TMDB Box Office Prediction

Janice Li

4/29/2019

Project Introduction

When we think of a movie that earns top box office revenue, we often find these elements in it: all-star casts, renowned director, popular franchise, etc. However, there has never been a concrete answer about how much each of these elements affects movies' revenue. This project, TMDB Box Office Revenue Prediction, intends to investigate this question through explanatory data analysis and build machine learning models to predict movies' revenue according to their features.

Data Overview

```
data=read.csv("/Users/janiceli/Downloads/train.csv")
dim(data)
```

```
## [1] 3000   23
```

Data Cleaning and Feature Engineering

The dataset contains records of 3000 movies from The Movie Database(TMDB) and embodies 23 variables, including the one we aim to predict, revenue. As the first step of data cleaning, I got rid of features that obvious do not relate with the response variable: id, homepage(hmtl),imdb_id and poster_path(html). Some features may be investigated with advanced NLP methods, like title/original_title, overview and tagline, but those are beyond the scope of this project. Therefore, I deleted those variables, too.

```
data=data[,c(2:4,7,10,12:17,20:23)]
colnames(data)
```

```
##  [1] "belongs_to_collection" "budget"
##  [3] "genres"                "original_language"
##  [5] "popularity"            "production_companies"
##  [7] "production_countries"  "release_date"
##  [9] "runtime"               "spoken_languages"
## [11] "status"                "Keywords"
## [13] "cast"                  "crew"
## [15] "revenue"
```

We will clean and adjust these variables one by one.

Belongs_to_collection

This veriable indicates whether the movie belongs to a specific collection/franchise. As you see above, the data was stored as python dictionaries with lots of irrelevant information. I used functions in “stringr” library to extract the collection names and save them as a variable “Collection_Name”. Furthermore, the fact that a movie belongs to a series might also be useful information. Thus I created a boolean varaible “Has_Collection” to indicate whether the movie is part of any franchise.

```
library("stringr")
data$Collection_Name=str_extract_all(data$belongs_to_collection,"(?<=name\\'\\:\\s{1}\\').+(?=\\'\\,\\s{1}\\'post
er)")
data$Collection_Name[data$Collection_Name=="character(0)"]=NA
data$Has_Collection=!is.na(data$Collection_Name)
data$Collection_Name=sub(" Collection","",data$Collection_Name)
data$Collection_Name=unlist(data$Collection_Name)
```

Unfortunately, I cannot use “Collection_Name” as model inputs since it contains too many category levels for the machine to learn. Alternatively, I created a variable, “belongs_to_popular_collection”, to reflect whether the collection is one of the 117 most popular franchises that earned the most average box office revenue. The list of popular franchises was scraped from https://www.the-numbers.com/movies/franchises (https://www.the-numbers.com/movies/franchises). All the rankings are supported by worldwide box office record.

```
Popular_Collection_List=c('Marvel Cinematic Universe','Star Wars',"J.K. Rowling's Wizarding World",'Avengers','Disney Live Action Reimaginings','Batman','Harry Potter','X-Men','James Bond','Spider-Man','DC Extended Universe','Wonder Woman','Middle Earth','Jurassic Park','Transformers','The Fast and the Furious','Pirates of the Caribbean','The Hunger Games','Shrek','Star Trek','Twilight','Despicable Me','The Dark Knight Trilogy','Mission: Impossible','Superman','The Lord of the Rings','Iron Man','Indiana Jones','Toy Story','The Incredibles','Captain America','Finding Nemo','The Hobbit','Bourne','Planet of the Apes','Ice Age','Avatar','Star Wars Anthology','Guardians of the Galaxy','Thor','Deadpool','Madagascar','Alvin and the Chipmunks','The Hangover','Men in Black','Terminator','LEGO','Fockers','The Matrix','Alien','Cars','Teenage Mutant Ninja Turtles','Madea',"Ocean's 11",'How to Train Your Dragon','The Mummy','The Conjuring','Night at the Museum','Wolverine','The Chronicles of Narnia','Planet of the Apes (2011-2017)','Monsters, Inc.','Kung Fu Panda','Rush Hour','Die Hard','Jack Ryan','Home Alone','Lethal Weapon','Hotel Transylvania','Austin Powers','Ghostbusters','Halloween','Scary Movie','Saw','King Kong','Beverly Hills Cop','The Karate Kid','Hannibal Lecter','Back to the Future','Alice in Wonderland','Independence Day','American Pie','Jumanji','Jaws','Paranormal Activity','Frozen','Ant-Man','Sherlock Holmes','Fantastic Beasts','National Treasure','Wreck-It Ralph','Robert Langdon','Fifty Shades','Friday the 13th','Taken','Nightmare on Elm Street','Godzilla and Kong Universe','The Santa Clause','The Secret Life of Pets','Jackass','Pitch Perfect Trilogy','Godzilla','Spy Kids','Divergent','Unbreakable','Fantastic Four','The Muppets','Minions','Aquaman','Scream','Jump Street','It','300','Predator','Crocodile Dundee','My Big Fat Greek Wedding','Ted')


data$belongs_to_popular_collection=data$Collection_Name %in% Popular_Collection_List
```

Take a look at the 3 new variables:

```
head(data[,16:18],5)
```

```
##        Collection_Name Has_Collection belongs_to_popular_collection
## 1 Hot Tub Time Machine           TRUE                         FALSE
## 2 The Princess Diaries           TRUE                         FALSE
## 3                   NA          FALSE                         FALSE
## 4                   NA          FALSE                         FALSE
## 5                   NA          FALSE                         FALSE
```

We see that whether a movie belongs to franchise significantly affects box office revenue. Moreover, whether it belongs to a popular franchise shows even more apparent correlation.

```
par(mfrow=c(1:2))
boxplot(data$revenue~data$Has_Collection,ylim=c(0,1000000000),main="Revenue ~ Has_Collection",ylab="Revenue")
```

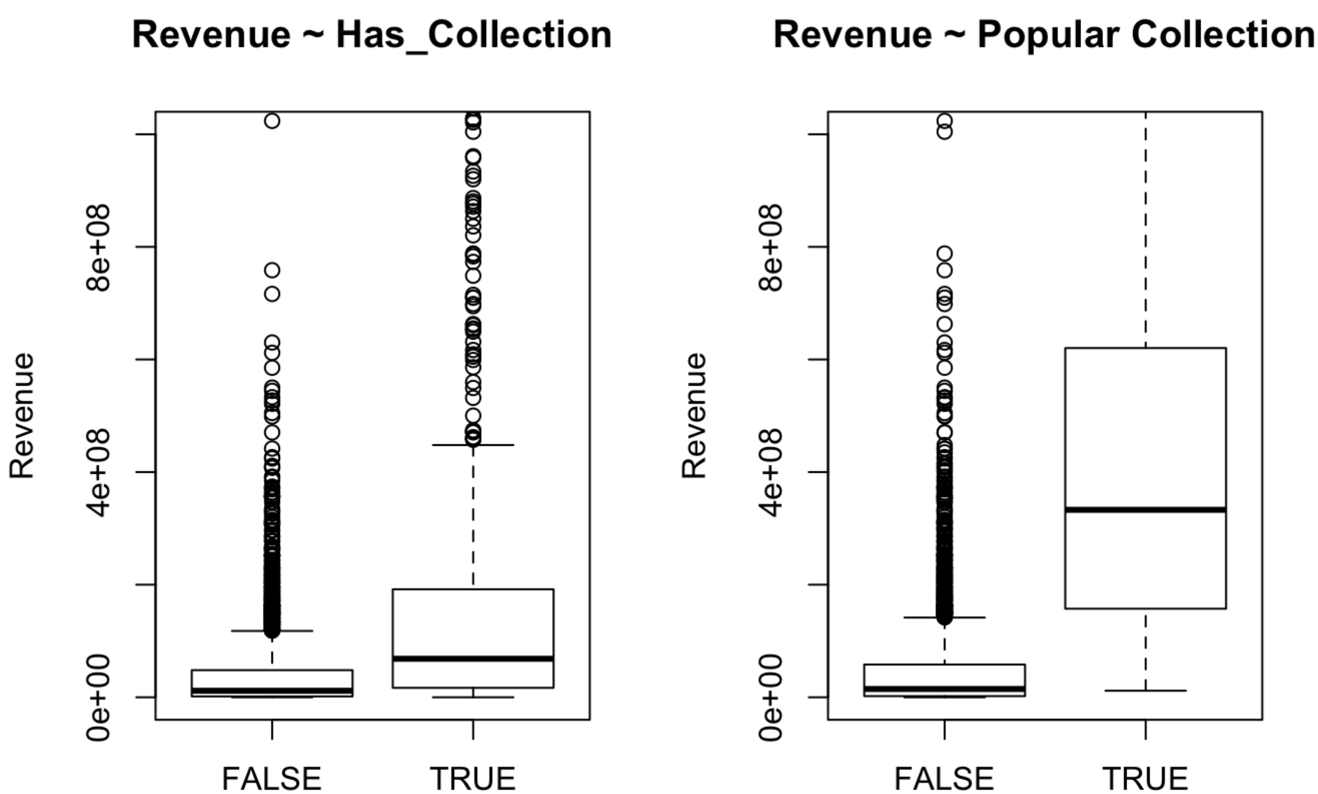
```
## Warning in x[floor(d)] + x[ceiling(d)]: NAs produced by integer overflow

## Warning in x[floor(d)] + x[ceiling(d)]: NAs produced by integer overflow
```

```
boxplot(data$revenue~data$belongs_to_popular_collection,ylim=c(0,1000000000),main="Revenue ~ Popular Collection",
ylab="Revenue")
```

```
## Warning in x[floor(d)] + x[ceiling(d)]: NAs produced by integer overflow

## Warning in x[floor(d)] + x[ceiling(d)]: NAs produced by integer overflow
```



Genres

Similar with “belongs_to_collection”, “Genres” is also saved as python dictionary format and mingled with irrelevant information. What makes to task more difficult is the fact that movies can have multiple genres, and I cannot store them all in a single variable. To solve this, I decided to create a sparse matrix with genre names being the column names and store boolean values in it to show whether a movie belongs to each genre. I also created a new variable “Genre_Count” to store the number of genres that the movies belong to since it might also be meaningful information.

```
data$Genre_Count=str_count(data$genres,"\\{")
Genres=as.data.frame(str_split_fixed(data$genres,"\\}\\,\\s\\{",data$Genre_Count),stringsAsFactors = FALSE)
Genres_2 <- as.data.frame(sapply(Genres, function(x) str_extract(x, "(?<=name\\'\\:\\s{1}\\').+(?=\\')")), string
sAsFactors = F)
```

```
head(Genres_2)
```

```
##          V1        V2     V3      V4   V5   V6   V7
## 1    Comedy      <NA>   <NA>    <NA> <NA> <NA> <NA>
## 2    Comedy     Drama Family Romance <NA> <NA> <NA>
## 3     Drama      <NA>   <NA>    <NA> <NA> <NA> <NA>
## 4  Thriller     Drama   <NA>    <NA> <NA> <NA> <NA>
## 5    Action  Thriller   <NA>    <NA> <NA> <NA> <NA>
## 6 Animation Adventure Family    <NA> <NA> <NA> <NA>
```

```
library("dplyr")
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
Genre_Titles=c("Action","Adventure","Animation","Comedy","Crime","Documentary","Drama","Family","Fantasy","Foreig
n","History","Horror","Music","Mystery","Romance","Science_Fiction","Thriller","TV_Movie","War","Western")
Genres_3=data.frame(matrix(nrow=0,ncol=length(Genre_Titles)))
colnames(Genres_3)=Genre_Titles
for (i in 1:length(Genre_Titles)) {
  for (j in 1:nrow(Genres_2)) {
      Genres_3[j,i]=Genre_Titles[i] %in% Genres_2[j,]
  }
}
data=cbind(data,as.data.frame(Genres_3))
```

This is how the sparse matrix looks like. I combined it with “data” to make the 17 genre names new features.

```
head(Genres_3,5)
```

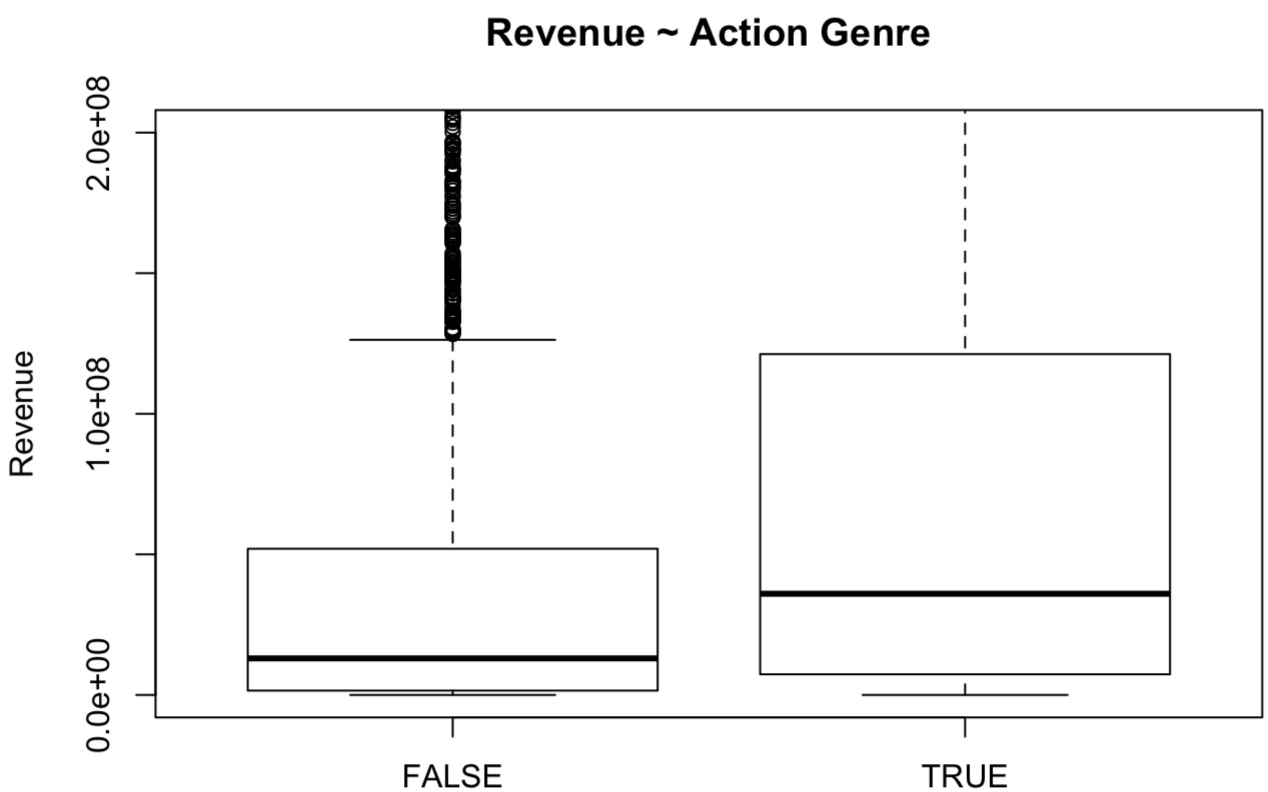
```
##   Action Adventure Animation Comedy Crime Documentary Drama Family Fantasy
## 1  FALSE     FALSE     FALSE   TRUE FALSE       FALSE FALSE  FALSE   FALSE
## 2  FALSE     FALSE     FALSE   TRUE FALSE       FALSE  TRUE   TRUE   FALSE
## 3  FALSE     FALSE     FALSE  FALSE FALSE       FALSE  TRUE  FALSE   FALSE
## 4  FALSE     FALSE     FALSE  FALSE FALSE       FALSE  TRUE  FALSE   FALSE
## 5   TRUE     FALSE     FALSE  FALSE FALSE       FALSE FALSE  FALSE   FALSE
##   Foreign History Horror Music Mystery Romance Science_Fiction Thriller
## 1   FALSE   FALSE  FALSE FALSE   FALSE   FALSE           FALSE    FALSE
## 2   FALSE   FALSE  FALSE FALSE   FALSE    TRUE           FALSE    FALSE
## 3   FALSE   FALSE  FALSE FALSE   FALSE   FALSE           FALSE    FALSE
## 4   FALSE   FALSE  FALSE FALSE   FALSE   FALSE           FALSE     TRUE
## 5   FALSE   FALSE  FALSE FALSE   FALSE   FALSE           FALSE     TRUE
##   TV_Movie   War Western
## 1    FALSE FALSE   FALSE
## 2    FALSE FALSE   FALSE
## 3    FALSE FALSE   FALSE
## 4    FALSE FALSE   FALSE
## 5    FALSE FALSE   FALSE
```

Taking “Adventure” as an example. We see that genres are significant indicators of the films’ box office revenue.

```
boxplot(data$revenue~data$Action,ylim=c(0,200000000),main="Revenue ~ Action Genre",ylab="Revenue")
```

```
## Warning in x[floor(d)] + x[ceiling(d)]: NAs produced by integer overflow

## Warning in x[floor(d)] + x[ceiling(d)]: NAs produced by integer overflow
```



Production Companies

The same data cleaning method was applied to "production_companies". I extracted the company names and map them to the list of top 100 entertainment companies that earned top average box office revenue. The list was scraped from https://www.the-numbers.com/movies/production-companies/ (https://www.the-numbers.com/movies/production-companies/). Notice that though some movies were made by multiple companies, I only extracted the first one mentioned, which reasonably plays a major role in production. I cannot make sparse matrix for this variable as I did for “genres”, since the abundance of companies in the data could hugely raise the number of columns and causes dimenality problems. Also, some research in the film industry told me that production companies do not influence audiences’s decision too much, so it should be enough to only consider the major company.

```
data$Company=str_extract_all(data$production_companies,"(?<=name\\'\\:\\s{1}\\').+(?=\\'\\,\\s{1}\\'id)")
data$Company=gsub("'.*","",data$Company)
data$Company[data$Company=="character(0)"]=NA
```

```
Popular_Company_List=c('Warner Bros.','Universal Pictures','Columbia Pictures','Walt Disney Pictures','Marvel Stu
dios','Paramount Pictures','20th Century Fox','Dune Entertainment','Legendary Pictures','Relativity Media','Dream
Works Animation','Amblin Entertainment','DreamWorks Pictures','New Line Cinema','Disney-Pixar','Regency Enterpris
es','Village Roadshow Productions','Metro-Goldwyn-Mayer Pictures','Heyday Films','Lucasfilm','RatPac Entertainmen
t','Walt Disney Animation Studios','Lionsgate','Summit Entertainment','Touchstone Pictures','di Bonaventura Pictu
res','Working Title Films','Jerry Bruckheimer','Original Film','Eon Productions','Wingnut Films','Illumination En
tertainment','1492 Pictures','TSG Entertainment','Bad Robot','The Kennedy/Marshall Company','Fox 2000 Pictures',
'Skydance Productions','Imagine Entertainment','Twentieth Century Fox','Perfect World Pictures','Temple Hill Ente
rtainment','Ingenious Film Partners','Hasbro Studios','Blue Sky Studios','PDI','Syncopy','One Race Films','Sony P
ictures Animation','Donners’ Company','Canal Plus','Atlas Entertainment','Silver Pictures','Spyglass Entertainmen
t','Blumhouse','Ingenious Media','New Regency','Scott Free Films','Chernin Entertainment','Dentsu Inc.','Walden M
edia','SunsweptEntertainment','Cruel and Unusual Films','Happy Madison','Zancuk Company','Davis Entertainment','L
Star Capital','Scott Rudin Productions','StudioCanal','Tri-Star Pictures','Centropolis Entertainment','Overbrook
 Entertainment','GK Films','Kinberg Genre','Color Force','Brian Grazer Productions','Roth Films','Chris Meledandr
i','Mandeville Films','Screen Gems','The Safran Company','Revolution Studios','Participant Media','Castle Rock En
tertainment','Weinstein Company','Cruise-Wagner','Lightstorm Entertainment','Fox Searchlight Pictures','United Ar
tists','Plan B Entertainment','China Film Company','Bad Hat Harry Productions','Laura Ziskin Productions','Focus
 Features','20th Century Fox Animation','Parkes+Macdonald Productions','Vertigo Entertainment','Fairview Entertai
nment','Wanda Media','EuropaCorp')
data$belongs_to_popular_company=data$Company %in% Popular_Company_List
```

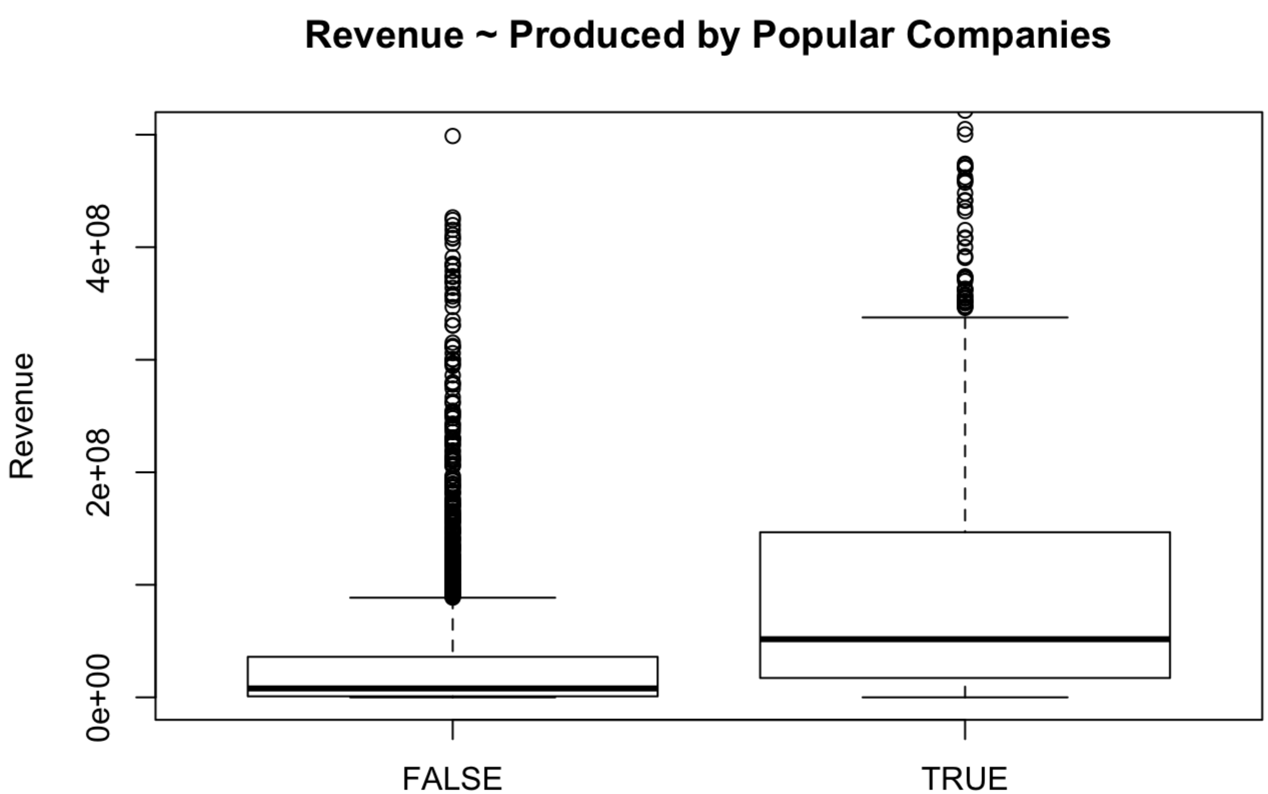
```
head(data[,39:40])
```

```
##   Western              Company
## 1   FALSE   Paramount Pictures
## 2   FALSE Walt Disney Pictures
## 3   FALSE           Bold Films
## 4   FALSE                 <NA>
## 5   FALSE                 <NA>
## 6   FALSE                 <NA>
```

We see that the variable “belongs_to_popular_company” does explain some variance in revenue.

```
boxplot(data$revenue~data$belongs_to_popular_company,ylim=c(0,500000000),main="Revenue ~ Produced by Popular Comp
anies",ylab="Revenue")
```

```
## Warning in x[floor(d)] + x[ceiling(d)]: NAs produced by integer overflow
```



Cast

Cast is argubly one of the most influencial elements to suggest film popularity. To leverage this indicator, I created two variables, “Cast_Count” and “Popular_Cast_Count”. The former is simply the number of casts that acted in the movie, and the latter shows how many of them are among the top 1000 actors/actresses, ranked by the total amount of worldwide revenue generated by all the movies a star has appeared in over their lifetime. The data source is https://www.the-numbers.com/box-office-star-records/worldwide/lifetime-acting/top-grossing-stars (https://www.the-numbers.com/box-office-star-records/worldwide/lifetime-acting/top-grossing-stars).

```
data$Cast_Count=str_count(data$cast,"\\{")
Casts=as.data.frame(str_split_fixed(data$cast,"\\}\\,\\s\\{",data$Cast_Count),stringsAsFactors = FALSE)
Casts_2 <- as.data.frame(sapply(Casts, function(x) str_extract(x, "(?<=name\\'\\:\\s{1}\\').+(?=\\')")), stringsA
sFactors = F)
Casts_3 <- as.data.frame(sapply(Casts_2, function(x) sub("\'.*","",x)), stringsAsFactors = F)
```

```
Popular_Cast_List=read.csv("/Users/janiceli/Desktop/cast.csv")
Casts_4=Casts_3
for (i in 1:dim(Casts_3)[2]) {
  Casts_4[,i]=tolower(Casts_3[,i]) %in% tolower(Popular_Cast_List$Cast.Name)
}
data$Popular_Cast_Count=rowSums(Casts_4)
```

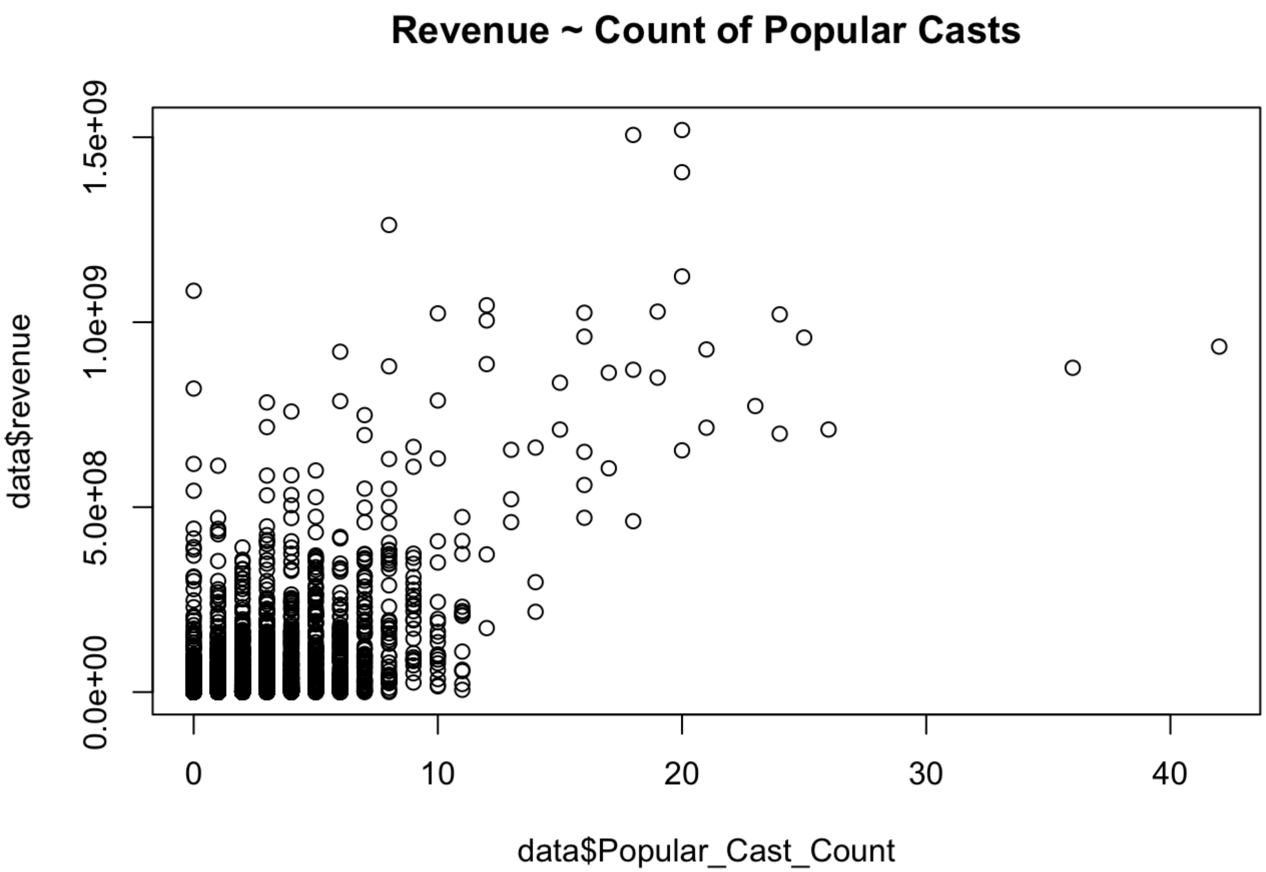
We see that nearly 1/3 of movies do not have any “popular casts”, while another 1/3 have 2-3 “popular casts”. The rest films have more than 3 “popular casts” participating.

```
table(data$Popular_Cast_Count)
```

```
## 
##   0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17 
## 968 544 423 326 246 167 105  73  52  25  20  13   5   3   3   2   5   2 
##  18  19  20  21  23  24  25  26  36  42 
##   3   2   4   2   1   2   1   1   1   1
```

We plot “Popular_Cast_Count” V.S. “Revenue”. Though it is not clear, we do see a positive correlation between the two variables.

```
plot(data$Popular_Cast_Count,data$revenue,main="Revenue ~ Count of Popular Casts")
```



Production Country

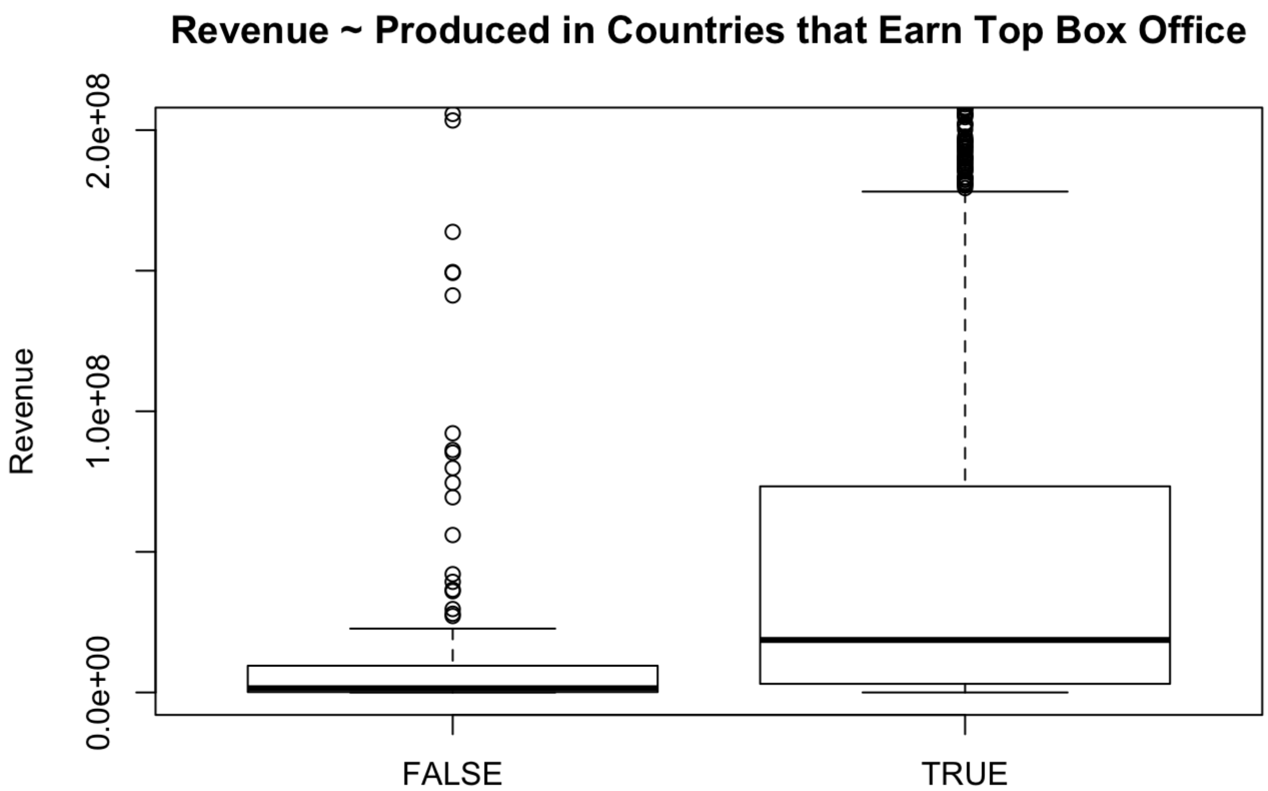
I created a variable “belongs_to_country_list” to store whether the movie is produced in one of the 19 countries that earned the most revenue per movie. The list was scraped from https://www.the-numbers.com/movies/production-countries/#tab=territory (https://www.the-numbers.com/movies/production-countries/#tab=territory).

```
production_countries=str_extract_all(data$production_countries,"(?<=name\\'\\:\\s{1}\\').+(?=\\')")
data$Country=gsub("\'.*","",production_countries)
data$Country[data$Country=="character(0)"]=NA
Popular_Country_List=c('United States of America','United Kingdom','China','France','Japan','Germany','New Zealan
d','Australia','South Korea','Canada','India','Hong Kong','Italy','Spain','Russia','Belgium','Mexico','Sweden','N
etherlands')
data$belongs_to_country_list=data$Country %in% Popular_Country_List
```

The correlation is significant.

```
boxplot(data$revenue~data$belongs_to_country_list,ylim=c(0,200000000),main="Revenue ~ Produced in Countries that
 Earn Top Box Office",ylab="Revenue")
```

```
## Warning in x[floor(d)] + x[ceiling(d)]: NAs produced by integer overflow
```

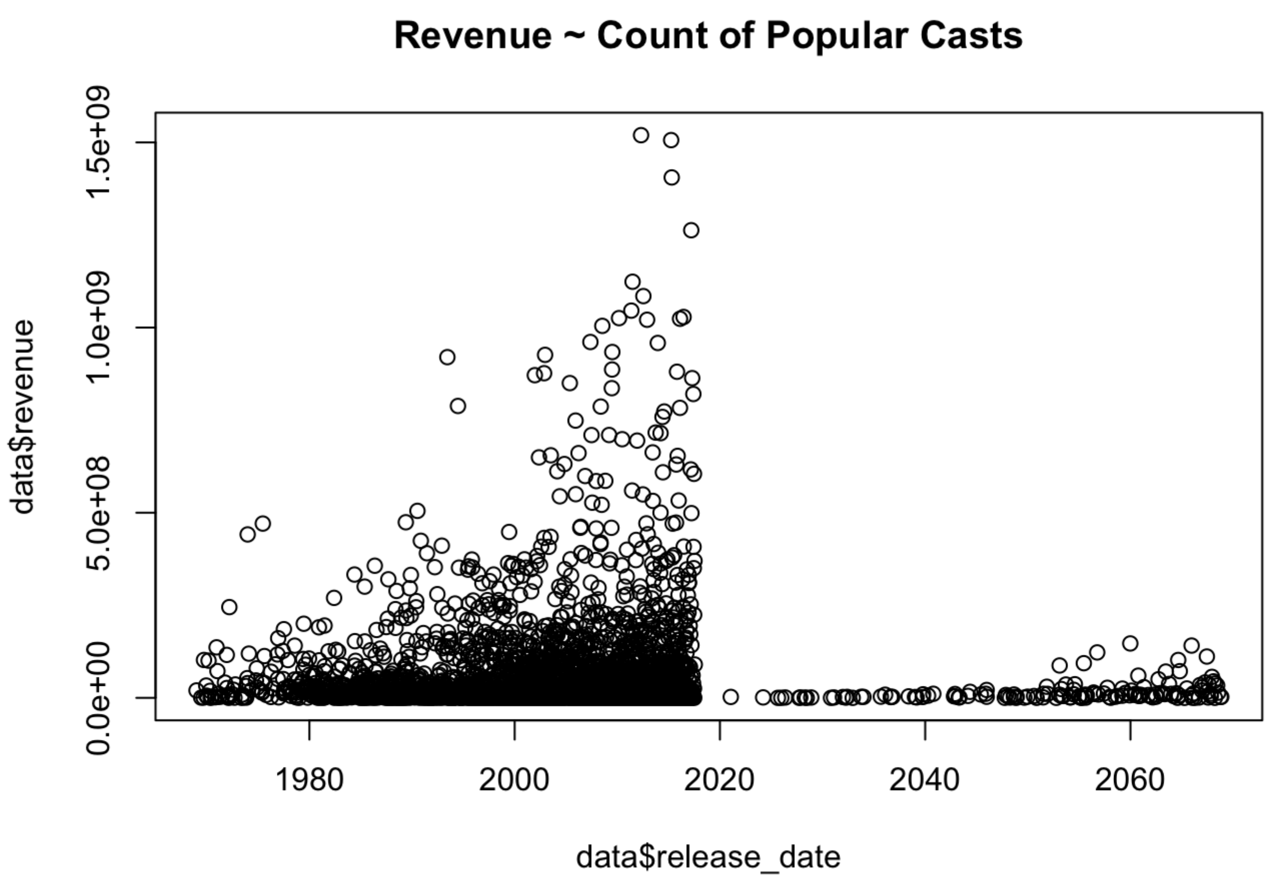


Released Date

Released dates reasonably plays a part in deciding a movie's box office revenue since audiences' buying power and appreciation of cinematic arts have been changing over time. So I converted it into a continuous variable.

```
data$release_date=as.character(data$release_date)
data$release_date=as.Date(data$release_date,format="%m/%d/%y")
```

```
plot(data$release_date,data$revenue,main="Revenue ~ Count of Popular Casts")
```



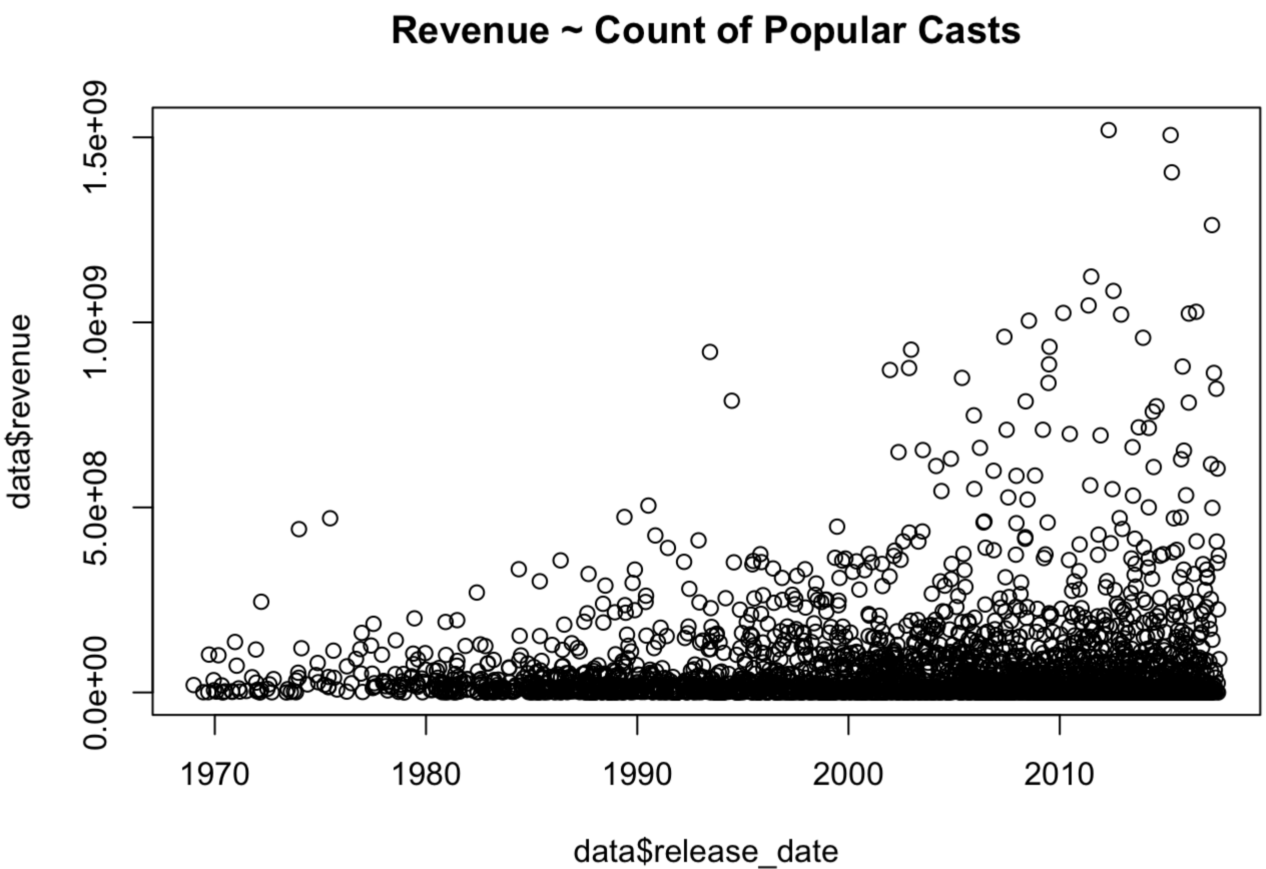
We do see a corelation between movie released date and revenue. However, how come that some movies have released dates in the future? I suppose those are either data entry error or unproved prediction. Let's dig into these movies that are released in the future:

```
Future_Movie=data[which(data$release_date>as.Date("01/01/2019",format="%m/%d/%y")),]
dim(Future_Movie)
```

```
## [1] 146  45
```

Since there are only have 146 data points that have abnormal released dates, I believe that deleting them would not cause much information loss, so I did it.

```
data=data[-which(data$release_date>as.Date("01/01/2019",format="%m/%d/%y")),]
plot(data$release_date,data$revenue,main="Revenue ~ Count of Popular Casts")
```



Keyword

To handle the Keyword variable, I created two variables “Keyword_Count” and “Popular_Keyword_Count”. The first shows the number of keywords that the movie contains, while the second shows the number of “popular keywords”. The list of 1000 “Popular keywords” was scraped from https://www.the-numbers.com/ (https://www.the-numbers.com/).

```
keywords_list=read.csv("/Users/janiceli/Desktop/Popular Keywords.csv")
data$Keyword_Count=str_count(data$Keywords,"\\{")
Keywords=as.data.frame(str_split_fixed(data$Keywords,"\\}\\,\\s\\{",data$Keyword_Count),stringsAsFactors = FALSE)
```

```
Keywords_2 <- as.data.frame(sapply(Keywords, function(x) str_extract(x, "(?<=name\\'\\:\\s{1}\\').+(?=\\')")), st
ringsAsFactors = F)
Keywords_3=Keywords_2
for (i in 1:dim(Keywords_3)[2]) {
  Keywords_3[,i]=tolower(Keywords_2[,i]) %in% tolower(keywords_list$Keyword)
}
data$Popular_Keyword_Count=rowSums(Keywords_3)
```

```
#Distribution of Popular_Keyword_Count
table(data$Popular_Keyword_Count)
```

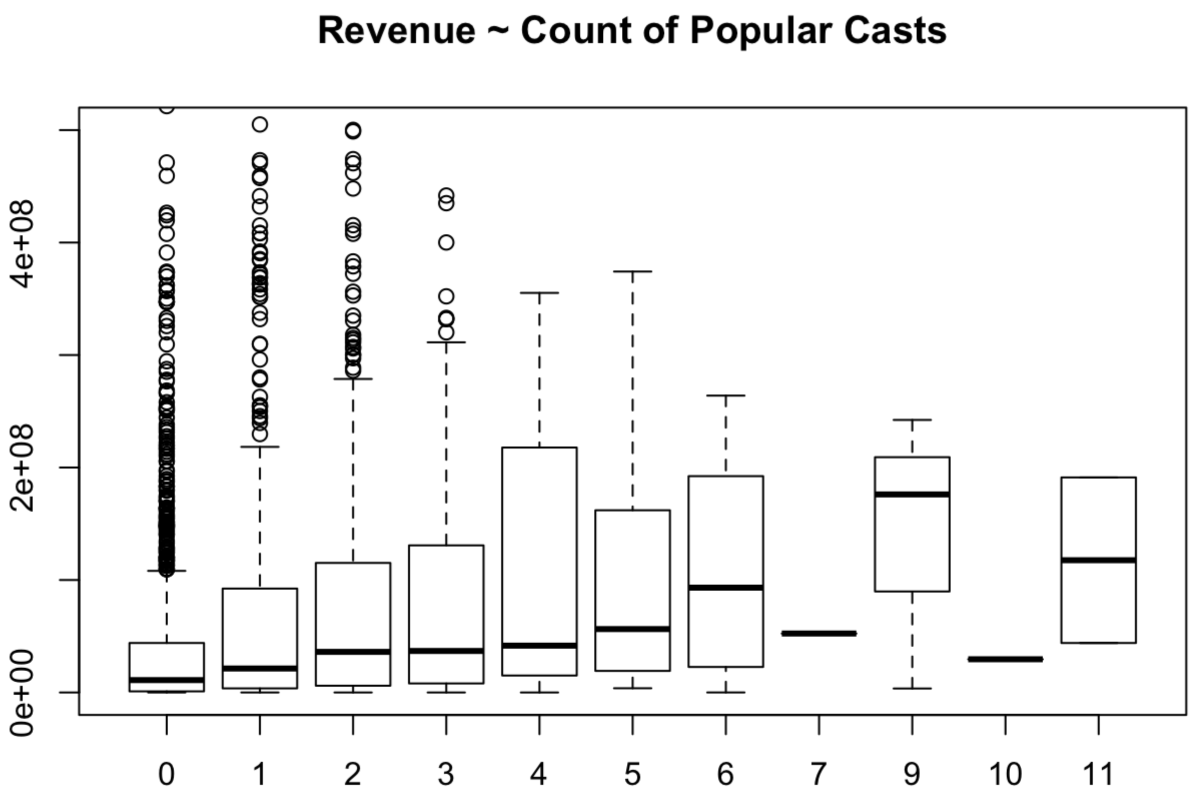
```
##
##    0    1    2    3    4    5    6    7    9   10   11
## 1395  782  440  142   61   17   10    1    3    1    2
```

```
boxplot(data$revenue~data$Popular_Keyword_Count,main="Revenue ~ Count of Popular Casts",ylim=c(0,500000000))
```

```
## Warning in x[floor(d)] + x[ceiling(d)]: NAs produced by integer overflow

## Warning in x[floor(d)] + x[ceiling(d)]: NAs produced by integer overflow

## Warning in x[floor(d)] + x[ceiling(d)]: NAs produced by integer overflow
```



There is an linear-like correlation between the variables, though the relationship becomes weak as the number of popular keywords rises.

Crew

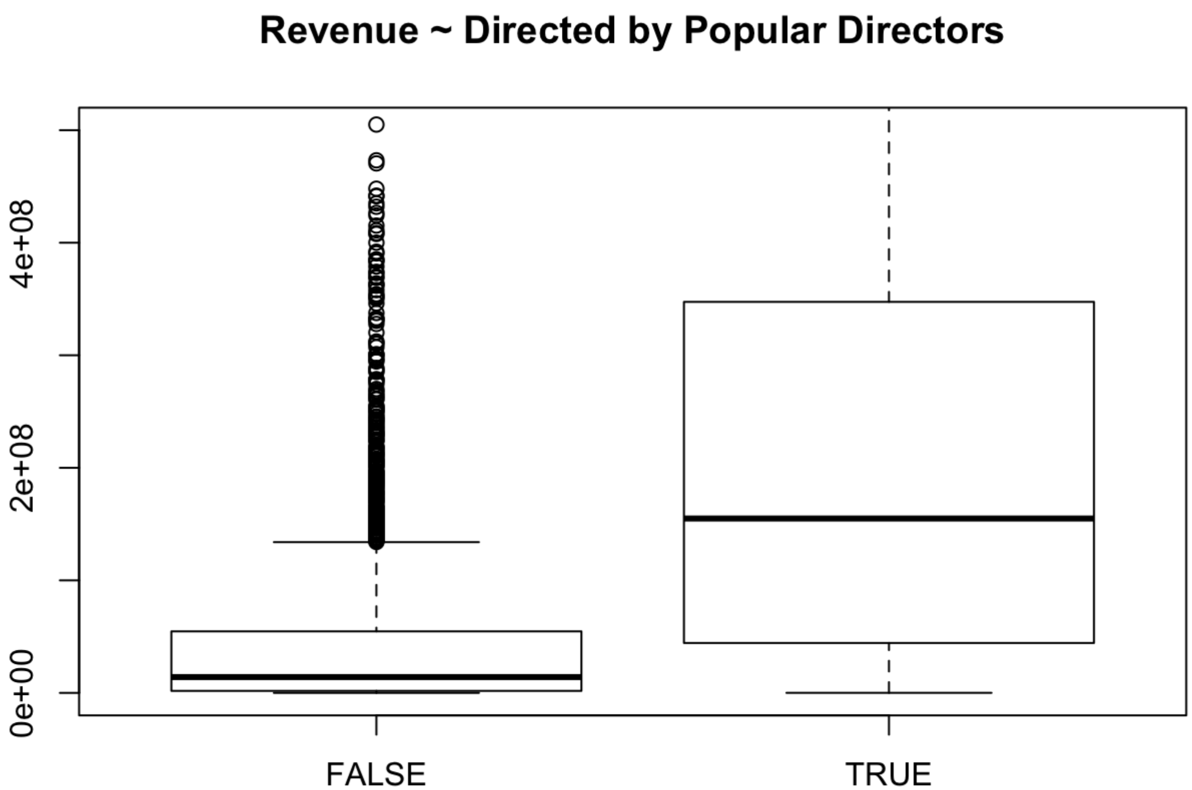
The “crew” variable contains hundreds of participates of each movie production. Since that is too much information to decipher, I only extracted the directors’ names, and mapped them to the top 100 average box office director list. The data source is https://www.the-numbers.com/box-office-star-records/worldwide/lifetime-specific-technical-role/director (https://www.the-numbers.com/box-office-star-records/worldwide/lifetime-specific-technical-role/director).

```
data$Director=str_extract_all(data$crew,"(?<=\\'Director\\', \\'name\\'\\:\\s{1}\\').+(?=\\')")
data$Director=gsub("'.*","",data$Director)
Popular_Director_List=c('Steven Spielberg','Joe Russo','Anthony Russo','Peter Jackson','Michael Bay','James Camer
on','David Yates','Christopher Nolan','Tim Burton','Robert Zemeckis','Ron Howard','Ridley Scott','Chris Columbus'
,'Roland Emmerich','Pierre Coffin','Bryan Singer','Gore Verbinski','James Wan','J.J. Abrams','George Lucas','Brad
Bird','Francis Lawrence','Sam Raimi','Clint Eastwood','Zack Snyder','Carlos Saldanha','M. Night Shyamalan','Bill
 Condon','Joss Whedon','Andrew Stanton','Chris Renaud','Tom McGrath','Sam Mendes','Jon Favreau','Andrew Adamson',
'John Lasseter','Eric Darnell','Barry Sonnenfeld','Shawn Levy','Justin Lin','Steven Soderbergh','Jon Turteltaub',
'Conrad Vernon','Kyle Balda','Brett Ratner','Martin Scorsese','Pete Docter','Rob Marshall','David Fincher','Tony
 Scott','Todd Phillips','Andy Wachowski','Rich Moore','Martin Campbell','Rob Minkoff','F. Gary Gray','Byron Howar
d','Richard Donner','Lee Unkrich','Raja Gosnell','Wolfgang Petersen','Ivan Reitman','Dennis Dugan','James Mangol
d','Mike Newell','Garry Marshall','Ron Clements','Jay Roach','Joe Johnston','John Musker','Mike Mitchell','Peyton
Reed','Christopher McQuarrie','James Gunn','Guy Ritchie','Joel Schumacher','Alfonso Cuarón','Colin Trevorrow','Ke
nneth Branagh','Dean DeBlois','Marc Forster','Rob Letterman','Quentin Tarantino','Robert Rodriguez','Gareth Edwar
ds','Peter Berg','Tom Shadyac','Paul Greengrass','Marc Webb','Ryan Coogler','Stephen Sommers','Ang Lee','Peter Fa
rrelly','Juan Antonio Bayona','Chris Weitz','Rian Johnson','Doug Liman','Kelly Asbury','Shane Black')
data$directed_by_famous_director=data$Director %in% Popular_Director_List
```

```
boxplot(data$revenue~data$directed_by_famous_director,main="Revenue ~ Directed by Popular Directors",ylim=c(0,500
000000))
```

```
## Warning in x[floor(d)] + x[ceiling(d)]: NAs produced by integer overflow

## Warning in x[floor(d)] + x[ceiling(d)]: NAs produced by integer overflow
```



Data Modeling

```
names(data)
```

```
##  [1] "belongs_to_collection"         "budget"
##  [3] "genres"                        "original_language"
##  [5] "popularity"                    "production_companies"
##  [7] "production_countries"          "release_date"
##  [9] "runtime"                       "spoken_languages"
## [11] "status"                        "Keywords"
## [13] "cast"                          "crew"
## [15] "revenue"                       "Collection_Name"
## [17] "Has_Collection"                "belongs_to_popular_collection"
## [19] "Genre_Count"                   "Action"
## [21] "Adventure"                     "Animation"
## [23] "Comedy"                        "Crime"
## [25] "Documentary"                   "Drama"
## [27] "Family"                        "Fantasy"
## [29] "Foreign"                       "History"
## [31] "Horror"                        "Music"
## [33] "Mystery"                       "Romance"
## [35] "Science_Fiction"               "Thriller"
## [37] "TV_Movie"                      "War"
## [39] "Western"                       "Company"
## [41] "belongs_to_popular_company"    "Cast_Count"
## [43] "Popular_Cast_Count"            "Country"
## [45] "belongs_to_country_list"       "Keyword_Count"
## [47] "Popular_Keyword_Count"         "Director"
## [49] "directed_by_famous_director"
```

Establish the training dataset by extracting the meaning variables from our original data, most of which are those I created.

```
train=data[,c(2,5,8,9,15,17,18:39,41:43,45:47,49)]
train[which(is.na(train$runtime)),"runtime"]=median(train$runtime,na.rm = T)
names(train)
```

```
##  [1] "budget"                        "popularity"
##  [3] "release_date"                  "runtime"
##  [5] "revenue"                       "Has_Collection"
##  [7] "belongs_to_popular_collection" "Genre_Count"
##  [9] "Action"                        "Adventure"
## [11] "Animation"                     "Comedy"
## [13] "Crime"                         "Documentary"
## [15] "Drama"                         "Family"
## [17] "Fantasy"                       "Foreign"
## [19] "History"                       "Horror"
## [21] "Music"                         "Mystery"
## [23] "Romance"                       "Science_Fiction"
## [25] "Thriller"                      "TV_Movie"
## [27] "War"                           "Western"
## [29] "belongs_to_popular_company"    "Cast_Count"
## [31] "Popular_Cast_Count"            "belongs_to_country_list"
## [33] "Keyword_Count"                 "Popular_Keyword_Count"
## [35] "directed_by_famous_director"
```

```
summary(train)
```

```
##      budget            popularity       release_date
##  Min.   :         0   Min.   :  0.000   Min.   :1969-01-01
##  1st Qu.:         0   1st Qu.:  4.126   1st Qu.:1995-03-31
##  Median :  9000000   Median :  7.423   Median :2005-05-15
##  Mean   : 23537644   Mean   :  8.560   Mean   :2002-08-27
##  3rd Qu.: 30000000   3rd Qu.: 10.936   3rd Qu.:2011-09-23
##  Max.   :380000000   Max.   :294.337   Max.   :2017-07-20
##     runtime          revenue           Has_Collection
##  Min.   :  0.0   Min.   :1.000e+00   Mode :logical
##  1st Qu.: 94.0   1st Qu.:2.437e+06   FALSE:2288
##  Median :104.0   Median :1.788e+07   TRUE :566
##  Mean   :107.4   Mean   :6.934e+07
##  3rd Qu.:117.0   3rd Qu.:7.323e+07
##  Max.   :338.0   Max.   :1.520e+09
##  belongs_to_popular_collection  Genre_Count      Action
##  Mode :logical                 Min.   :0.000   Mode :logical
##  FALSE:2733                    1st Qu.:2.000   FALSE:2140
##  TRUE :121                     Median :2.000   TRUE :714
##                                Mean   :2.505
##                                3rd Qu.:3.000
##                                Max.   :7.000
##  Adventure       Animation         Comedy           Crime
##  Mode :logical   Mode :logical   Mode :logical   Mode :logical
##  FALSE:2437      FALSE:2715      FALSE:1857      FALSE:2403
##  TRUE :417       TRUE :139       TRUE :997       TRUE :451
##
##
##
##  Documentary       Drama           Family          Fantasy
##  Mode :logical   Mode :logical   Mode :logical   Mode :logical
##  FALSE:2767      FALSE:1423      FALSE:2601      FALSE:2627
##  TRUE :87        TRUE :1431      TRUE :253       TRUE :227
##
##
##
##   Foreign         History          Horror           Music
##  Mode :logical   Mode :logical   Mode :logical   Mode :logical
##  FALSE:2823      FALSE:2738      FALSE:2565      FALSE:2765
##  TRUE :31        TRUE :116       TRUE :289       TRUE :89
##
```

```
##
##
##   Mystery          Romance          Science_Fiction  Thriller
##  Mode :logical   Mode :logical   Mode :logical   Mode :logical
##  FALSE:2642      FALSE:2327      FALSE:2854      FALSE:2084
##  TRUE :212       TRUE :527                       TRUE :770
##
##
##
##   TV_Movie          War            Western
##  Mode :logical   Mode :logical   Mode :logical
##  FALSE:2854      FALSE:2769      FALSE:2826
##                  TRUE :85        TRUE :28
##
##
##
##  belongs_to_popular_company   Cast_Count     Popular_Cast_Count
##  Mode :logical               Min.   :  0.00   Min.   : 0.000
##  FALSE:1939                  1st Qu.: 11.00   1st Qu.: 0.000
##  TRUE :915                   Median : 16.00   Median : 2.000
##                              Mean   : 20.39   Mean   : 2.445
##                              3rd Qu.: 24.00   3rd Qu.: 4.000
##                              Max.   :156.00   Max.   :42.000
##  belongs_to_country_list Keyword_Count     Popular_Keyword_Count
##  Mode :logical           Min.   :  0.000   Min.   : 0.000
##  FALSE:183               1st Qu.:  3.000   1st Qu.: 0.000
##  TRUE :2671              Median :  6.000   Median : 1.000
##                          Mean   :  7.141   Mean   : 0.891
##                          3rd Qu.: 10.000   3rd Qu.: 1.000
##                          Max.   :149.000   Max.   :11.000
##  directed_by_famous_director
##  Mode :logical
##  FALSE:2549
##  TRUE :305
##
##
##
```

The data looks nice and clean with no missing value(I imputed 2 NAs of “runtime”).

I firstly fitted a random forest model to the data.

```
library(randomForest)
```

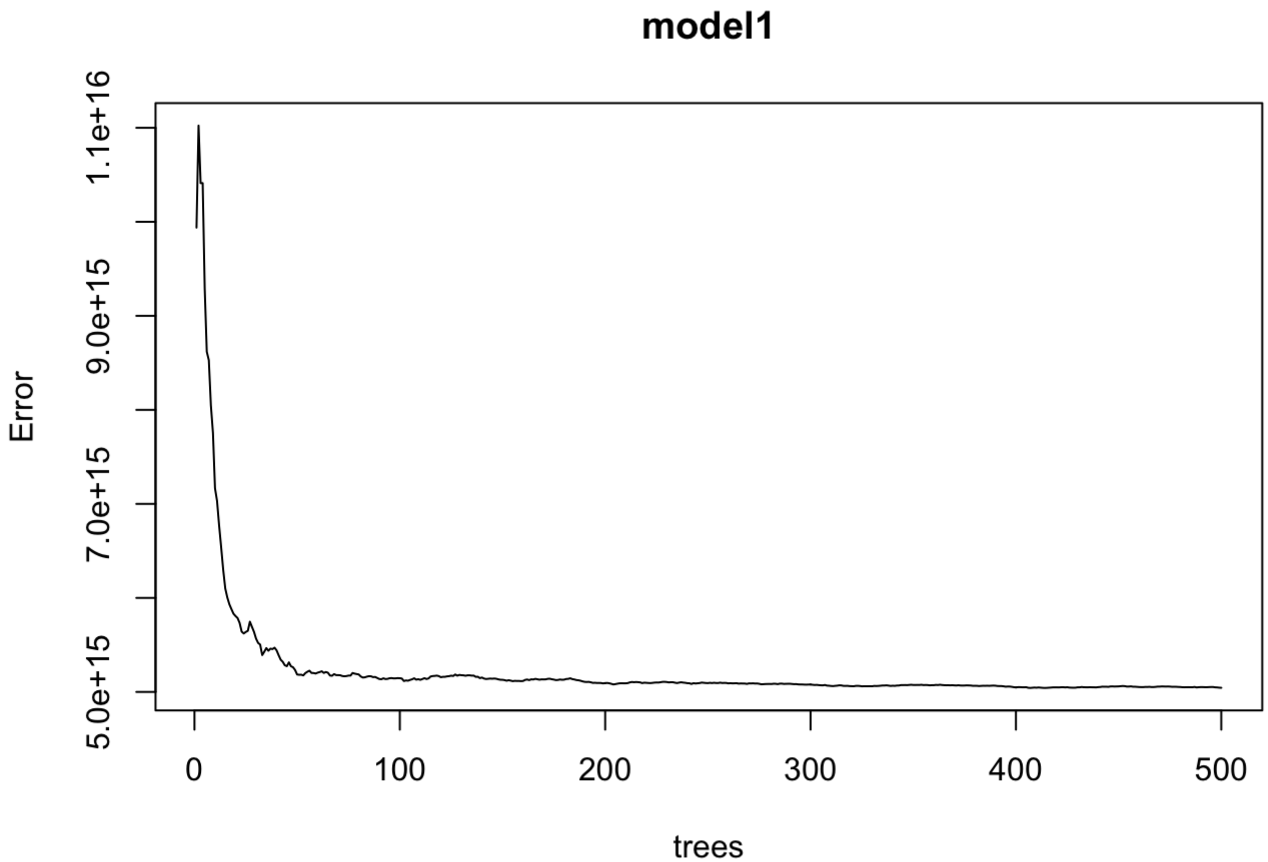
```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
model1=randomForest(formula=revenue~.,data=train,na.action = na.omit)
plot(model1)
```

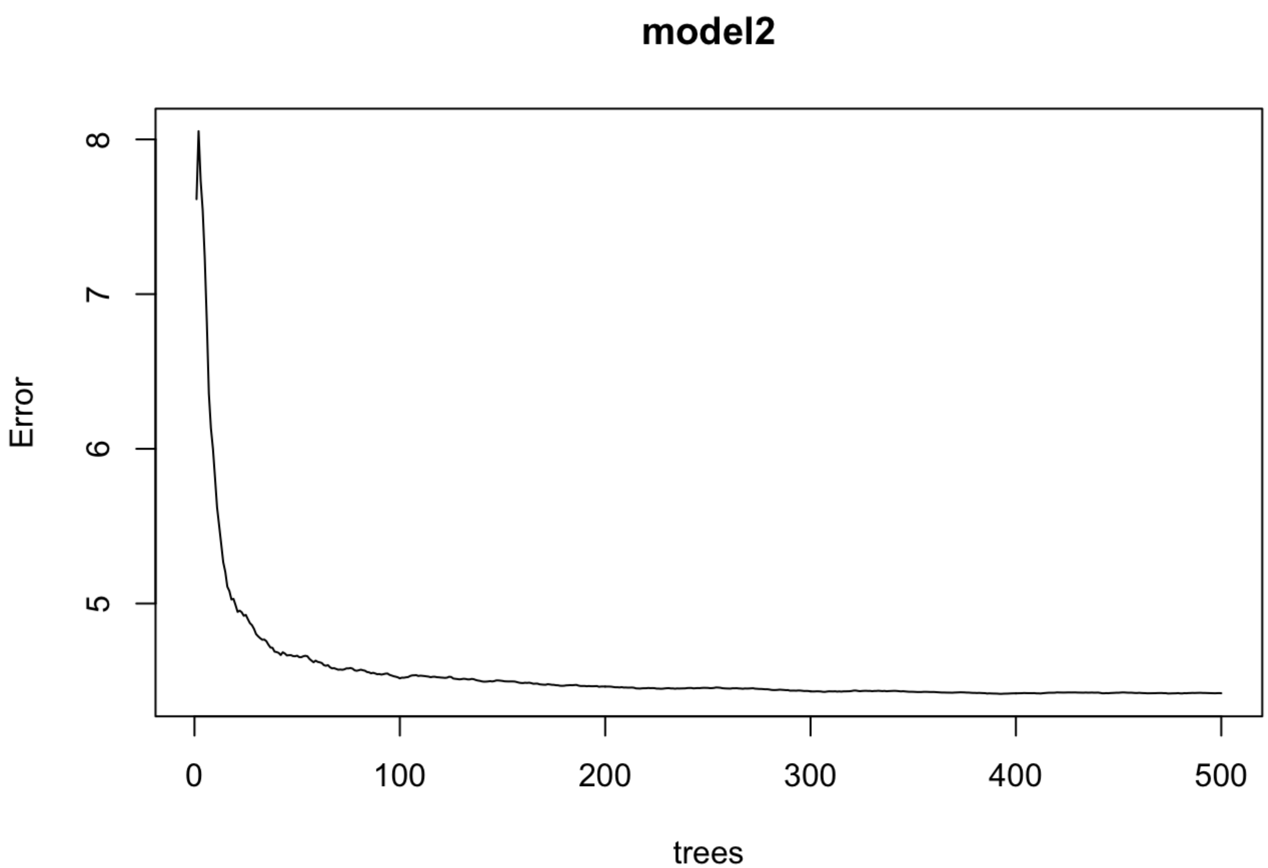



```
model1
```

```
##
## Call:
##  randomForest(formula = revenue ~ ., data = train, na.action = na.omit)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 11
##
##           Mean of squared residuals: 5.04386e+15
##                     % Var explained: 74.4
```

The RMSE value is very large considering revenue can be easily of million or billion units. However, the model is able to explain 74.38% of variance, which is quite remarkable given the vagueness of our raw data. To remediate the effect of large revenue numbers, I used log and scale functions:

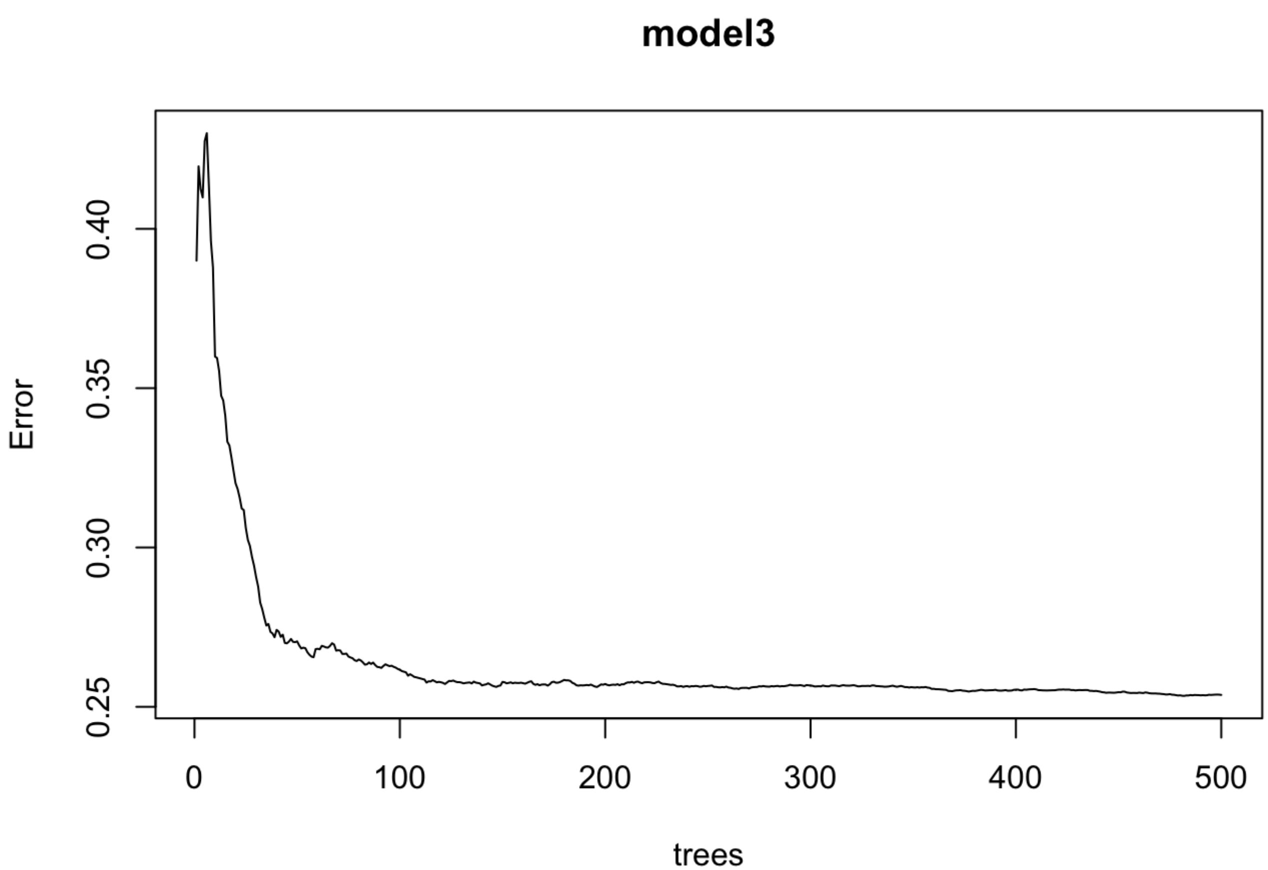
```
model2=randomForest(formula=log(revenue)~.,data=train,na.action = na.omit)
plot(model2)
```




```
model2
```

```
##
## Call:
##  randomForest(formula = log(revenue) ~ ., data = train, na.action = na.omit)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 11
##
##           Mean of squared residuals: 4.420112
##                     % Var explained: 52.84
```

```
model3=randomForest(formula=scale(revenue)~.,data=train,na.action = na.omit)
plot(model3)
```



```
model3
```

```
##
## Call:
##  randomForest(formula = scale(revenue) ~ ., data = train, na.action = na.omit)
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 11
##
##           Mean of squared residuals: 0.2537349
##                     % Var explained: 74.62
```

Secondly, I fitted xgboost model to the data. However, the model tends to produce extreme values like negative revenue, which do not make sense.

```
library("xgboost")
```

```
## Warning: package 'xgboost' was built under R version 3.5.2
```

```
##
## Attaching package: 'xgboost'
```

```
## The following object is masked from 'package:dplyr':
##
##     slice
```

```
model3=xgboost(data=as.matrix(train[,-3]),label = train$revenue,nrounds = 100,objective="reg:linear")
```

```
## [1]  train-rmse:111996712.000000
## [2]  train-rmse:80387120.000000
## [3]  train-rmse:57908224.000000
## [4]  train-rmse:41861644.000000
## [5]  train-rmse:30397740.000000
## [6]  train-rmse:22209590.000000
## [7]  train-rmse:16384172.000000
## [8]  train-rmse:12181664.000000
## [9]  train-rmse:9162117.000000
## [10] train-rmse:6995142.500000
## [11] train-rmse:5423594.500000
## [12] train-rmse:4268829.000000
## [13] train-rmse:3400142.500000
## [14] train-rmse:2732303.250000
## [15] train-rmse:2251175.500000
## [16] train-rmse:1884262.000000
## [17] train-rmse:1601797.500000
## [18] train-rmse:1391443.750000
## [19] train-rmse:1237129.250000
## [20] train-rmse:1108835.875000
## [21] train-rmse:1012287.187500
## [22] train-rmse:945977.625000
## [23] train-rmse:877904.562500
## [24] train-rmse:834511.062500
## [25] train-rmse:804228.937500
## [26] train-rmse:770948.312500
## [27] train-rmse:735779.625000
## [28] train-rmse:705297.125000
## [29] train-rmse:684071.875000
## [30] train-rmse:659524.562500
## [31] train-rmse:654730.875000
## [32] train-rmse:634664.937500
## [33] train-rmse:614138.437500
## [34] train-rmse:611459.250000
## [35] train-rmse:593985.625000
## [36] train-rmse:589394.312500
## [37] train-rmse:571905.062500
## [38] train-rmse:559407.250000
## [39] train-rmse:551991.500000
## [40] train-rmse:548368.562500
```

```
## [41] train-rmse:538852.437500
## [42] train-rmse:513772.687500
## [43] train-rmse:496044.187500
## [44] train-rmse:489171.093750
## [45] train-rmse:479433.281250
## [46] train-rmse:454393.687500
## [47] train-rmse:438737.625000
## [48] train-rmse:422319.906250
## [49] train-rmse:406897.187500
## [50] train-rmse:399405.468750
## [51] train-rmse:393420.593750
## [52] train-rmse:379240.312500
## [53] train-rmse:362619.625000
## [54] train-rmse:355753.062500
## [55] train-rmse:353212.718750
## [56] train-rmse:345348.718750
## [57] train-rmse:341790.906250
## [58] train-rmse:338605.718750
## [59] train-rmse:327806.593750
## [60] train-rmse:318311.375000
## [61] train-rmse:316548.937500
## [62] train-rmse:304078.687500
## [63] train-rmse:302953.968750
## [64] train-rmse:297694.875000
## [65] train-rmse:285633.625000
## [66] train-rmse:280748.656250
## [67] train-rmse:275624.062500
## [68] train-rmse:272645.062500
## [69] train-rmse:267691.875000
## [70] train-rmse:261845.500000
## [71] train-rmse:253587.984375
## [72] train-rmse:247581.890625
## [73] train-rmse:243145.703125
## [74] train-rmse:242567.500000
## [75] train-rmse:237434.156250
## [76] train-rmse:233064.765625
## [77] train-rmse:230321.484375
## [78] train-rmse:227323.359375
## [79] train-rmse:224429.750000
## [80] train-rmse:222002.843750
## [81] train-rmse:215925.375000
```

```
## [82] train-rmse:208659.343750
## [83] train-rmse:205958.812500
## [84] train-rmse:201261.671875
## [85] train-rmse:199373.078125
## [86] train-rmse:198028.625000
## [87] train-rmse:196836.250000
## [88] train-rmse:192568.531250
## [89] train-rmse:188482.656250
## [90] train-rmse:185736.843750
## [91] train-rmse:180551.453125
## [92] train-rmse:178117.906250
## [93] train-rmse:171378.531250
## [94] train-rmse:169236.937500
## [95] train-rmse:166143.703125
## [96] train-rmse:162379.265625
## [97] train-rmse:157278.187500
## [98] train-rmse:156174.343750
## [99] train-rmse:155776.750000
## [100]    train-rmse:153762.953125
```

Summary

While working on this project, I performed a lot of data cleaning and feature engineering. Most of the variables I used for modeling are those I created with the help of external data. Therefore, I gained the understanding that these messy and tedious tasks are often the essential parts of data mining. I also learned a lot about the film industry, including its revenue indicators and marketing strategies.

As the ultimate result, I built a random forest model that explains 74.38% of the data variance with RMSE of 0.256(scaled data).

Areas that can be improved

1. There could be better use of external data when I map it to the raw data. For instance, I could have weigh some features like casts and directors according to the average box office revenue they are affliated to, instead of just setting boolean value of whether they are "popular". In this way, I can transform some boolean values into numeric ones that indicate more linear correlations with revenue.
2. I only tried random forest and xgboost when fitting model, while there are a lot of other machine learning methods. Also, I should spend more time training the model and optimize the parameters.
3. As I mentioned at the begining, some advance NPL methods may be adopted to analyze movies' titles, overviews and taglines. However, since I am not an expert in the realm(currently), this part will be saved for future exploration.