Project 1: Explore and Prepare Data

Code

(Student GT acount name: xtao41) CSE6242 - Data and Visual Analytics - Spring 2017 Due: Sunday, March 5, 2017 at 11:59 PM UTC-12:00 on T-Square

Note: This project involves getting data ready for analysis and doing some preliminary investigations. Project 2 will involve modeling and predictions, and will be released at a later date. Both projects will have equal weightage towards your grade.

Data

In this project, you will explore a dataset that contains information about movies, including ratings, budget, gross revenue and other attributes. It was prepared by Dr. Guy Lebanon, and here is his description of the dataset:

> The file `movies_merged` (https://s3.amazonaws.com/content.udacity-data.com/courses/gt-cs6242/project/movies_merged) contains a dataframe with the same name that has 40K rows and 39 columns. Each row represents a movie title and each column represents a descriptor such as `Title`, `Actors`, and `Budget`. I collected the data by querying IMDbâs API (see www.omdbapi.com (http://www.omdbapi.com/)) and joining it with a separate dataset of movie budgets and gross earnings (unknown to you). The join key was the movie title. This data is available for personal use, but IMDbâs terms of service do not allow it to be used for commercial purposes or for creating a competing repository.

Objective

Your goal is to investigate the relationship between the movie descriptors and the box office success of movies, as represented by the variable `Gross`. This task is extremely important as it can help a studio decide which titles to fund for production, how much to bid on produced movies, when to release a title, how much to invest in marketing and PR, etc. This information is most useful before a title is released, but it is still very valuable after the movie is already released to the public (for example it can affect additional marketing spend or how much a studio should negotiate with on-demand streaming companies for âsecond windowâ streaming rights).

Instructions

This is an R Markdown (http://rmarkdown.rstudio.com) Notebook. Open this file in RStudio to get started.

When you execute code within the notebook, the results appear beneath the code. Try executing this chunk by clicking the *Run* button within the chunk or by placing your cursor inside it and pressing *Cmd+Shift+Enter*.

Hide

```
x = 1:10
print(x^2)
```

```
 [1]   1   4   9  16  25  36  49  64  81 100
```

Plots appear inline too:

Hide

```
plot(x, x^2, 'o')
```

Add a new chunk by clicking the *Insert Chunk* button on the toolbar or by pressing *Cmd+Option+I*.

When you save the notebook, an HTML file containing the code and output will be saved alongside it (click the *Preview* button or press *Cmd+Shift+K* to preview the HTML file).

Please complete the tasks below and submit this R Markdown file (as **pr1.Rmd**) as well as a PDF export of it (as **pr1.pdf**). Both should contain all the code, output, plots and written responses for each task.

Setup

Load data

Make sure you've downloaded the `movies_merged` (https://s3.amazonaws.com/content.udacity-data.com/courses/gt-cs6242/project/movies_merged) file and it is in the current working directory. Now load it into memory:

Hide

```
load('movies_merged')
```

This creates an object of the same name (`movies_merged`). For convenience, you can copy it to `df` and start using it:

Hide

```
df = movies_merged
cat("Dataset has", dim(df)[1], "rows and", dim(df)[2], "columns", end="\n", fil
e="")
```

```
Dataset has 40789 rows and 39 columns
```

Hide

```
colnames(df)
```

```
 [1] "Title"            "Year"             "Rated"            "Released"
"Runtime"
 [6] "Genre"             "Director"         "Writer"           "Actor
s"            "Plot"
[11] "Language"          "Country"          "Awards"           "Poste
r"            "Metascore"
[16] "imdbRating"        "imdbVotes"        "imdbID"           "Typ
e"              "tomatoMeter"
[21] "tomatoImage"       "tomatoRating"     "tomatoReviews"    "tomatoFres
h"       "tomatoRotten"
[26] "tomatoConsensus"   "tomatoUserMeter"  "tomatoUserRating" "tomatoUserRev
iews" "tomatoURL"
[31] "DVD"               "BoxOffice"        "Production"       "Websit
e"           "Response"
[36] "Budget"            "Domestic_Gross"   "Gross"            "Dat
e"
```

Load R packages

Load any R packages that you will need to use. You can come back to this chunk, edit it and re-run to load any additional packages later.

Hide

```
library(ggplot2)
library(GGally)
```

If you are loading any non-standard packages (ones that have not been discussed in class or explicitly allowed for this project), please mention them below. Include any special instructions if they cannot be installed using the regular `install.packages('<pkg name>')` command.

Non-standard packages used: None

Tasks

Each task below is worth **10** points, and is meant to be performed sequentially, i.e. do step 2 after you have processed the data as described in step 1. Total points: **100**

Complete each task by implementing code chunks as described by `TODO` comments, and by responding to questions ("**Q**:") with written answers ("**A**:"). If you are unable to find a meaningful or strong relationship in any of the cases when requested, explain why not by referring to appropriate plots/statistics.

It is OK to handle missing values below by omission, but please omit as little as possible. It is worthwhile to invest in reusable and clear code as you may need to use it or modify it in project 2.

1. Remove non-movie rows

The variable `Type` captures whether the row is a movie, a TV series, or a game. Remove all rows from `df` that do not correspond to movies.

Hide

```
# TODO: Remove all rows from df that do not correspond to movies
df<-df[(df$Type=="movie"),]
```

Q: How many rows are left after removal? *Enter your response below.*

A: 40,000

2. Process `Runtime` column

The variable `Runtime` represents the length of the title as a string. Write R code to convert it to a numeric value (in minutes) and replace `df$Runtime` with the new numeric column.

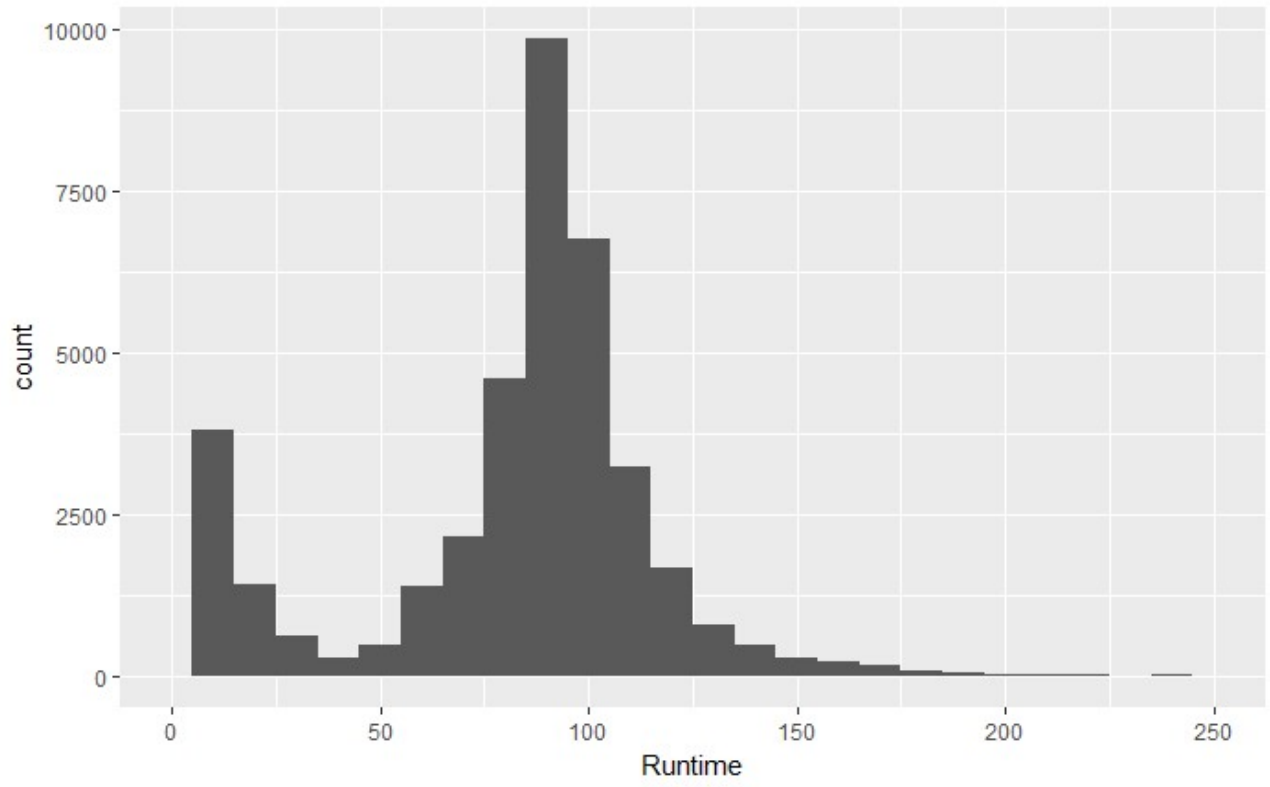
Hide

```
# TODO: Replace df$Runtime with a numeric column containing the runtime in minu
tes
convert_time<-function(runtime){
  if(is.na(runtime)){return(NA)}
  if(!grepl("h",runtime)){
    n_runtime<-as.numeric(runtime)
  }
  else{
   a=(strsplit(runtime, " h "))[[1]][1]
   b=(strsplit(runtime, " h "))[[1]][2]
   a<-as.numeric(a)
   b<-as.numeric(b)
   n_runtime<-a*60+b
  }
    return(n_runtime)
}
df$Runtime<-gsub("min","", df$Runtime)
df$Runtime<-sapply(df$Runtime,function(x) convert_time(x))
```

Now investigate the distribution of `Runtime` values and how it changes over years (variable `Year`, which you can bucket into decades) and in relation to the budget (variable `Budget`). Include any plots that illustrate.

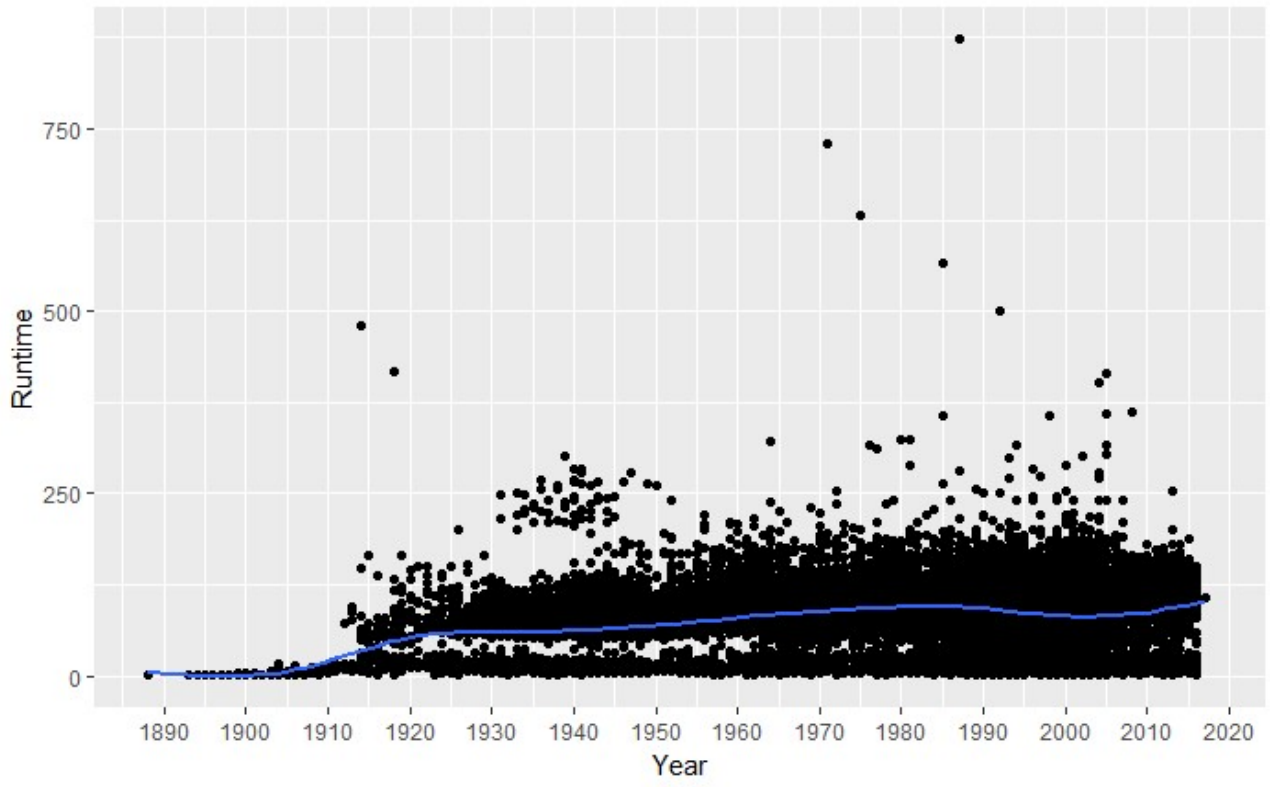
Hide

```
# TODO: Investigate the distribution of Runtime values and how it varies by Yea
r and Budget
#distribution of 'Runtime'values
ggplot(df, aes(Runtime)) + geom_histogram(binwidth = 10)+xlim(c(0,250))
```



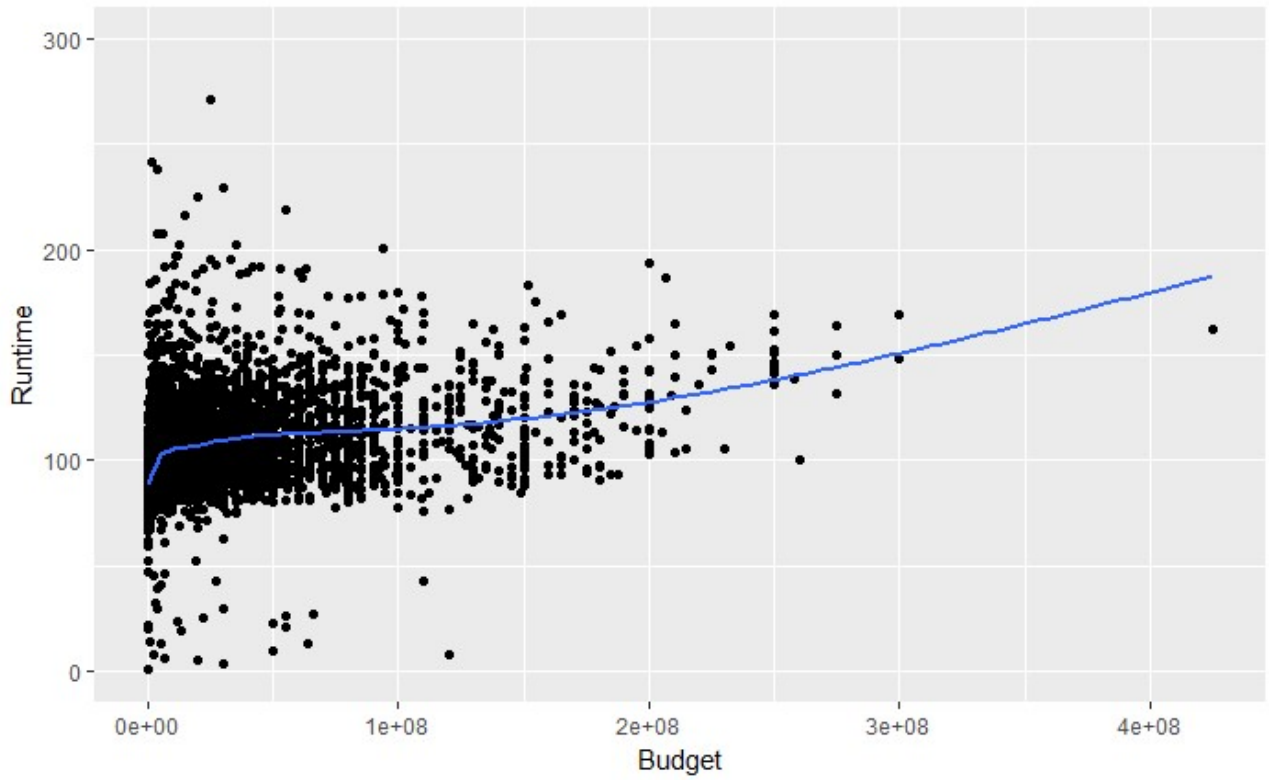
Hide

```
# by Year bucket into decdes
library(ggplot2)
ggplot(df, aes(x=Year, y=Runtime)) +
geom_point()+  scale_x_continuous(breaks = round(seq(min(df$Year), max(df$Yea
r), 10),digits = -1))+ stat_smooth(se=FALSE)
```



Hide

```
# by buget
ggplot(df, aes(x=Budget, y=Runtime)) + geom_point() +scale_y_continuous(limits=
c(0, 300))+ stat_smooth(se=FALSE)
```



Feel free to insert additional code chunks as necessary.

Q: Comment on the distribution as well as relationships. Are there any patterns or trends that you can observe?

A: 1) Distribution of Runtime: the Runtime of largest number of movies are around 90-100 minutes.There are also a few movies that have short runtime (about 10 to 20 minutes). None or few movies have Runtime above 250 miniutes.

2. Runtime vs Year: from 1890 to 1900 the Runtime did not change much.However, this is a sharp increase from 1905 to 1920. Afterwards, it increases at a slower rate till 1980. From 1980 to 2016, runtime stoped increases and became more steady. There was also a sharp incrase of the variation of runtime in 1913, which increases overally till around 2000.

3. Runtime vs.Budget: There was a steep increase of Runtime as budget increases when budget is below 20,000,000. When budget is higher than that, runtime increases at a much slower rate with increasing budget. Most movies have budgets below 1e08.

3. Encode `Genre` column

The column `Genre` represents a list of genres associated with the movie in a string format. Write code to parse each text string into a binary vector with 1s representing the presence of a genre and 0s the absence, and add it to the dataframe as additional columns. Then remove the original `Genre` column.

For example, if there are a total of 3 genres: Drama, Comedy, and Action, a movie that is both Action and Comedy should be represented by a binary vector <0, 1, 1>. Note that you need to first compile a dictionary of all possible genres and then figure out which movie has which genres (you can use the R `tm` package to create the dictionary).

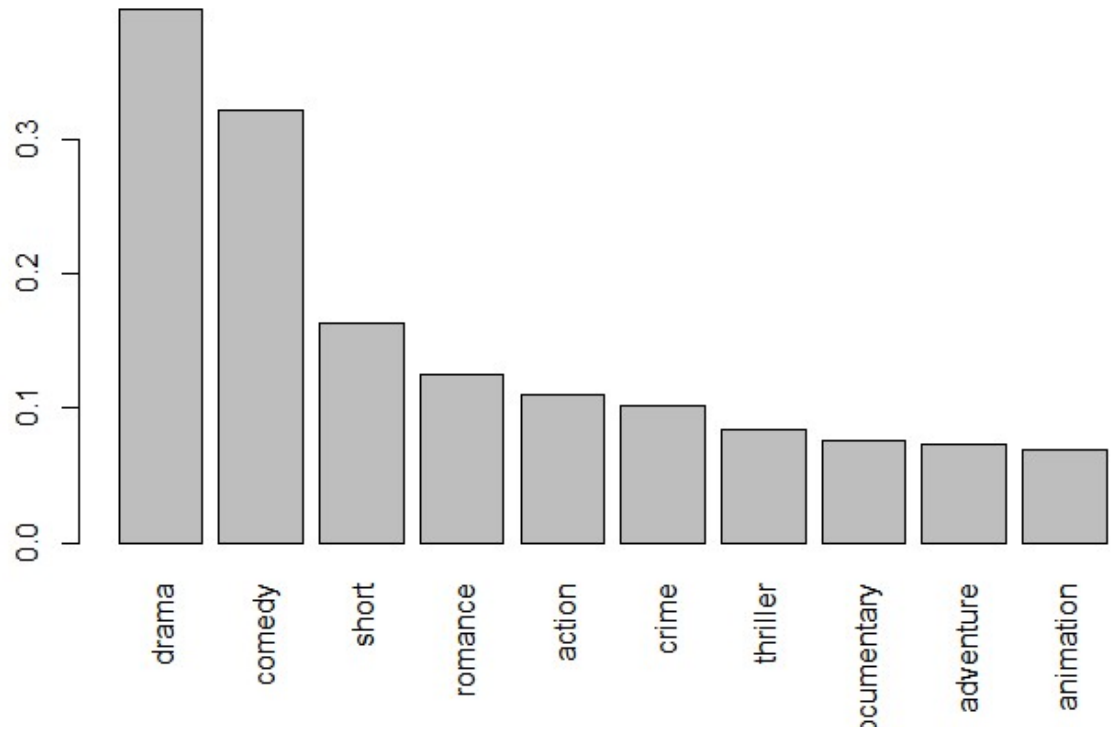
Hide

```
# TODO: Replace Genre with a collection of binary columns
# remove all comma in Genre
#install.packages("tm")
df$Genre<-gsub(","," ", df$Genre)
Col_genre<-c(df$Genre)
library(tm)
Corpus_genre <- Corpus(VectorSource(Col_genre))
TDM_genre<-TermDocumentMatrix(Corpus_genre, control = list(minWordLength = 1))
Gen_cols<- data.frame(t(as.matrix(TDM_genre)))
df<-cbind(df,Gen_cols)
df$Genre<-NULL
```

Plot the relative proportions of movies having the top 10 most common genres.

Hide

```
# TODO: Select movies from top 10 most common genres and plot their relative pr
oportions
top=-sort(-colSums(Gen_cols)/nrow(Gen_cols))[0:10]
barplot(top,las=3)
```

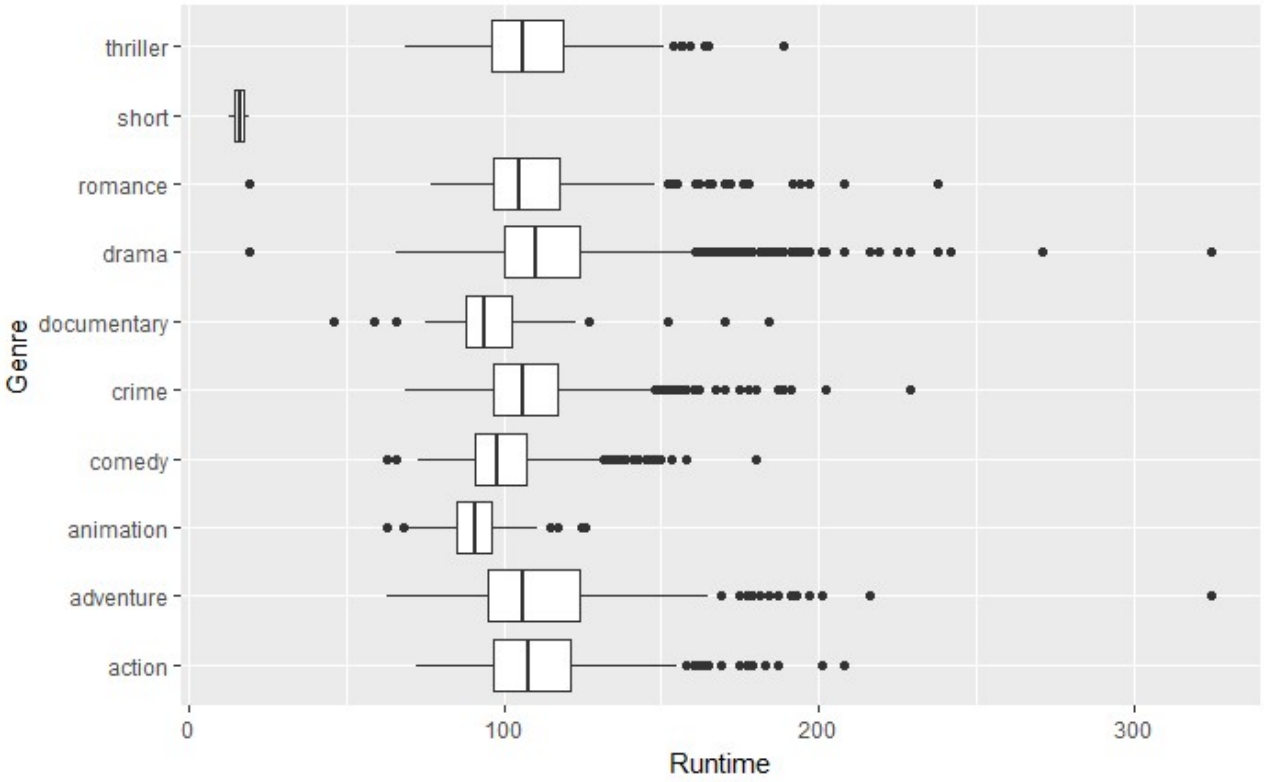


Examine how the distribution of `Runtime` changes across genres for the top 10 most common genres.

Hide

```
# TODO: Plot Runtime distribution for top 10 most common genres
top_genres=c('comedy','short','romance','action','crime','thriller','documentar
y','adventure','animation')
temp_df = df[df$drama==1, ]
temp_df$Genre<-'drama'
df_gen<-temp_df

for (i in c(1:9)){
  temp_df<-df[df[,top_genres[i]]==1,]
  temp_df$Genre<-top_genres[i]
  df_gen<-rbind(df_gen,temp_df)
}
ggplot(data = na.omit(df_gen), aes(Genre, Runtime)) + geom_boxplot()+coord_flip
()
```



Q: Describe the interesting relationship(s) you observe. Are there any expected or unexpected trends that are evident?

A: 1) Among the top 10 most common genres, drama has the largest relative proportion and comedy has the second largest proportion. Though animation has the lowest proportion, it does not different much from the proportion of adventure and documentary. 2) the boxplot of Runtime across genres show the runtime of short is much shorter than others. Among the rest, animation and documentary are

the next shortest movie types. Though drama has the largest median of runtime, it does not differ much from action, adventure and crime. It is worth to notice that the spread of drama and adventure is also high compared to other movie types.

4. Eliminate mismatched rows

The dataframe was put together by merging two different sources of data and it is possible that the merging process was inaccurate in some cases (the merge was done based on movie title, but there are cases of different movies with the same title). The first sourceâs release time was represented by the column `Year` (numeric representation of the year) and the second by the column `Released` (string representation of release date).

Find and remove all rows where you suspect a merge error occurred based on a mismatch between these two variables. To make sure subsequent analysis and modeling work well, avoid removing more than 10% of the rows that have a `Gross` value present.

Hide

```
# TODO: Remove rows with Released-Year mismatch
df_match_year<-df
df_match_year$Released<-as.numeric(format(df$Released,'%Y'))
df_match_year<-subset(df_match_year,is.na(df_match_year$Released)| (df_match_ye
ar$Released>=df_match_year$Year))
```

Q: What is your precise removal logic and how many rows did you end up removing?

A: 19 rows were removed (less than 10% of the rows that have a ‘Gross’ value).The released date is the last time that the movie was released, so it should be equal or later than the “Year”. The discrepancies between them were likely because of movies released in different times in different region. So removal was performed for columns with “Released” less than “Year”, which is considered a mismatch.

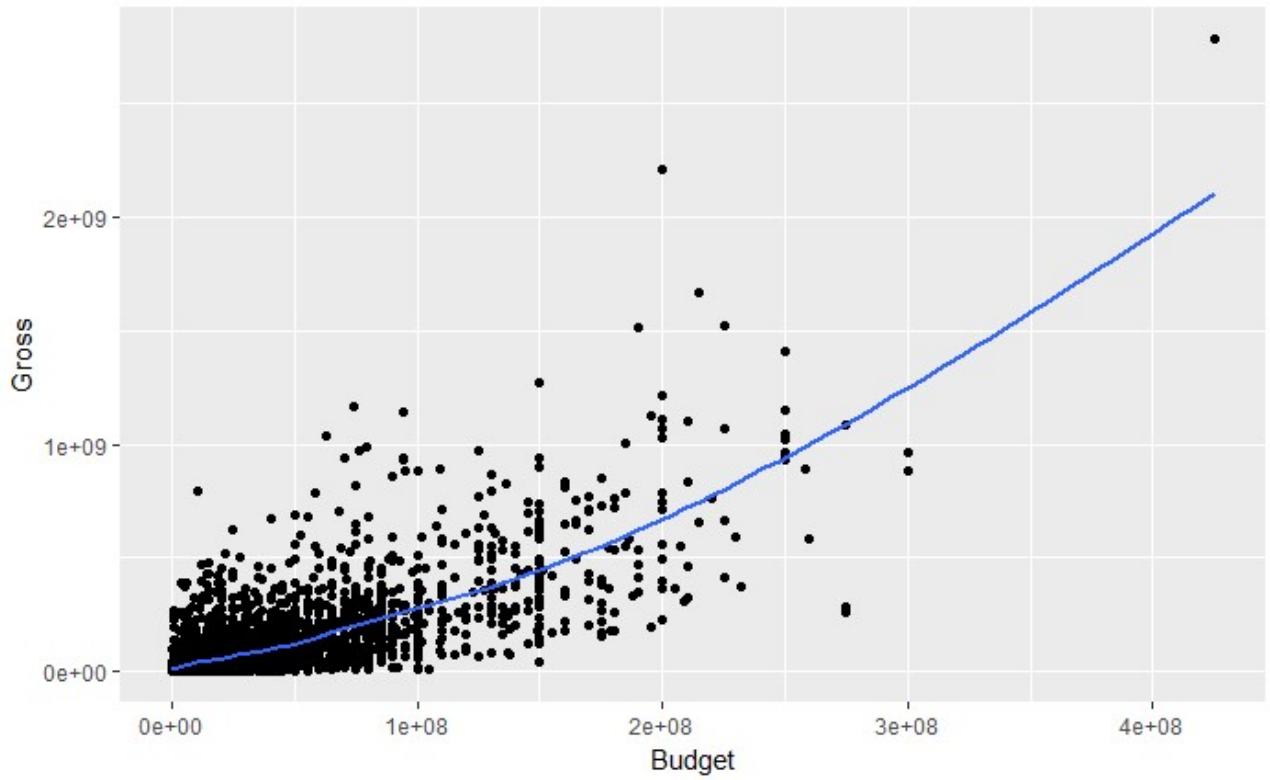
5. Explore `Gross` revenue

For the commercial success of a movie, production houses want to maximize Gross revenue. Investigate if Gross revenue is related to Budget, Runtime or Genre in any way.

Note: To get a meaningful relationship, you may have to partition the movies into subsets such as short vs. long duration, or by genre, etc.

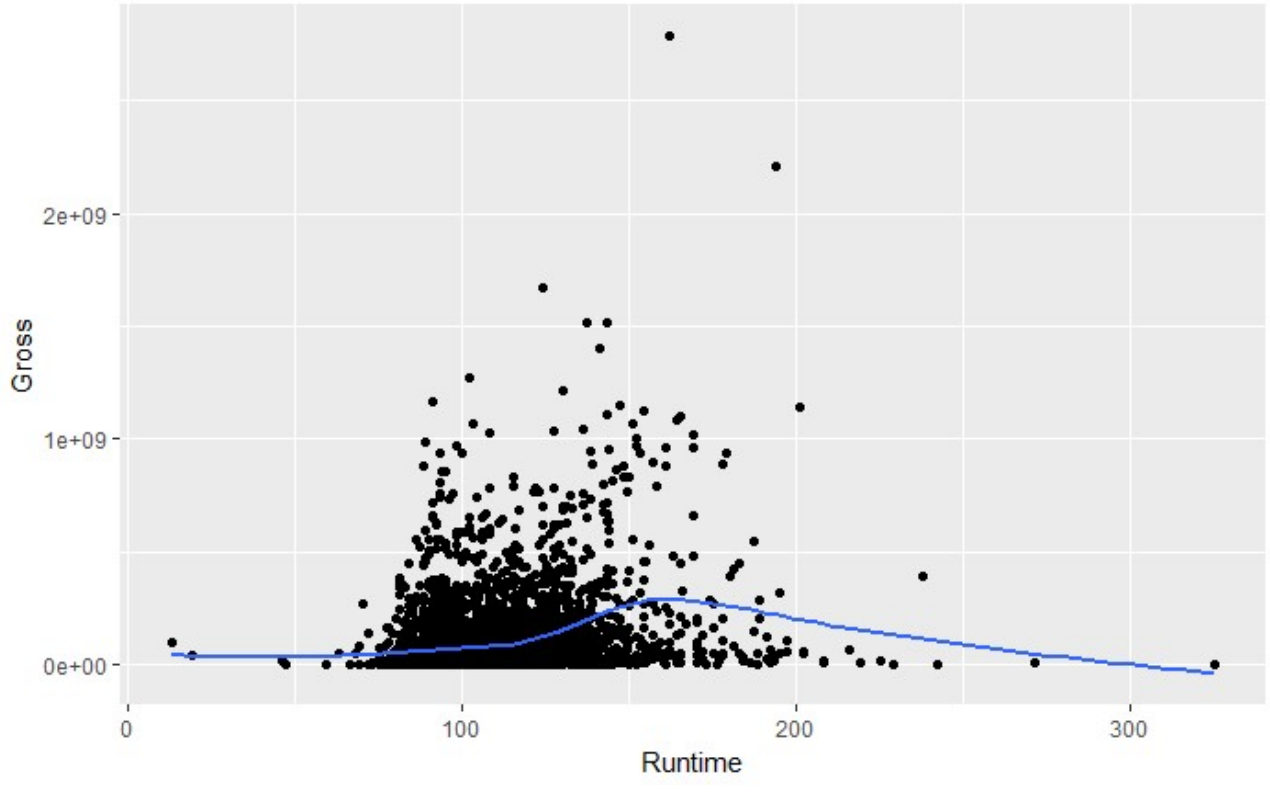
Hide

```
# TODO: Investigate if Gross Revenue is related to Budget, Runtime or Genre
ggplot(na.omit(df_match_year), aes(Budget, Gross)) + geom_point()+ stat_smooth
(se=FALSE)
```



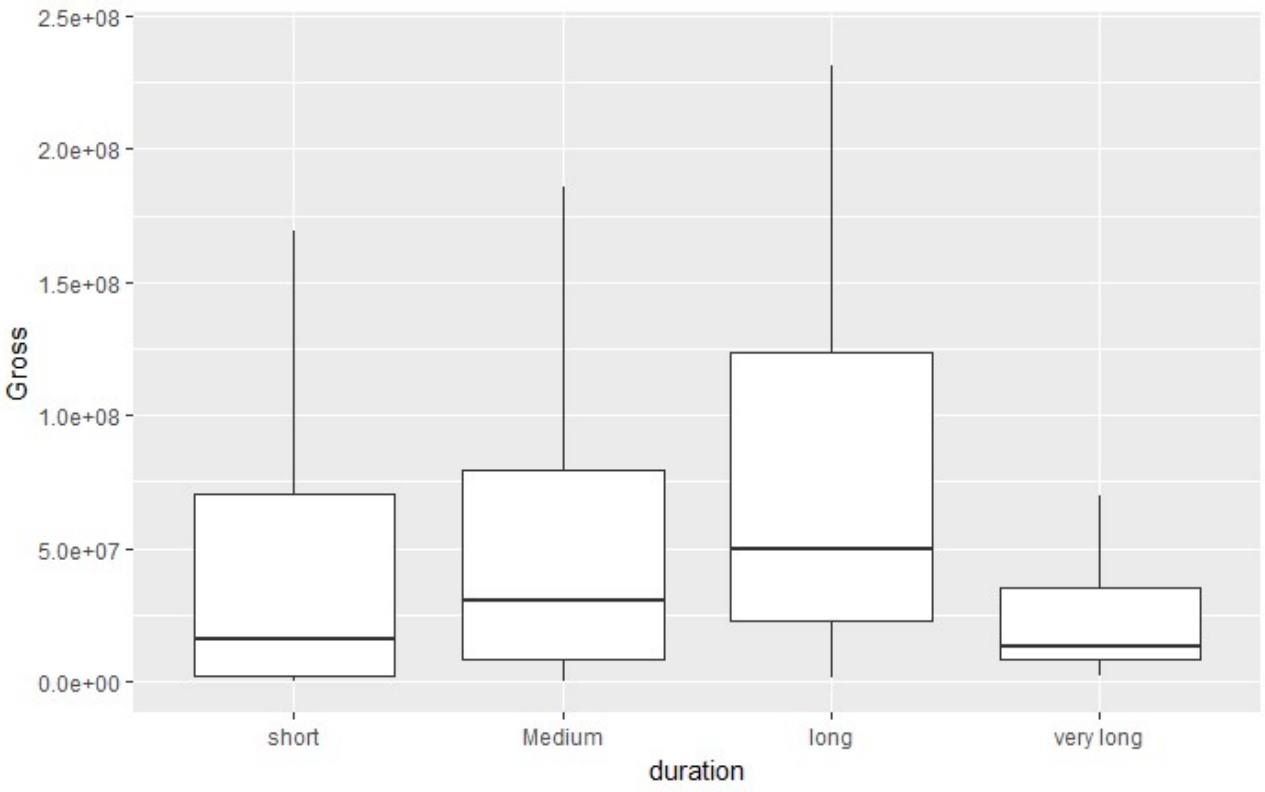
Hide

```
ggplot(na.omit(df_match_year), aes(Runtime, Gross)) + geom_point()+ stat_smooth
(se=FALSE)
```



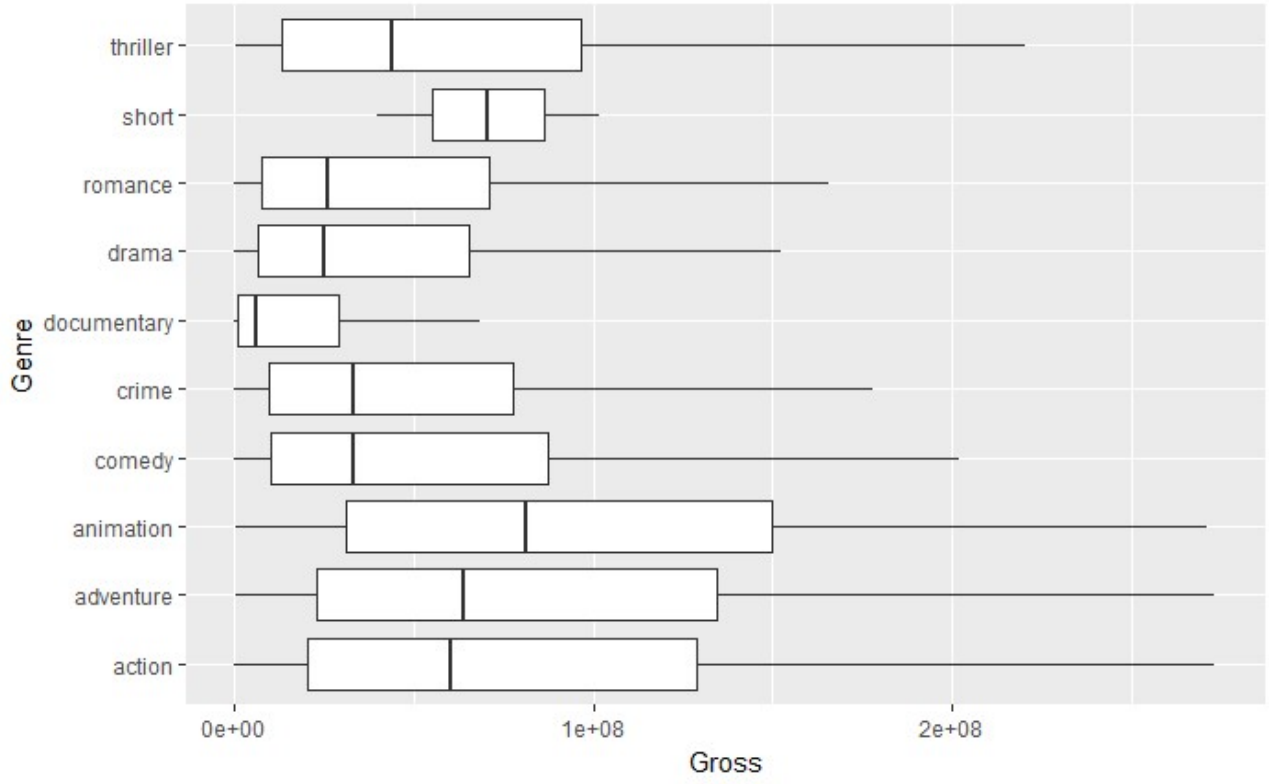
Hide

```
#partition Short vs long
df_match_year$duration[df_match_year$Runtime<80]='short'
df_match_year$duration[df_match_year$Runtime>=80 & df_match_year$Runtime<160]
='Medium'
df_match_year$duration[df_match_year$Runtime>=160 & df_match_year$Runtime<200]
='long'
df_match_year$duration[df_match_year$Runtime>=200]='very long'
df_match_year$duration <- factor(df_match_year$duration, levels = c('short','Me
dium', 'long', 'very long'),ordered = TRUE)
ggplot(data = na.omit(df_match_year), aes(duration, Gross)) + geom_boxplot(outl
ier.size = NA) +
scale_y_continuous(limits = quantile(df_match_year$Gross, c(0.1, 0.9), na.rm =
TRUE))
```



Hide

```
#partition by genre
if (class(df_gen$Released)!="numeric"){
  df_gen$Released<-as.numeric(format(df_gen$Released,'%Y'))
}
df_gen<-subset(df_gen,is.na(df_gen$Released)| (df_gen$Released>=df_gen$Year))
ggplot(data = na.omit(df_gen), aes(Genre, Gross)) + geom_boxplot(outlier.size
= NA)+coord_flip()+scale_y_continuous(limits = quantile(df_gen$Gross, c(0.1, 0.
9), na.rm = TRUE))
```

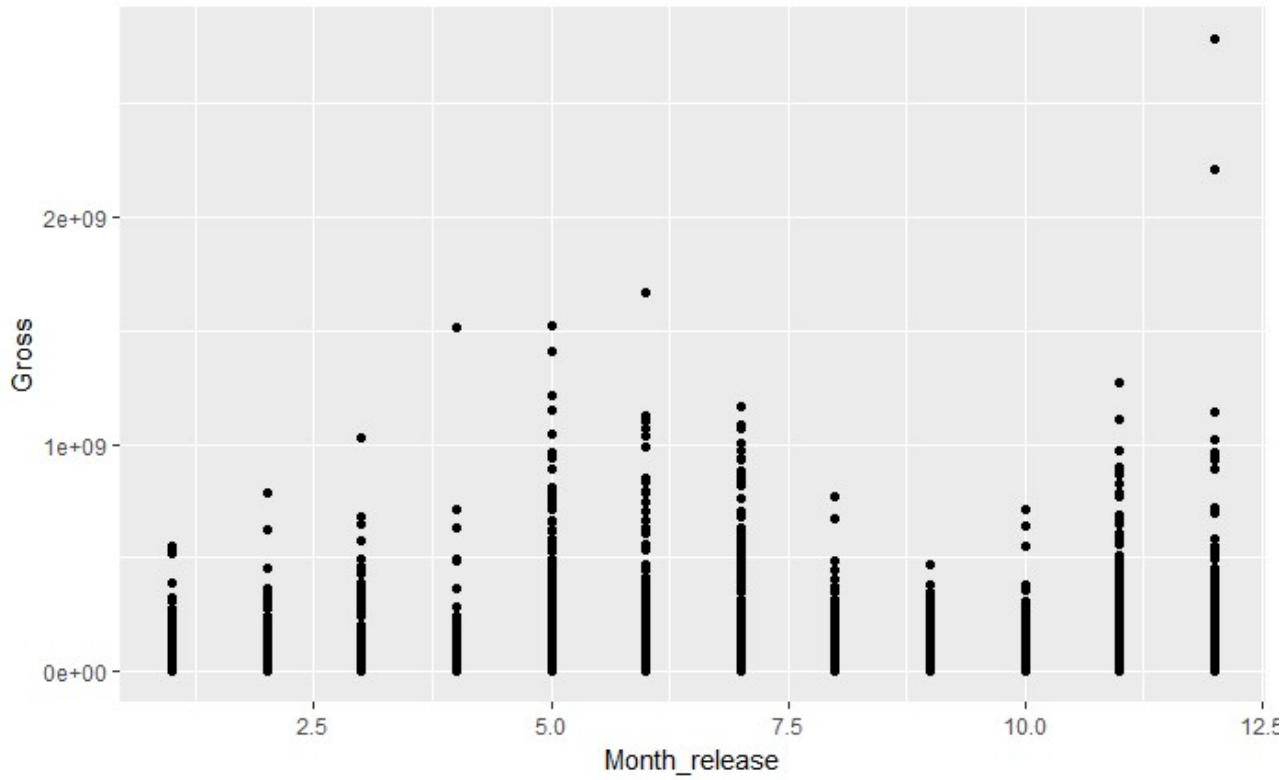


Q: Did you find any observable relationships or combinations of Budget/Runtime/Genre that result in high Gross revenue? If you divided the movies into different subsets, you may get different answers for them - point out interesting ones.

A: Overall, gross revenue increases as budget increases. For runtime below around 150 mins, gross revenue increases with runtime, but decreases when runtime is too long. As for Genre, the gross revenue of animation has the highest median and documentray has the lowest. Adventure and action also have good revenues. Thus, the combinations of runtime of 150 to 200 mins (relatively long movie from box plot) with genre of animation/adventure/action often yield high gross revenue. If budget is not a concern, of coures the higher the better.

Hide

```
# TODO: Investigate if Gross Revenue is related to Release Month
df_mon<-df
df_mon$Month_release = as.numeric(format(df$Released, "%m"))
df_mon$Year_release = as.numeric(format(df$Released, "%Y"))
df_mon<-subset(df_mon,is.na(df_mon$Year_release)| (df_mon$Year_release>=df_mon
$Year))
ggplot(na.omit(df_mon), aes(Month_release, Gross)) + geom_point()
```



Q: Is Gross Revenue is related to Release Month?

A: Yes, It is clear that during summer season (May, June, July) and winter holiday season (Nov.-Dec), the gross revenue are high compare to other months.

6. Process `Awards` column

The variable `Awards` describes nominations and awards in text format. Convert it to 2 numeric columns, the first capturing the number of wins, and the second capturing nominations. Replace the `Awards` column with these new columns, and then study the relationship of `Gross` revenue with respect to them.

Note that the format of the `Awards` column is not standard; you may have to use regular expressions to find the relevant values. Try your best to process them, and you may leave the ones that don't have enough information as NAs or set them to 0s.

Hide

```
# TODO: Convert Awards to 2 numeric columns: wins and nominations
get_wins<-function(str){
  wins=0
  if(str=="N/A" | is.na(str)){return(0)}
  else {
    s=unlist(strsplit(str,split="[\\&\\.]"))
    for(i in 1:length(s)){
      if (grepl("win",s[i],ignore.case = TRUE) | grepl("won",s[i],ignore.case
= TRUE)){
        n=as.numeric(strsplit(s[i],"\\D+")[[1]])
        n=n[length(n)]
        wins=wins+n
          }
    }
  }
  return(wins)
}
get_nomin<-function(str){
  nomin=0
  if(str=="N/A" | is.na(str)){return(0)}
  else {
    s=unlist(strsplit(str,split="[\\&\\.]"))
    for(i in 1:length(s)){
      if (grepl("nomin",s[i],ignore.case = TRUE)){
        n=as.numeric(strsplit(s[i],"\\D+")[[1]])
        n=n[length(n)]
        nomin=nomin+n
          }
    }
  }
  return(nomin)
}
df_match_year$Wins<-sapply(df_match_year$Awards,get_wins)
df_match_year$Nominations<-sapply(df_match_year$Awards,get_nomin)
nrow(subset(df_match_year,df_match_year$Wins>0))
```

```
[1] 10942
```

Hide

```
nrow(subset(df_match_year,df_match_year$Nominations>0))
```

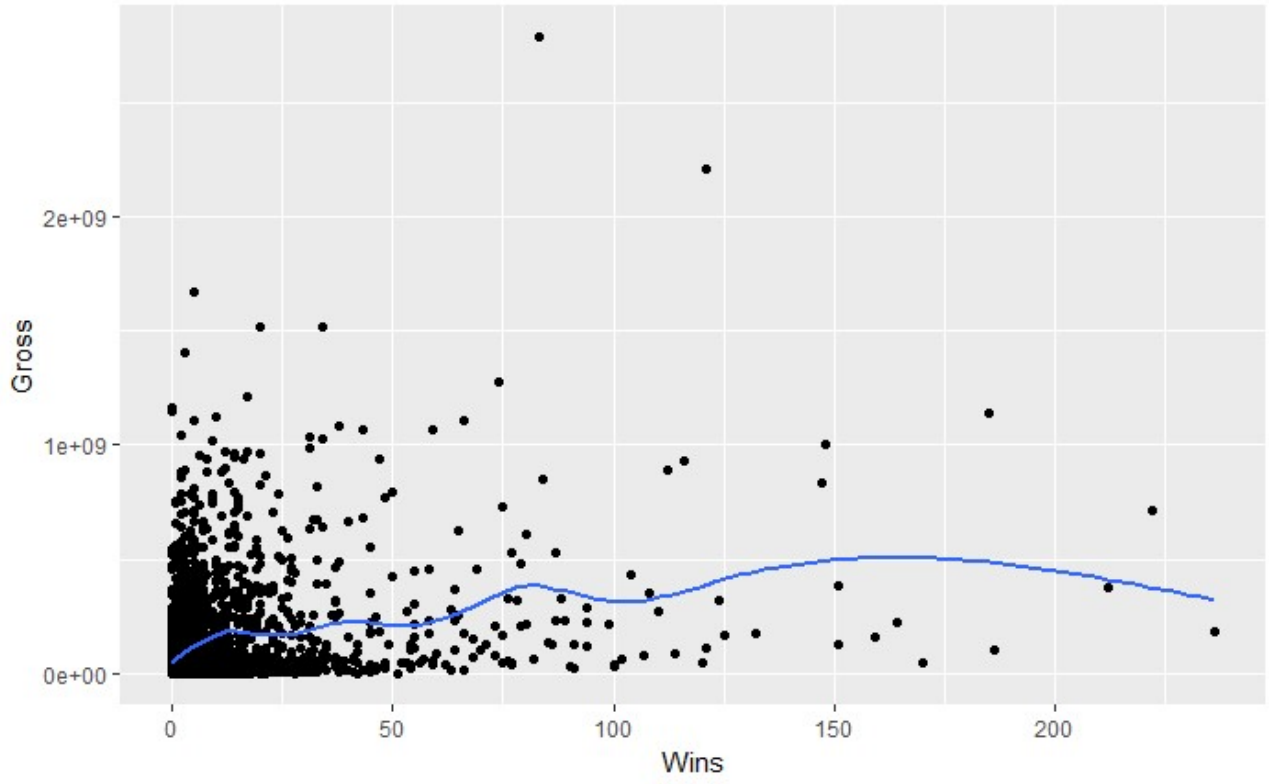
```
[1] 11416
```

Q: How did you construct your conversion mechanism? How many rows had valid/non-zero wins or nominations?

A: First split the string in “Awards” by deliminaters “&” and “.” since I observed that all “wins” or “nominations” are separated in this way. Then find out either this splitted part is a “win” or “normination” by string matching “win” (or “won”) and “nomin”. Lastly, extract the number in the splitted part to the new column based on whehter it is “win” or “normination”. 10942 rows had valid/non-zero wins and 11416 rows had valid/non-zero nominations.

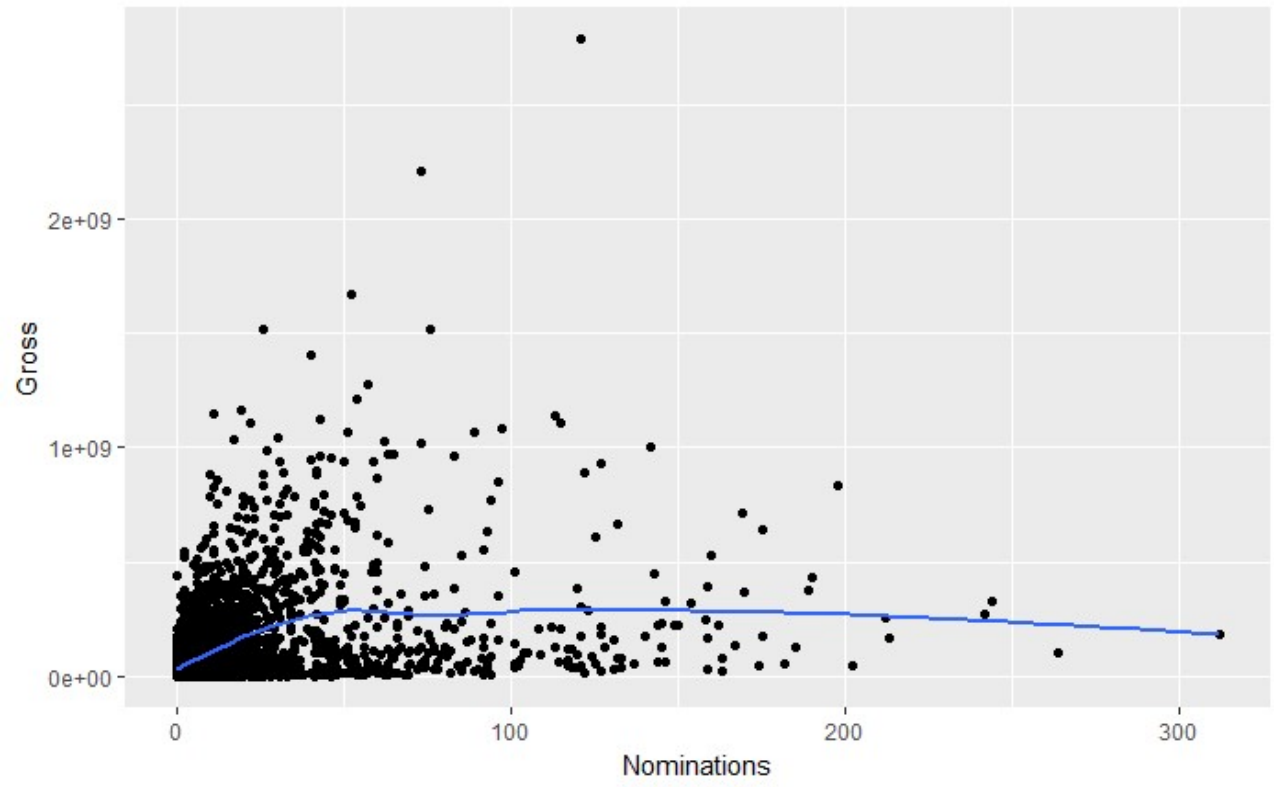
Hide

```
# TODO: Plot Gross revenue against wins and nominations
ggplot(na.omit(df_match_year), aes(Wins, Gross)) + geom_point()+ stat_smooth(se
=FALSE)
```



Hide

```
ggplot(na.omit(df_match_year), aes(Nominations, Gross)) + geom_point()+ stat_sm
ooth(se=FALSE)
```



Q: How does the gross revenue vary by number of awards won and nominations received?

A: The gross revenue increases as the number of wins increase. This is more obvious for wins less than 100.The gross revenue also increases with number of nominations received when that number is less than 30, after which the gross revenue becomes more flat.

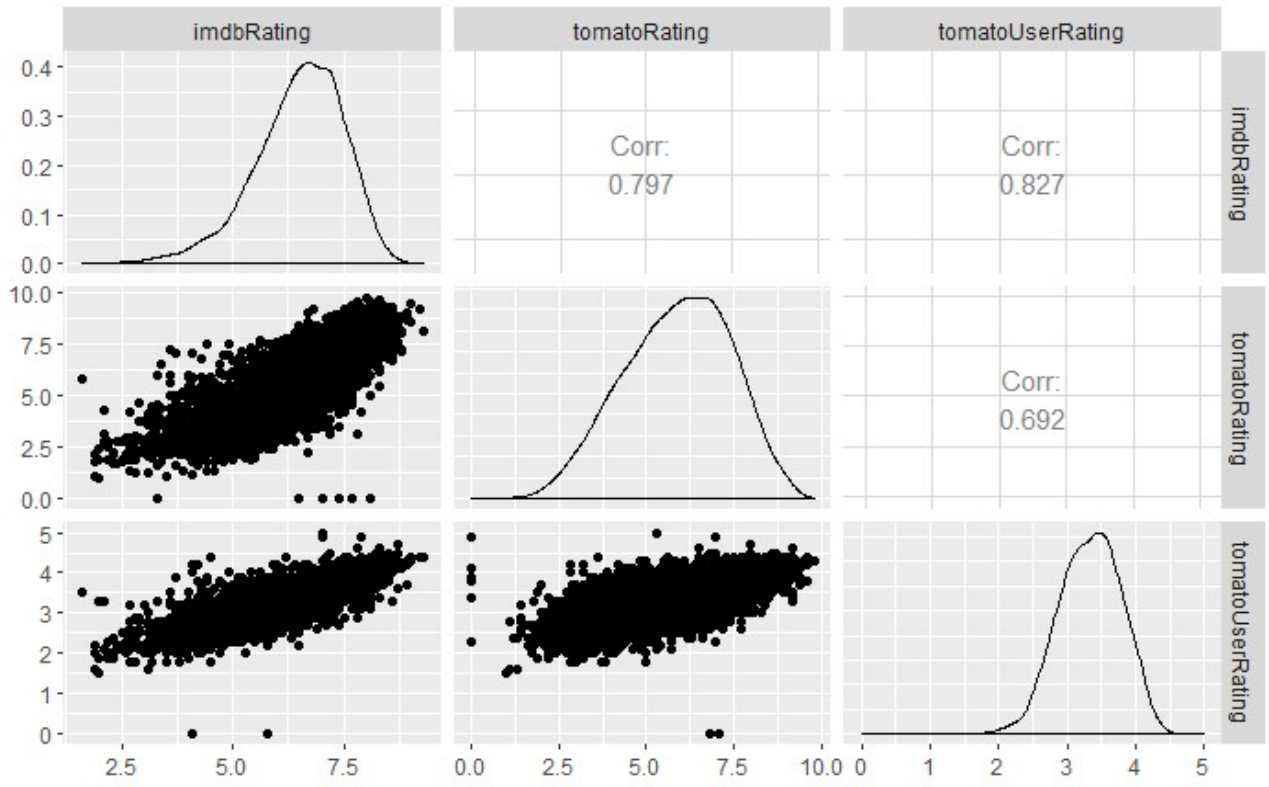
7. Movie ratings from IMDb and Rotten Tomatoes

There are several variables that describe ratings, including IMDb ratings (`imdbRating` represents average user ratings and `imdbVotes` represents the number of user ratings), and multiple Rotten Tomatoes ratings (represented by several variables pre-fixed by `tomato`). Read up on such ratings on the web (for example rottentomatoes.com/about (https://www.rottentomatoes.com/about) and www.imdb.com/help/show_leaf?votestopfaq (http://%20www.imdb.com/help/show_leaf?votestopfaq)).

Investigate the pairwise relationships between these different descriptors using graphs.

Hide

```
# TODO: Illustrate how ratings from IMDb and Rotten Tomatoes are related
library(GGally)
df_rating <- df_match_year[, c("imdbRating", "tomatoRating", "tomatoUserRatin
g")]
ggpairs(na.omit(df_rating))
```



Q: Comment on the similarities and differences between the user ratings of IMDb and the critics ratings of Rotten Tomatoes.

A: Pariwise relationship among “imdbRating”, “tomatoRating” and “tomatoUserRating” reveals that they have strong correlations. Among them, “tomatoUserRating” and “imdbRating” has the strongest corrlation. Thus the ratings between IMDB rating and critics rating from “tomatoUserRating” is the most similar, while “tomatoRating” and “tomatoUserRating” has more differences than other pairs, although all pairs shows pretty resonable correlations.

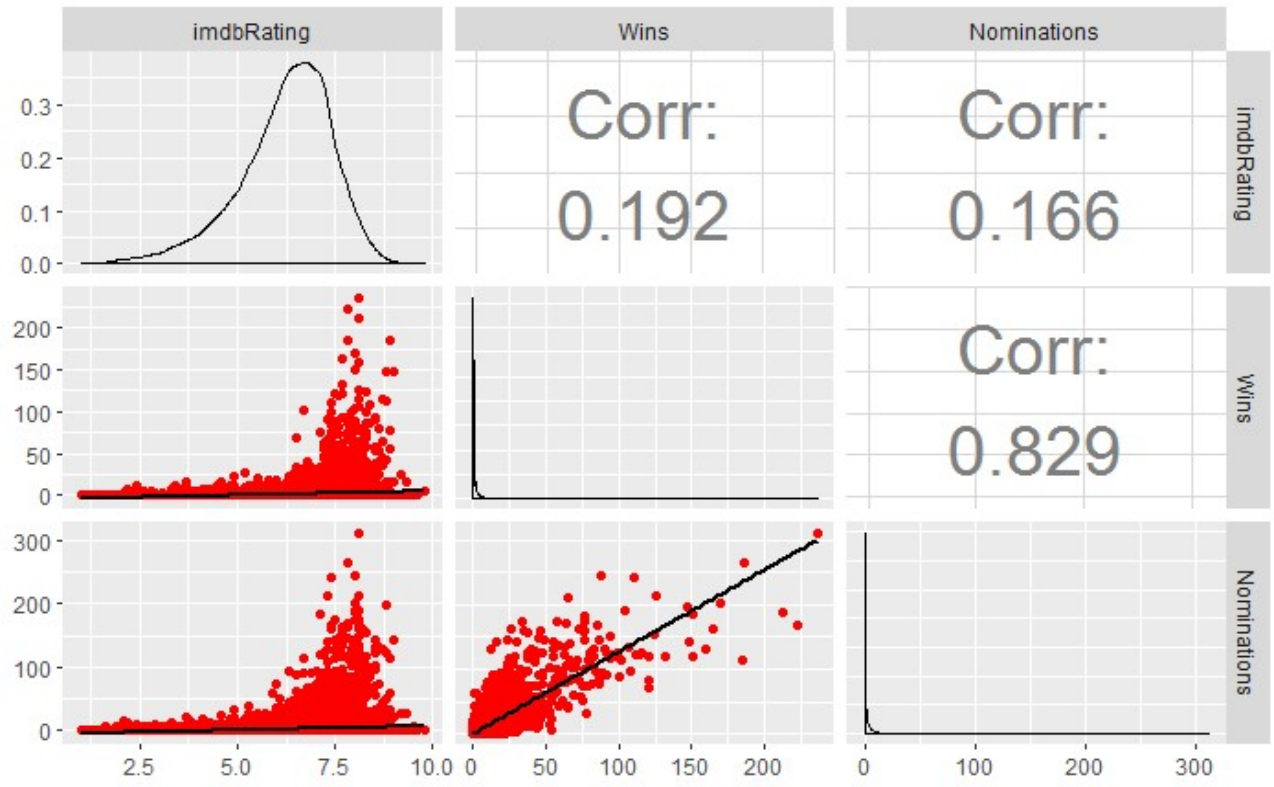
8. Ratings and awards

These ratings typically reflect the general appeal of the movie to the public or gather opinions from a larger body of critics. Whereas awards are given by professional societies that may evaluate a movie on specific attributes, such as artistic performance, screenplay, sound design, etc.

Study the relationship between ratings and awards using graphs (awards here refers to wins and/or nominations).

Hide

```
# TODO: Show how ratings and awards are related
library(GGally)
df_rating_awards <- df_match_year[, c("imdbRating", "Wins", "Nominations")]
ggpairs(na.omit(df_rating_awards),upper = list(continuous = wrap("cor",size = 1
0)), lower = list(continuous = wrap("smooth", color="red")))
```



Q: How good are these ratings in terms of predicting the success of a movie in winning awards or nominations? Is there a high correlation between two variables?

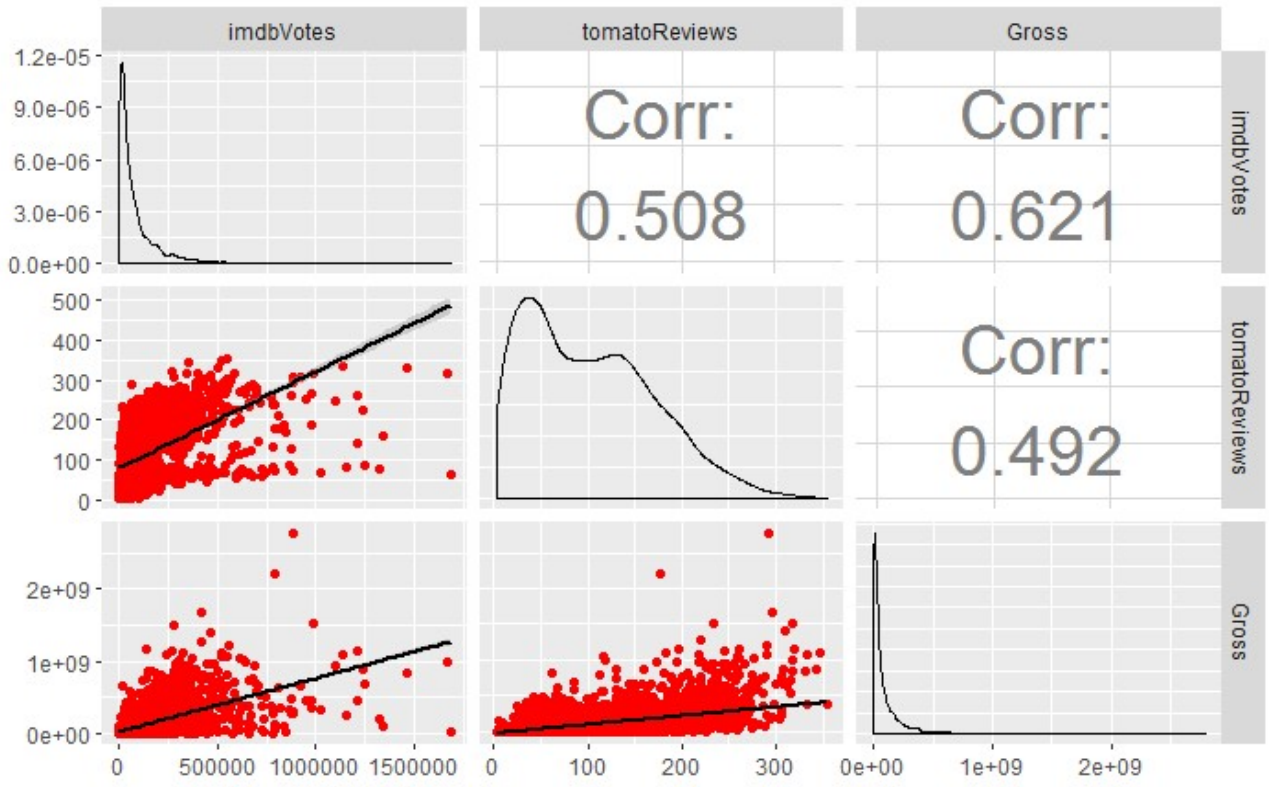
A: There is no high correlation between the rating and number of awards or nominations received. So the rating is not a good predictor of the awards (correlation coefficient are less than 0.2).

9. Expected insights

Come up with two new insights (backed up by data and graphs) that is expected. Here ânewâ means insights that are not an immediate consequence of one of the above tasks. You may use any of the columns already explored above or a different one in the dataset, such as `Title`, `Actors`, etc.

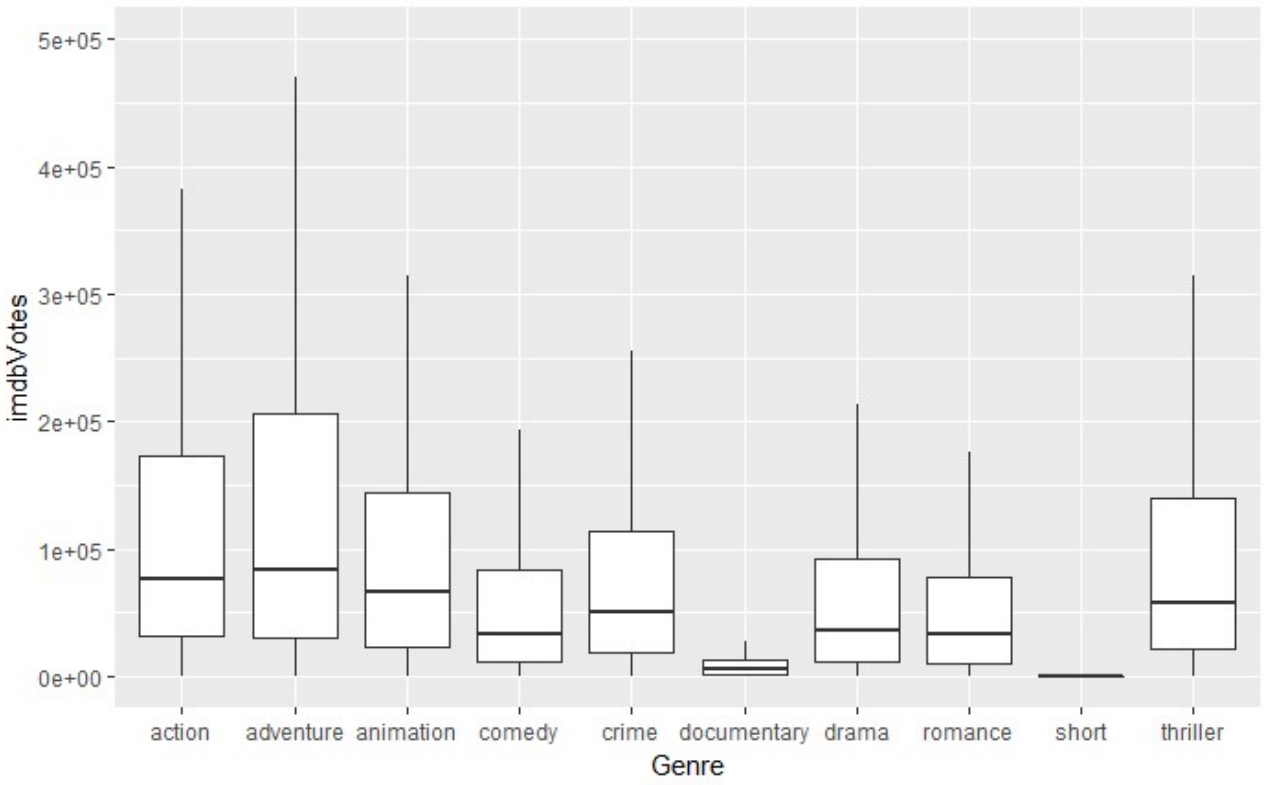
Hide

```
# TODO: Find and illustrate two expected insights
## Ratings vs gross revenue
library(GGally)
df_rating_votes <- df_match_year[, c("imdbVotes", "tomatoReviews", "Gross")]
ggpairs(na.omit(df_rating_votes),upper = list(continuous = wrap("cor",size = 1
0)), lower = list(continuous = wrap("smooth", color="red")))
```



Hide

```
#what type of movie is the most popular?
ggplot(data = na.omit(df_gen), aes(Genre, imdbVotes)) + geom_boxplot(outlier.si
ze = NA)+coord_flip()+coord_cartesian(ylim = c(0, 500000))
```



Q: Expected insight #1.The votes from viewers affected the gross revenue

A: The votes of movie viewers (“imdbVotes”) has a relatively strong correlation with gross revenue. It makes sense because the more people voted the movie, the more people are likely to watch it, then higher the gross revenue. Similarly, the more reviews (“tomatoReviews”) one movie received, the more likely more people watched it.

Q: Expected insight #2.what type of movie is the most popular?

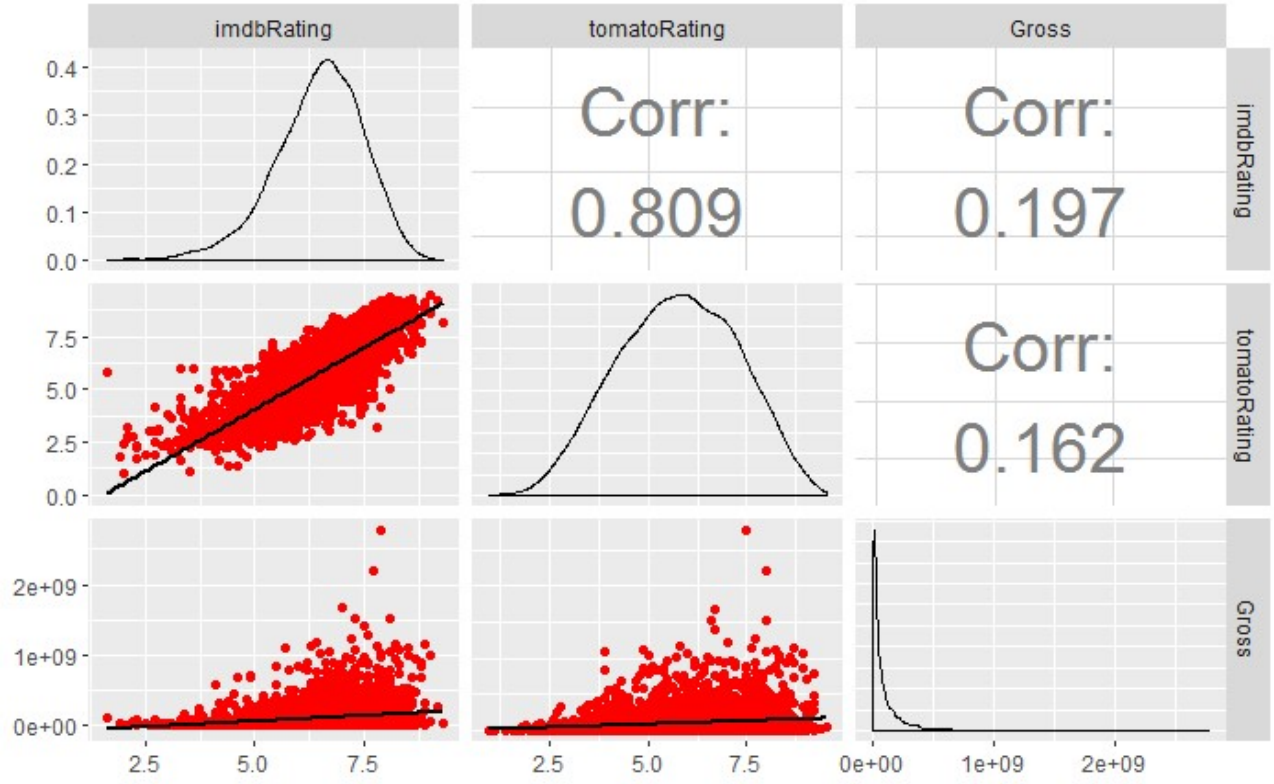
A: It is not suprise that adventure movie is the most voted genre.The second and third are action and animation. On the other hand, documentary and short movies are not popular.

10. Unexpected insight

Come up with one new insight (backed up by data and graphs) that is unexpected at first glance and do your best to motivate it. Same instructions apply as the previous task.

Hide

```
# TODO: Find and illustrate one unexpected insight
# Ratings vs gross revenue
library(GGally)
df_rating_gross <- df_match_year[, c("imdbRating", "tomatoRating", "Gross")]
ggpairs(na.omit(df_rating_gross),upper = list(continuous = wrap("cor",size = 1
0)), lower = list(continuous = wrap("smooth", color="red")))
```



Q: Unexpected insight.I was expecting the user rating would has a correlation with the gross revenue since movies with good ratings should attract more people to watch.

A: However, the results were unexpected because the correlation values are low (<0.2). So the rating is not a good prediction for gross revenue.