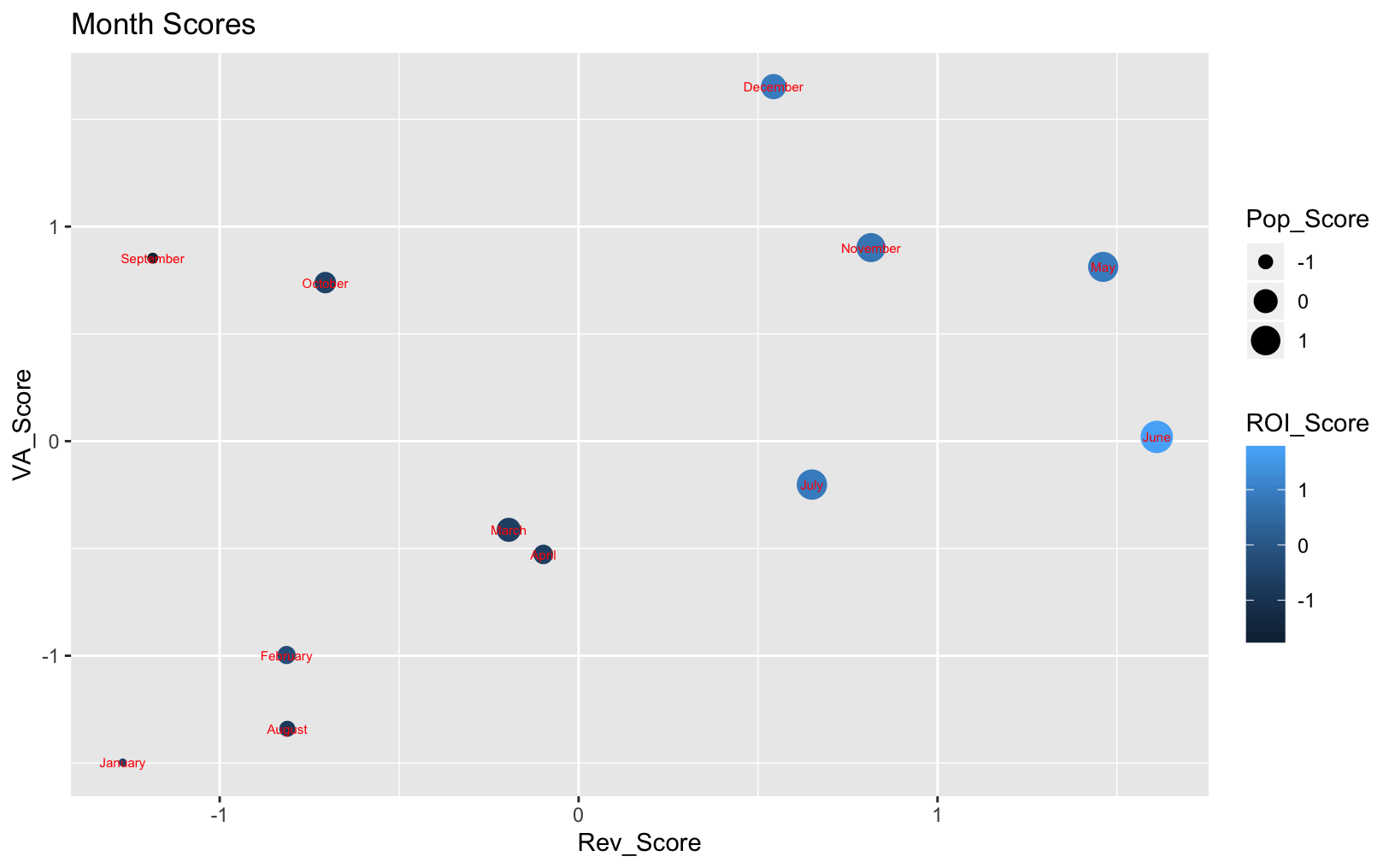
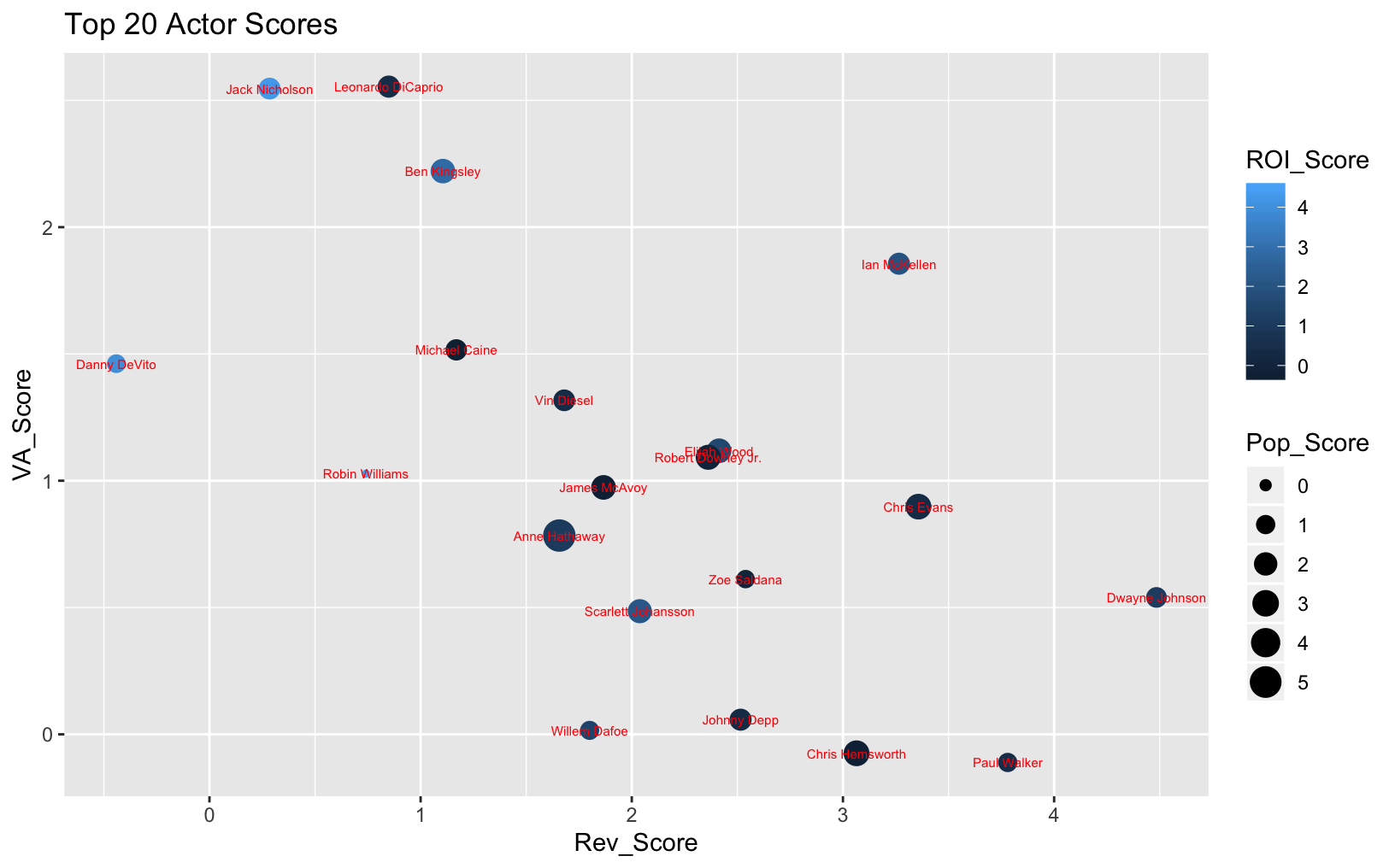
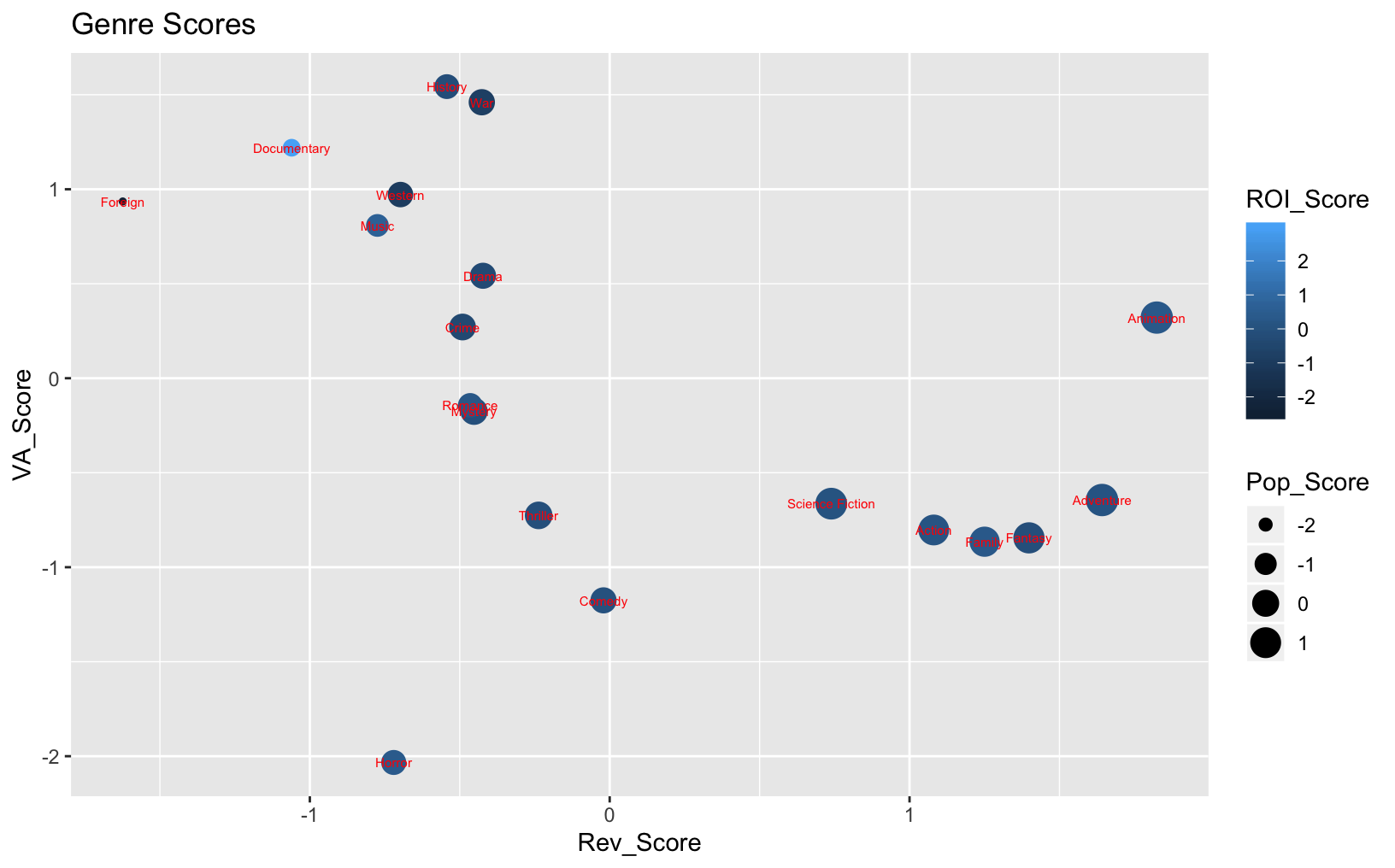
This model is not a good model to predict ROIl. Though the model has low p-values indicating statistical significance of independent variable showing that changes in revenue and budget are strong predictors of a change in ROI. The model however, also has a very low Adjusted R-squared of 0.002864 which indicates that changes in revenue and budget only explain .2% of the change in ROI. This suggests that there are other external factors driving ROI which are not reflected in the data.

After running regression analysis, we decided to look at a different way to measure box office success. A box office bomb is defined as a movie that returns a net loss at the box office. We created a variable that assigned each movie as a box office bomb or not. Using this variable we ran a KSVM model to attempt to predict whether a movie is a box office bomb or not. KSVM is a support vector mechanism. It uses the data to try and separate different data points based on the criteria of being a box office bomb or not. The variables we used to predict bomb or success were the budget, the popularity, the vote average and vote count, the runtime and the release month. If the model is successful, it should provide an ability to predict which movies will flop at the box office. To create the model, we took a random two-thirds of our data and used it to train the computer. The computer uses this “training” data to create its model that can be used to predict bomb or not. The model’s accuracy can then be tested using the remaining third of the data.

Another method we used to attempt to predict successful movies was to look at the discrete variables that make movies unique. We wanted to assign a score to the unique values of these variables to determine which ones most often lead to successful movies. We assigned a score for how well each individual performed in revenue generation, return on investment, critical vote average and popularity. To create these scores we used the scale function to determine how many standard deviations each individual was from the mean of the data. We did this for each individual and each success measure to create a score for each. We then summed the scores to create a total score. For example, each genre was given a score based on Revenue, ROI, Vote Average, Popularity and a Total Score. Scores were created for each genre in the data set, each month a movie was released, each actor who appeared in at least ten movies in the dataset and each director that directed at least 8 movies. This was done to prevent actors and directors from being over-scored by a small sample size that may include only one movie or a small series of movies. We then created charts to compare the scores visually. The goal is for these scores to provide us with information on which individual characteristics most often lead to successful movies.

1. Analysis

After creating the KSVM model, we took a random third of the data to test it’s accuracy. The model predicted if each of the test data was a bomb or not. Using these predictions we are able to compare the model’s predictions with the actual dataset to determine how accurate the model was. The model predicted 31 movies as bomb’s that were not and 28 movies as successes that were. Out of the the 1,062 movies in the test data, the model predicted 94.4% of the data correctly. This appears to be a very strong model with high accuracy, but there is an issue with using it. By examining the overall dataset, we saw that only about 24 percent of movies were box office bombs. This means that being a box office bomb is more rare than not and the model could struggle to predict bombs as it is likely to predict everything as being a success. The model would be better if the percentage of bombs to successes in the dataset was close to even.

By looking at the scores for individual genres, months, actors and directors we can begin to see which are more likely to contribute to a successful movie. The top total scoring genres were Animation, Adventure and Fantasy. The top scoring release months were June, May and November. The top scoring actors were Anne Hathaway, Ian McKellan and Jack Nicholson. The top scoring directors were Bobby Farrely, Alfred Hitchcock and Bryan Singer. However, it was rare that these individuals were among the top five scores for each category. We realized that there were relationships amongst the scores and that at times one measure of success had a negative relationship with another. Genres and actors that scored better in terms of producing revenue generally had lower vote-average scores. For example, Adventure movies make the second most money on average, but they score below-average when it comes to critical review. Documentaries are below average in revenue score, but do very well with critics and have a higher return on investment. These relationships show that it may be difficult to create a movie that earns a lot of revenue while also becoming critically acclaimed. However, our analysis of the individual scores could provide insight on how to create a movie that can accomplish this. The charts below show the relationships between the scores and provide information on which individuals score better in each category. For the actors and directors, the top 20 overall total scores were used in the plots to prevent overcrowding. The individual scores in conjunction with the charts outline a specific cast, director, genre and release month that can potentially lead to a successful movie. 

1. Results

The findings of our research and analysis show that predicting a movies success can be very challenging. There are many different variables that can impact how a movie performs at the box office and success may have a different meaning to different people. Overall, gross revenue is most likely affected by the budget spent on the movie. If you spend more money to create your movie, you would expect to make more money at the box office. However, success may be defined by how much money your movie makes relative to its budget. Can you predict if your small budget film will make millions at the box office? Return on investment is very unpredictable. The data shows that there are a few extreme outliers that cannot be explained and regression analysis is unreliable. It could help to look at success more broadly and try to predict if a movie will flop or not at the box office and net a loss. While the model we used to predict this is very accurate, most movies do not bomb at the box office and the model could prove to be unreliable for movies that may be close to being bombs. More modeling is needed to attempt to predict box office bombs.

The most useful results from our analysis are the individual scores we assigned. Using these scores we can suggest different combinations of actors, directors, genres and a release month that could lead to a successful movie. If you are looking to make a movie that generates the most revenue, the movie should be an animated, action, adventure movie that stars Dwayne Johnson, Chris Evans and Ian McKellon. The director should be Sam Mendes and it should be released in June. To create a movie that will be critically acclaimed and do well come awards season, the movie should be a western, historical war story that stars Leonardo DiCaprio, Jennifer Connelly and Ben Kingsley. It should be directed by Bobby Farrely and be released in December. If you are just concerned with making a movie that is really popular and buzz-worthy, you should create an animated, science fiction adventure movie starring Anne Hathaway, Chris Hemsworth and Matthew McCounaughy. The movie should be released in June and be directed by Bobby Farrely. To make the best overall movie that succeeds in all facets, you should make an animated, western adventure movie. The movie should star Dwayne Johnson, Anne Hathaway, Ian McKellon, Leonardo DiCaprio and Chris Evans. Evans is actually the only actor to have a top five scores in multiple categories while Hathaway and McKellon are the top two overall rated actors. The movie should definitely be released in June or May as they are clearly the two most successful months to release a movie. Finally, the movie should be directed by Bobby Farrely, the top rated director overall and the only director to have a top five score in three categories. These suggested movies are obviously just that; a suggestion. They use our calculated scores to try and predict combinations of factors that would have a higher probability of creating a successful movie. However, like most of our other analysis shows, a movie’s success can be unpredictable and depend on variables outside of the producer’s control. We can only try to provide information to increase the probability of success.

**Conclusions**

Our group, *Fusion, INC.*, analyzed the TMDB 5000 Movie Dataset. This is a large dataset that examines movie data pulled from IMDb (the Internet Movie Database). The main question we wanted to answer with this data was: is there a way to predict a movie’s success based on critical review and public review? To analyze this data we deployed several data analysis tools to better read this dataset, which included financial analytics, descriptive analytics, linear regression models to test multicollinearity based on the variance inflation factor (VIF), and a KSVM model. Each of these tools gave us the ability to come to several conclusions that would be convenient for streaming service executives as well as movie producers to know. With the linear regression model we were able to examine the financial aspects of the dataset to attempt to predict trends that can lead to answers to our main question. The KSVM machine learning algorithm was able to compile information together and break things down further to pair with the linear regression model to start making finalizing points from our dataset. Our results indicate gross revenue is most influenced by the movie’s budget. Remaining questions, such as cast, ROI, creative team, etc., were much more difficult to determine a movie’s success due to extreme outliers, unreliable analyses and vague data definitions. Future analyses may aim to better nominalize descriptive data and more specifically focus on the definition of success.

**References**

The Dataset: <https://www.kaggle.com/tmdb/tmdb-movie-metadata>

**Appendix**

**Descriptive Analysis & Data Visualization Tools**

```{r}

df <- read\_excel("~/Desktop/tmdb\_5000\_movies.xlsx")

View(df)

#remove columns 17-33

str(df)

plot(df$revenue ~ df$budget, xlab="budget", ylab="revenue")

rev.bud <- lm(formula = revenue ~ budget, data=df)

summary.lm(rev.bud)

plot(df$vote\_average ~ df$budget, xlab="vote\_average", ylab="revenue")

rev.votavg <- lm(formula = revenue ~ vote\_average, data=df)

summary.lm(rev.votavg)

plot(df$revenue ~ df$popularity, xlab="popularity", ylab="revenue")

rev.pop <- lm(formula = revenue ~ popularity, data=df)

summary.lm(rev.pop)

```

```{r}

MovieData <- df

MovieData = subset(MovieData, select = -c(17:33) )

ggplot(MovieData, aes(x=popularity)) +

geom\_histogram(binwidth=20, color="white", fill="black") +

ylab("Popularity") + xlab("Revenue") + xlim(-10,150) + ggtitle("Revenue vs. Popularity")

```

```{r}

NewMovieData <- data.frame(MovieData$title,MovieData$popularity,MovieData$vote\_count,MovieData$revenue)

str(NewMovieData)

ggplot(NewMovieData,aes(x=MovieData.title, y=MovieData.revenue)) + geom\_point(aes(size=MovieData.popularity, color=MovieData.vote\_count)) + ggtitle("Popularity According to the Viewers")

**Linear Regression and Multicollinearity Test**

```{r}

tmdb\_5000\_movies <- read\_xlsx("C:/Users/stena/Desktop/tmdb\_5000\_movies.xlsx")

head(tmdb\_5000\_movies)

str(tmdb\_5000\_movies)

summary(tmdb\_5000\_movies)

str(tmdb\_5000\_movies)

## Step 2 remove columns 18 through 34

tmdb\_5000\_movies <- tmdb\_5000\_movies[,-18:-34]

str(tmdb\_5000\_movies)

summary(tmdb\_5000\_movies)

tmdb\_5000\_movies[,12]

## Step 3 clean up function (Tim)

LoadMovies <- function(){

Movies <- tmdb\_5000\_movies

Movies <- Movies[,1:17]

Movies$release\_date <- as.Date(Movies$release\_date,origin="1899-12-30")

Movies$release\_month <- format(Movies$release\_date,"%b")

Movies$release\_year <- format(Movies$release\_date, "%Y")

Movies <-Movies[!(Movies$budget==0 | Movies$popularity==0 | Movies$revenue==0 | Movies$runtime==0 | Movies$vote\_average==0 | Movies$vote\_count==0),]

Movies <- Movies[complete.cases(Movies[,c(3,6:8,11,12)]),]

Movies$budget <- ifelse(Movies$budget<200,Movies$budget\*1000000,Movies$budget)

Movies$revenue <- ifelse(Movies$revenue<200,Movies$revenue\*1000000,Movies$revenue)

Percent\_Budget <- (Movies$revenue/Movies$budget)\*100

cbind(Movies,Percent\_Budget)

return(Movies)

}

## Step 4 store as df

LoadMoviesdf <- LoadMovies()

str(LoadMoviesdf)

summary(LoadMoviesdf)

##Net Revenue

Net\_Revenue <- LoadMoviesdf$revenue - LoadMoviesdf$budget

Net\_Revenue

#Which movie generated a profit(yes or no)

Profit\_Loss <- ifelse(tmdb\_5000\_movies$revenue > tmdb\_5000\_movies$budget, "yes", "no")

Profit\_Loss

## Movie ROI (Revenue - cost)/cost

MovieROI <- (LoadMoviesdf$revenue - LoadMoviesdf$budget)/LoadMoviesdf$budget

MovieROI

##create linear model

model <- lm(formula = revenue ~ budget + vote\_average + vote\_count + runtime + popularity, data=LoadMoviesdf)

summary(model)

##re run model linear model removing independent variables that aren't statistically significant (vote\_average and runtime)

## these indepedent variables are highly statistically significant close to 100%, we

## reject the null hypothesis, which allows us to conclude that there is a relationship

## revenue and budget, vote\_count and popularity where these 3 independent variables are drives of revenue.

## However, only indicates an R squared of 5694 which infers that changes in budget, vote\_count and popularity can explain

## only 56.94 of the change in revenues, which suggests that there are other drivers not reflected in the data that would impact movie revenues.

model2 <- lm(formula = revenue ~ budget + vote\_count + popularity, data=LoadMoviesdf)

summary(model2)

##re run model linear model removing independent variables that aren't statistically significant (vote\_average and runtime)

model3 <- lm(formula = revenue ~ budget + popularity, data=LoadMoviesdf)

summary(model3)

##re run model linear model removing independent variables that aren't statistically significant (vote\_average and runtime)

model4 <- lm(formula = revenue ~ budget + vote\_count, data=LoadMoviesdf)

summary(model4)

##re run model linear model removing independent variables that aren't statistically significant (vote\_average and runtime)

model5 <- lm(formula = revenue ~ budget, data=LoadMoviesdf)

summary(model5)

## to review if multicolinearity exists . No evidence or issue with multicolinearity because variance inflation factor is lower than 10, so very low.

library(faraway)

vif(model)

## Conclude that model2 is the best model

model2 <- lm(formula = revenue ~ budget + vote\_count + popularity, data=LoadMoviesdf)

summary(model2)

##Plot revenue versus budget

plot(LoadMoviesdf$budget, LoadMoviesdf$revenue)

model5 <- lm(formula = revenue ~ budget, data=LoadMoviesdf)

summary(model5)

#Fit

abline(model5)

library(ggplot2)

g <- ggplot(LoadMoviesdf, aes(x = budget, y = revenue)) + geom\_point() + stat\_smooth(method = "lm", col = "red") + ggtitle("Revenue versus Budget")

##Plot lm - ggplot

ggplot(LoadMoviesdf, aes(x = budget, y = revenue)) + geom\_point() + stat\_smooth(method = "lm", col = "red") + ggtitle("Revnue vs. Budget")

##Plot revenue versus vote\_count. Vote count has the strongest relationship with

## revenue.

plot(LoadMoviesdf$vote\_count, LoadMoviesdf$revenue)

model6 <- lm(formula = revenue ~ vote\_count, data=LoadMoviesdf)

summary(model6)

#Fit

abline(model6)

g2 <- ggplot(LoadMoviesdf, aes(x = vote\_count, y = revenue)) + geom\_point() + stat\_smooth(method = "lm", col = "red") + ggtitle("Revenue versus Vote Count")

##Plot revenue versus popularity

plot(LoadMoviesdf$popularity, LoadMoviesdf$revenue)

model7 <- lm(formula = revenue ~ popularity, data=LoadMoviesdf)

summary(model7)

#Fit

abline(model7)

g3 <- ggplot(LoadMoviesdf, aes(x = popularity, y = revenue)) + geom\_point() + stat\_smooth(method = "lm", col = "red") + ggtitle("Revenue versus Popularity")

library(gridExtra)

grid.arrange(g, g2, g3, nrow=2, ncol=2)

str(LoadMoviesdf)

## Added columns Net\_Revenue and ROI

LoadMoviesdfNew <- cbind(LoadMoviesdf, Net\_Revenue, MovieROI)

LoadMoviesdfNew

str(LoadMoviesdfNew)

summary(LoadMoviesdfNew)

##Movie with the highest ROI - Paranomal Activity

LoadMoviesdfNew[which.max(LoadMoviesdfNew$MovieROI), 1]

max(LoadMoviesdfNew$MovieROI)

##Movie with the Lowest ROI - "The Adventurer: The Curse of the Midas Box"

LoadMoviesdfNew[which.min(LoadMoviesdfNew$MovieROI), 1]

min(LoadMoviesdfNew$MovieROI)

##Movie with the highest Revenue - Avatar - $2,787,965,087

LoadMoviesdfNew[which.max(LoadMoviesdfNew$revenue), 1]

max(LoadMoviesdfNew$revenue)

##Movie with the highest Revenue - Avatar

LoadMoviesdfNew[which.max(LoadMoviesdfNew$revenue-LoadMoviesdf$budget), 1]

#Movie with the highest Net\_Revenue - Avatar, $ 2,550,965,087

LoadMoviesdfNew[which.max(LoadMoviesdfNew$Net\_Revenue), 1]

max(LoadMoviesdfNew$revenue-LoadMoviesdfNew$budget)

max(LoadMoviesdfNew$Net\_Revenue)

##Movie with the lowest Net\_Revenue - "The Lone Ranger", $ -165,710,090

LoadMoviesdfNew[which.min(LoadMoviesdfNew$Net\_Revenue), 1]

min(LoadMoviesdfNew$revenue-LoadMoviesdfNew$budget)

min(LoadMoviesdfNew$Net\_Revenue)

##Movie with the Lowest Revenue - "I Married a Strange Person", $203

LoadMoviesdfNew[which.min(LoadMoviesdfNew$revenue), 1]

min(LoadMoviesdfNew$revenue)

#Movie with the highest Net\_Revenue - Avatar, $ 2,550,965,087

LoadMoviesdfNew[which.max(LoadMoviesdfNew$Net\_Revenue), 1]

max(LoadMoviesdfNew$Net\_Revenue)

##Movie with the Lowest Net\_Revenue - "The Loan Ranger", Net loss of -$165,70,090

LoadMoviesdfNew[which.min(LoadMoviesdfNew$Net\_Revenue), 1]

min(LoadMoviesdfNew$Net\_Revenue)

##Movie with the highest budget - "Pirates of the Caribbean: On Stranger Tides", $3.8e+08

LoadMoviesdfNew[which.max(LoadMoviesdfNew$budget), 1]

max(LoadMoviesdfNew$budget)

##Movie with the lowest budget - "I Married A Stranger", $250

LoadMoviesdfNew[which.min(LoadMoviesdfNew$budget), 1]

min(LoadMoviesdfNew$budget)

##Order by Revnue

OrderedLoadMoviesdfNew2 <- LoadMoviesdfNew[order(LoadMoviesdfNew$revenue),]

head(OrderedLoadMoviesdfNew2)

tail(OrderedLoadMoviesdfNew2)

##Order by ROI

OrderedLoadMoviesdfNew <- LoadMoviesdfNew[order(LoadMoviesdfNew$MovieROI),]

head(OrderedLoadMoviesdfNew)

tail(OrderedLoadMoviesdfNew)

##Order by Budget

OrderedLoadMoviesdfNew3 <- LoadMoviesdfNew[order(LoadMoviesdfNew$MovieROI),]

head(OrderedLoadMoviesdfNew3)

tail(OrderedLoadMoviesdfNew3)

##Order by Net\_Revenue

OrderedLoadMoviesdfNew4 <- LoadMoviesdfNew[order(LoadMoviesdfNew$Net\_Revenue),]

head(OrderedLoadMoviesdfNew4)

tail(OrderedLoadMoviesdfNew4)

##This model is not a good model. Though low p-value indicating statistical signifcance showing that changes in revenue and budget

# are strong predictors of a change in ROI, however only explains .2% of the change in ROI. This suggest that there are

#other external factors exist which are not reflected in the data.

model8 <- lm(formula = MovieROI ~ revenue + budget, data=LoadMoviesdfNew)

summary(model8)

summary(LoadMoviesdfNew)

str(LoadMoviesdfNew)

##Plot Revenue versus ROI

g4 <- ggplot(LoadMoviesdfNew, aes(x = revenue, y = MovieROI)) + geom\_line() + ggtitle("Revenue versus ROI")

library(ggplot2)

g4

```

**KSVM Machine Learning Algorithm**

library(jsonlite)

library(reshape2)

library(moments)

library(scales)

library(tidyverse)

library(sqldf)

library(readxl)

library(kernlab)

library(e1071)

library(ggplot2)

credits <- read\_csv("Downloads/tmdb\_5000\_credits.csv",col\_names=TRUE,na="NA")

crew <- credits %>%

filter(nchar(crew)>2) %>%

mutate(

js = lapply(crew, fromJSON)

) %>%

unnest(js)

crew <- crew[,c(2,9,10)]

crew <- sqldf("select title, name from crew where job='Director'group by title")

colnames(crew) <- c("title","director")

cast <- credits %>%

filter(nchar(cast)>2) %>%

mutate(js = lapply(cast, fromJSON)) %>%

unnest(js) %>%

select(-cast, -crew, -credit\_id) %>%

rename(actor=name, movie\_cast\_id=cast\_id, actor\_id=id) %>%

mutate\_if(is.character, factor)

cast <- cast %>% filter(order %in% c(0, 1, 2)) %>% select(movie\_id, title, order, actor)

cast$order[1] <- 0

for (i in 1:(nrow(cast)-1)){

if(cast$movie\_id[i+1]!=cast$movie\_id[i]){

cast$order[i+1] <- 0

} else {cast$order[i+1] <- cast$order[i]+1}

}

cast <- cast %>% filter(order %in% c(0, 1, 2)) %>%

spread(key=order, value=actor)

cast <- cast[,c(2:5)]

colnames(cast) <- c("title","actor\_1","actor\_2","actor\_3")

Cast\_Crew <- merge(cast,crew,by="title",all=T)

tmdb\_5000\_movies <- read\_excel("Documents/tmdb\_5000\_movies.xls")

Movies <- tmdb\_5000\_movies

Movies <- Movies[,1:17]

Movies$release\_date <- as.Date(Movies$release\_date,origin="1899-12-30")

Movies$release\_month <- months(Movies$release\_date)

Movies$release\_year <- format(Movies$release\_date, "%Y")

Movies <-Movies[!(Movies$budget==0 | Movies$popularity==0 | Movies$revenue==0 | Movies$runtime==0 | Movies$vote\_average==0 | Movies$vote\_count==0),]

Movies <- Movies[complete.cases(Movies[,c(3,6:8,11,12)]),]

Movies$budget <- ifelse(Movies$budget<200,Movies$budget\*1000000,Movies$budget)

Movies$revenue <- ifelse(Movies$revenue<200,Movies$revenue\*1000000,Movies$revenue)

Movies$Percent\_Budget <- (Movies$revenue/Movies$budget)\*100

Movies$Difference <- Movies$revenue-Movies$budget

Movies <- merge(Movies,Cast\_Crew,by="title",all=T)

Movies <- Movies[complete.cases(Movies[,2]),]

Summary\_Stats <- function(column){

Mean <- mean(column)

Median <- median(column)

SD <- sd(column)

Max <- max(column)

Min <- min(column)

Skewness <- skewness(column)

Vector <- c(Mean,Median,SD,Max,Min,Skewness)

return(Vector)

}

Vote\_Average <- Summary\_Stats(Movies$vote\_average)

Budget <- Summary\_Stats(Movies$budget)

Revenue <- Summary\_Stats(Movies$revenue)

Runtime <- Summary\_Stats(Movies$runtime)

Popularity <- Summary\_Stats(Movies$popularity)

Percent\_Budget <- Summary\_Stats(Movies$Percent\_Budget)

Difference <- Summary\_Stats(Movies$Difference)

Labels <- c("Mean","Median","SD","Max","Min","Skew")

Summary\_Statistics <- data.frame(Labels,Vote\_Average,Budget,Revenue,Runtime,Popularity,Percent\_Budget,Difference)

GenreMeans <- function(column){

Genre1Avg <- aggregate(x=column,list(Movies$genre1),mean)

Genre2Avg <- aggregate(x=column,list(Movies$genre2),mean)

Genre3Avg <- aggregate(x=column,list(Movies$genre3),mean)

GenreAvg <- merge(Genre1Avg,Genre2Avg, by="Group.1",all =TRUE, no.dups = T)

GenreAvg <- merge(GenreAvg,Genre3Avg, by="Group.1",all =TRUE, no.dups = T)

Average <- rowMeans(cbind(GenreAvg$x.x,GenreAvg$x.y,GenreAvg$x),na.rm=TRUE)

GenreAvg <- cbind(GenreAvg,Average)

GenreAvg <- GenreAvg[,c(1,5)]

colnames(GenreAvg)[colnames(GenreAvg)=="Group.1"] <- "Genre"

return(GenreAvg)

}

Genre\_VA\_Mean <-GenreMeans(Movies$vote\_average)

Genre\_Budget\_Mean <- GenreMeans(Movies$budget)

Genre\_Revenue\_Mean <- GenreMeans(Movies$revenue)

Genre\_Pop\_Mean <- GenreMeans(Movies$popularity)

Genre\_Run\_Mean <- GenreMeans(Movies$runtime)

Genre\_PB\_Mean <- GenreMeans(Movies$Percent\_Budget)

Genre\_Dif\_Mean <- GenreMeans(Movies$Difference)

GenreMedians <- function(column){

Genre1Med <- aggregate(x=column,list(Movies$genre1),median)

Genre2Med <- aggregate(x=column,list(Movies$genre2),median)

Genre3Med <- aggregate(x=column,list(Movies$genre3),median)

GenreMed <- merge(Genre1Med,Genre2Med, by="Group.1",all =TRUE, no.dups = T)

GenreMed <- merge(GenreMed,Genre3Med, by="Group.1",all =TRUE, no.dups = T)

Median <- rowMeans(cbind(GenreMed$x.x,GenreMed$x.y,GenreMed$x),na.rm=TRUE)

GenreMed <- cbind(GenreMed,Median)

GenreMed <- GenreMed[,c(1,5)]

colnames(GenreMed)[colnames(GenreMed)=="Group.1"] <- "Genre"

return(GenreMed)

}

Genre\_VA\_Median <-GenreMedians(Movies$vote\_average)

Genre\_Budget\_Median <- GenreMedians(Movies$budget)

Genre\_Revenue\_Median <- GenreMedians(Movies$revenue)

Genre\_Pop\_Median <- GenreMedians(Movies$popularity)

Genre\_Run\_Median <- GenreMedians(Movies$runtime)

Genre\_PB\_Median <- GenreMedians(Movies$Percent\_Budget)

Genre\_Dif\_Median <- GenreMedians(Movies$Difference)

MergeGenre <- function(df){

G1 <- merge(Genre\_VA\_Mean,Genre\_VA\_Median,by="Genre")

G2 <- merge(G1,Genre\_Budget\_Mean, by ="Genre")

G3 <- merge(G2,Genre\_Budget\_Median, by ="Genre")

G4 <- merge(G3,Genre\_Revenue\_Mean, by ="Genre")

G5 <- merge(G4,Genre\_Revenue\_Median, by ="Genre")

G6 <- merge(G5,Genre\_Pop\_Mean, by ="Genre")

G7 <- merge(G6,Genre\_Pop\_Median, by ="Genre")

G8 <- merge(G7,Genre\_Run\_Mean, by ="Genre")

G9 <- merge(G8,Genre\_Run\_Median, by ="Genre")

G10 <- merge(G9,Genre\_PB\_Mean, by ="Genre")

G11 <- merge(G10,Genre\_PB\_Median, by ="Genre")

G12 <- merge(G11,Genre\_Dif\_Mean, by ="Genre")

Genre\_Summary <- merge(G12,df, by ="Genre")

colnames(Genre\_Summary) <- c("Genre","VA\_mean","VA\_med","Budget\_mean","Budget\_med","Revenue\_mean","Revenue\_med","Pop\_mean","Pop\_med","Run\_mean","Run\_med","PB\_mean","PB\_med","Dif\_mean","Dif\_med")

return(Genre\_Summary)

}

Genre\_Summary <- MergeGenre(Genre\_Dif\_Median)

Genre\_Summary$Rev\_Score <- scale(Genre\_Summary$Revenue\_mean)

Genre\_Summary$VA\_Score <- scale(Genre\_Summary$VA\_mean)

Genre\_Summary$Pop\_Score <- scale(Genre\_Summary$Pop\_mean)

Genre\_Summary$ROI\_Score <- scale(Genre\_Summary$PB\_med)

Genre\_Summary$Total\_Score <- rowSums(Genre\_Summary[,c(16:19)])

GenreScoreplot <- ggplot(Genre\_Summary,aes(x=Rev\_Score,y=VA\_Score))+geom\_point(aes(color=ROI\_Score,size=Pop\_Score))+geom\_text(aes(label=Genre),size=2,color="red")+ggtitle("Genre Scores")

MonthMeans <- function(column){

MonthAvg <- aggregate(x=column,list(Movies$release\_month),mean)

colnames(MonthAvg)[colnames(MonthAvg)=="Group.1"] <- "Month"

colnames(MonthAvg)[colnames(MonthAvg)=="x"] <- "Average"

return(MonthAvg)

}

Month\_VA\_Mean <-MonthMeans(Movies$vote\_average)

Month\_Budget\_Mean <- MonthMeans(Movies$budget)

Month\_Revenue\_Mean <- MonthMeans(Movies$revenue)

Month\_Pop\_Mean <- MonthMeans(Movies$popularity)

Month\_Run\_Mean <- MonthMeans(Movies$runtime)

Month\_PB\_Mean <- MonthMeans(Movies$Percent\_Budget)

Month\_Dif\_Mean <- MonthMeans(Movies$Difference)

MonthMedians <- function(column){

MonthMed <- aggregate(x=column,list(Movies$release\_month),median)

colnames(MonthMed)[colnames(MonthMed)=="Group.1"] <- "Month"

colnames(MonthMed)[colnames(MonthMed)=="x"] <- "Median"

return(MonthMed)

}

Month\_VA\_Median <-MonthMedians(Movies$vote\_average)

Month\_Budget\_Median <- MonthMedians(Movies$budget)

Month\_Revenue\_Median <- MonthMedians(Movies$revenue)

Month\_Pop\_Median <- MonthMedians(Movies$popularity)

Month\_Run\_Median <- MonthMedians(Movies$runtime)

Month\_PB\_Median <- MonthMedians(Movies$Percent\_Budget)

Month\_Dif\_Median <- MonthMedians(Movies$Difference)

MergeMonth <- function(df){

G1 <- merge(Month\_VA\_Mean,Month\_VA\_Median,by="Month")

G2 <- merge(G1,Month\_Budget\_Mean, by ="Month")

G3 <- merge(G2,Month\_Budget\_Median, by ="Month")

G4 <- merge(G3,Month\_Revenue\_Mean, by ="Month")

G5 <- merge(G4,Month\_Revenue\_Median, by ="Month")

G6 <- merge(G5,Month\_Pop\_Mean, by ="Month")

G7 <- merge(G6,Month\_Pop\_Median, by ="Month")

G8 <- merge(G7,Month\_Run\_Mean, by ="Month")

G9 <- merge(G8,Month\_Run\_Median, by ="Month")

G10 <- merge(G9,Month\_PB\_Mean, by ="Month")

G11 <- merge(G10,Month\_PB\_Median, by ="Month")

G12 <- merge(G11,Month\_Dif\_Mean, by ="Month")

Month\_Summary <- merge(G12,df, by ="Month")

colnames(Month\_Summary) <- c("Month","VA\_mean","VA\_med","Budget\_mean","Budget\_med","Revenue\_mean","Revenue\_med","Pop\_mean","Pop\_med","Run\_mean","Run\_med","PB\_mean","PB\_med","Dif\_mean","Dif\_med")

return(Month\_Summary)

}

Month\_Summary <- MergeMonth(Month\_Dif\_Median)

Month\_Summary$Rev\_Score <- scale(Month\_Summary$Revenue\_mean)

Month\_Summary$VA\_Score <- scale(Month\_Summary$VA\_mean)

Month\_Summary$Pop\_Score <- scale(Month\_Summary$Pop\_mean)

Month\_Summary$ROI\_Score <- scale(Month\_Summary$PB\_med)

Month\_Summary$Total\_Score <- rowSums(Month\_Summary[,c(16:19)])

MonthScoreplot <- ggplot(Month\_Summary,aes(x=Rev\_Score,y=VA\_Score))+geom\_point(aes(color=ROI\_Score,size=Pop\_Score))+geom\_text(aes(label=Month),size=2,color="red")+ggtitle("Month Scores")

YearMeans <- function(column){

YearAvg <- aggregate(x=column,list(Movies$release\_year),mean)

colnames(YearAvg)[colnames(YearAvg)=="Group.1"] <- "Year"

colnames(YearAvg)[colnames(YearAvg)=="x"] <- "Average"

return(YearAvg)

}

Year\_VA\_Mean <- YearMeans(Movies$vote\_average)

Year\_Budget\_Mean <- YearMeans(Movies$budget)

Year\_Revenue\_Mean <- YearMeans(Movies$revenue)

Year\_Pop\_Mean <- YearMeans(Movies$popularity)

Year\_Run\_Mean <- YearMeans(Movies$runtime)

Year\_PB\_Mean <- YearMeans(Movies$Percent\_Budget)

Year\_Dif\_Mean <- YearMeans(Movies$Difference)

YearMedians <- function(column){

YearMed <- aggregate(x=column,list(Movies$release\_year),median)

colnames(YearMed)[colnames(YearMed)=="Group.1"] <- "Year"

colnames(YearMed)[colnames(YearMed)=="x"] <- "Median"

return(YearMed)

}

Year\_VA\_Median <-YearMedians(Movies$vote\_average)

Year\_Budget\_Median <- YearMedians(Movies$budget)

Year\_Revenue\_Median <- YearMedians(Movies$revenue)

Year\_Pop\_Median <- YearMedians(Movies$popularity)

Year\_Run\_Median <- YearMedians(Movies$runtime)

Year\_PB\_Median <- YearMedians(Movies$Percent\_Budget)

Year\_Dif\_Median <- YearMedians(Movies$Difference)

ProdCompMeans <- function(column){

PC1Avg <- aggregate(x=column,list(Movies$production\_company),mean)

PC2Avg <- aggregate(x=column,list(Movies$production\_company2),mean)

PC3Avg <- aggregate(x=column,list(Movies$production\_company3),mean)

PCAvg <- merge(PC1Avg,PC2Avg, by="Group.1",all =TRUE, no.dups = T)

PCAvg <- merge(PCAvg,PC3Avg, by="Group.1",all =TRUE, no.dups = T)

Average <- rowMeans(cbind(PCAvg$x.x,PCAvg$x.y,PCAvg$x),na.rm=TRUE)

PCAvg <- cbind(PCAvg,Average)

PCAvg <- PCAvg[,c(1,5)]

colnames(PCAvg)[colnames(PCAvg)=="Group.1"] <- "Production\_Company"

return(PCAvg)

}

Company\_VA\_Mean <- ProdCompMeans(Movies$vote\_average)

Company\_Budget\_Mean <- ProdCompMeans(Movies$budget)

Company\_Revenue\_Mean <- ProdCompMeans(Movies$revenue)

Company\_Pop\_Mean <- ProdCompMeans(Movies$popularity)

Company\_Run\_Mean <- ProdCompMeans(Movies$runtime)

Company\_PB\_Mean <- ProdCompMeans(Movies$Percent\_Budget)

Company\_Dif\_Mean <- ProdCompMeans(Movies$Difference)

ProdCompMedians <- function(column){

PC1Med <- aggregate(x=column,list(Movies$production\_company),median)

PC2Med <- aggregate(x=column,list(Movies$production\_company2),median)

PC3Med <- aggregate(x=column,list(Movies$production\_company3),median)

PCMed <- merge(PC1Med,PC2Med, by="Group.1",all =TRUE, no.dups = T)

PCMed <- merge(PCMed,PC3Med, by="Group.1",all =TRUE, no.dups = T)

Median <- rowMeans(cbind(PCMed$x.x,PCMed$x.y,PCMed$x),na.rm=TRUE)

PCMed <- cbind(PCMed,Median)

PCMed <- PCMed[,c(1,5)]

colnames(PCMed)[colnames(PCMed)=="Group.1"] <- "Production\_Company"

return(PCMed)

}

Company\_VA\_Median <- ProdCompMedians(Movies$vote\_average)

Company\_Budget\_Median <- ProdCompMedians(Movies$budget)

Company\_Revenue\_Median <- ProdCompMedians(Movies$revenue)

Company\_Pop\_Median <- ProdCompMedians(Movies$popularity)

Company\_Run\_Median <- ProdCompMedians(Movies$runtime)

Company\_PB\_Median <- ProdCompMedians(Movies$Percent\_Budget)

Company\_Dif\_Median <- ProdCompMedians(Movies$Difference)

CountryMeans <- function(column){

CountryAvg <- aggregate(x=column,list(Movies$production\_country),mean)

colnames(CountryAvg)[colnames(CountryAvg)=="Group.1"] <- "Country"

colnames(CountryAvg)[colnames(CountryAvg)=="x"] <- "Average"

return(CountryAvg)

}

Country\_VA\_Mean <- CountryMeans(Movies$vote\_average)

Country\_Budget\_Mean <- CountryMeans(Movies$budget)

Country\_Revenue\_Mean <- CountryMeans(Movies$revenue)

Country\_Pop\_Mean <- CountryMeans(Movies$popularity)

Country\_Run\_Mean <- CountryMeans(Movies$runtime)

Country\_PB\_Mean <- CountryMeans(Movies$Percent\_Budget)

Country\_Dif\_Mean <- CountryMeans(Movies$Difference)

CountryMedians <- function(column){

CountryMed <- aggregate(x=column,list(Movies$production\_country),median)

colnames(CountryMed)[colnames(CountryMed)=="Group.1"] <- "Country"

colnames(CountryMed)[colnames(CountryMed)=="x"] <- "Median"

return(CountryMed)

}

Country\_VA\_Median <- CountryMedians(Movies$vote\_average)

Country\_Budget\_Median <- CountryMedians(Movies$budget)

Country\_Revenue\_Median <- CountryMedians(Movies$revenue)

Country\_Pop\_Median <- CountryMedians(Movies$popularity)

Country\_Run\_Median <- CountryMedians(Movies$runtime)

Country\_PB\_Median <- CountryMedians(Movies$Percent\_Budget)

Country\_Dif\_Median <- CountryMedians(Movies$Difference)

Movies$boxoffice <- ifelse(Movies$Difference<=0,"Bomb","Not Bomb")

length(which(Movies$boxoffice == "Bomb"))

length(which(Movies$boxoffice == "Bomb"))/length(Movies$title)

ActorSum <- function(column){

A1Sum <- aggregate(x=column,list(Movies$actor\_1),sum)

A2Sum <- aggregate(x=column,list(Movies$actor\_2),sum)

A3Sum <- aggregate(x=column,list(Movies$actor\_3),sum)

ASum <- merge(A1Sum,A2Sum, by="Group.1",all =TRUE, no.dups = T)

ASum <- merge(ASum,A3Sum, by="Group.1",all =TRUE, no.dups = T)

Sum <- rowSums(cbind(ASum$x.x,ASum$x.y,ASum$x),na.rm=TRUE)

ASum <- cbind(ASum,Sum)

ASum <- ASum[,c(1,5)]

colnames(ASum)[colnames(ASum)=="Group.1"] <- "Actor"

colnames(ASum)[colnames(ASum)=="Sum"] <- "Total\_Revenue"

return(ASum)

}

ActorRevenue <- ActorSum(Movies$revenue)

ActorRevenue <- ActorRevenue[order(-ActorRevenue$Total\_Revenue),]

DirectorSum <- function(column){

DSum <- aggregate(x=column,list(Movies$director),sum)

colnames(DSum)[colnames(DSum)=="Group.1"] <- "Director"

colnames(DSum)[colnames(DSum)=="x"] <- "Total\_Revenue"

return(DSum)

}

DirectorRevenue <- DirectorSum(Movies$revenue)

DirectorRevenue <- DirectorRevenue[order(-DirectorRevenue$Total\_Revenue),]

randIndex <- sample(1:dim(Movies)[1])

cut\_point <- floor(2\*dim(Movies)[1]/3)

Moviestrain <- Movies[randIndex[1:cut\_point],]

Moviestest <- Movies[randIndex[(cut\_point+1):dim(Movies)[1]],]

KSVMmod <- ksvm(boxoffice~budget+revenue+release\_month+popularity+vote\_average+vote\_count+runtime,data=Moviestrain,C=25)

KSVM\_Predict <- predict(KSVMmod, Moviestest)

compTable <- data.frame(Moviestest$boxoffice,KSVM\_Predict)

T <- table(compTable)

(T[1,1]+T[2,2])/nrow(Moviestest)\*100

Moviestrain$boxnum <- as.factor(ifelse(Moviestrain$boxoffice=="Bomb",0,1))

Moviestest$boxnum <- as.factor(ifelse(Moviestest$boxoffice=="Bomb",0,1))

NBmod <- naiveBayes(boxnum~budget+release\_month+popularity+vote\_average+vote\_count+runtime,data=Moviestrain)

NB\_Predict <- predict(NBmod, Moviestest)

compTable2 <- data.frame(Moviestest$boxnum,NB\_Predict)

T2 <- table(compTable2)

(T2[1,1]+T2[2,2])/nrow(Moviestest)\*100

Moviestest$predict <- compTable[,2]

KSVMplot <- ggplot(Moviestest, aes(x=Difference,y=Percent\_Budget)) + geom\_point(aes(shape=boxoffice,color=predict))

KSVMplot <- KSVMplot + xlim(-2000000,20000000)+ylim(0,500)

KSVMplot <- KSVMplot + xlab("Profit") +ylab("ROI")+ggtitle("KSVM Prediction Plot")

Act <- as.data.frame(table(Movies$actor\_1))

Act <- Act[order(-Act$Freq),]

Act2 <- as.data.frame(table(Movies$actor\_2))

Act2 <- Act2[order(-Act2$Freq),]

Act3 <- as.data.frame(table(Movies$actor\_3))

Act3 <- Act3[order(-Act3$Freq),]

Actor\_Freq <- merge(Act,Act2,by = "Var1")

Actor\_Freq <- merge(Actor\_Freq,Act3,by = "Var1")

Actor\_Freq$Frequency <- Actor\_Freq$Freq.x+Actor\_Freq$Freq.y+Actor\_Freq$Freq

Actor\_Freq <- Actor\_Freq[,c(1,5)]

Actor\_Freq <- Actor\_Freq[which(Actor\_Freq$Frequency>=10),]

colnames(Actor\_Freq)<- c("actor","number\_movies")

Movies\_Actors <- filter(Movies,actor\_1 | actor\_2 | actor\_3 %in% Actor\_Freq$actor)

ActorScore <- function(col){

A <- aggregate(col,list(Movies\_Actors$actor\_1),mean)

B <- aggregate(col,list(Movies\_Actors$actor\_2),mean)

C <- aggregate(col,list(Movies\_Actors$actor\_3),mean)

D <- merge(A,B, by = "Group.1",all=TRUE)

E <- merge(D,C, by="Group.1",all=TRUE)

E$average <- rowMeans(E[2:4],na.rm =TRUE)

E <- E[,c(1,5)]

colnames(E) <- c("actor","average")

E <- filter(E,actor%in%Actor\_Freq$actor)

return(E)

}

ARev <- ActorScore(Movies\_Actors$revenue)

AVA <-ActorScore(Movies\_Actors$vote\_average)

APop <- ActorScore(Movies\_Actors$popularity)

AROI <- ActorScore(Movies\_Actors$Percent\_Budget)

Actor\_Freq <- Actor\_Freq[-125,]

TopActors <- data.frame(Actor\_Freq$actor,ARev$average,AVA$average,APop$average,AROI$average)

colnames(TopActors) <- c("Actor","Avg\_Rev","Avg\_VA","Avg\_Pop","Avg\_ROI")

TopActors$Rev\_Score <- scale(TopActors$Avg\_Rev)

TopActors$VA\_Score <- scale(TopActors$Avg\_VA)

TopActors$Pop\_Score <- scale(TopActors$Avg\_Pop)

TopActors$ROI\_Score <- scale(TopActors$Avg\_ROI)

TopActors$Actor\_Score <- rowSums(TopActors[,c(6:9)])

TopActors <- TopActors[order(-TopActors$Actor\_Score),]

Dir <- as.data.frame(table(Movies$director))

Dir <- Dir[order(-Dir$Freq),]

Dir <- Dir[which(Dir$Freq>=6),]

colnames(Dir)<- c("director","number\_movies")

Movies\_Director <- filter(Movies,director %in% Dir$director)

DirectorScore <- function(col){

A <- aggregate(col,list(Movies\_Director$director),mean)

colnames(A) <- c("director","average")

A <- filter(A,director%in%Dir$director)

return(A)

}

DRev <- DirectorScore(Movies\_Director$revenue)

DVA <- DirectorScore(Movies\_Director$vote\_average)

DPop <- DirectorScore(Movies\_Director$popularity)

DROI <- DirectorScore(Movies\_Director$Percent\_Budget)

TopDirectors <- data.frame(Dir$director,DRev$average,DVA$average,DPop$average,DROI$average)

colnames(TopDirectors) <- c("Director","Avg\_Rev","Avg\_VA","Avg\_Pop","Avg\_ROI")

TopDirectors$Rev\_Score <- scale(TopDirectors$Avg\_Rev)

TopDirectors$VA\_Score <- scale(TopDirectors$Avg\_VA)

TopDirectors$Pop\_Score <- scale(TopDirectors$Avg\_Pop)

TopDirectors$ROI\_Score <- scale(TopDirectors$Avg\_ROI)

TopDirectors$Director\_Score <- rowSums(TopDirectors[,c(6:9)])

TopDirectors <- TopDirectors[order(-TopDirectors$Director\_Score),]

TopActors1 <- head(TopActors[order(-TopActors$Avg\_Rev),],10)

Actplot <- ggplot(TopActors1, aes(Actor,Avg\_Rev))

ActRevplot <- Actplot + geom\_col() + theme(axis.text.x = element\_text(angle=70,hjust =1),axis.text =element\_text(size=7),axis.title =element\_text(size = 8))

fx <- function(x){

x/1000000

}

ActorScoreplot <- ggplot(TopActors,aes(x=Rev\_Score,y=VA\_Score))+geom\_point(aes(color=ROI\_Score,size=Pop\_Score))+geom\_text(aes(label=ifelse(Rev\_Score>4|VA\_Score<(-4)|ROI\_Score>4|Pop\_Score>5,as.character(Actor),"")))

TopActors20 <- head(TopActors,20)

TopDirectors20 <-head(TopDirectors, 20)

ActorScoreplot20 <- ggplot(TopActors20,aes(x=Rev\_Score,y=VA\_Score))+geom\_point(aes(color=ROI\_Score,size=Pop\_Score))+geom\_text(aes(label=Actor),size=2,color="red")+ggtitle("Top 20 Actor Scores")

DirectorScoreplot20 <- ggplot(TopDirectors20,aes(x=Rev\_Score,y=VA\_Score))+geom\_point(aes(color=ROI\_Score,size=Pop\_Score))+geom\_text(aes(label=Director),size=2,color="red")+ggtitle("Top 20 Director Scores")

```

**Data Dictionary**

