Project 1: Explore and Prepare Data

CSE6242 - Data and Visual Analytics - Spring 2017 Due: Sunday, March 5, 2017 at 11:59 PM UTC-12:00 on T-Square Author : Melisande Zonta Roudes GT account name : mzr3

Contents

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` 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 IMDb???s API (see 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 IMDb???s 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 ???second window??? streaming rights).

Instructions

This is an R Markdown Notebook. Open this file in RStudio to get started.

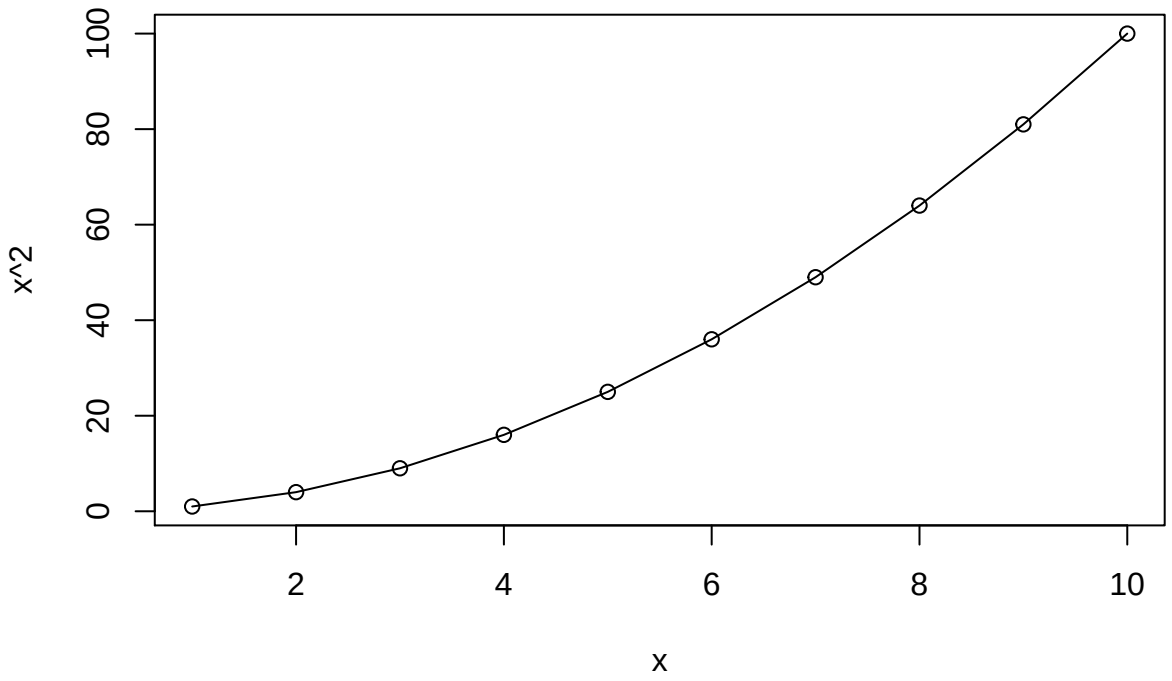
When you execute code within the notebook, the results appear beneath the code. Try executing this chunk by clicking the *Run* button within the chunk or by placing your cursor inside it and pressing *Cmd+Shift+Enter*.

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

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

Plots appear inline too:

```
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` file and it is in the current working directory. Now load it into memory:

```
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:

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

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

```
colnames(df)
```

```
##  [1] "Title"             "Year"              "Rated"
##  [4] "Released"          "Runtime"           "Genre"
##  [7] "Director"          "Writer"            "Actors"
## [10] "Plot"              "Language"          "Country"
## [13] "Awards"            "Poster"            "Metascore"
## [16] "imdbRating"        "imdbVotes"         "imdbID"
## [19] "Type"              "tomatoMeter"       "tomatoImage"
## [22] "tomatoRating"      "tomatoReviews"     "tomatoFresh"
## [25] "tomatoRotten"      "tomatoConsensus"   "tomatoUserMeter"
## [28] "tomatoUserRating"  "tomatoUserReviews" "tomatoURL"
## [31] "DVD"               "BoxOffice"         "Production"
## [34] "Website"           "Response"          "Budget"
## [37] "Domestic_Gross"    "Gross"             "Date"
```

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.

```
library(ggplot2)
library(reshape)
library(GGally)
library(stringr)
library(splitstackshape)
library(grid)
library(gridExtra)
library(plyr)
```

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: install.packages('stringr') and install.packages('splitstackshape')

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.

```
# TODO: Remove all rows from df that do not correspond to movies
df1 = subset(df,Type == "movie")
dim(df1)
```

```
## [1] 40000    39
```

```
dim(df)
```

```
## [1] 40789    39
```

```
df = df1
```

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

A: 789 examples were of type "non movie". As the dataset had 40789, 40000 observations remain.

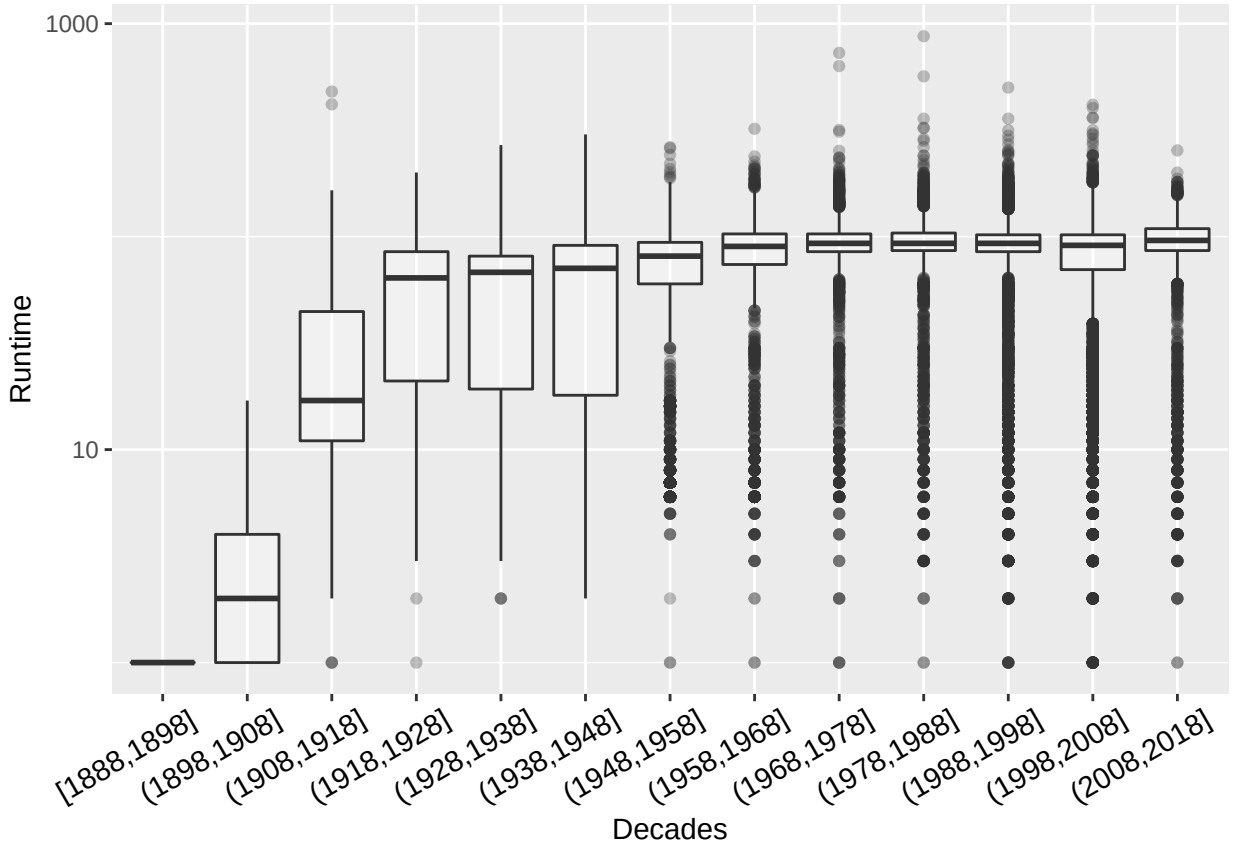
2. Process `Runtime` column

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.

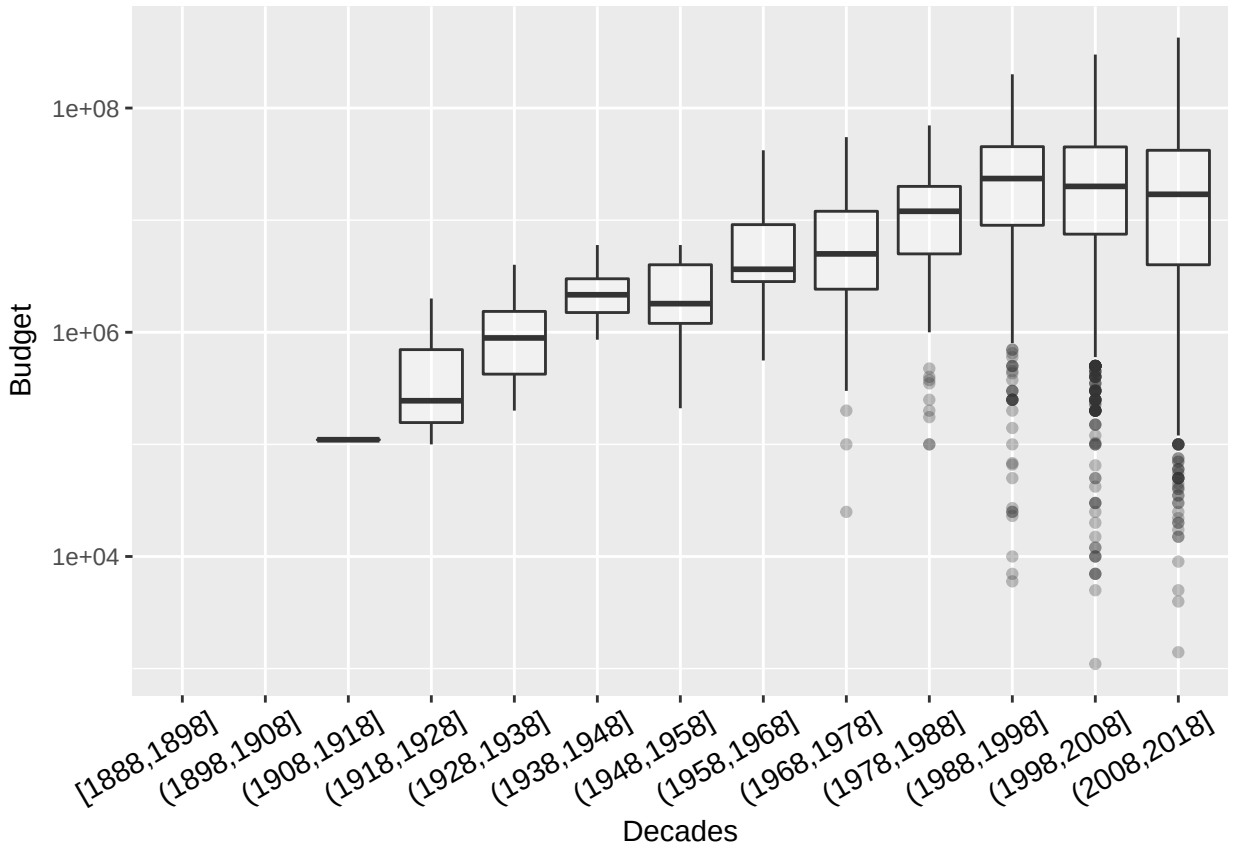
```
# TODO: Replace df$Runtime with a numeric column containing the runtime in minutes

# Remove_min function allowing to delete the 'mins'

remove_min = function(x){
if (x != 'NA'){
   return(str_sub(x,start = 1,end = -5))
}
}

# Apply the previous function and cConversion to numeric

new = lapply(df$Runtime,remove_min)
df$Runtime = as.numeric(new)
```

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.

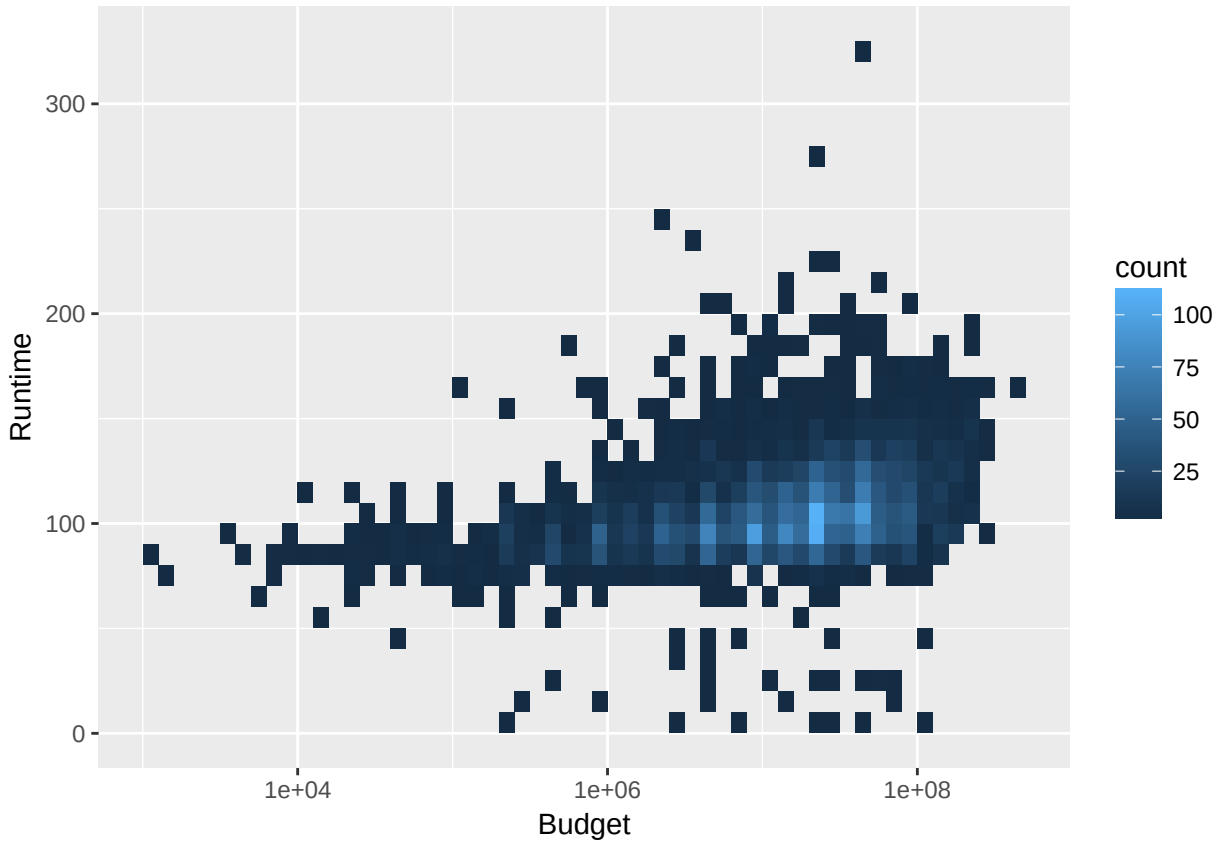
```r
# Year into decades

df$decades <- cut(df$Year, breaks=(seq(min(df$Year),max(df$Year),by = 10)), dig.lab = 4, include.lowest

# Investigation between Runtime and Decades

ggplot(df, aes(reorder(decades, -Runtime, median), Runtime))+
     geom_boxplot(alpha = 0.3)+
     scale_y_log10()+
     theme(axis.text.x=element_text(color = "black", size=11, angle=30, vjust=.8, hjust=0.8))+
     xlab("Decades")+ylab("Runtime")
```



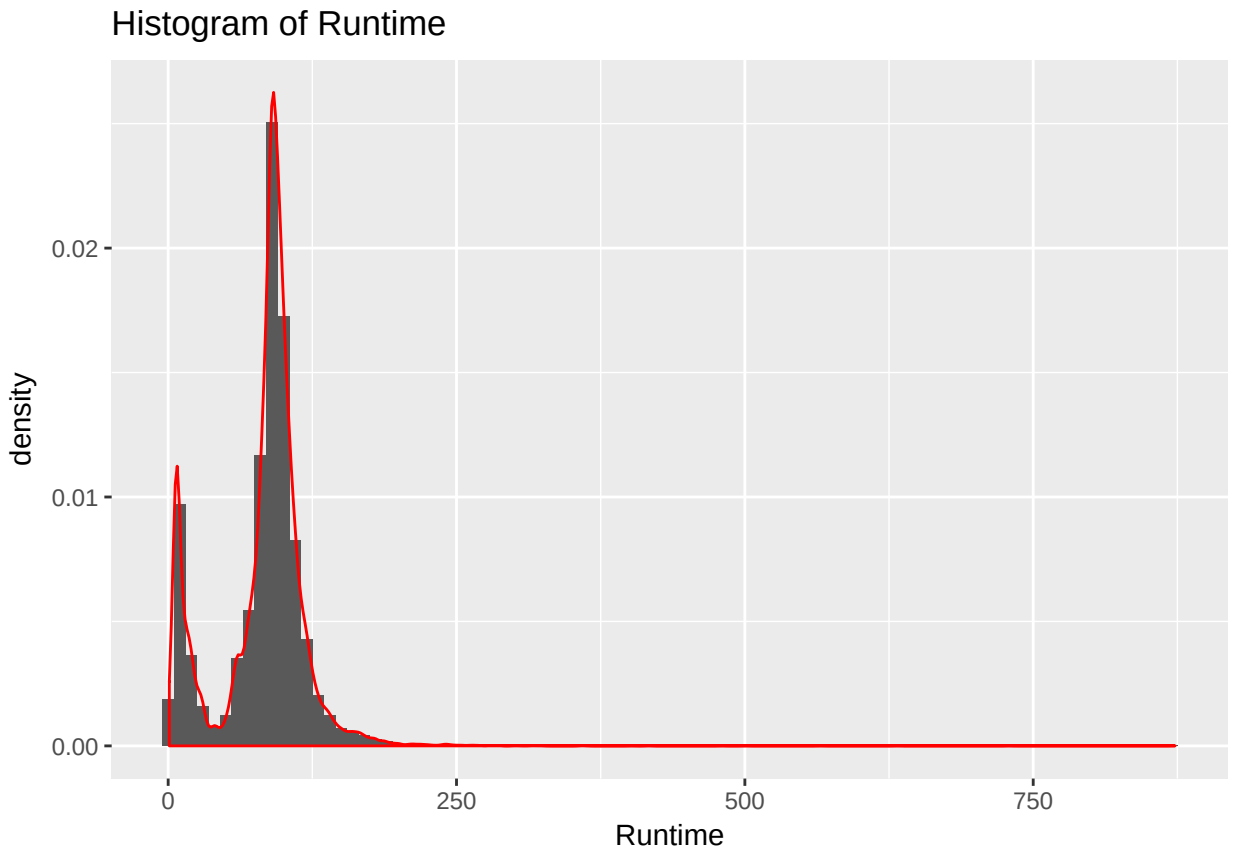

```r
# Investigation between Budget and Decades

ggplot(df, aes(reorder(decades, -Budget, median), Budget))+
     geom_boxplot(alpha = 0.3)+
     scale_y_log10()+
     theme(axis.text.x=element_text(color = "black", size=11, angle=30, vjust=.8, hjust=0.8))+
     xlab("Decades")+ylab("Budget")
```



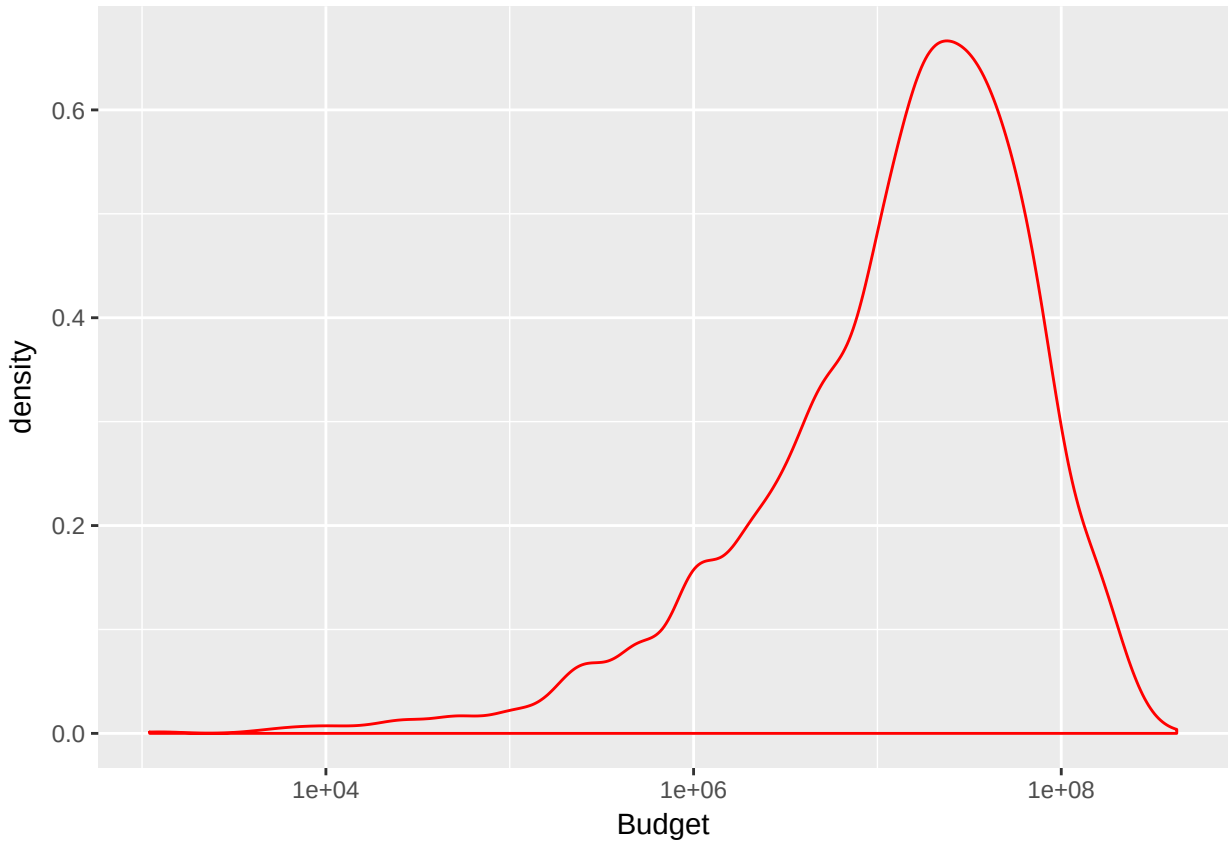

```
# TODO: Investigate the distribution of Runtime values and how it varies by Budget
ggplot(df,aes(x = df$Budget, y = df$Runtime)) +
geom_bin2d(binwidth = c(0.1, 10))+
scale_x_log10()+
xlab('Budget')+
ylab('Runtime')
```




```
# Representation of the distributions

ggplot(df, aes(x = Runtime, y = ..density..)) +
     geom_histogram(binwidth = 10) +
     geom_density(size = .5, color = "red")+
     ggtitle(label = "Histogram of Runtime")
```



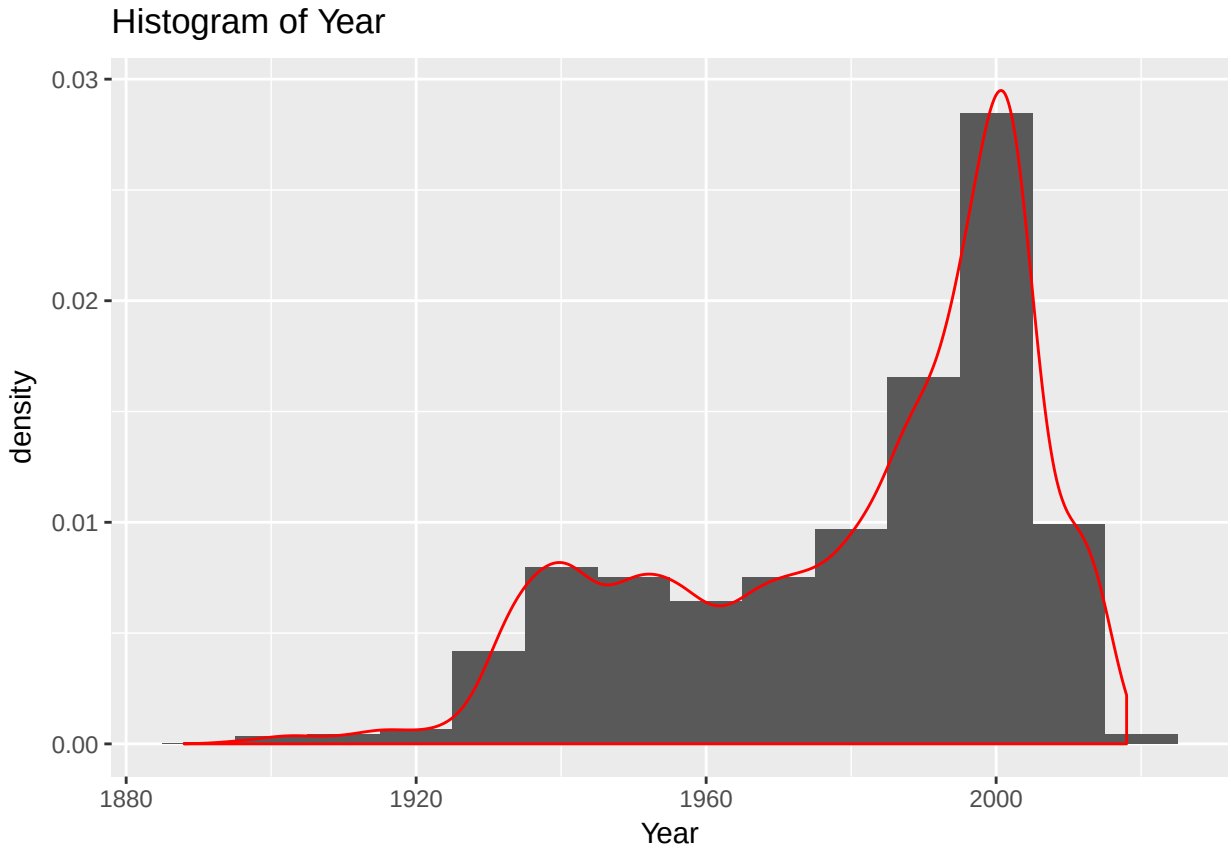

```
ggplot(df, aes(x = Budget, y = ..density..)) +
     geom_density(size = .5, color = "red")+
     scale_x_log10()
```




```
     ggtitle(label = "Histogram of Budget")
```

```
## $title
## [1] "Histogram of Budget"
##
## $subtitle
## NULL
##
## attr(,"class")
## [1] "labels"
```

```
ggplot(df, aes(x = Year, y = ..density..)) +
     geom_histogram(binwidth = 10) +
     geom_density(size = .5, color = "red")+
     ggtitle(label = "Histogram of Year")
```



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: The Runtime distribution is bimodal : one small around 8 mins and one larger around 100 mins. The first one is typical of short films and the second one is current for films. The Budget distribution is unimodal around 50 millions of dollars which is normal for a production. Finally, the Year distribution presents an increasing evolution with a peak in 2000 which is consistent with the dataset focused on the 2000 decades. A boom in the production films occured in the fourties because of the after second war.

The boxplot between runtime and year shows that the median keeps increasing over the decades, but we can observe that during the first decade (1898-1908) a film was 5 minutes long and that one century after the duration has been multiplied by 20. One can see that the duration began to stabilize around the 1958 decade indeed the spread of boxplots fall in a range between 20 mins and 130 mins which means that films begin to answer to some standards.

The boxplot between budget and year shows that the median is growing exponentially over the decades and that the budget of the beginning of the 19th century and the 20th century was multiplied by 500. Furthermore, the diversity of films was increased as the spread of boxplots and the presence of outliers starting from 1988 demonstrates indeed in the 2008 decade, the spread is of 50 millions.

The map between Runtime and Budget does not put forward a real relationship between the two attributes, we can only conlude from the importance of the count that the film the most produced has a duration around 100 min and a budget around 50 millions.

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).

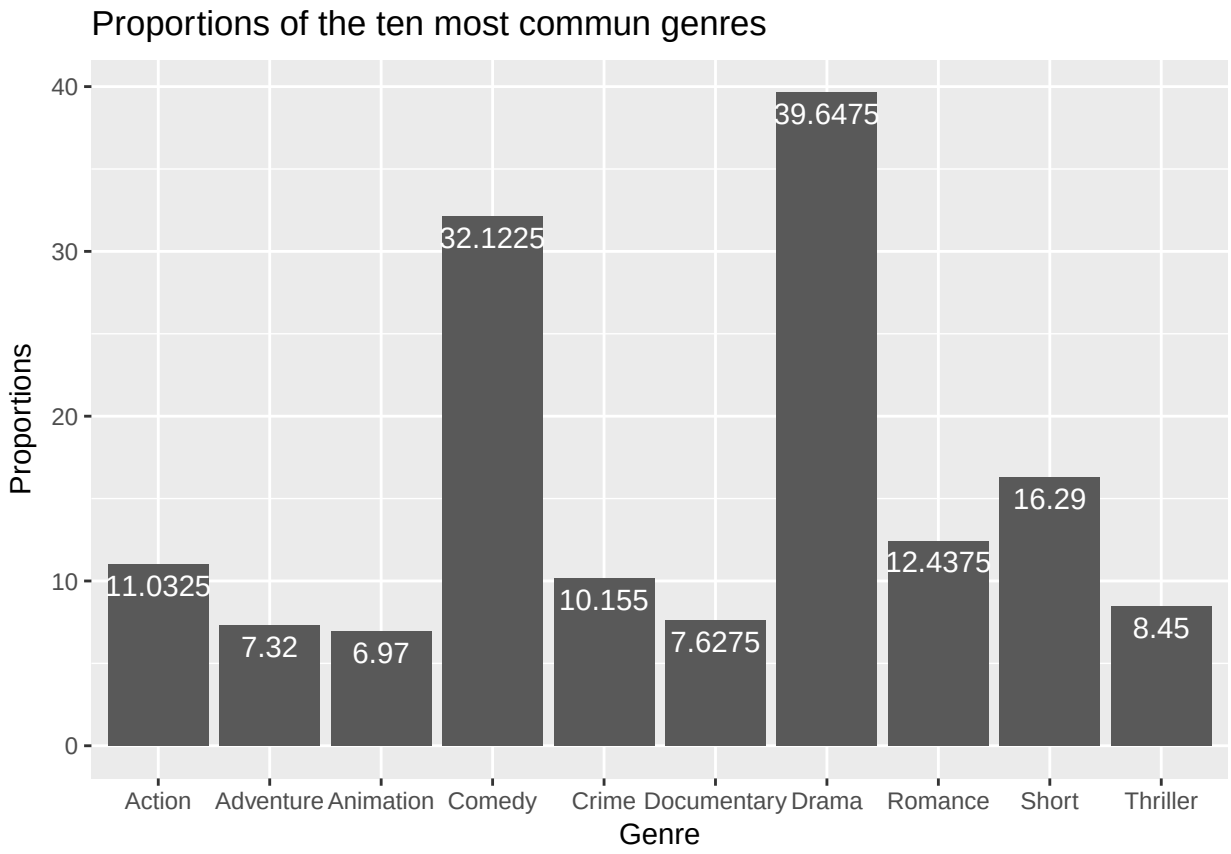
```
# Suppress spaces in Genre
df$Genre = gsub(" ", "", df$Genre)
```

```
# TODO: Replace Genre with a collection of binary columns

list_genre = c()
temp = c()

# Concatenate genres of all films in a list
for (i in seq(1,length(df$Genre))) {
  temp = unlist(strsplit(df$Genre[i],","))
  list_genre = c(list_genre,temp)
}

# Remove duplicated values in the previous list

duplicated.values = which(duplicated(list_genre))
list.genre.bis = list_genre[-duplicated.values]
list.genre.bis = sort(list.genre.bis)

map = matrix(0,length(df$Genre),length(list.genre.bis))

# Create a map of 1 and 0 for each genre
for (j in seq(1,length(df$Genre))){

  temp = unlist(strsplit(df$Genre[j],","))

  for (k in seq(1,length(temp))){
    index = which(temp[k] == list.genre.bis)
    map[j,index] = 1
  }
}

# Remove the old Genre column
df$Genre <- NULL
dim_old = (dim(df)[2])+1
data_frame = data.frame(map)
names(data_frame) = list.genre.bis

# Create the new data frame with the 29 columns added
df = cbind(df,data_frame)
```

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

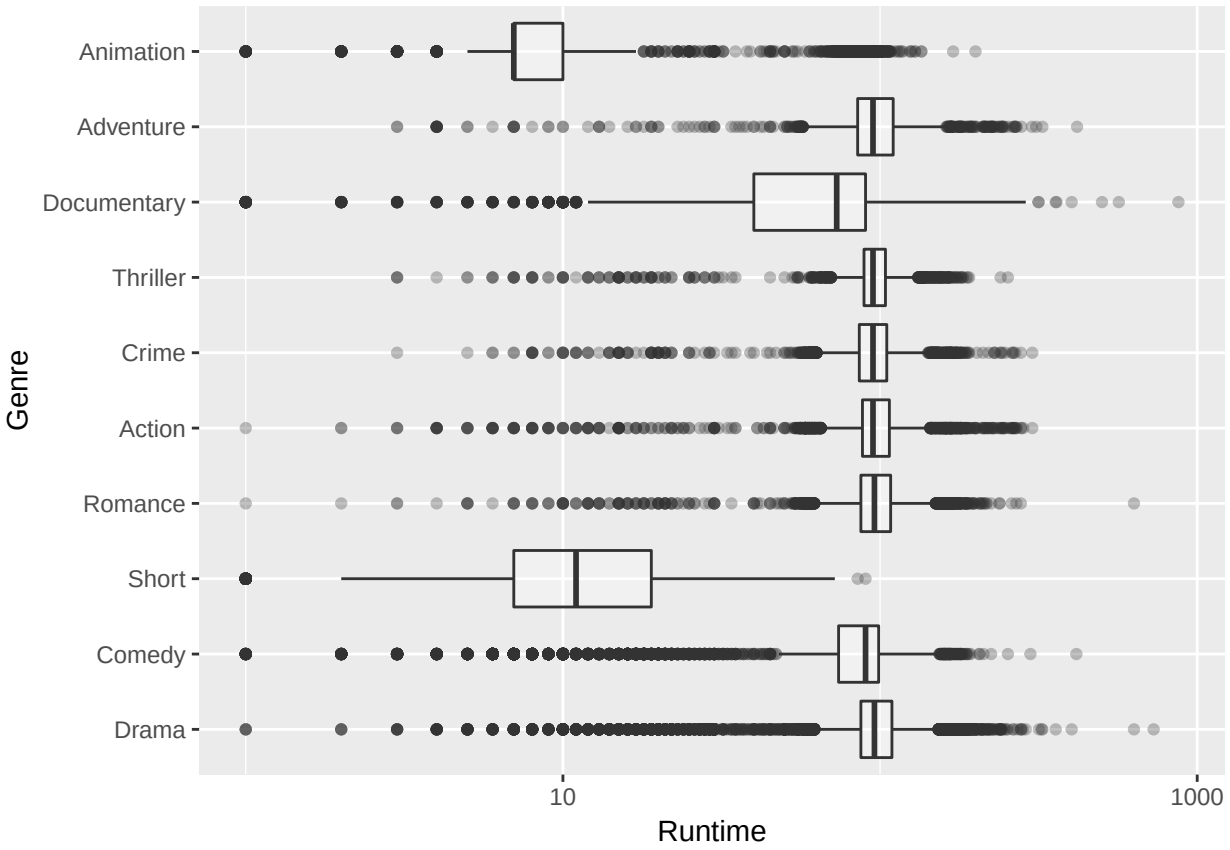
```r
# TODO: Select movies from top 10 most common genres and plot their relative proportions

# Sum all the 1 in each column

list = lapply(df[,dim_old:length(df)],sum)

# Order the list in decreasing sum
list = list[rev(order(sapply(list,'[[',1)))]

# Create the data frame for the 10 most common genres
list.10.first = head(list,10)
data.10.first = as.data.frame(list.10.first)
data_total = as.data.frame(list)
sum_total = dim(df)[1]
data.10.first.relative = (data.10.first/sum_total)*100
temp = c()
for (i in seq(1,length(data.10.first.relative))){
  temp = c(temp,data.10.first.relative[,i])
}

data_frame.10.first.relative = data.frame(genre = names(list.10.first),proportions = temp)
```

```r
# TODO: Plot their relative proportions

ggplot(data=data_frame.10.first.relative, aes(x = data_frame.10.first.relative$genre, y=data_frame.10.f
geom_bar(stat="identity")+
geom_text(aes(label=data_frame.10.first.relative$proportions), vjust=1.5, colour="white")+
ggtitle("Proportions of the ten most commun genres") +
xlab("Genre") +
ylab("Proportions")
```



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

```
# TODO: Plot Runtime distribution for top 10 most common genres

# Select a dataframe with only the rows of genre and runtime
temp.genre = data.frame(Runtime = df$Runtime,df[,names(list.10.first)])

data.genre.plus.runtime = data.frame(Genre = rep(names(list.10.first)[1],length(temp.genre$Runtime[whic

for (i in (seq(2,length(names(list.10.first))))){
data2 = data.frame(Genre = rep(names(list.10.first)[i],length(temp.genre$Runtime[which(temp.genre[[name
data.genre.plus.runtime = rbind(data.genre.plus.runtime,data2)
}

p <- ggplot(data.genre.plus.runtime, aes(reorder(Genre, -Runtime, median), Runtime))
p + geom_boxplot(alpha = 0.3) +
    coord_flip() +
    scale_y_log10()+
    scale_x_discrete("Genre")
```



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

A: We can observe on the barchart reprensenting the proportions of the ten most common genres that two genres stand out of the crowd that is drama (around 40%) and comedy (around 32%). This feature is expected since most of films are classified in one of these categories. What can be surprising is the proportion of short films, indeed it represents the third most common genre and in my opinion, it's far less common than a thriller movie.

The representation of Genre according to the Runtime provides expected features such as the duration of short films. Except Animation films which duration is intriguing since in my mind it lasts as long as comedies or others films, all the others categories have the median duration of a film around 100 minutes. Here all categories have disjoint values from Animations films which duration can be shorter than short movies.

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 source???s 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.

```
# TODO: Remove rows with Released-Year mismatch

# Function to select only the year in the released year
```

```
released.in.year = function(x){
if (is.na(x) == FALSE){
   x = as.character(x)
   return(as.numeric(str_sub(x,start = 1,end = 4)))
}
  else {
    return(x)
  }
}

# Function to select only the month in the released year
released.in.month = function(x){
if (is.na(x) == FALSE){
   x = as.character(x)
   return(as.numeric(str_sub(x,start = 6,end = 7)))
}
  else {
    return(x)
  }
}

# Create lists where the previous were applied
new_year = c()
new_month = c()
for (i in 1:length(df$Released)){
    new_year = c(new_year,released.in.year(df$Released[i]))
    new_month = c(new_month,released.in.month(df$Released[i]))
}

# Select the index of the lists (the films) which don't abide by the condition
index_remove = c()
for (i in seq(1,length(new_year))){
  if (is.na(new_year[i])== FALSE){
    if ((((new_year[i] == (df$Year[i]+1)) && (new_month[i] > 3))||
         (new_year[i] > (df$Year[i]+1)))
      || (((new_year[i] == (df$Year[i]-1)) && (new_month[i] < 10))||
          (new_year[i] < (df$Year[i]-1))))
{
        index_remove = c(index_remove,i)
}
  }
}
# Remove the rows
df4 = df[-index_remove,]

# Calculate the percentage of gross value present removed
percent.gross.present =
(( length(df$Gross[is.na(df$Gross) == FALSE]) - length(df4$Gross[is.na(df4$Gross) == FALSE])) /length(d

cat("Before removal the number of rows was", dim(df)[1], "and after it is", dim(df4)[1], ".", "Whereas
```

```
## Before removal the number of rows was 40000 and after it is 35991 . Whereas the percentage of remove
```

```
df = df4
```

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

A: The first method which seemed logical at first sight was to remove all the titles which present a mismatch between release_year and year but this was too rude regarding the removal of gross values present (16%). So I constrained less the conditions and removed all the films which have a difference equal or more that 2 years. As the resulting percentage of removed gross values present was around 2%, I released the constraint to a difference of more than 3 months. And it removed 4009 rows and the percentage of removed gross values present is less than 9%.

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.

```
# TODO: Investigate if Gross Revenue is related to Runtime

# Partitionning of the runtime values
summary(df$Runtime)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    1.00   69.00   90.00   80.82  101.00  873.00     883
```

```
first.quartile = subset(df,df$Runtime < summary(df$Runtime)[2])
dim(first.quartile)
```

```
## [1] 8708   68
```

```
second.quartile = subset(df,((df$Runtime >= summary(df$Runtime)[2]) & (df$Runtime < summary(df$Runtime)
dim(second.quartile)
```

```
## [1] 8181   68
```

```
third.quartile = subset(df,((df$Runtime >= summary(df$Runtime)[3]) & (df$Runtime < summary(df$Runtime)[
dim(third.quartile)
```

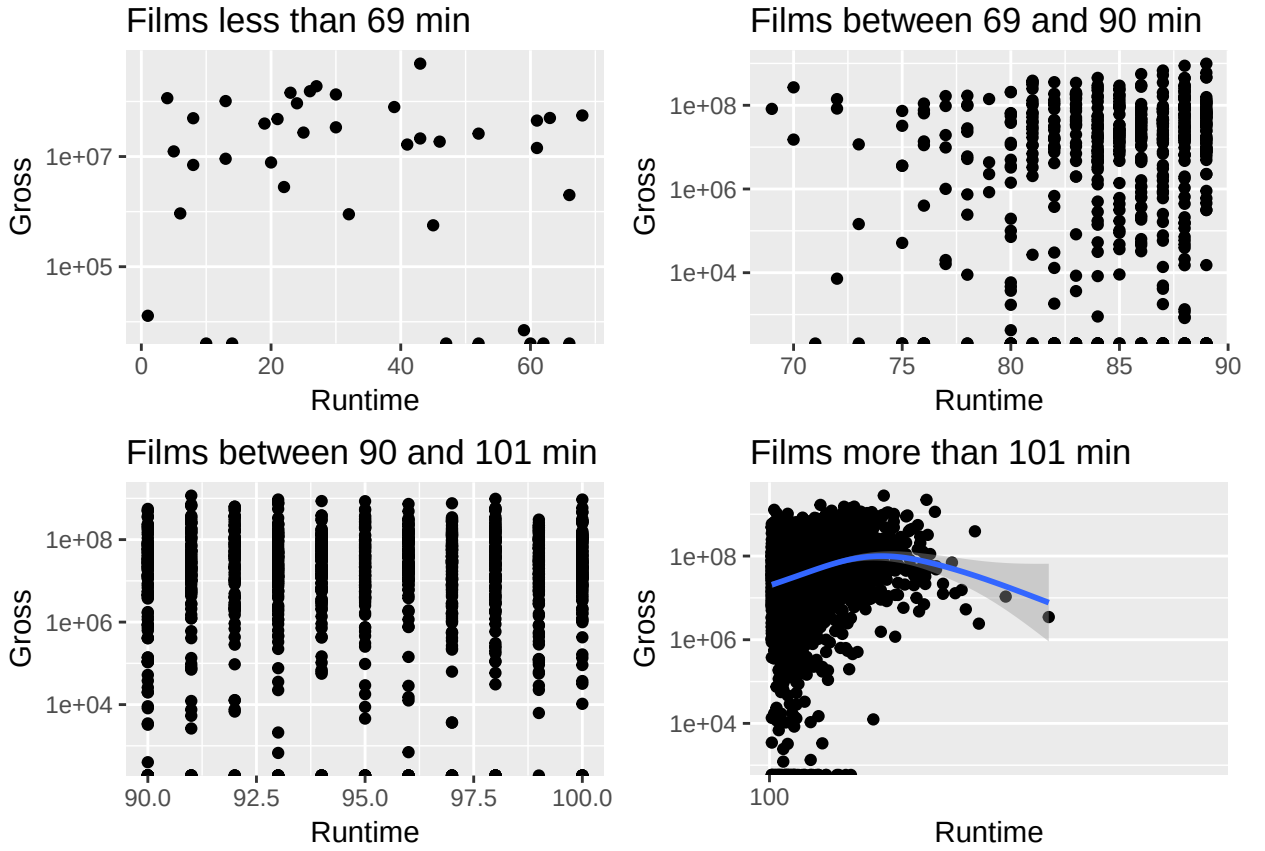
```
## [1] 9333   68
```

```
fourth.quartile = subset(df,((df$Runtime >= summary(df$Runtime)[5]) & (df$Runtime < summary(df$Runtime)
dim(fourth.quartile)
```

```
## [1] 8885   68
```

```
p1 <- ggplot(first.quartile, aes(x = Runtime, y = Gross))+
    geom_point()+
    ggtitle('Films less than 69 min')+
    scale_y_log10()+
    xlab("Runtime")+
    ylab("Gross")
p2 <- ggplot(second.quartile, aes(x = Runtime, y = Gross))+
    geom_point()+
    ggtitle('Films between 69 and 90 min')+
    scale_y_log10()+
    xlab("Runtime")+
```

```
    ylab("Gross")
p3 <- ggplot(third.quartile, aes(x = Runtime, y = Gross))+
    geom_point()+
    ggtitle('Films between 90 and 101 min')+
    scale_y_log10()+
    xlab("Runtime")+
    ylab("Gross")
p4 <- ggplot(fourth.quartile, aes(x = Runtime, y = Gross))+
    geom_point()+
    geom_smooth()+
    ggtitle('Films more than 101 min')+
    scale_y_log10()+
    scale_x_log10()+
    xlab("Runtime")+
    ylab("Gross")
grid.newpage()
pushViewport(viewport(layout = grid.layout(2, 2)))
vplayout <- function(x, y) viewport(layout.pos.row = x, layout.pos.col = y)
print(p1, vp = vplayout(1, 1))
print(p2, vp = vplayout(1, 2))
print(p3, vp = vplayout(2, 1))
print(p4, vp = vplayout(2, 2))
```

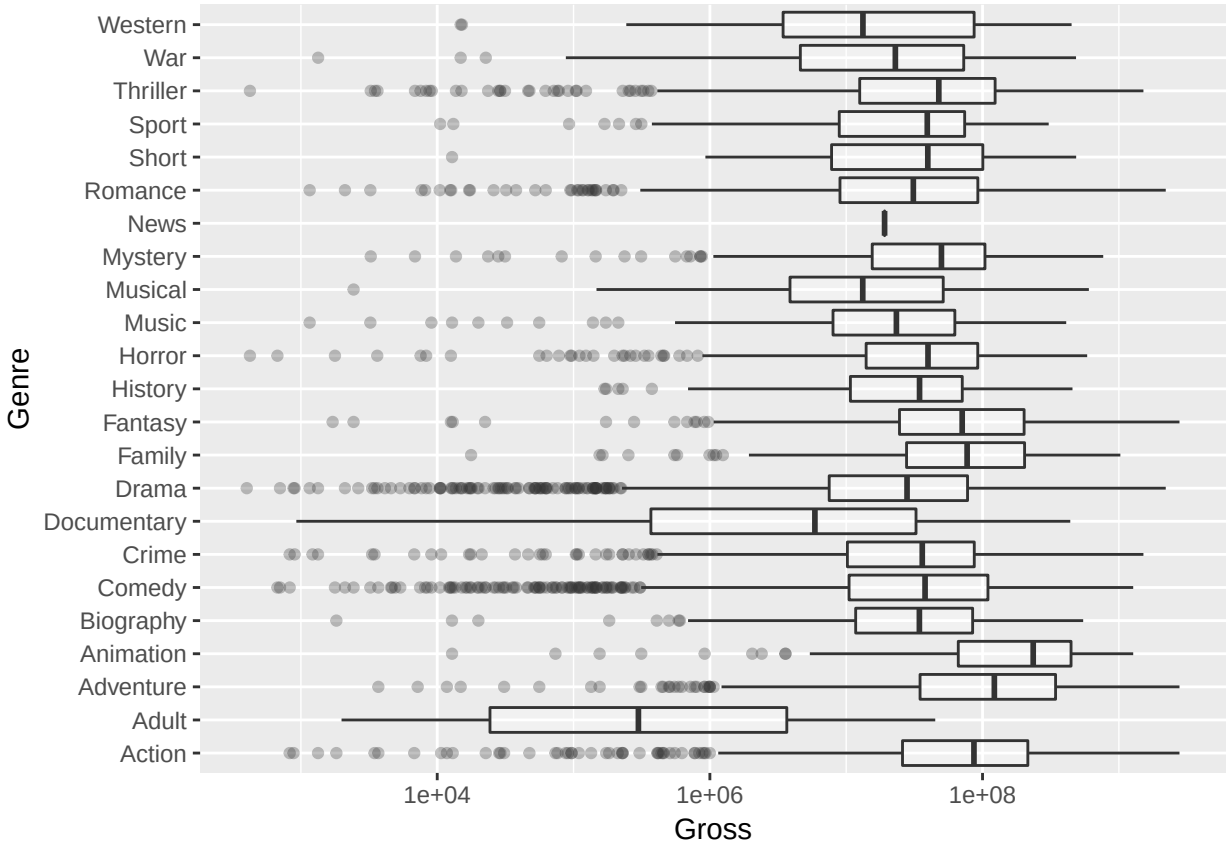



```
# TODO: Investigate if Gross Revenue is related to Genre

# Representation of relationship between Gross Revenue and Genre through boxplots
temp.genre.bis = data.frame(Gross = df$Gross,df[,list.genre.bis])
```

```
data.genre.plus.gross = data.frame(Genre = rep((list.genre.bis)[1],length(temp.genre.bis$Gross[which(ter

for (i in (seq(2,length(list.genre.bis)))){
data3 = data.frame(Genre = rep((list.genre.bis)[i],length(temp.genre.bis$Gross[which(temp.genre.bis[[(li
data.genre.plus.gross = rbind(data.genre.plus.gross,data3)
}

p <- ggplot(data.genre.plus.gross, aes(reorder(Genre, Gross, median), Gross))
p + geom_boxplot(alpha = 0.3) +
    coord_flip() +
    scale_y_log10()+
    scale_x_discrete("Genre")
```

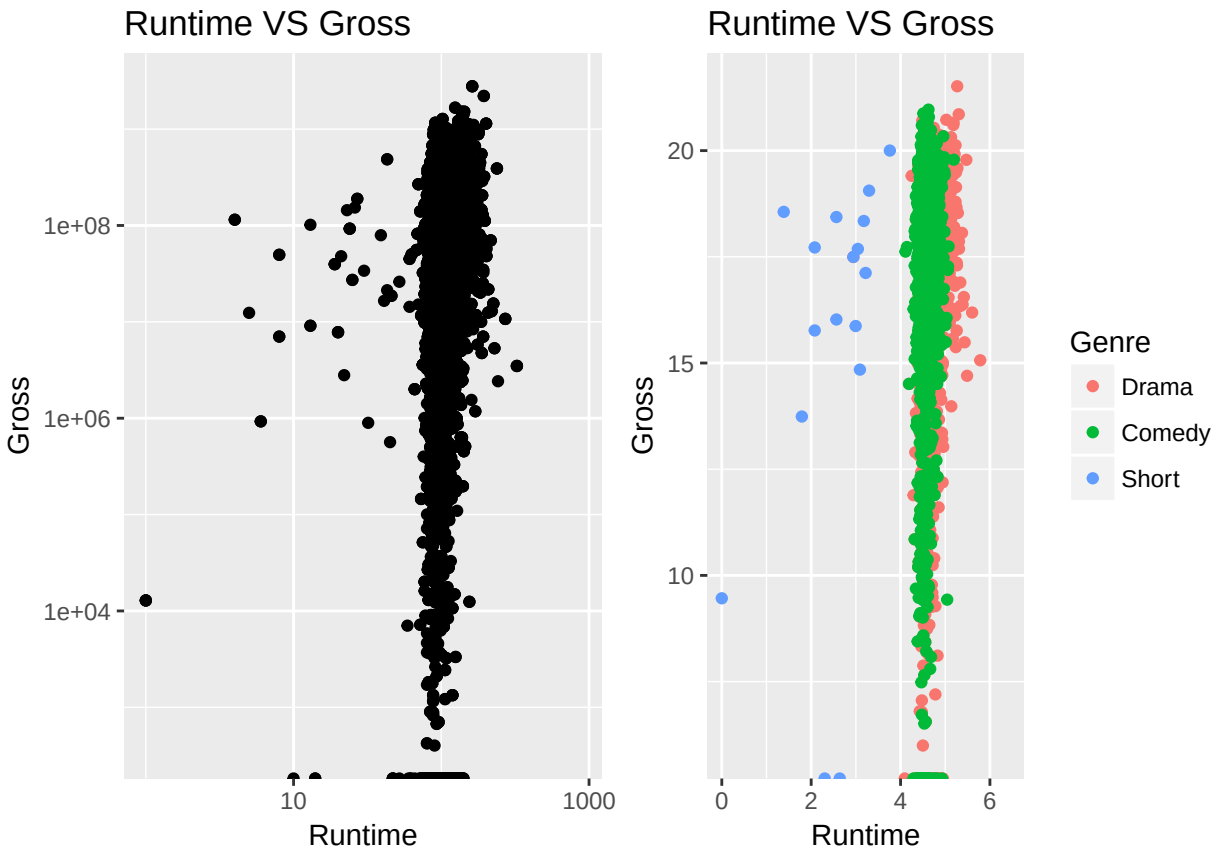
```
## Warning: Transformation introduced infinite values in continuous y-axis
```

```
## Warning: Removed 60206 rows containing non-finite values (stat_boxplot).
```



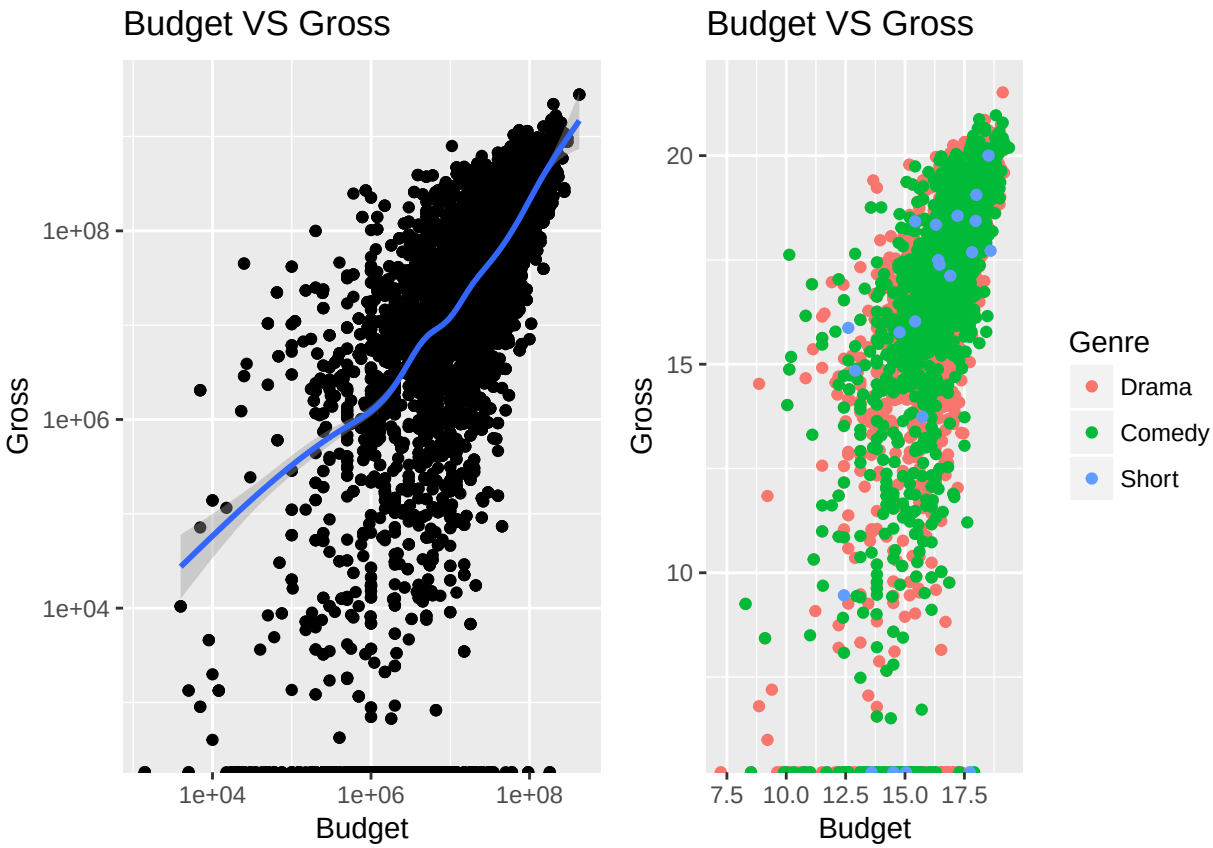

```
# Representation of relationship between Gross Revenue and Runtime through scatter plots

# Gross Revenue against Runtime
temp.genre.bis = data.frame(Gross = df$Gross, Runtime = df$Runtime,df[,list.genre.bis])

data.runtime.plus.gross.plus.genre = data.frame(Genre = rep((list.genre.bis)[1],length(temp.genre.bis$Gr

for (i in (seq(2,length((list.genre.bis))))){
data3 = data.frame(Genre = rep((list.genre.bis)[i],length(temp.genre.bis$Gross[which(temp.genre.bis[[(li
data.runtime.plus.gross.plus.genre = rbind(data.runtime.plus.gross.plus.genre,data3)
```

```
}

p1 <- ggplot(data = data.runtime.plus.gross.plus.genre, aes(x = Runtime, y = Gross ))+
     geom_point() +
     scale_x_log10()+
     scale_y_log10()+
     ggtitle("Runtime VS Gross")

# Gross Revenue against Runtime for 3 genres

data.runtime.plus.gross.plus.drama = data.frame(Genre = rep((list.genre.bis)[9],length(temp.genre.bis$G

data.runtime.plus.gross.plus.comedy = data.frame(Genre = rep((list.genre.bis)[6],length(temp.genre.bis$

data.runtime.plus.gross.plus.short = data.frame(Genre = rep((list.genre.bis)[24],length(temp.genre.bis$

data.runtime.plus.gross.plus.drama.comedy.short = rbind(data.runtime.plus.gross.plus.drama,
                                                        data.runtime.plus.gross.plus.comedy,
                                                        data.runtime.plus.gross.plus.short)
p2 <- qplot(x = log(Runtime),
      y = log(Gross),
      data = data.runtime.plus.gross.plus.drama.comedy.short,
      color = Genre,
      main = "Runtime VS Gross",
      xlab = "Runtime",
      ylab = "Gross")

grid.newpage()
pushViewport(viewport(layout = grid.layout(1, 2)))
vplayout <- function(x, y) viewport(layout.pos.row = x, layout.pos.col = y)
print(p1, vp = vplayout(1, 1))
print(p2, vp = vplayout(1, 2))
```




```
# Gross Revenue against Budget

temp.genre.bis = data.frame(Gross = df$Gross, Budget = df$Budget,df[,list.genre.bis])

data.Budget.plus.gross.plus.genre = data.frame(Genre = rep((list.genre.bis)[1],length(temp.genre.bis$Gr

for (i in (seq(2,length((list.genre.bis))))){
data3 = data.frame(Genre = rep((list.genre.bis)[i],length(temp.genre.bis$Gross[which(temp.genre.bis[[(l
data.Budget.plus.gross.plus.genre = rbind(data.Budget.plus.gross.plus.genre,data3)
}

p1 <- ggplot(data = data.Budget.plus.gross.plus.genre, aes(x = Budget, y = Gross ))+
     geom_point() +
     geom_smooth()+
     scale_x_log10()+
     scale_y_log10()+
     ggtitle("Budget VS Gross")

# Gross Revenue against Budget for 3 genres

data.Budget.plus.gross.plus.drama = data.frame(Genre = rep((list.genre.bis)[9],length(temp.genre.bis$Gr

data.Budget.plus.gross.plus.comedy = data.frame(Genre = rep((list.genre.bis)[6],length(temp.genre.bis$G

data.Budget.plus.gross.plus.short = data.frame(Genre = rep((list.genre.bis)[24],length(temp.genre.bis$G

data.Budget.plus.gross.plus.drama.comedy.short = rbind(data.Budget.plus.gross.plus.drama,
```

```
                                                    data.Budget.plus.gross.plus.comedy,
                                                    data.Budget.plus.gross.plus.short)
p2 <- qplot(x = log(Budget),
      y = log(Gross),
      data = data.Budget.plus.gross.plus.drama.comedy.short,
      color = Genre,
      main = "Budget VS Gross",
      xlab = "Budget",
      ylab = "Gross")

grid.newpage()
pushViewport(viewport(layout = grid.layout(1, 2)))
vplayout <- function(x, y) viewport(layout.pos.row = x, layout.pos.col = y)
print(p1, vp = vplayout(1, 1))
print(p2, vp = vplayout(1, 2))
```



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.

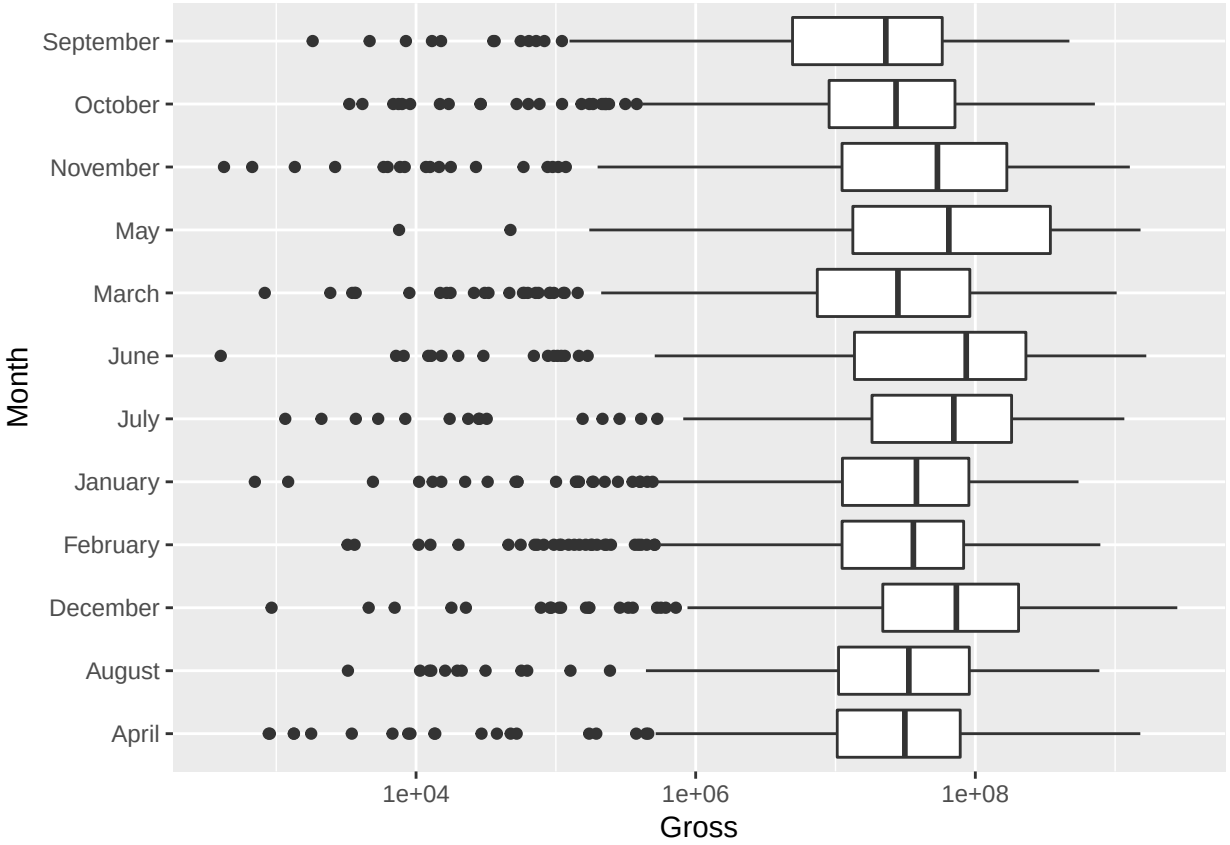
A: Facing the issue of finding relationships between Budget/Runtime/Genre that result in high Gross revenue, the first method that came to my mind was to plot a pair-wise graph as ggpairs provides one. But its lack of clarity convinced me to pursue with independent graphs.

The boxplot between genre and gross shows that the Gross is the same for almost all genres, since the medians are all positionned around the same value except for Adult films and Film Noirs which are not really beneficial according to their low median and documentary which spread is particularly high compared to the others categories.

In order to compare Budget to Gross, a log-log scale transformation was applied in order to visualize some model, we can see that a linear model for high value is present which is logical that huge budget provides a lot of publicity to the film and then gross whereas the little budget's films has more random success. To visualize the difference between genres, the same scatter plot is made according to the 3 most commons genres (drama,comedy and short). The values for drama and comedy are such the same that they overlap each other. So genre have no effect on the distribution of gross according to budget.

The comparison between Runtime and Gross was first realized brutely. The log-log scale representation provides us the conclusion that runtime is constant, the variations according to the same three genre than previously shows us that the "outliers" present correspond to short films which explained the small duration. In order to deepen this analysis, we partition runtime into intervals. To make statistical analysis easier, quartiles were chosen as breaks of intervals. The quartiles (between 69 and 90 mins and between 90 and 101 mins) just prove that the no matter the runtime, the range of values of Gross are the same but if we group these two quartiles we could see that the trend is increasing. Indeed the longer the film is, the larger the gross is, this observation remains lonly between 69 and 101 mins. The first quartile can't add information and the last quartile shows that after having reached huge gross for films greater than 101 mins, the too long films have less success because they are too length and sometime less attractive to the audience.

```
# TODO: Investigate if Gross Revenue is related to Release Month
df.rating = df[is.na(df$Released) == FALSE,]
new_month = c()
for (i in 1:length(df.rating$Released)){
    new_month = c(new_month,released.in.month(df.rating$Released[i]))
}
list_mois = c("January","February","March","April","May","June","July","August","September","October","
for (i in seq(1,12)){new_month[which(new_month == i)] = list_mois[i]}

data.gross.plus.month = data.frame(Gross = df.rating$Gross,Month = new_month )


p <- ggplot(data.gross.plus.month, aes(reorder(Month, -Gross, median), Gross))
p + geom_boxplot() +
    coord_flip() +
    scale_y_log10()+
    scale_x_discrete("Month")
```



A. The Boxplot of Gross realized over the months show that the Gross is rather contant all the year, but we can observe that there is an increase at the end of the year and around the summer months (June). It seems logical since a lot of boxoffice movies (Star Wars for example) made their release around December.

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.

```
# TODO: Convert Awards to 2 numeric columns: wins and nominations

# Convert to lowercase
df$Awards = tolower(df$Awards)

# Select all the patterns "win" and "wins" and return the number associated
wins = as.numeric(str_match(df$Awards, "(\\d*) (w)")[,2])
wins[is.na(wins) == TRUE] = 0

# Select the pattern "won "and return the number associated
more_wins = as.numeric(str_match(df$Awards, "won (\\d*)")[,2])
more_wins[is.na(more_wins) == TRUE] = 0
```

```
# Sum the two previous lists
wins = wins + more_wins
length_wins = length(wins[which(wins != 0)])
sprintf("The number of valid/non zeros wins is %i",length_wins)
```

```
## [1] "The number of valid/non zeros wins is 9380"
```

```
# Select all the patterns "nomination" and "nominations" and return the number associated
nominations = as.numeric(str_match(df$Awards, "(\\d*) (n)")[,2])
nominations[is.na(nominations) == TRUE] = 0

# Select the pattern "nominated for "and return the number associated
more_nominations = as.numeric(str_match(df$Awards, "nominated for (\\d*)")[,2])
more_nominations[is.na(more_nominations) == TRUE] = 0

# Sum the two previous lists
nominations = nominations + more_nominations


length_nominations = length(nominations[which(nominations != 0)])
length_wins_nominations = length((which(nominations != 0 | wins != 0)))
sprintf("The number of valid/non zeros nominations is %i",length_nominations)
```

```
## [1] "The number of valid/non zeros nominations is 9867"
```

```
sprintf("The number of valid/non zeros nominations or wins is %i",length_wins_nominations)
```

```
## [1] "The number of valid/non zeros nominations or wins is 12725"
```

```
df$Wins = wins
df$Nominations = nominations
```

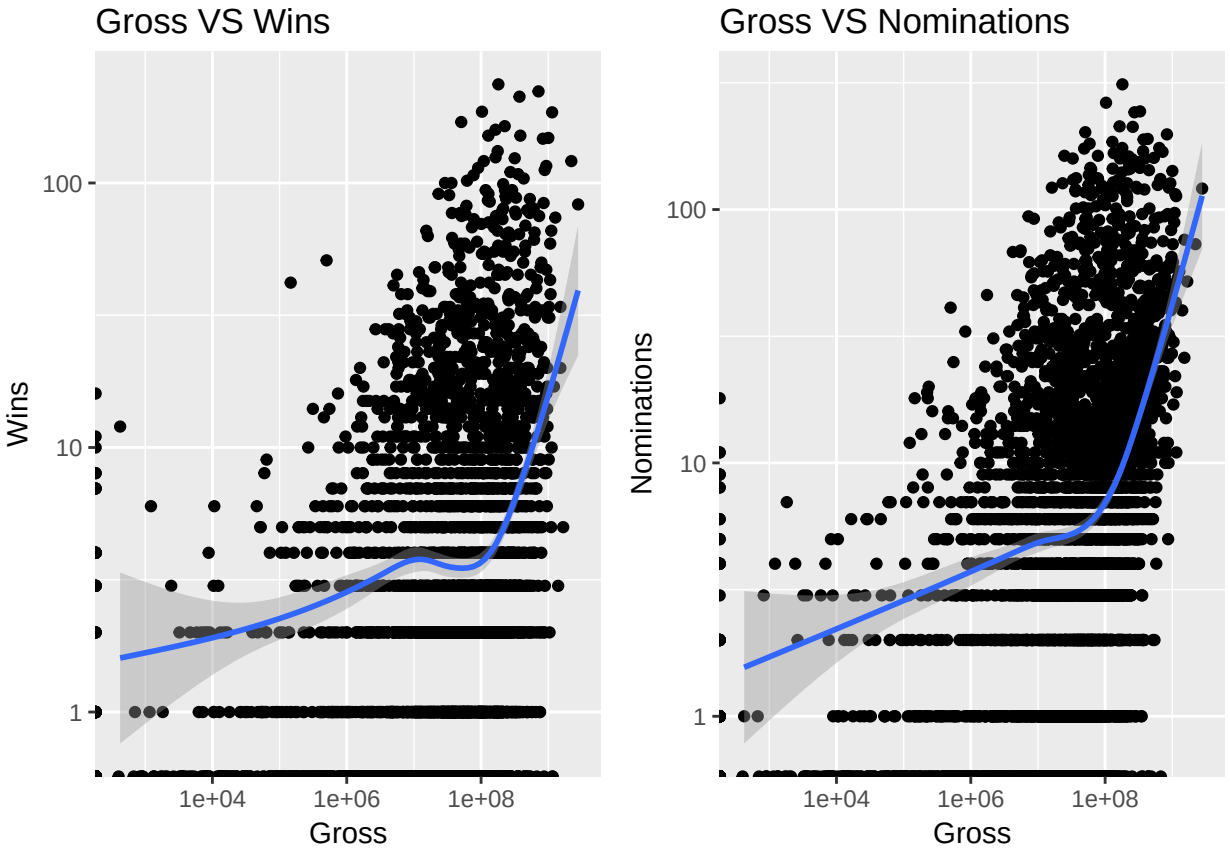
Q: How did you construct your conversion mechanism? How many rows had valid/non-zero wins or nominations?

A: The conversion mechanism was enabled by the package "stringr", it handles regular expressions indeed it allows to recover automatically all the patterns beginning by a number and followed by 'w' or 'n' and returns the number. These two cases were stored in different columns named respectively wins and nominations. As other frames were present, two others regular expressions were created allowing to recover the number of others nominations and wins not included in the previous numbers. These were stored in more_wins and more_nominations. In order to provide figures for global wins and global nominations, wins was summed with more wins and the same was done for nominations. Concerning the films which had no wins or nominations, they were filled with zeros.

The number of rows which had valid/non zero wins or nominations is 12725.

```
# TODO: Plot Gross revenue against wins and nominations

Gross.plus.wins.nominations = data.frame(Gross = df$Gross,Wins = df$Wins, Nominations = df$Nominations)
p1 <- ggplot(Gross.plus.wins.nominations,aes(x = Gross ,y = Wins)) +
      geom_point() +
      geom_smooth() +
      ggtitle('Gross VS Wins') +
      xlab("Gross") +
      ylab ("Wins") +
      scale_x_log10() +
      scale_y_log10()
p2 <- ggplot(Gross.plus.wins.nominations,aes(x = Gross ,y = Nominations)) +
```

```
      geom_point() +
      geom_smooth() +
      ggtitle('Gross VS Nominations') +
      xlab("Gross") +
      ylab ("Nominations") +
      scale_x_log10() +
      scale_y_log10()

grid.newpage()
pushViewport(viewport(layout = grid.layout(1, 2)))
vplayout <- function(x, y) viewport(layout.pos.row = x, layout.pos.col = y)
print(p1, vp = vplayout(1, 1))
print(p2, vp = vplayout(1, 2))
```



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

A: The two scatter plots show that neither the number of wins or nominations are related to Gross. The difference between those two graphs only stands in the larger number of nominations over wins. We can observe that when the number of wins or nominations is high, the Gross associated is high also which can be explained by the fact that when films earn awards, it leads the audience to watch films and then make Gross increasing. Whereas, films whith low gross do not follow any model, they can earn awards or not.

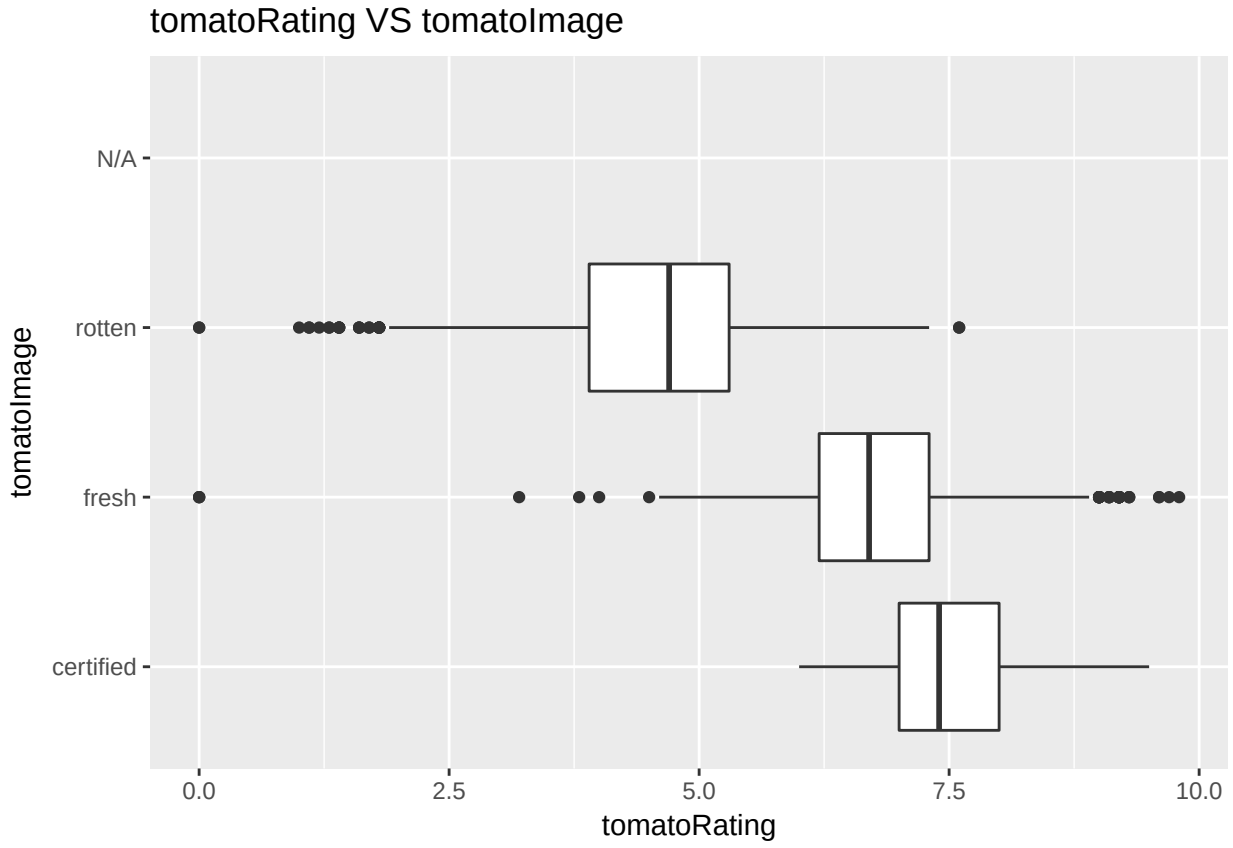
7. Movie ratings from IMDb and Rotten Tomatoes

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

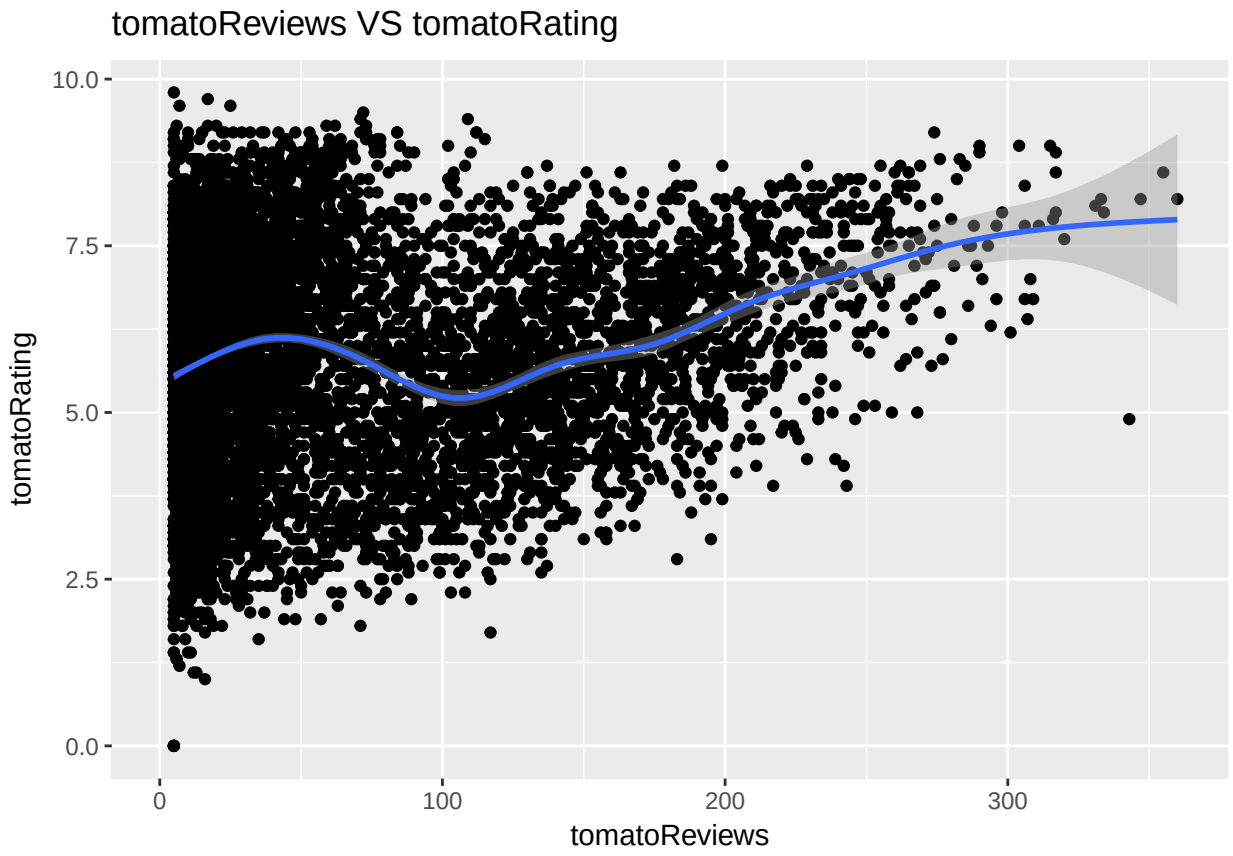
(represented by several variables pre-fixed by `tomato`). Read up on such ratings on the web (for example rottentomatoes.com/about and www.imdb.com/help/show_leaf?votestopfaq).

Investigate the pairwise relationships between these different descriptors using graphs.

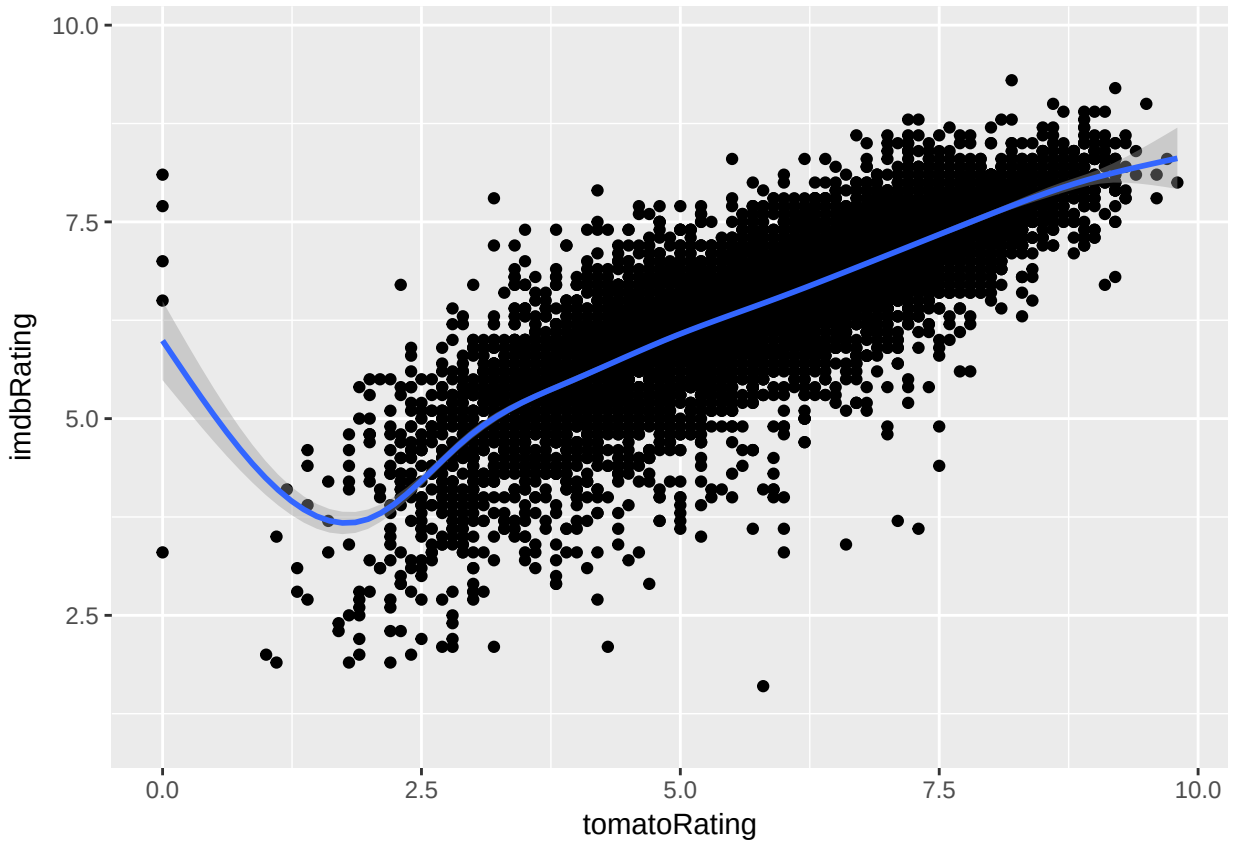
```
# TODO: Illustrate how ratings from IMDb and Rotten Tomatoes are related

df.rating = df[which(df$tomatoImage != 'N/A'),]
tomato.plus.imdb = data.frame(tomatoImage = df$tomatoImage,tomatoRating =  df$tomatoRating, imdbRating =

ggplot(tomato.plus.imdb, aes(reorder(tomatoImage, -tomatoRating, median), tomatoRating))+
    geom_boxplot() +
    coord_flip() +
    scale_x_discrete("tomatoImage")+
    ggtitle('tomatoRating VS tomatoImage')
```




```
ggplot(tomato.plus.imdb, aes(x= tomatoReviews, y = tomatoRating))+
    geom_point() +
    geom_smooth() +
    ggtitle('tomatoReviews VS tomatoRating')
```



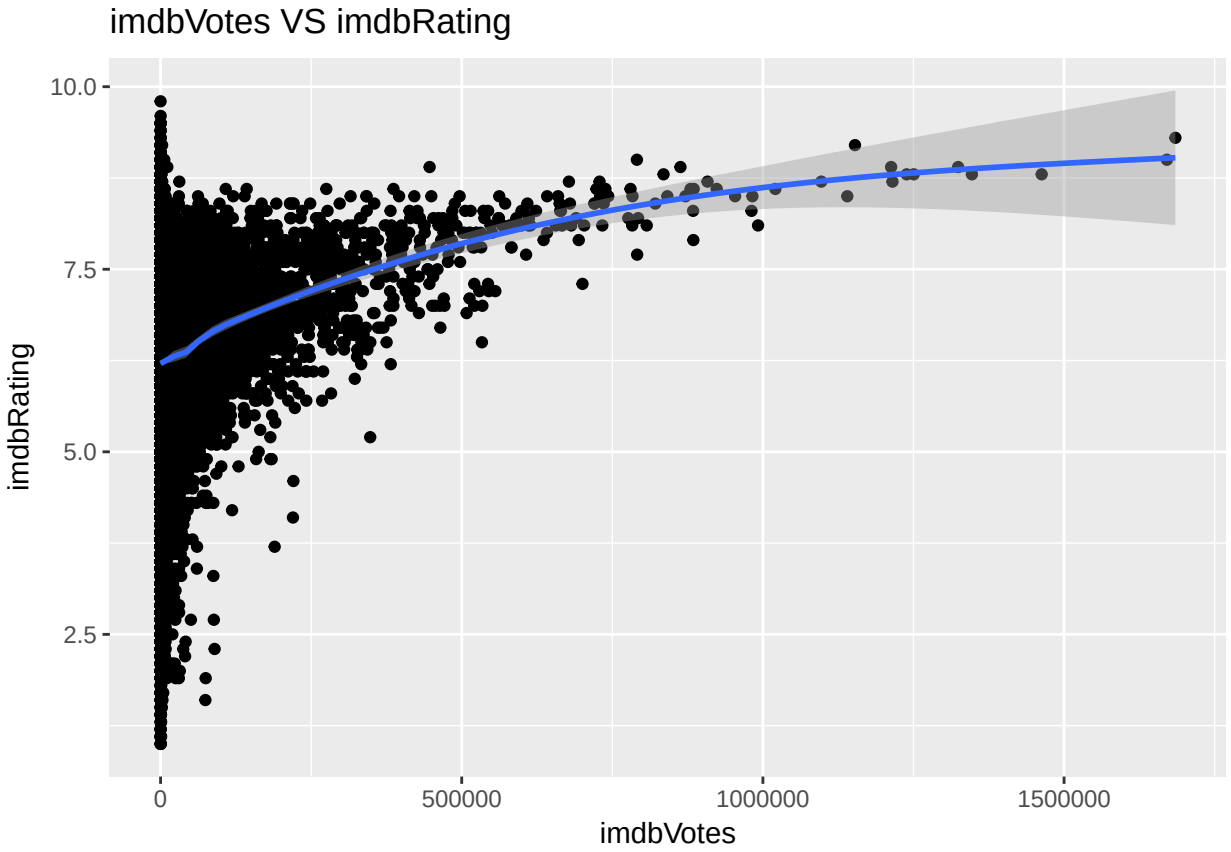

```
ggplot(tomato.plus.imdb, aes(x= tomatoRating, y = imdbRating))+
    geom_point() +
    geom_smooth()
```




```
    ggtitle('imdbRating VS tomatoRating')
```

```
## $title
## [1] "imdbRating VS tomatoRating"
##
## $subtitle
## NULL
##
## attr(,"class")
## [1] "labels"
```

```
ggplot(tomato.plus.imdb, aes(x= imdbVotes, y = imdbRating))+
    geom_point() +
    geom_smooth()+
    ggtitle('imdbVotes VS imdbRating')
```



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

A: The boxplot between the classification of tomatoImage and the rating given by tomato users provides us the insights given on the website of tomatos which said that a film is qualified as fresh if the rating is above 60% and is certified if the rating is above 75%. Furthermore, we can observe that the spread of rotten is larger than others, which show that this category is much more subjective than others.

The scatter plots which represents the relationships between the reviews and the rating as tomatoReviews and tomatoRating (resp. imdbVotes VS imdbRating) shows how biased or not are these ratings. Indeed we can see that the one for imdb demonstrates a biased point of view since the films that are the best graded are the ones for which the audience has voted the most. Whereas, the scatter plot for tomato is far more homogeneous even if the trend is an increasing one. This observation can be explained by the fact that the ratings are provided by professional critics on tomato whereas everyone can vote on imdb.

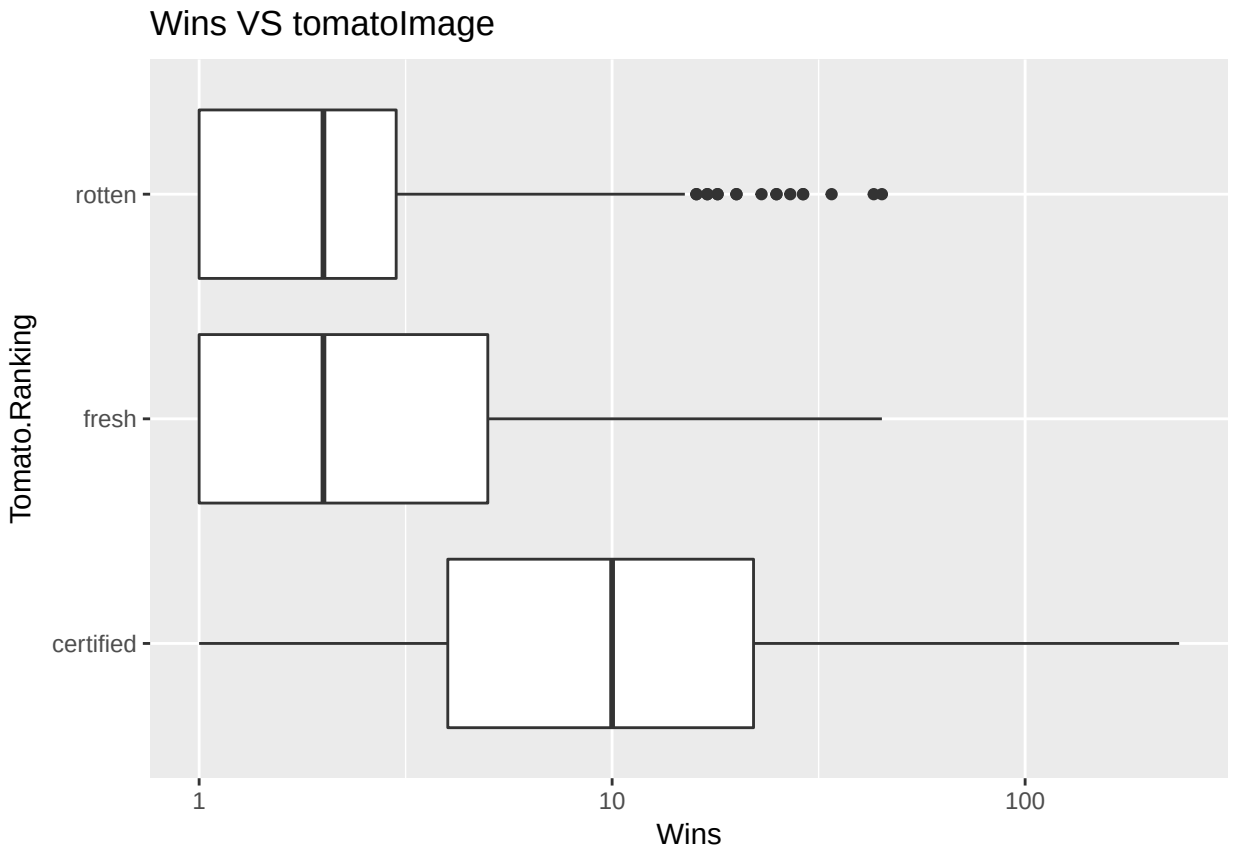
The scatter plot representing imdbRating and tomatoRating shows a linear relationship, which proves that the both are consistent with one another. Therefore, the advice of the large audience of imdb is consistent with the professional critics' conclusions.

8. Ratings and awards

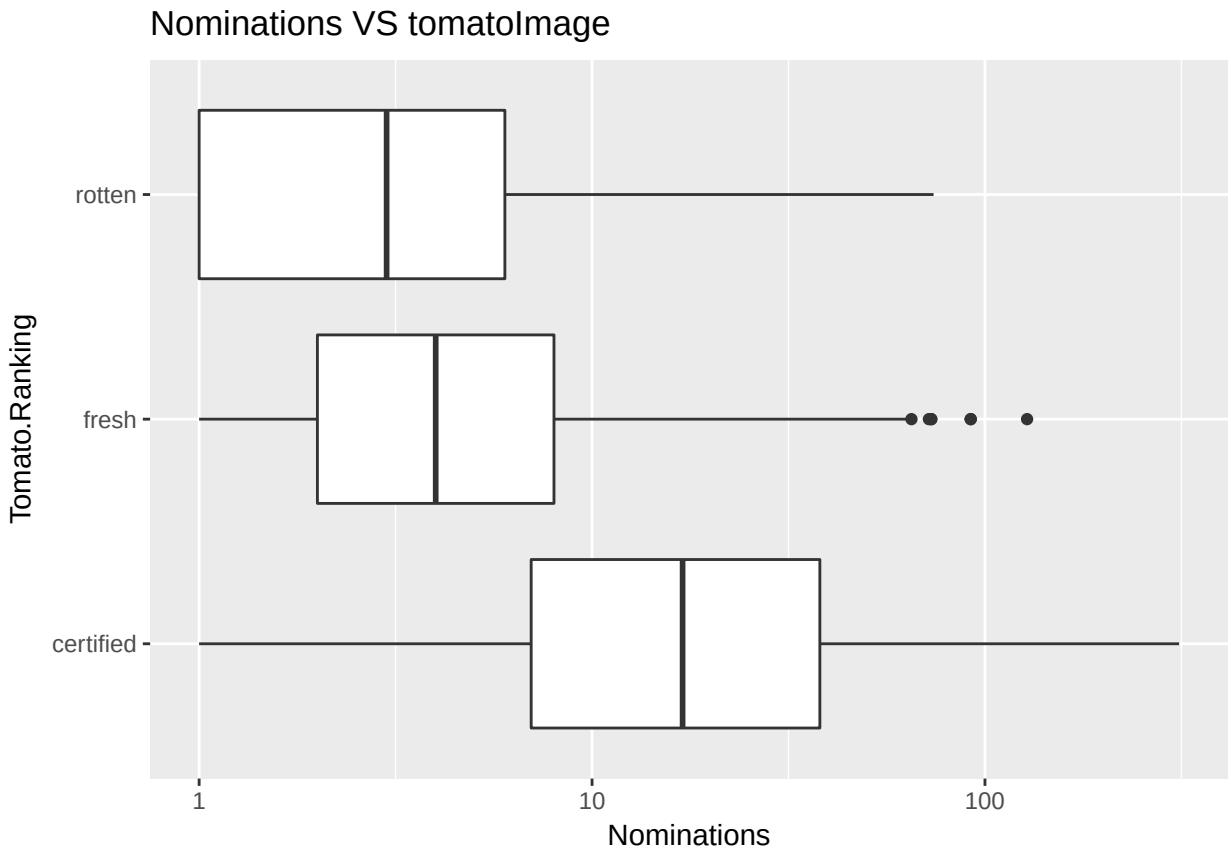
These ratings typically reflect the general appeal of the movie to the public or gather opinions from a larger body of critics. Whereas awards are given by professional societies that may evaluate a movie on specific attributes, such as artistic performance, screenplay, sound design, etc.

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

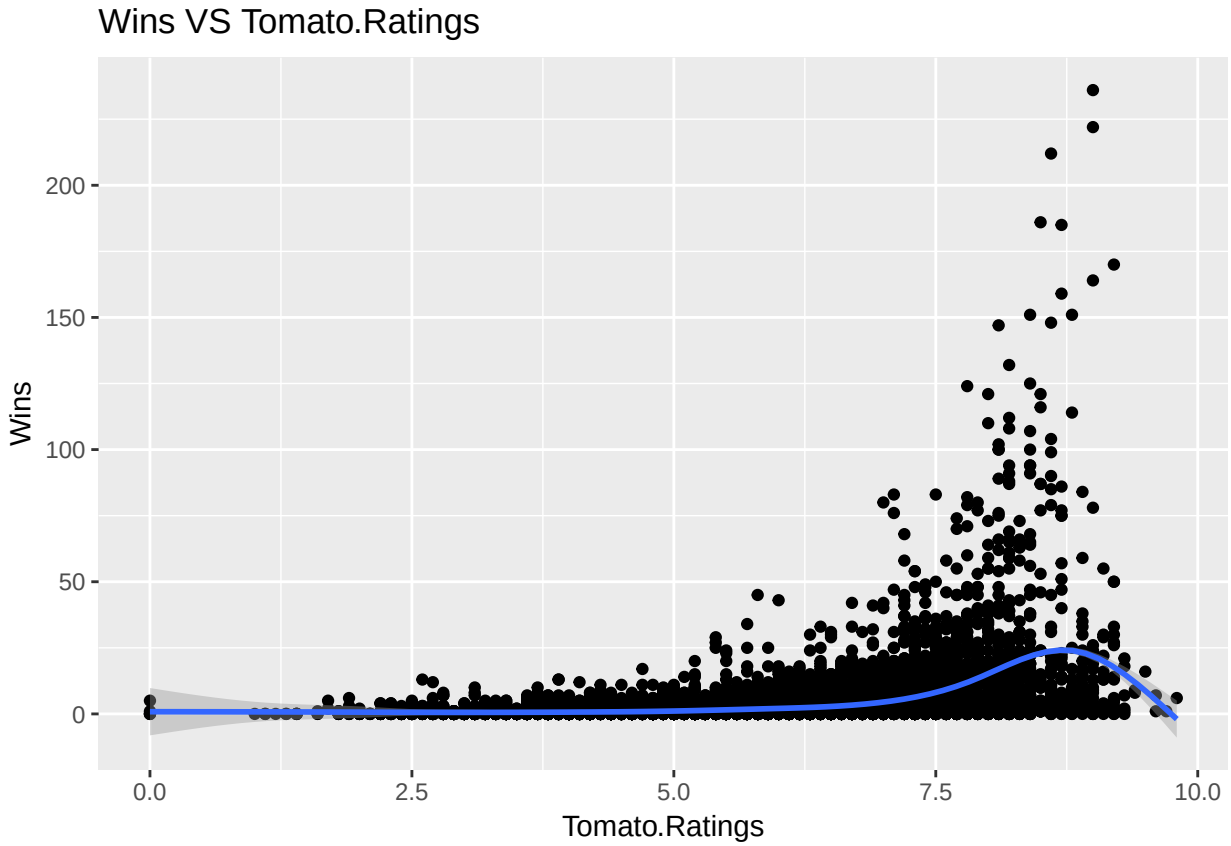
```r
# TODO: Show how ratings and awards are related

df.rating = df[which(df$tomatoImage != 'N/A'),]

data.ratings.plus.wins.nominations = data.frame(Tomato.Ratings = df.rating$tomatoRating, Wins = df.ratin

ggplot(data.ratings.plus.wins.nominations, aes(reorder(Tomato.Ranking, -Wins, median), Wins))+
    geom_boxplot() +
    coord_flip() +
    scale_y_log10()+
    scale_x_discrete("Tomato.Ranking")+
    ggtitle('Wins VS tomatoImage')
```



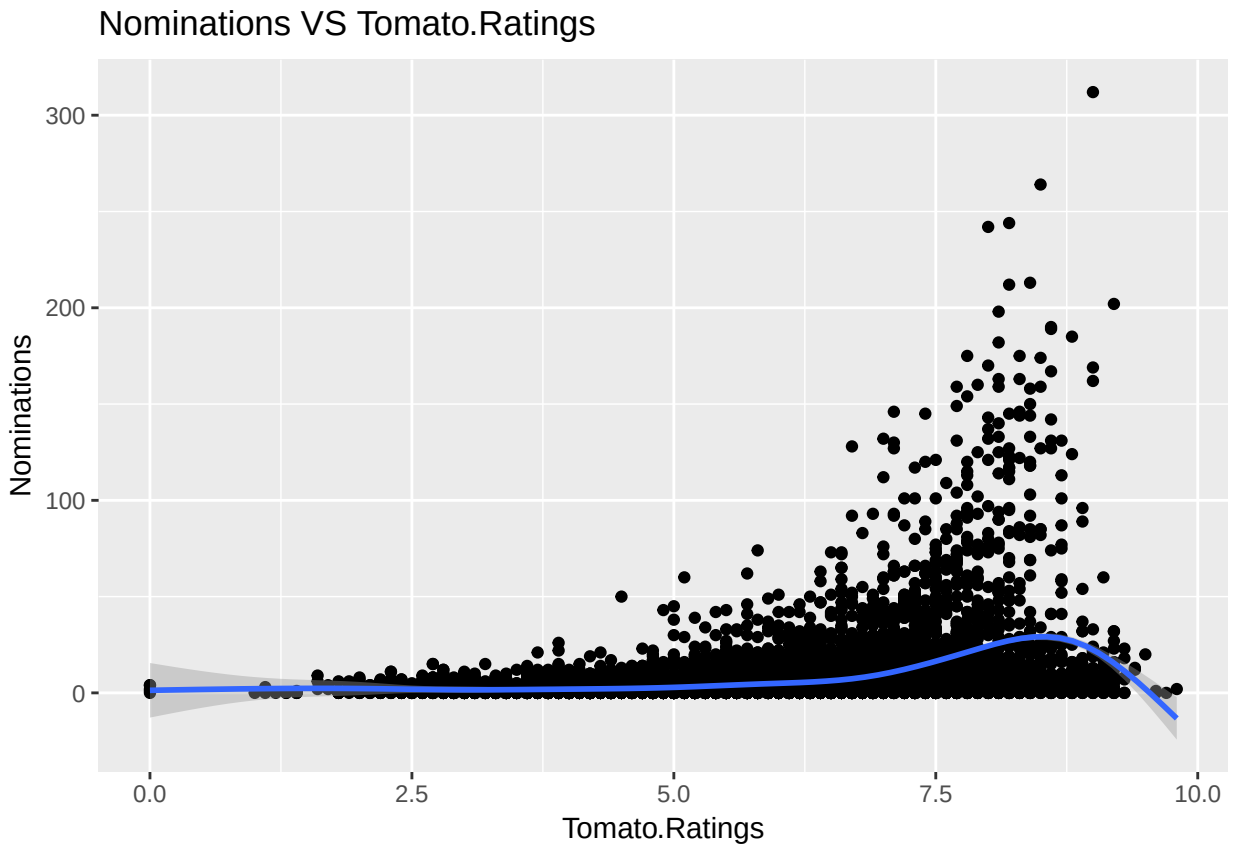

```r
ggplot(data.ratings.plus.wins.nominations, aes(reorder(Tomato.Ranking, -Nominations, median), Nominatio
    geom_boxplot() +
    coord_flip() +
    scale_y_log10()+
    scale_x_discrete("Tomato.Ranking")+
    ggtitle('Nominations VS tomatoImage')
```



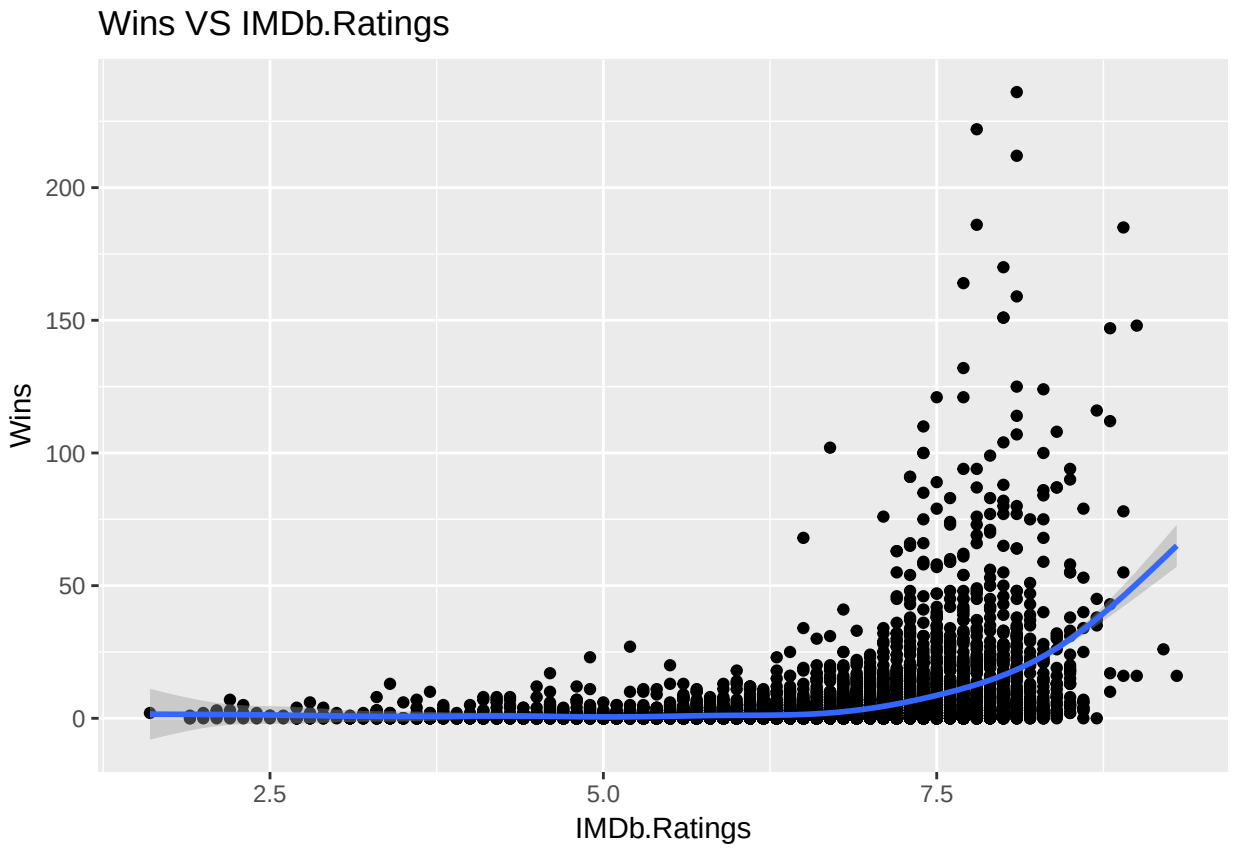

```
ggplot(data.ratings.plus.wins.nominations, aes(x= Tomato.Ratings, y = Wins))+
    geom_point() +
    geom_smooth()+
    ggtitle('Wins VS Tomato.Ratings')
```



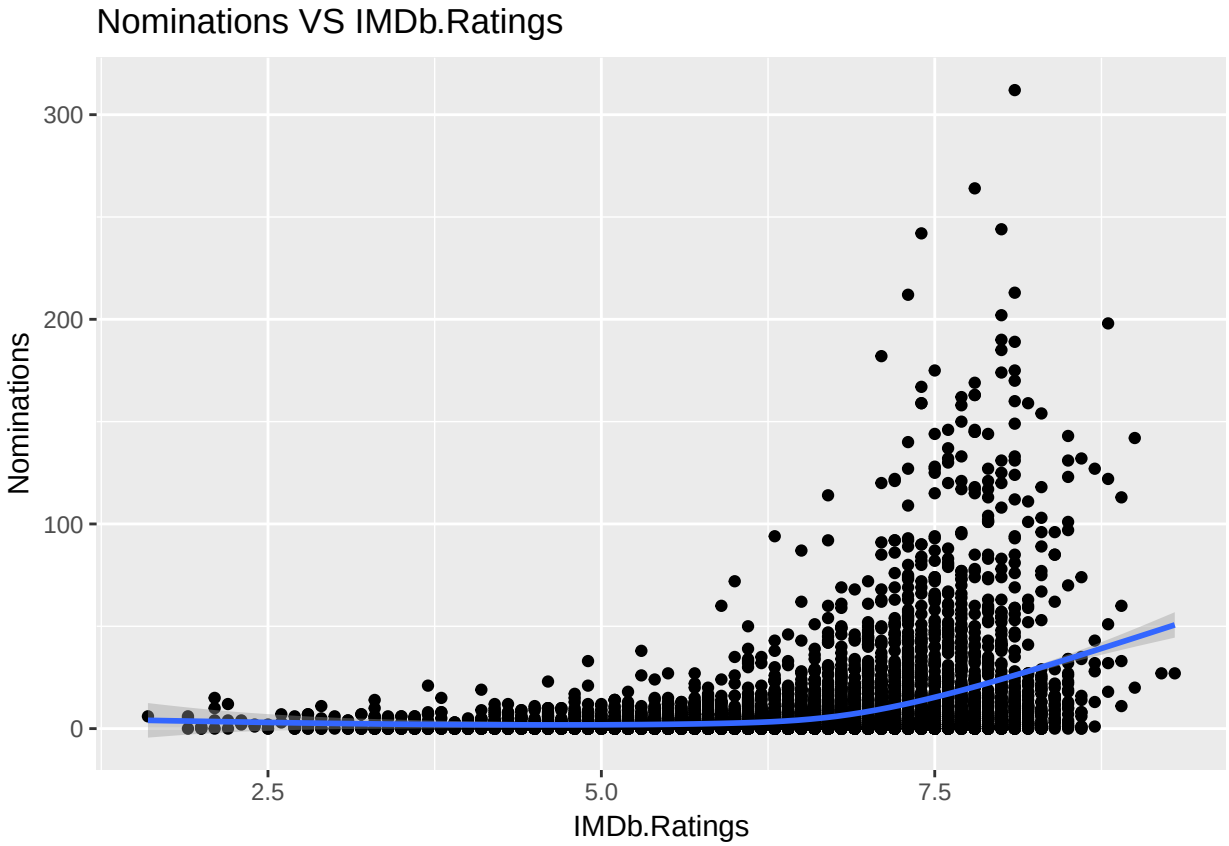

```
ggplot(data.ratings.plus.wins.nominations, aes(x= Tomato.Ratings, y = Nominations))+
    geom_point() +
    geom_smooth()+
    ggtitle('Nominations VS Tomato.Ratings')
```




```
ggplot(data.ratings.plus.wins.nominations, aes(x= IMDb.Ratings, y = Wins))+
    geom_point() +
    geom_smooth()+
    ggtitle('Wins VS IMDb.Ratings ')
```




```
ggplot(data.ratings.plus.wins.nominations, aes(x= IMDb.Ratings, y = Nominations))+
    geom_point() +
    geom_smooth()+
    ggtitle('Nominations VS IMDb.Ratings ')
```



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: The boxplot between wins and tomato ranking (that is classification rotten/fresh/certified) is consistent since the medians are ordered according to the number of wins. The quartiles of Certified Films are all positionned with more than 5 wins. Despite their rating, rotten films have a median of 3, which could mean that critics are severe sometimes.

The boxplot between nominations and tomato ranking leads to the same conclusions than previously but the numbers are higher.

The scatter plots between nominations/wins and both of ratings demonstrate no particular relationship, so the correlation is almost unexistent, we can only notice that the films which earn many wins/nominations have a good rating.

9. Expected insights

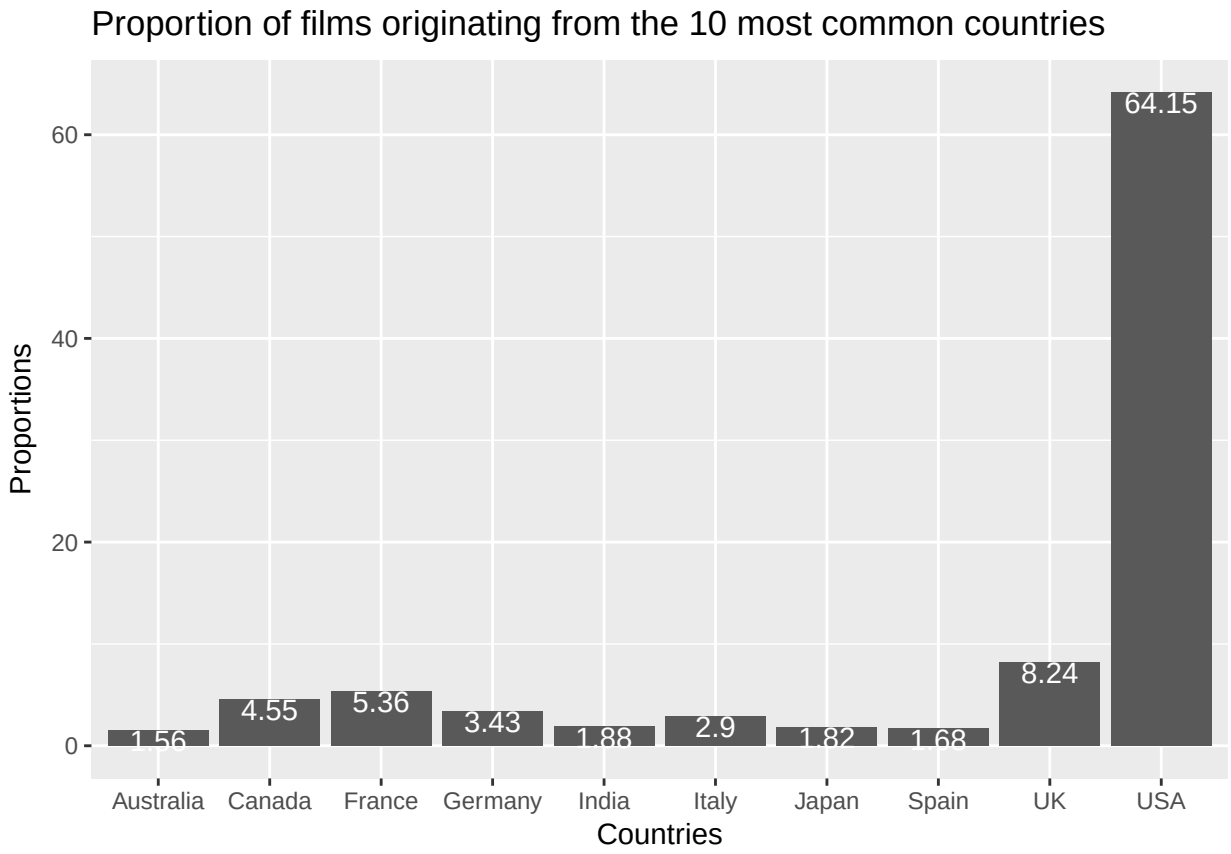
Come up with two new insights (backed up by data and graphs) that is expected. Here ???new??? means insights that are not an immediate consequence of one of the above tasks. You may use any of the columns already explored above or a different one in the dataset, such as `Title`, `Actors`, etc.

```
# TODO: Find and illustrate two expected insights

# First Insight : Countries

# Transform in binaries lists

# Suppress spaces in Countries
```

```r
df$Country = gsub(" ", "", df$Country)

list_countries = c()
temp_countries = c()

for (i in seq(1,length(df$Country))) {
  temp_countries = unlist(strsplit(df$Country[i],","))
  list_countries = c(list_countries,temp_countries)
}

duplicated.values = which(duplicated(list_countries))
list.countries.bis = list_countries[-duplicated.values]
list.countries.bis = sort(list.countries.bis)

map = matrix(0,length(df$Country),length(list.countries.bis))

for (j in seq(1,length(df$Country))){

  temp_countries = unlist(strsplit(df$Country[j],","))

  for (k in seq(1,length(temp_countries))){
    index = which(temp_countries[k] == list.countries.bis)
    map[j,index] = 1
  }
}

dim_old = (dim(df)[2])+1
data_frame = data.frame(map)
names(data_frame) = list.countries.bis
df = cbind(df,data_frame)
```

```r
# Find the ten most common countries and plot their relative proportions

list = lapply(df[,dim_old:length(df)],sum)
list = list[rev(order(sapply(list,'[[',1)))]
countries.10.first = head(list,10)
countries.10.first = as.data.frame(countries.10.first)
data_total = as.data.frame(list)
sum_total = dim(df)[1]
countries.10.first.relative = (countries.10.first/sum_total)*100
temp = c()
for (i in seq(1,length(countries.10.first))){
  temp = c(temp,countries.10.first.relative[,i])
}

data_frame.countries.10.first.relative = data.frame(Countries = names(countries.10.first),proportions =
```

```r
ggplot(data=data_frame.countries.10.first.relative, aes(x=data_frame.countries.10.first.relative$Countr
geom_bar(stat="identity")+
geom_text(aes(label= round(data_frame.countries.10.first.relative$proportions,2)), vjust=1, colour="whi
ggtitle("Proportion of films originating from the 10 most common countries") +
xlab("Countries") +
ylab("Proportions")
```

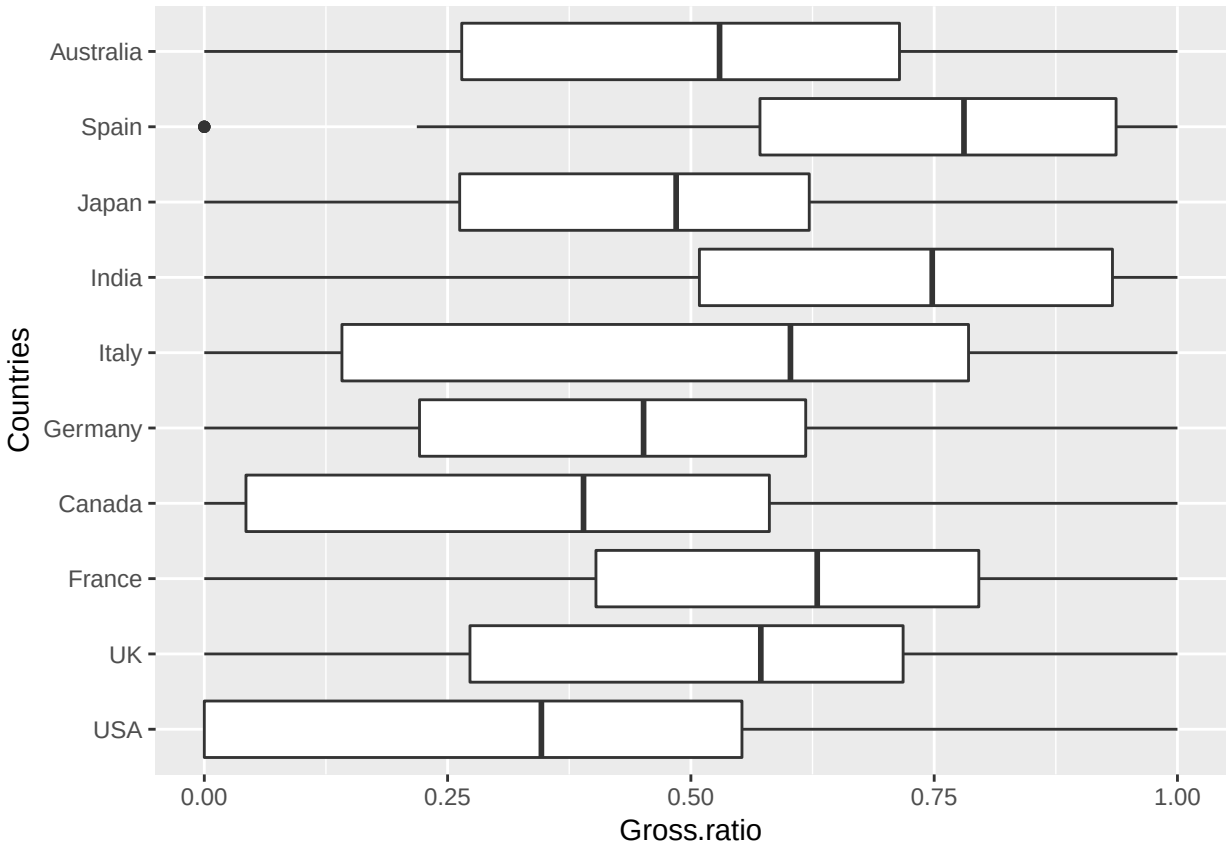


Q: Expected insight #1.

A: As we classified films according to genres and we found that the two most common genre were drama and comedy (expected features), the classification according to the tenth most common countries is interesting. The process of conversion into binary lists is the same as for genres.

The barcharts show that USA has the most important proportion and that the second one is UK, which were expected features, since most of BoxOffices are from USA and UK. Those proportions are totally consistent with this dataset, the ten countries are also the countries which are the most known in the cinematographic world, so no surprise in those results.

```
# Second Insight

# Calculate the "international" ratio
df$Gross_ratio = (df$Gross - df$Domestic_Gross)/df$Gross

temp_countries = data.frame(Gross_ratio = df$Gross_ratio,df[,names(countries.10.first)])

data.gross_ratio.plus.countries = data.frame(Countries = rep(names(countries.10.first)[1],length(temp_c

for (i in (seq(2,length(names(countries.10.first))))){
data2 = data.frame(Countries = rep(names(countries.10.first)[i],length(temp_countries$Gross_ratio[which
data.gross_ratio.plus.countries = rbind(data.gross_ratio.plus.countries,data2)
}

p <- ggplot(data.gross_ratio.plus.countries, aes(reorder(Countries, -Gross.ratio, median), Gross.ratio)
p + geom_boxplot() +
    coord_flip() +
    scale_x_discrete("Countries")
```



Q: Expected insight #2.

A: An interesting insight is to observe how the gross evolves according to countries and especially how the international gross evolves. Indeed, starting from Gross and Domestic Gross we can deduce the ratio of Gross that was earned outside of the country.

The resulting boxplot leads to unexpected conclusions at first sight because of the ratios obtained in Spain and USA. Indeed we observe that USA has International Gross Ratio the less important of all and Spain the more important which is after some thoughts quite logical. USA has a domestic gross much more important than the international gross (around 70% domestic gross) since the sucess of american films is assured by americans. Whereas for India it's the opposite since Bollywood has much more sucess internationally because all indian people can't afford to watch their own movies and Bollywood is one of the biggest film industries in the world in terms of the number of people employed and the number of films produced. The ratio concerning Spain is also explanable because of all the spanish speaking countries (South America) which are considered as international Gross. This observation worth also for french speaking country which assures the international consumption of french films.

10. Unexpected insight

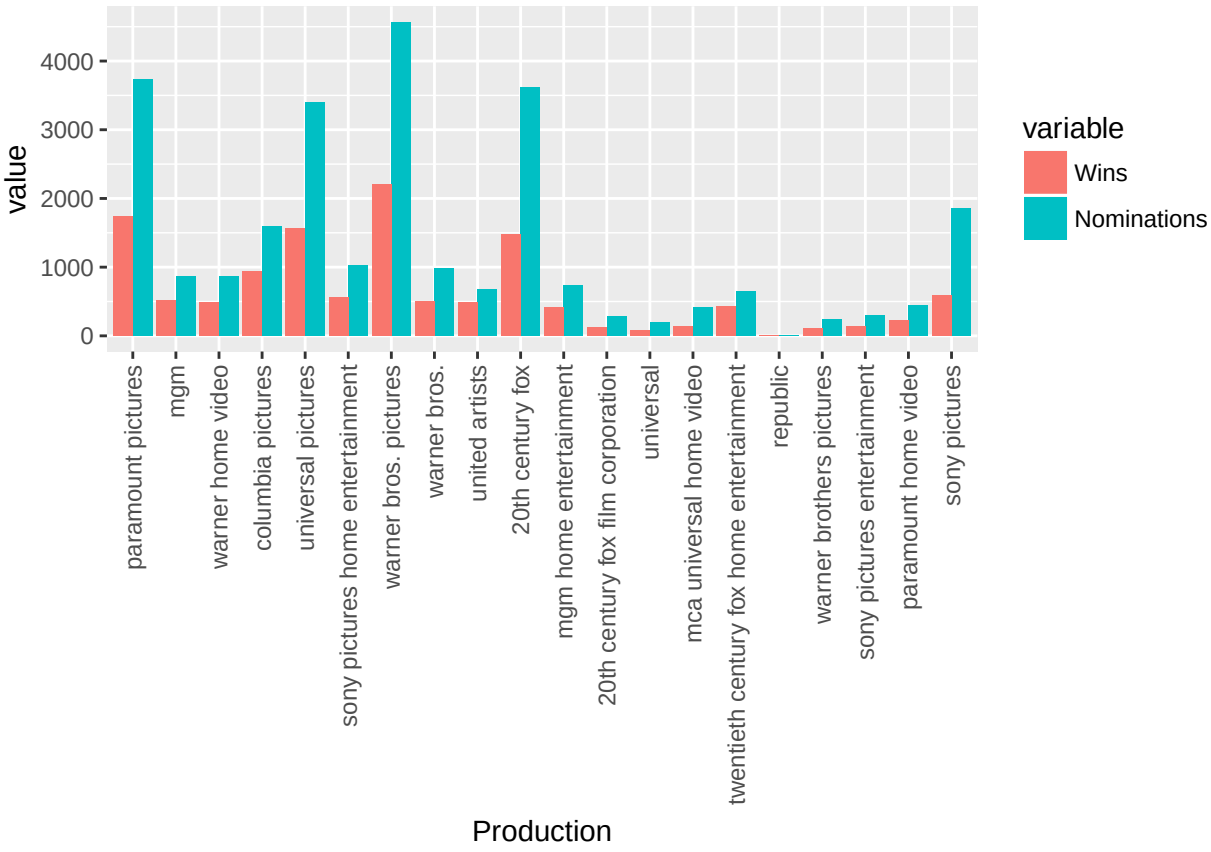
Come up with one new insight (backed up by data and graphs) that is unexpected at first glance and do your best to motivate it. Same instructions apply as the previous task.

```
# TODO: Find and illustrate one unexpected insight

df$Production = tolower(df$Production)

data.Production = as.data.frame(table(df$Production))
data.Production = data.Production[with(data.Production, order(-Freq)), ]
```

```r
# Select the most frequent 20 studios
data.frame.20.first = head(data.Production,21)
data.frame.20.first = data.frame.20.first[-1,]
data.frame.20.first
```

```
##                                        Var1 Freq
## 1976                      paramount pictures  679
## 1683                                     mgm  555
## 2795                    warner home video  450
## 621                      columbia pictures  449
## 2681                    universal pictures  444
## 2404       sony pictures home entertainment  425
## 2782                 warner bros. pictures  409
## 2779                          warner bros.  388
## 2659                        united artists  362
## 3                        20th century fox  356
## 1688                mgm home entertainment  312
## 7         20th century fox film corporation  284
## 2674                             universal  209
## 1641              mca universal home video  205
## 2634 twentieth century fox home entertainment  190
## 2187                               republic  159
## 2792              warner brothers pictures  158
## 2403           sony pictures entertainment  155
## 1975                 paramount home video  139
## 2400                       sony pictures  131
```

```r
df.20.first = df[df$Production %in% data.frame.20.first$Var1,]
df.20.first = df.20.first[,c("Production","Wins","Nominations")]


df.20.first = ddply(df.20.first, .(Production), summarise, Wins=sum(Wins),Nominations=sum(Nominations))
df.20.first$Production <- factor(df.20.first$Production, levels=data.frame.20.first$Var1)

# melt the data frame for plotting
data.m <- melt(df.20.first, id.vars='Production')

# plot everything
ggplot(data.m, aes(x = Production, y = value)) +
  geom_bar(aes(fill = variable), position = "dodge", stat="identity")+
  theme(axis.text.x=element_text(angle=90,hjust=1,vjust=0.5))
```



Q: Unexpected insight.

A: Giving another glance to attributes, it seemed to me that Production was an interesting feature to exploit. The combination betwen wins/nominations and Production could seem expected. So I combine on the barplot above, the frequency of each studio (number of films produced by each studio) and the number of wins/nominations associated to each studio. The order from left to right is consistent with the order of frequency of studio. Hence paramout picture has a frequency of 679 films, mgm of 555, etc. But we can observe that whereas warner bros. pictures has produced 409 films the number of wins and nominations are the most important for this studio and not for paramount picture. So the law of big numbers does not apply here. The number of films produced is not related to the number of wins/nominations.