Project 1: Explore and Prepare Data

CSE6242 - Data and Visual Analytics - Fall 2017 Due: Sunday, October 15, 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 on the same dataset, and will be released at a later date. Both projects will have equal weightage towards your grade. You may reuse some of the preprocessing/analysis steps from Project 1 in Project 2.

Name: Shijie Shi

GaTech ID: SSHI48

Email: shijie.shi@gatech.edu

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.

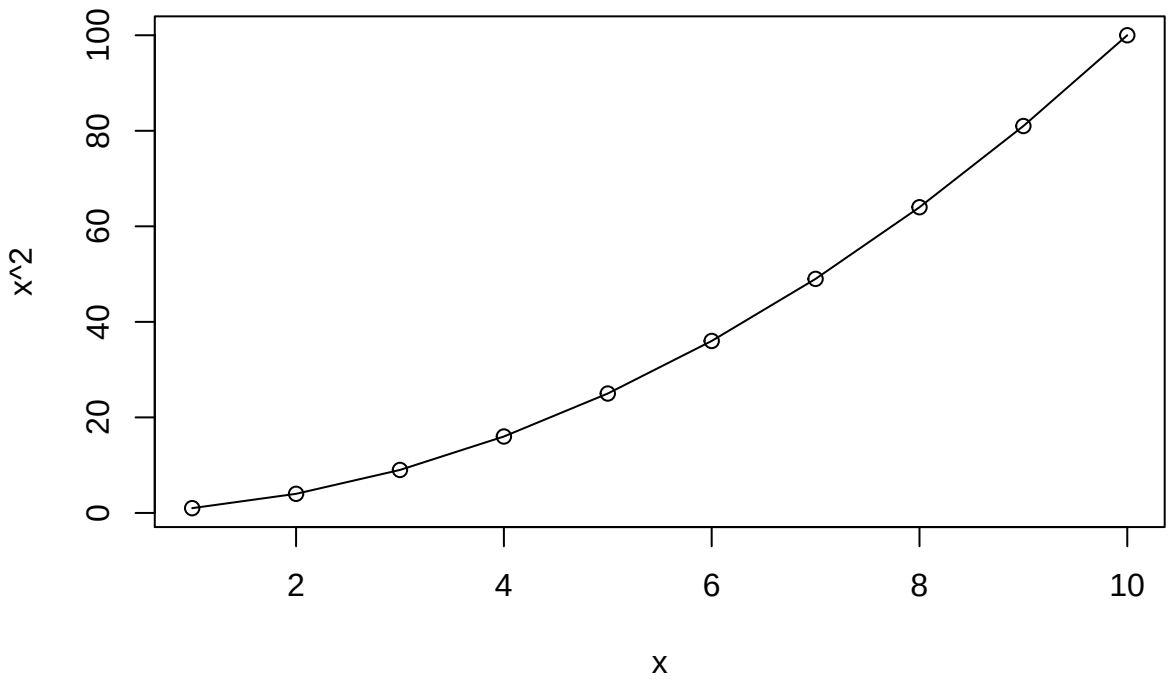
When you execute code within the notebook, the results appear beneath the code. Try executing this chunk by clicking the *Run* button within the chunk or by placing your cursor inside it and pressing *Cmd+Shift+Enter*.

```
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*. Enter some R code and run it.

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 all the tasks below by implementing code chunks that have a `TODO` comment in them, running all code chunks so that output and plots are displayed, and typing in answers to each question (**Q:** ...) next to/below the corresponding answer prompt (**A:**). Feel free to add code chunks/show additional output to support any of the answers.

When you are done, you will need to submit the final R markdown file (as **pr1.Rmd**) with all code chunks implemented and executed, and all text responses written in. You also need to submit a PDF export of the markdown file (as **pr1.pdf**), which should show your code, output, plots and written responses–this will be your project report. Compress these two files into a single .zip archive and upload it on T-Square.

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

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

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("Column names:", end="\n", file="")
```

```
## Column names:
```

```
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)
# install.packages("GGally")
library(GGally)
library(tm)
```

```
## Loading required package: NLP
```

```
##
## Attaching package: 'NLP'
```

```
## The following object is masked from 'package:ggplot2':
##
##     annotate
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following object is masked from 'package:base':
##
##     date
```

```
library(stringr)
```

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 okay 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
df2 <- df[df$Type == "movie",]
dim(df2)
```

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

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

A: 40000 rows are left after the removal.

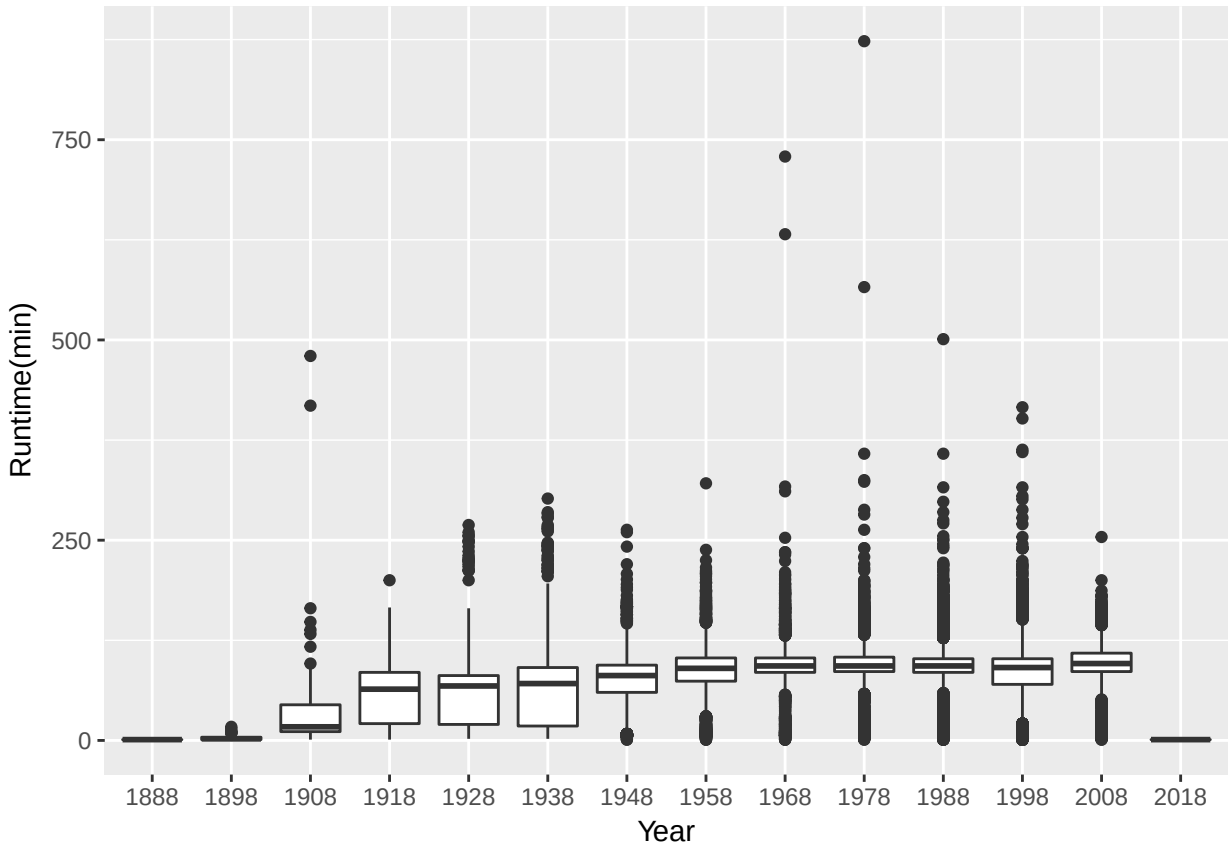
2. Process `Runtime` column

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.

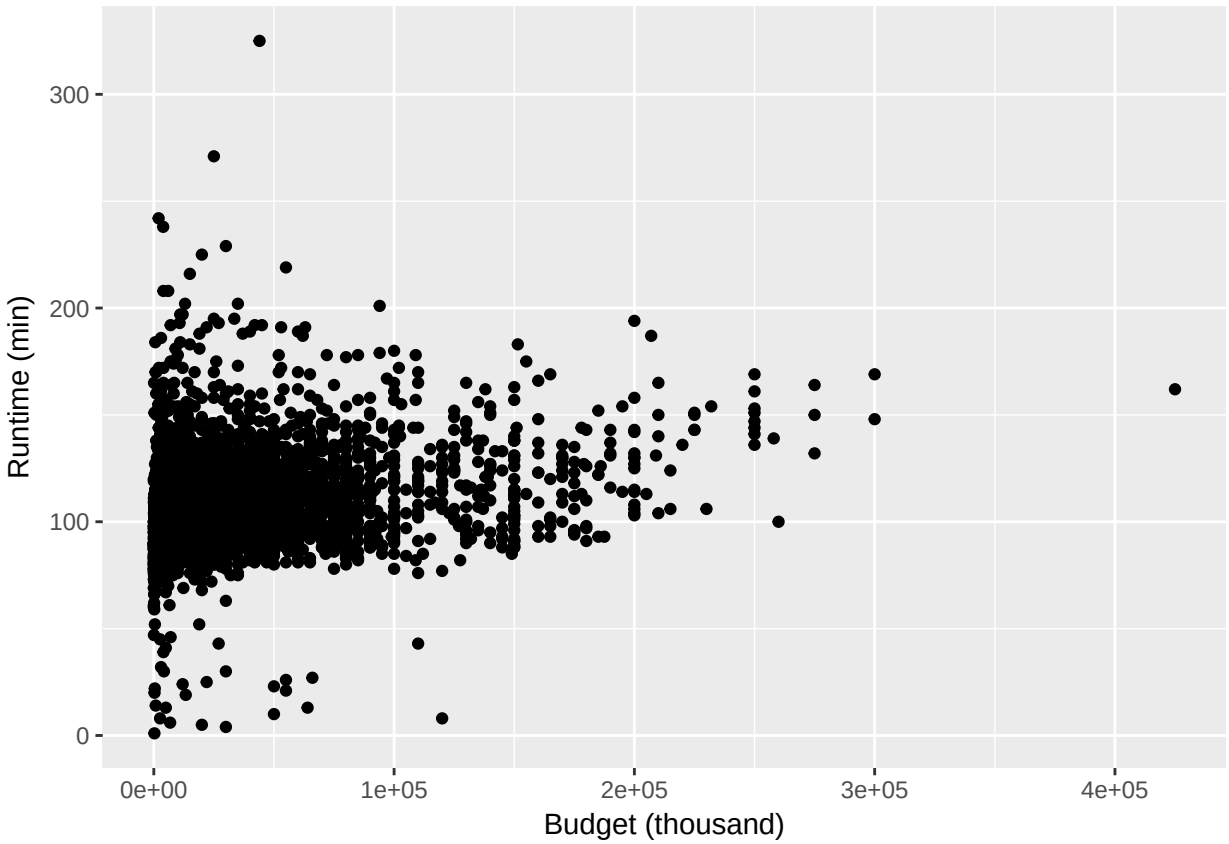
```
# TODO: Replace df$Runtime with a numeric column containing the runtime in minutes
conver_runtime <- function(str1){
  time <- gsub(" min","", str1)
  time <- strsplit(gsub(' h','', time), " ")[[1]]
  time <- rev(time)
  if( is.na(time[2]) ){
    time[2]=0
  }
  return(as.numeric(time[1])+as.numeric(time[2])*60)
}
df2$Runtime <- suppressWarnings(sapply(df2$Runtime,FUN=conver_runtime))
```

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.

```
# TODO: Investigate the distribution of Runtime values and how it varies by Year and Budget
# 1. Year and Runtime
year_runtime = data.frame(df2$Year, df2$Runtime)
year_runtime <- na.omit(year_runtime)
year_runtime$bin <- cut(year_runtime$df2.Year, seq(1888, 2018, by=10))

ggplot(year_runtime) + geom_boxplot(aes(bin, df2.Runtime), na.rm=TRUE) +  xlab("Year") +ylab("Runtime(m
```




```
# 2. Budget and Runtime
budget_runtime = data.frame(df2$Budget, df2$Runtime)
budget_runtime <- na.omit(budget_runtime)

ggplot(budget_runtime)  + geom_point(aes(df2.Budget/1000, df2.Runtime))+ guides(fill=FALSE) +  xlab("Bu
```



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:
Runtime and Year: Movies in late 1800 and early 1900 were shorter compare to nowwadays. Also, the lengths of old moview varies a lot. The movie industry seemed to be standardized since 1960s. Most of the mordern movies' lengths are a little less than 200 minutes. Runtime and Budget: I did not observe any relationship between runtime and budget.

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.

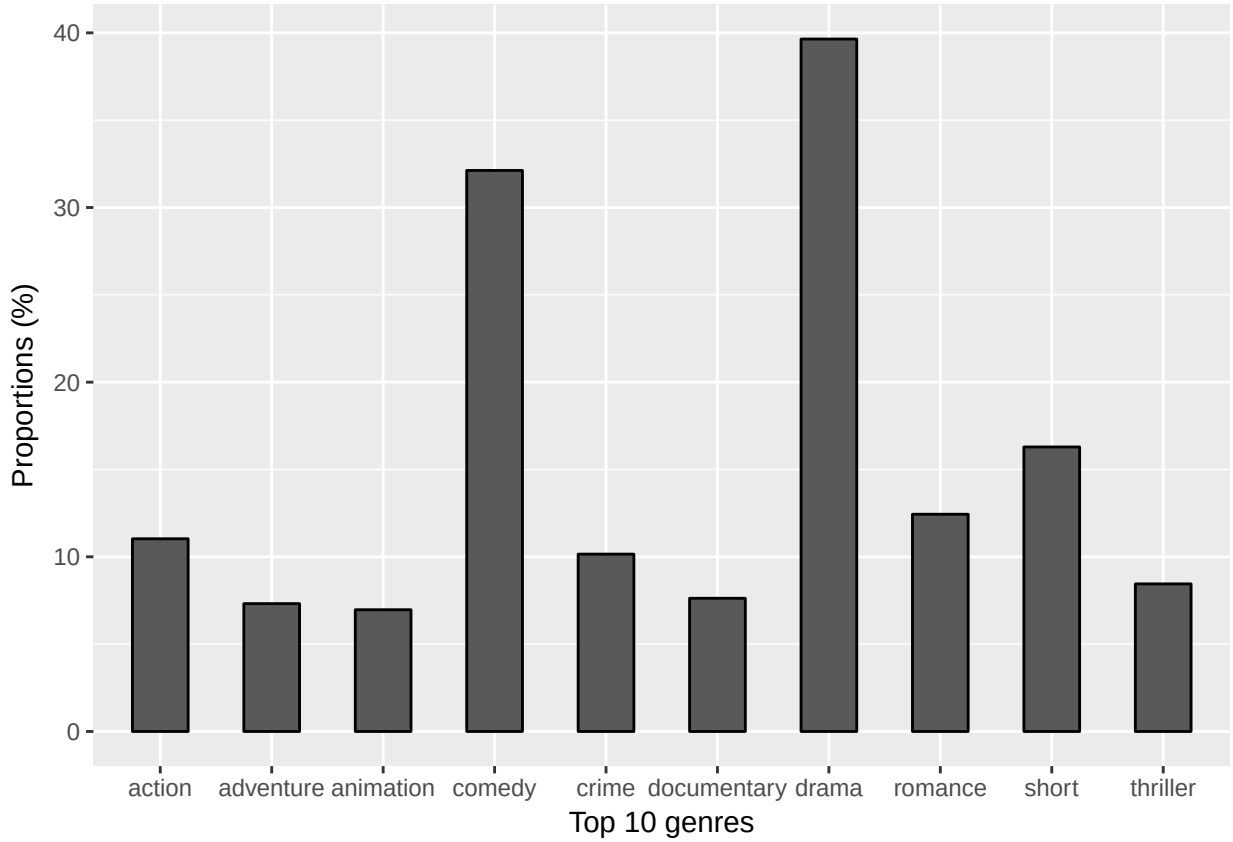
For example, if there are a total of 3 genres: Drama, Comedy, and Action, a movie that is both Action and Comedy should be represented by a binary vector <0, 1, 1>. Note that you need to first compile a dictionary of all possible genres and then figure out which movie has which genres (you can use the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

```
# TODO: Replace Genre with a collection of binary columns
data <- Corpus(VectorSource(df2$Genre))
dataTM<- DocumentTermMatrix( tm_map(data, removePunctuation)  )

binaryMatrix <- function(x) {
  r <- nrow(x)
```

```
  c <- ncol(x)
  y <- matrix(vector(typeof(x$v), 1 * r * c), r, c)
  y[cbind(x$i, x$j)] <- x$v
  dimnames(y) <- x$dimnames
  return(y)
}

genre_binary <- as.data.frame( binaryMatrix(dataTM) )
df2 <- cbind(df2, genre_binary)
```

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

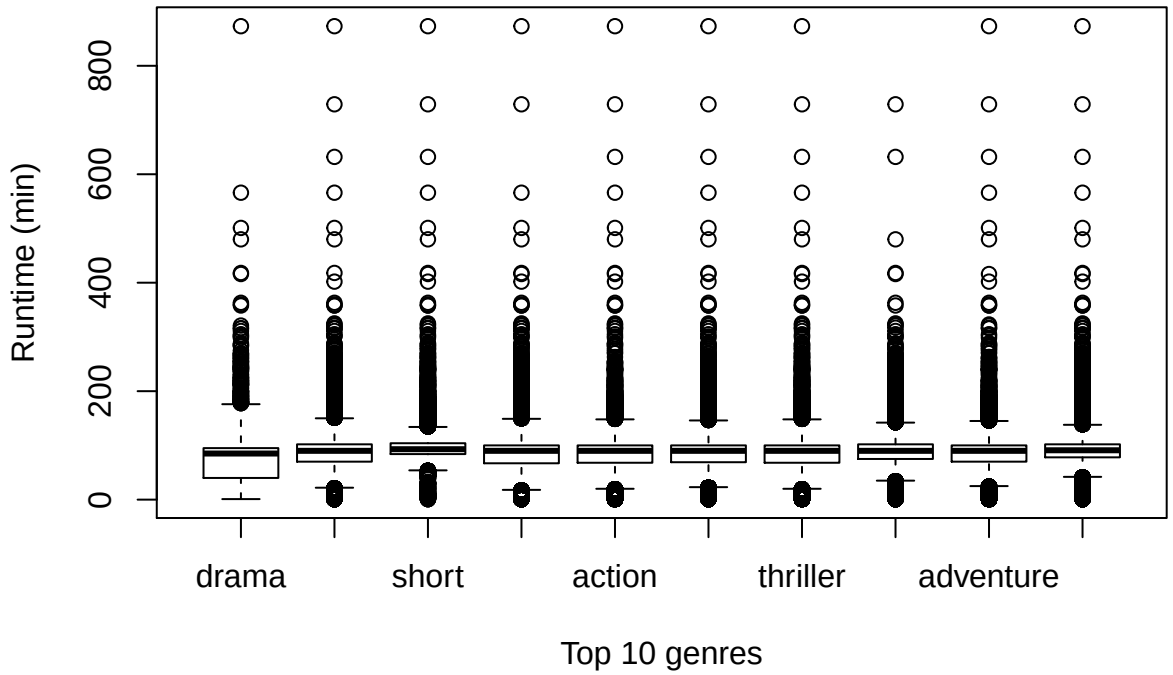
```
# TODO: Select movies from top 10 most common genres and plot their relative proportions
top10 <- data.frame(sort(colSums(df2[,40:67]),decreasing = TRUE)[1:10])
top10[, 2] <- top10[1]/nrow(df2[,40:67]) * 100
top10[, 3] <- rownames(top10)

colnames(top10) <- c("count","proportions", "genres")
top10$genres = as.factor(top10$genres)
ggplot(top10, aes(x = genres, y= proportions)) + geom_bar(colour="black", stat="identity", width = 0.5)
  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
genres <- rownames(top10)
runTimeTop10 <- (df2[genres]==0) *df2$Runtime
runTimeTop10[runTimeTop10 == 0] <- NA
```

```r
boxplot( runTimeTop10 ,names=colnames(runTimeTop10), xlab="Top 10 genres", ylab= "Runtime (min)")
```



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

A: I am surprised to see that there are much more drama (39%) and comedy(32%) movies than action (11%) movies. The length distribution of different type of the movies are about the same, with drama has a slightly wide distribution.

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). There are 3 columns that contain date information: `Year` (numeric year), `Date` (numeric year), and `Released` (string representation of the release date).

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

Note: Do not remove the rows with `Gross == NA` at this point, just use this a guideline.

```r
# TODO: Remove rows with Year/Date/Released mismatch
release_year <- as.numeric(substring(df2$Released, 1,4))
matched <- (abs(release_year-df2$Year) <= 1)
df2_matched<- df2[matched,]

removed <- nrow(df2)- nrow(df2_matched)
orig_gross<- sum(df2$Gross>0,na.rm = TRUE)
cur_gross <- sum(df2_matched$Gross > 0, na.rm = TRUE)
rate <- (orig_gross-cur_gross )/orig_gross*100
print(paste("Percentage of data removed ", round(rate, 2),"%."))
```

```
## [1] "Percentage of data removed  2.08 %."
```

```
print(paste("The number of rows remain in the resulting dataset: ", nrow(df2_matched) ))
```

```
## [1] "The number of rows remain in the resulting dataset:  38181"
```

Q: What is your precise removal logic, and how many rows remain in the resulting dataset?

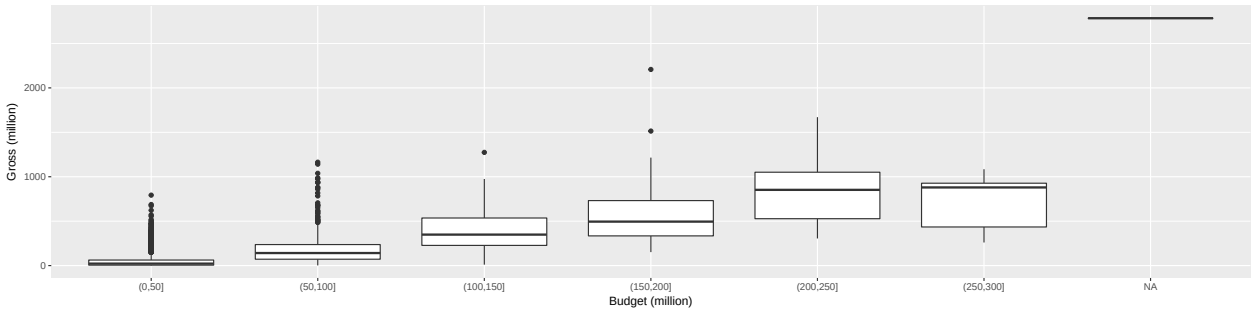
A: My logic is the year value in Released shoud be the same as the Year or one year after. A movie can be released on Dec 30, 2016 and thus marked as 2017 in Year. All unmatched data are removed. I have 38181 rows remain in my dataset.

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.

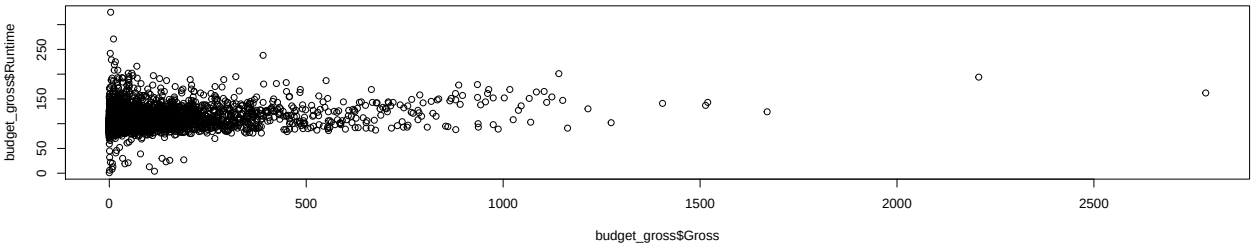
```
# TODO: Investigate if Gross Revenue is related to Budget, Runtime or Genre

# 1. Gross and Budget
budget_gross <- subset(df2_matched,df2_matched$Gross>0)
budget_gross$Gross <- budget_gross$Gross/1000000
budget_gross$Budget <- budget_gross$Budget/1000000
budget_gross$Budget_bin <- cut(budget_gross$Budget, seq(from=0, to=350, by=50))
ggplot(budget_gross, aes(x=Budget_bin,y=Gross)) + geom_boxplot()+ ylab("Gross (million)") + xlab("Budge
```




```
print(paste("The correlation between gross and budget is:", round(cor(budget_gross$Gross,budget_gross$Bu
```

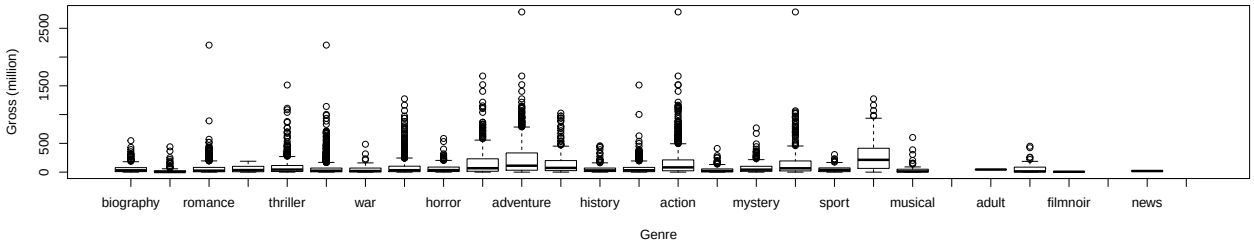
```
## [1] "The correlation between gross and budget is: 0.74"
```

```
# 2. Gross and Runtime
plot(budget_gross$Gross, budget_gross$Runtime)
```




```
# 3. Gross and Genre
genre_gross <- budget_gross[40:67]*budget_gross$Gross
```

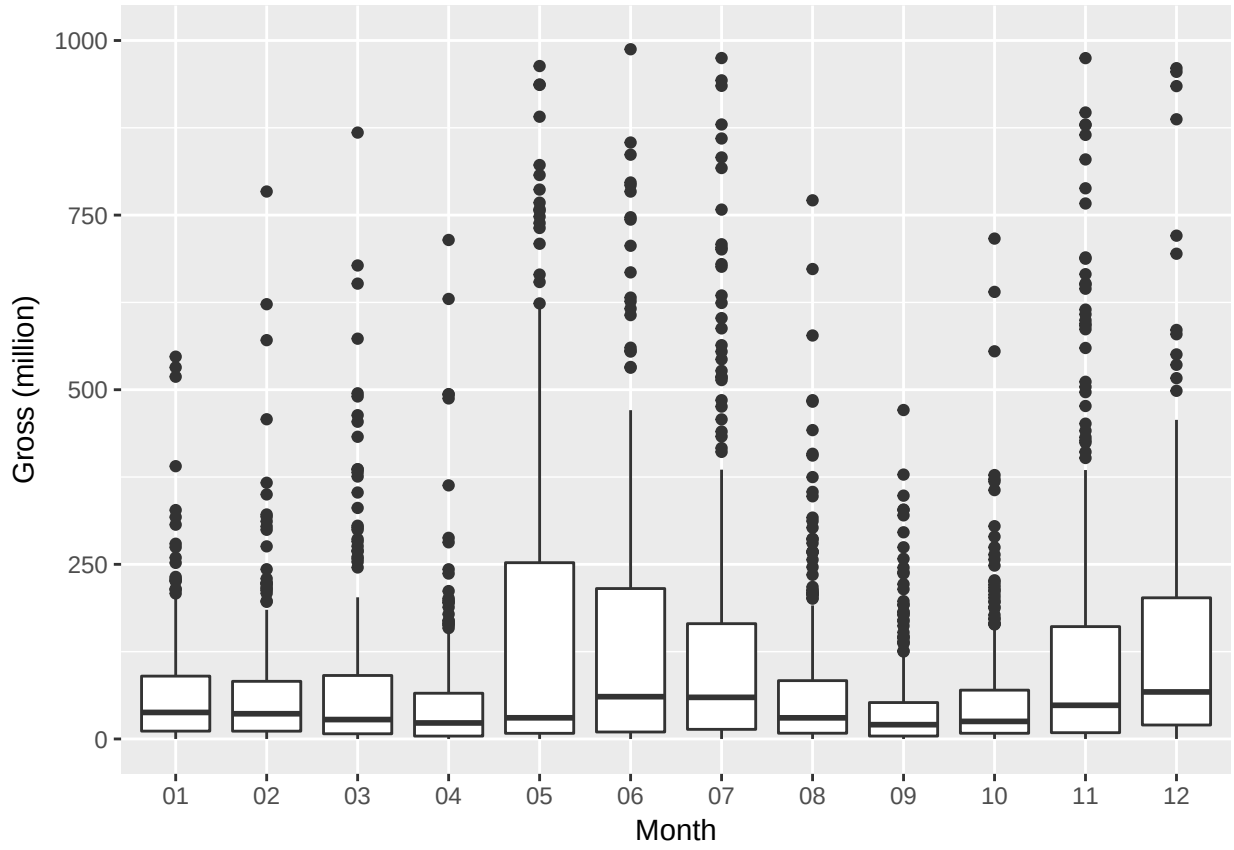
```
genre_gross[genre_gross == 0] <- NA
boxplot(genre_gross, names=colnames(genre_gross), xlab = "Genre", ylab="Gross (million)")
```



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: Budget and gross have a high correlation: 0.76. I did not find any relation between runtime and gross. This may because the length so movies are standardized in movie industry. I did not find a clear relationship between genres and gross, but I need to address that the adventure, animation and action genres have a larger range of gross revenue compare to the rest of genres.

```
# TODO: Investigate if Gross Revenue is related to Release Month
budget_gross$release_month <- substring(budget_gross$Released, 6,7)
ggplot(budget_gross,aes(x=release_month,y=Gross)) + geom_boxplot(na.rm = TRUE)+ ylim(0, 1000) + ylab("G
```



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: 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


# Construct two functions

countWin <- function(x){
  if(x!= "n/a"){
    count = 0
    if(str_detect(x,"win")){
      a <- as.numeric(gregexpr("win",x))
      b <- substring(x, a-3, a)
      c <- gregexpr('[0-9]+', b)
      count = count + as.numeric(regmatches(b,c)[[1]])
    }
    if(str_detect(x,"won")){
      a <- as.numeric(gregexpr("won",x))
      b <- substring(x, a+3, a+6)
      c <- gregexpr('[0-9]+', b)
      count = count + as.numeric(regmatches(b,c)[[1]])
    }
    return(count)
  }
  else(return(NA))
}

countNomi<- function(x){
  if(x!= "n/a"){
    n_nom = 0
    if(str_detect(x,"nomination")){
      a <- as.numeric(gregexpr("nomination",x))
      b <- substring(x, a-4, a)
      c <- gregexpr('[0-9]+', b)
      n_nom = n_nom + as.numeric(regmatches(b,c)[[1]])
    }
    if(str_detect(x,"nominated for")){
      a <- as.numeric(gregexpr("nominated for",x))
      b <- substring(x, a+13, a+16)
      c <- gregexpr('[0-9]+', b)
      n_nom = n_nom + as.numeric(regmatches(b,c)[[1]])
    }
    return(n_nom)
  }
  else(return(NA))
}
```

```
award <- budget_gross$Awards
award <- sapply(award, tolower)
budget_gross$Win <- sapply(award, countWin)
budget_gross$Nomination <- sapply(award, countNomi)
summary(budget_gross$Win>0)
```

```
##    Mode   FALSE    TRUE    NA's
## logical     858    2761     565
```

```
summary(budget_gross$Nomination>0)
```

```
##    Mode   FALSE    TRUE    NA's
## logical     142    3477     565
```

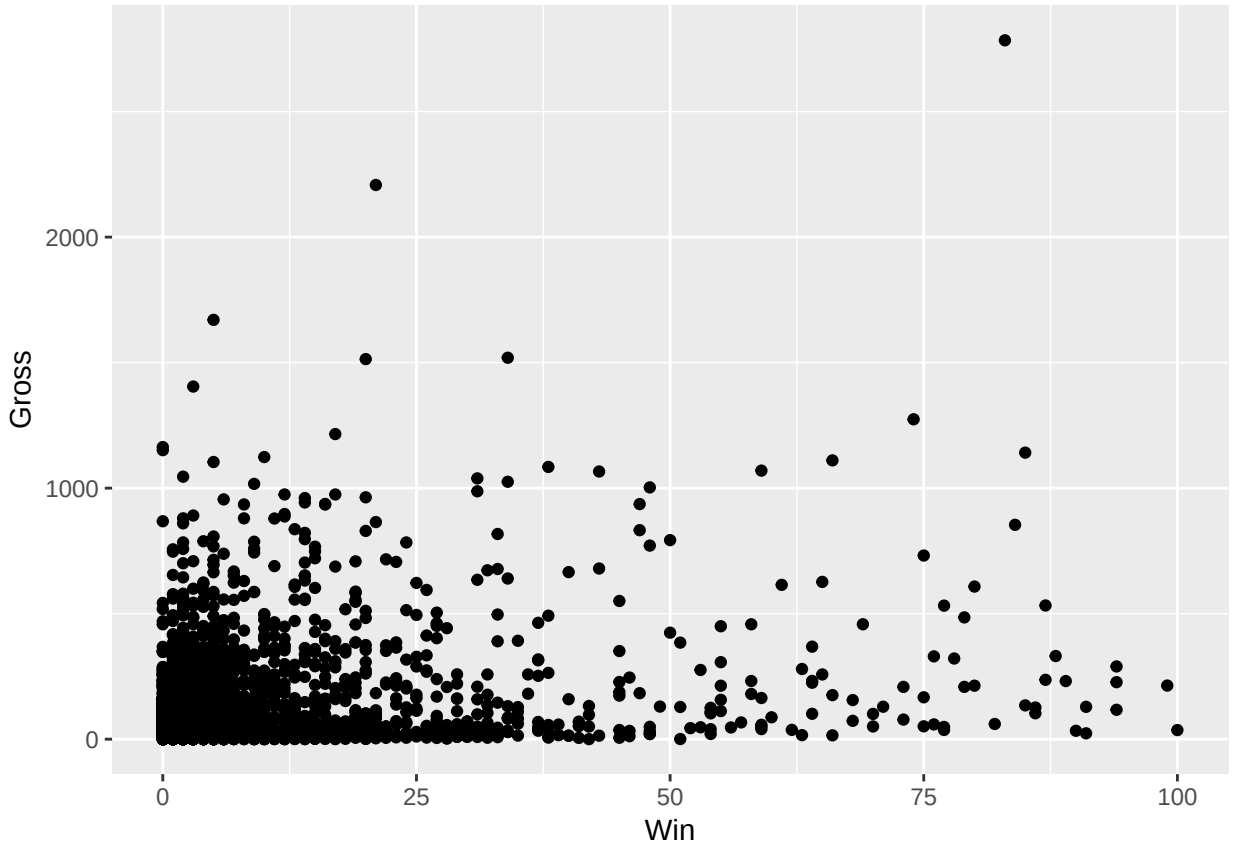
Q: How did you construct your conversion mechanism? How many rows had valid/non-zero wins or nominations?

A: I adopted str_detect, gregexpr, substring and regmatchese to find the target word "win" "won" and "nominated for". I saved the number before the target words as well as after the targted words. I found 2761 valid wins and 3477 valid nominations.

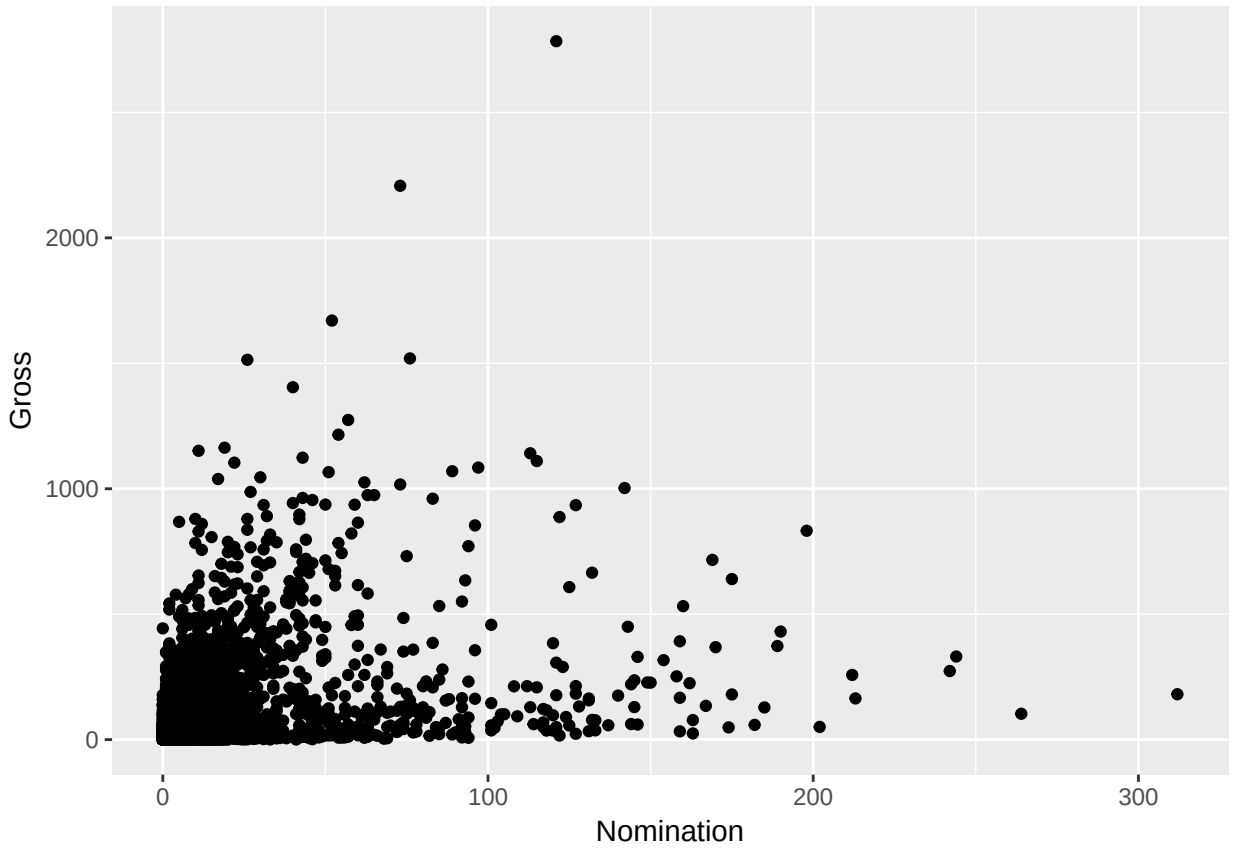
```
# TODO: Plot Gross revenue against wins and nominations
ggplot(budget_gross, aes(Win,Gross))+geom_point(a.rm=TRUE)
```

```
## Warning: Ignoring unknown parameters: a.rm
```

```
## Warning: Removed 565 rows containing missing values (geom_point).
```




```
ggplot(budget_gross, aes(Nomination,Gross))+geom_point(na.rm=TRUE)
```




```
cor_gross_win = cor(budget_gross$Gross, budget_gross$Win, use = "pairwise.complete.obs")
cor_gross_nomination = cor(budget_gross$Gross, budget_gross$Nomination, use = "pairwise.complete.obs")
cor_gross_win
```

```
## [1] 0.2748131
```

```
cor_gross_nomination
```

```
## [1] 0.3308615
```

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

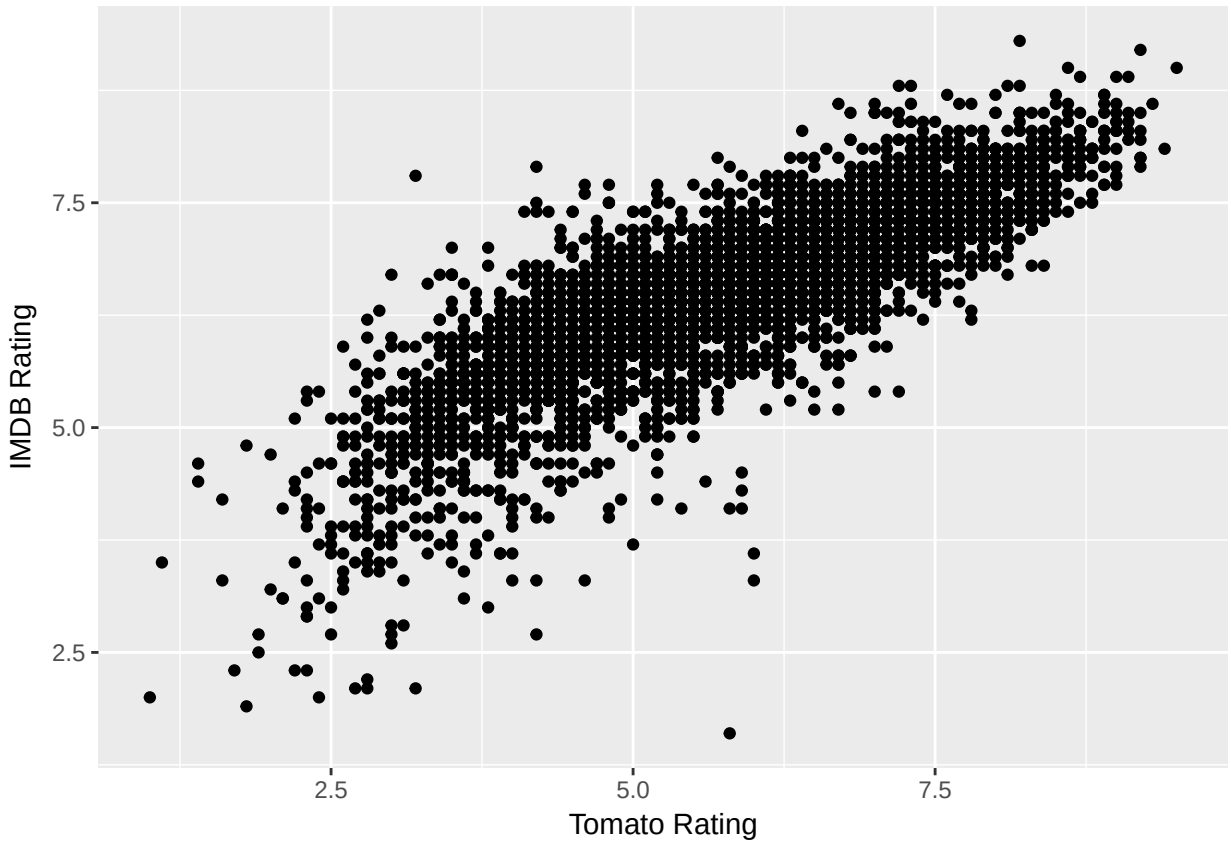
A: I did not observe any corelation from the graph. The computed coefficients also varified my observation. No strong corelations are found between gross revenue and the number of awards won and nominations received.

7. Movie ratings from IMDb and Rotten Tomatoes

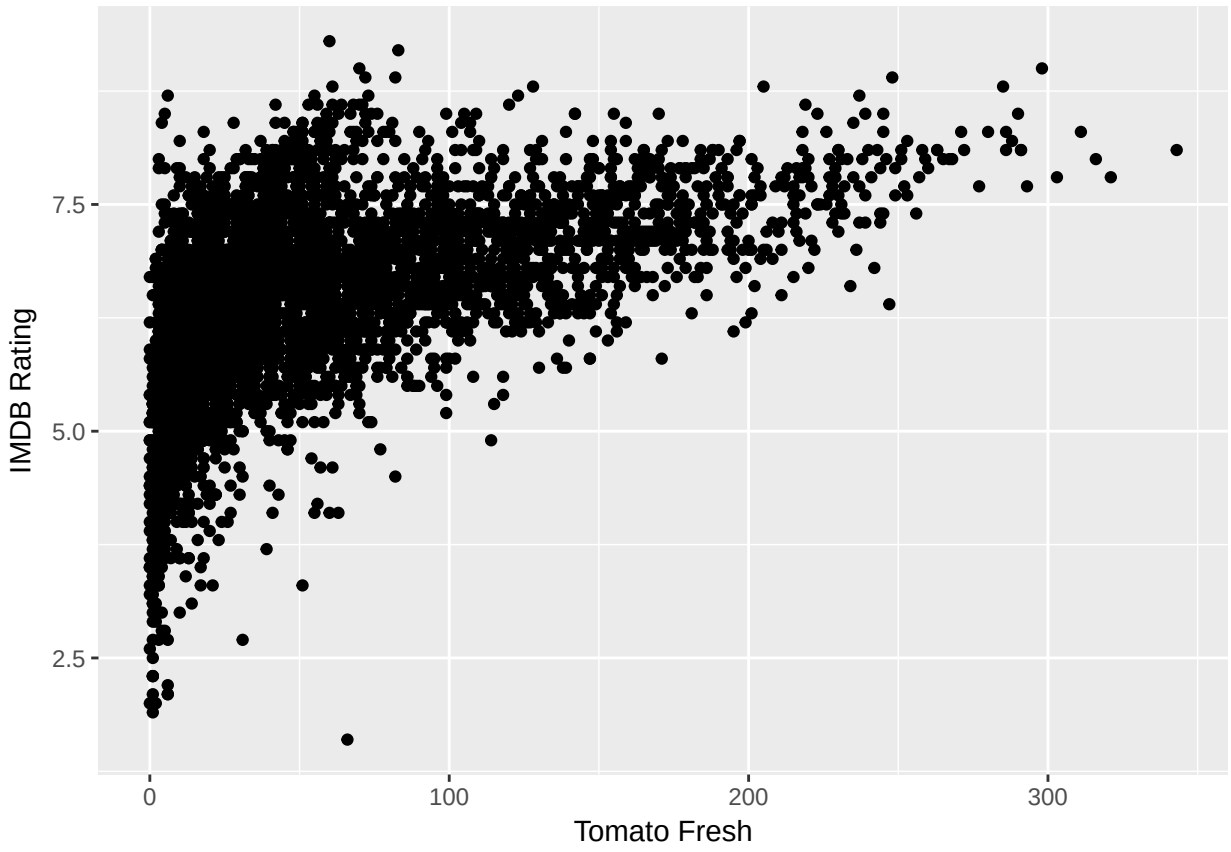
There are several variables that describe ratings, including IMDb ratings (`imdbRating` represents average user ratings and `imdbVotes` represents the number of user ratings), and multiple Rotten Tomatoes ratings (represented by several variables pre-fixed by `tomato`). Read up on such ratings on the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

Investigate the pairwise relationships between these different descriptors using graphs.

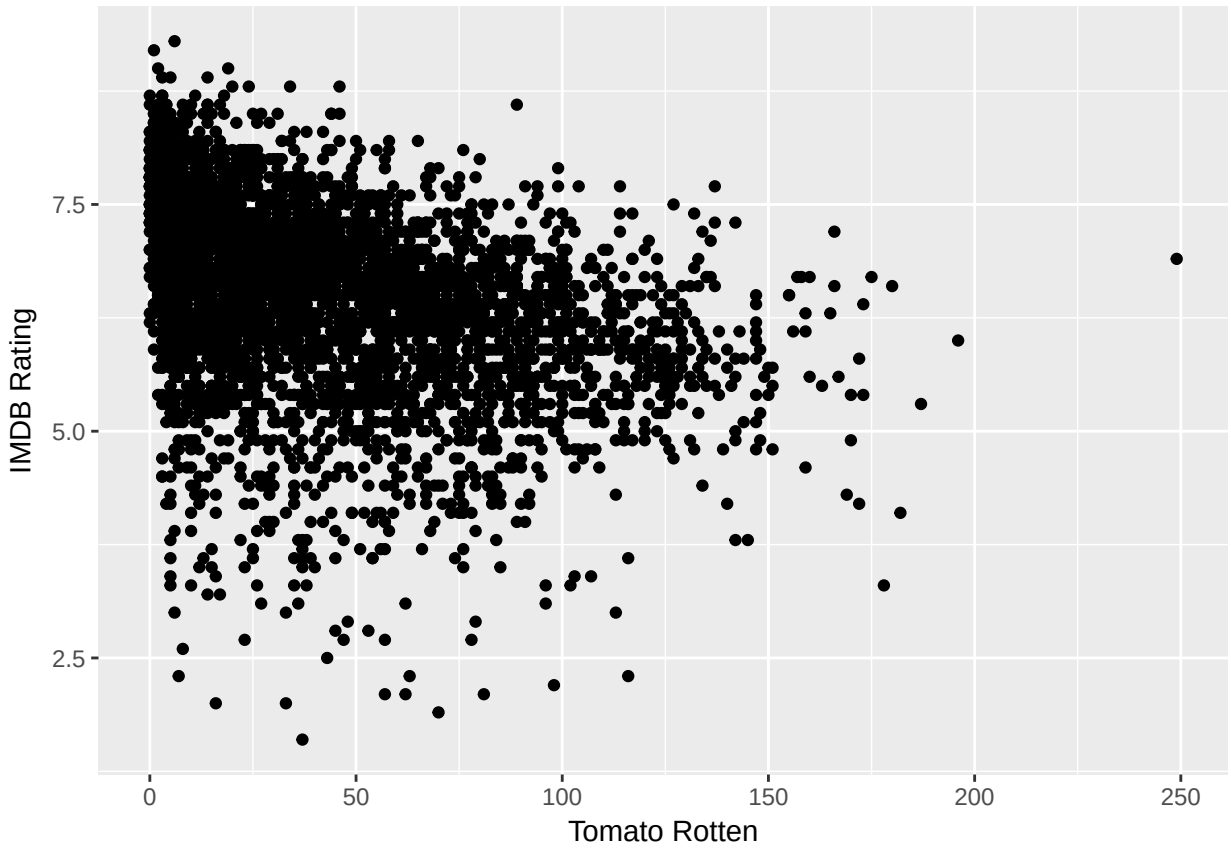
```
# TODO: Illustrate how ratings from IMDb and Rotten Tomatoes are related
ggplot(budget_gross, aes(tomatoRating,imdbRating)) + geom_point(na.rm=TRUE) + xlab("Tomato Rating") + yl
```



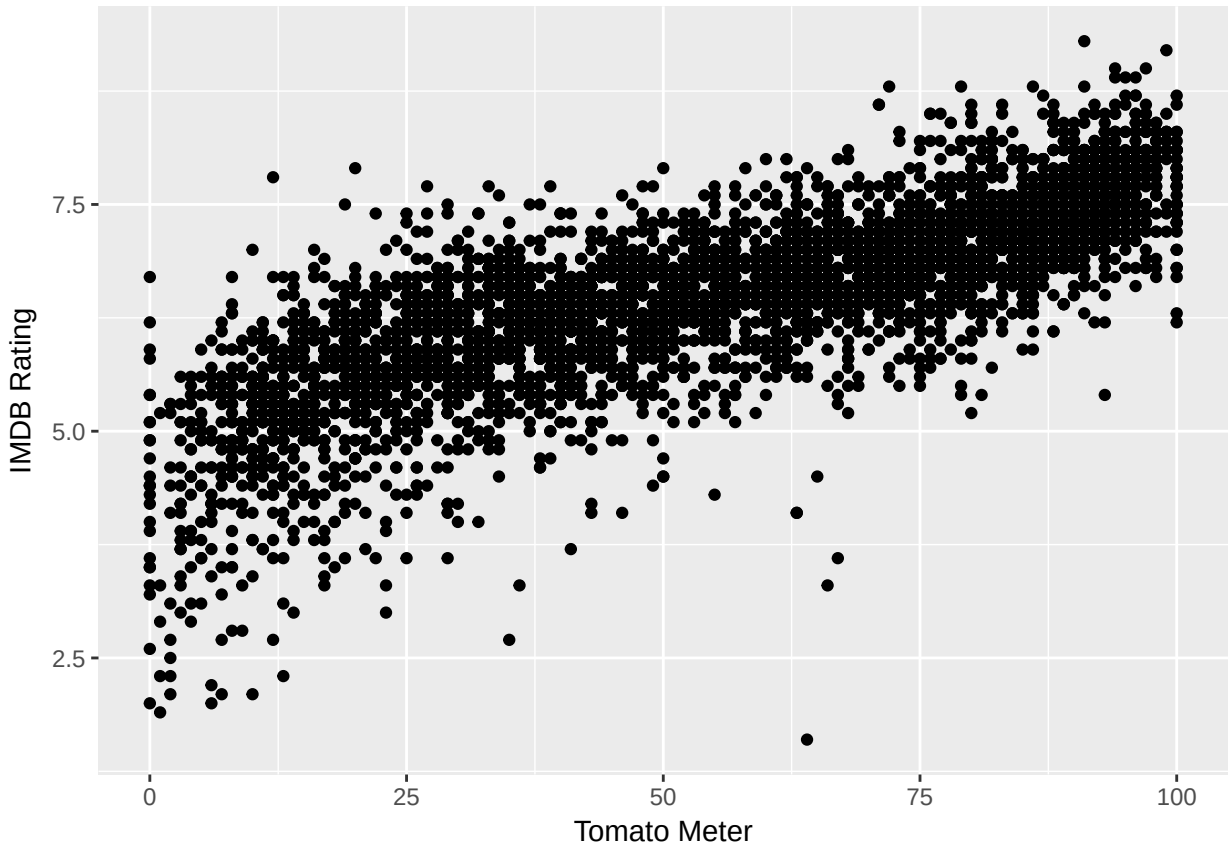

```
ggplot(budget_gross, aes(tomatoFresh,imdbRating)) + geom_point(na.rm=TRUE) + xlab("Tomato Fresh") + ylab
```



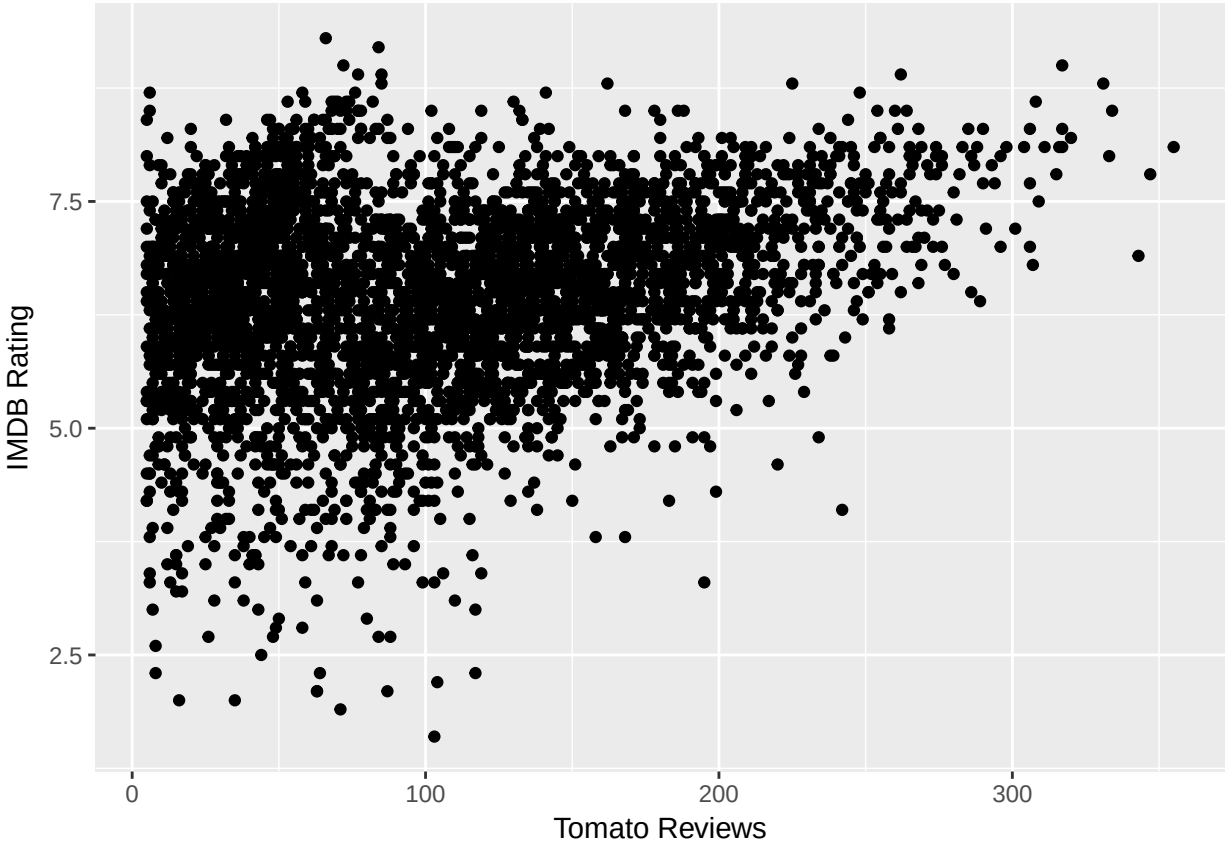

```
ggplot(budget_gross, aes(tomatoRotten,imdbRating)) + geom_point(na.rm=TRUE) + xlab("Tomato Rotten") + yl
```




```
ggplot(budget_gross, aes(tomatoMeter,imdbRating)) + geom_point(na.rm=TRUE) + xlab("Tomato Meter") + ylab
```




```
ggplot(budget_gross, aes(tomatoReviews,imdbRating)) + geom_point(na.rm=TRUE) + xlab("Tomato Reviews") +
```



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

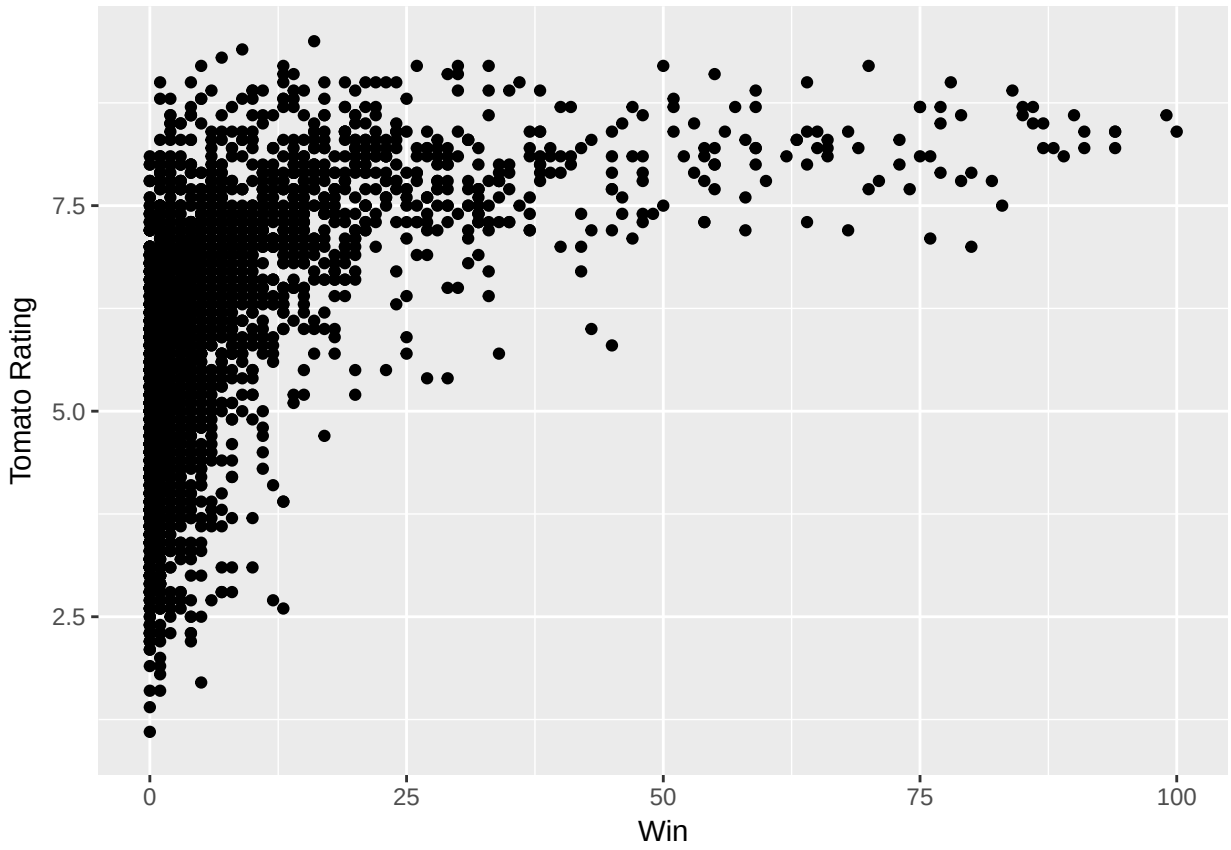
A: The user ratings of IMDb is very similar to the critics ratings of Rotten Tomatoes, especially Tomato Meter rating. The difference I observed is that user tend to mark high on all movies (most above 5/10) while critics tends to mark lower on the movies they don't like (under 100/250).

8. Ratings and awards

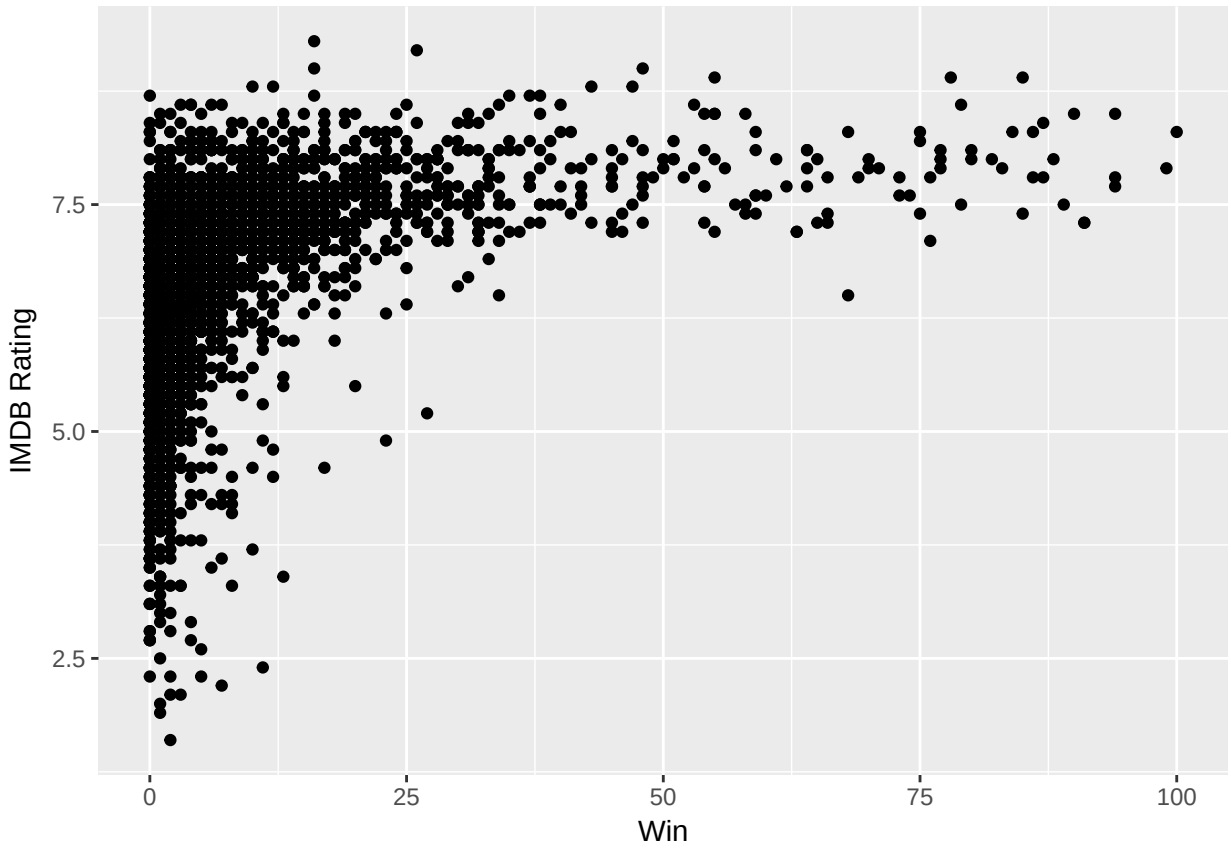
These ratings typically reflect the general appeal of the movie to the public or gather opinions from a larger body of critics. Whereas awards are given by professional societies that may evaluate a movie on specific attributes, such as artistic performance, screenplay, sound design, etc.

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

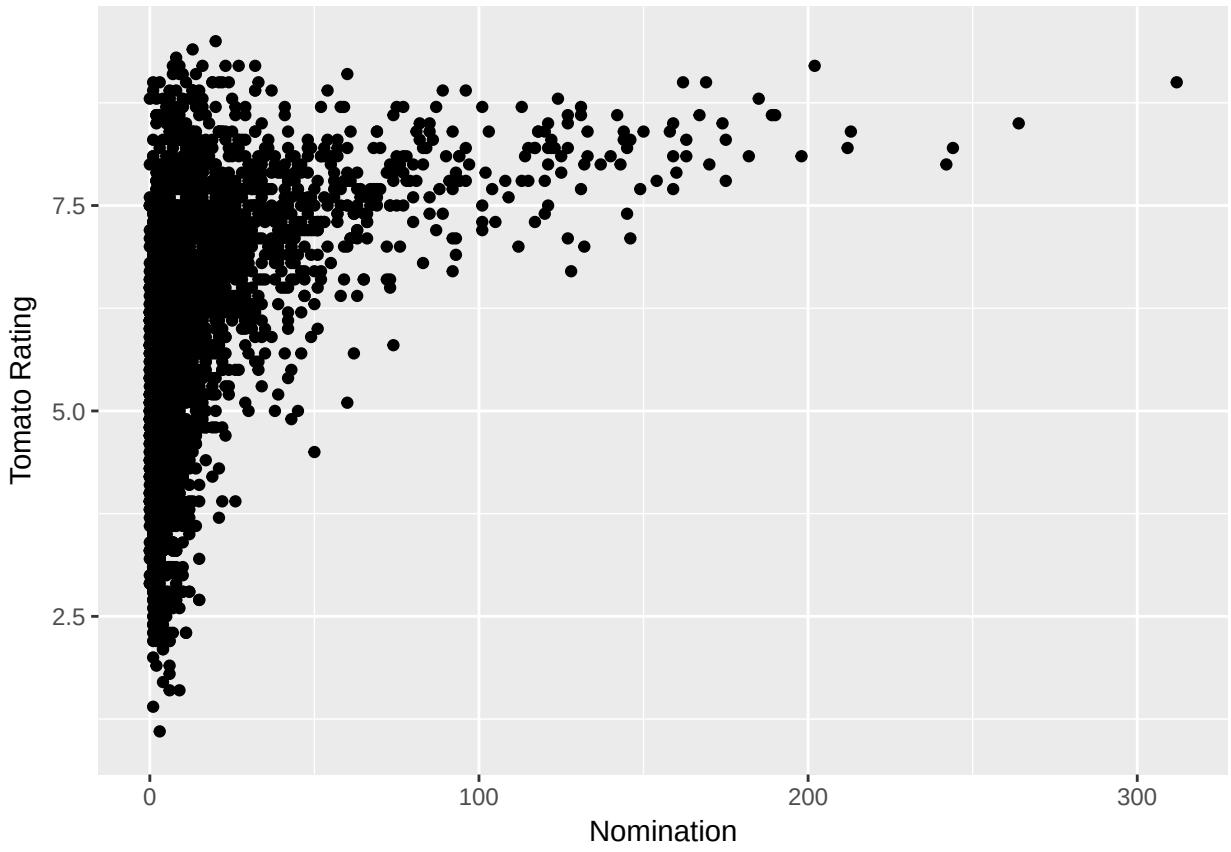
```
# TODO: Show how ratings and awards are related
ggplot(budget_gross, aes(Win,tomatoRating))+geom_point(na.rm=TRUE) + ylab("Tomato Rating")
```



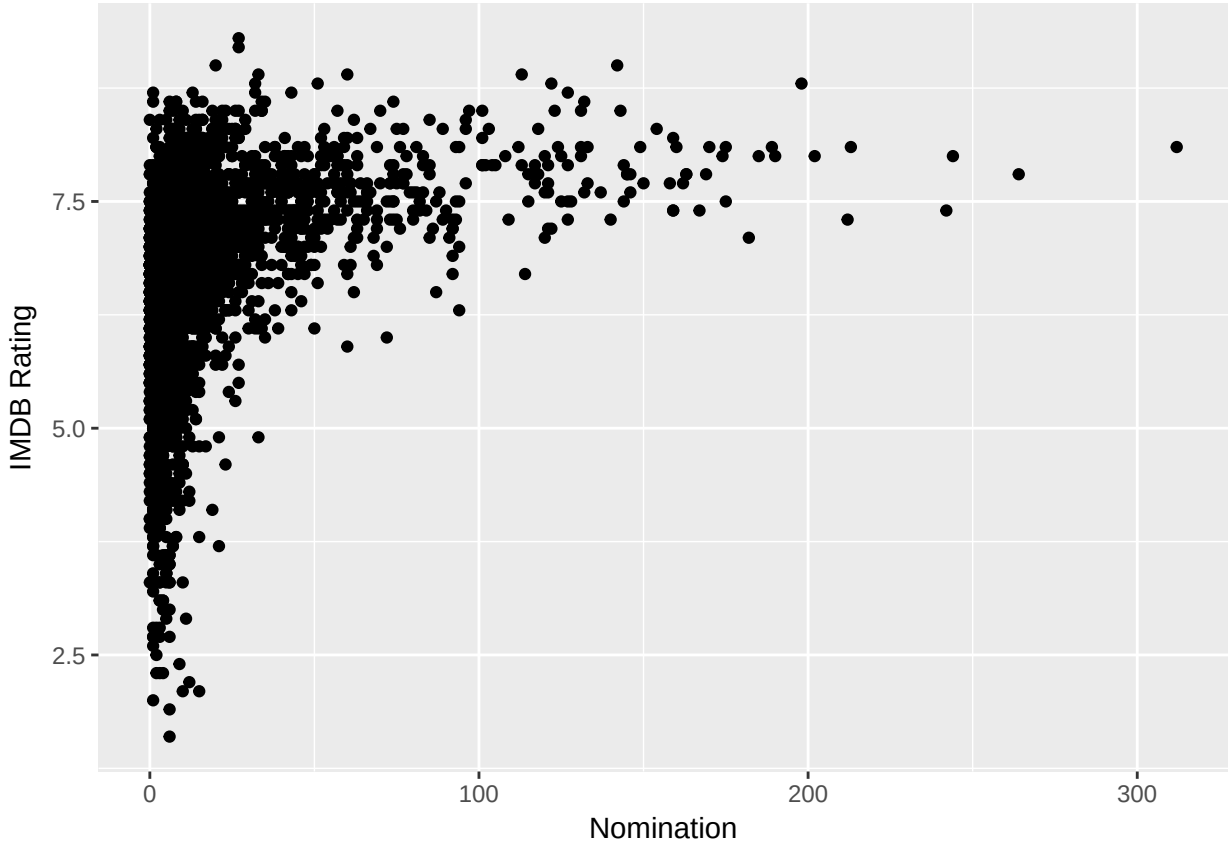

```
ggplot(budget_gross, aes(Win,imdbRating))+geom_point(na.rm=TRUE) + ylab("IMDB Rating")
```




```
ggplot(budget_gross, aes(Nomination,tomatoRating))+geom_point(na.rm=TRUE) + ylab("Tomato Rating")
```




```
ggplot(budget_gross, aes(Nomination,imdbRating))+geom_point(na.rm=TRUE) + ylab("IMDB Rating")
```




```
cor_Trating_win = cor(budget_gross$tomatoRating, budget_gross$Win, use = "pairwise.complete.obs")
cor_Trating_nomination = cor(budget_gross$tomatoRating, budget_gross$Nomination, use = "pairwise.comple
cor_Irating_win = cor(budget_gross$imdbRating, budget_gross$Win, use = "pairwise.complete.obs")
cor_Irating_nomination = cor(budget_gross$imdbRating, budget_gross$Nomination, use = "pairwise.complete

cor_Trating_win
```

```
## [1] 0.5104076
```

```
cor_Trating_nomination
```

```
## [1] 0.4673358
```

```
cor_Irating_win
```

```
## [1] 0.4422805
```

```
cor_Irating_nomination
```

```
## [1] 0.3958534
```

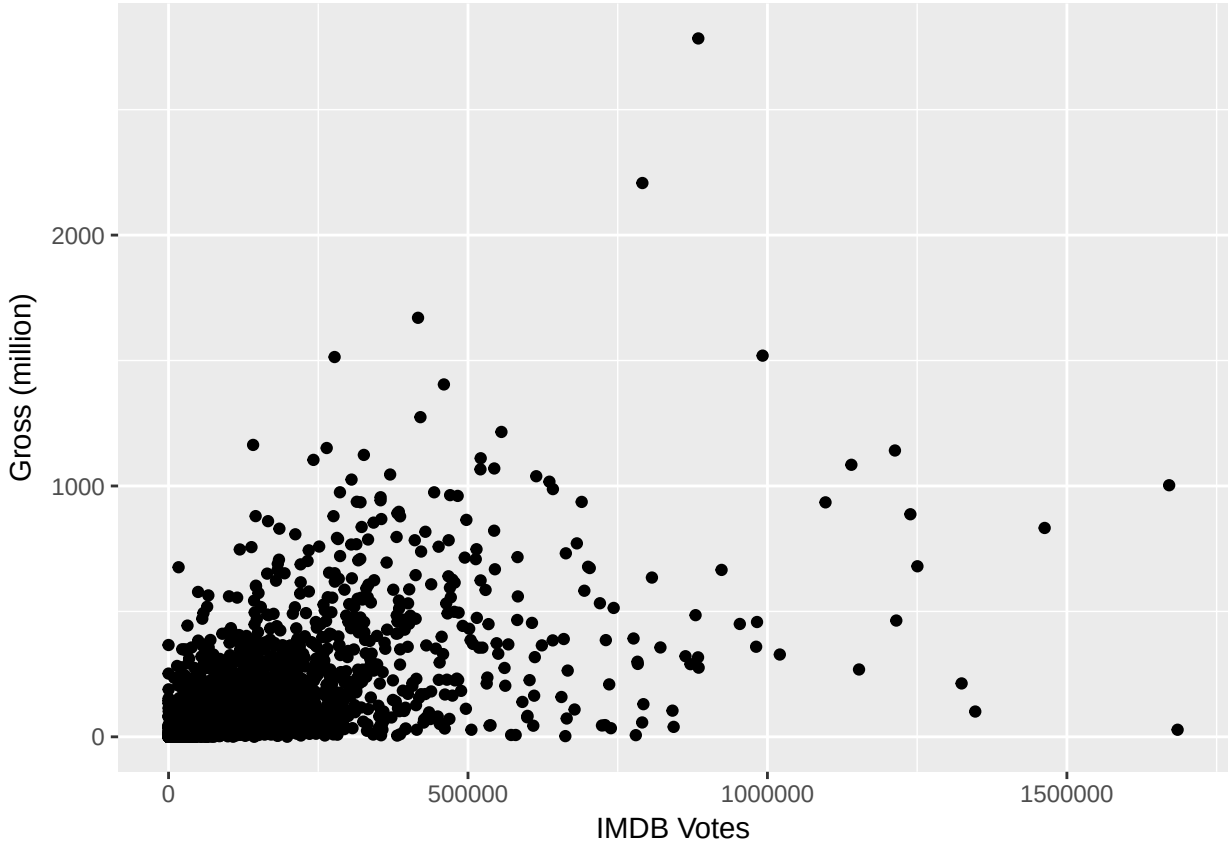
Q: How good are these ratings in terms of predicting the success of a movie in winning awards or nominations? Is there a high correlation between two variables?

A: All movies that received high number (over 50) of wins and nominations received high ratings from both Tomato and IMDB. However, for movies that received lower number (under 50) of wins and nominations, the relationship is not clear. The computed coefficients varifies my observation.

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

# Expected insight #1: IMDB Votes and Gross
ggplot(budget_gross, aes(imdbVotes,Gross)) + geom_point(na.rm=TRUE) + xlab("IMDB Votes")+ ylab("Gross (
```




```
cor_votes_gross = cor(budget_gross$imdbVotes, budget_gross$Gross, use = "pairwise.complete.obs")
cor_votes_gross
```

```
## [1] 0.6212329
```

```
# Expected insight #2: Tomato User reviews and Gross
top100_movies <- data.frame(sort(budget_gross$Gross, decreasing = TRUE) [1:100])
total_gross <- sum(budget_gross$Gross)
c = rep(total_gross, 100)
top100_movies[, 2] <- top100_movies[1]/c * 100
colnames(top100_movies) <- c("Gross Revenue","GrossRevenueProportion")
a<-colSums(top100_movies)
print(paste("The total gross revenue proportion of Top 100 movies are:", round(a[2] , 2),"%."))
```

```
## [1] "The total gross revenue proportion of Top 100 movies are: 21.76 %."
```

Q: Expected insight #1.

A: I expect that gross revenue is higly related with the imdbVotes. The computed coefficient proved my

observation.

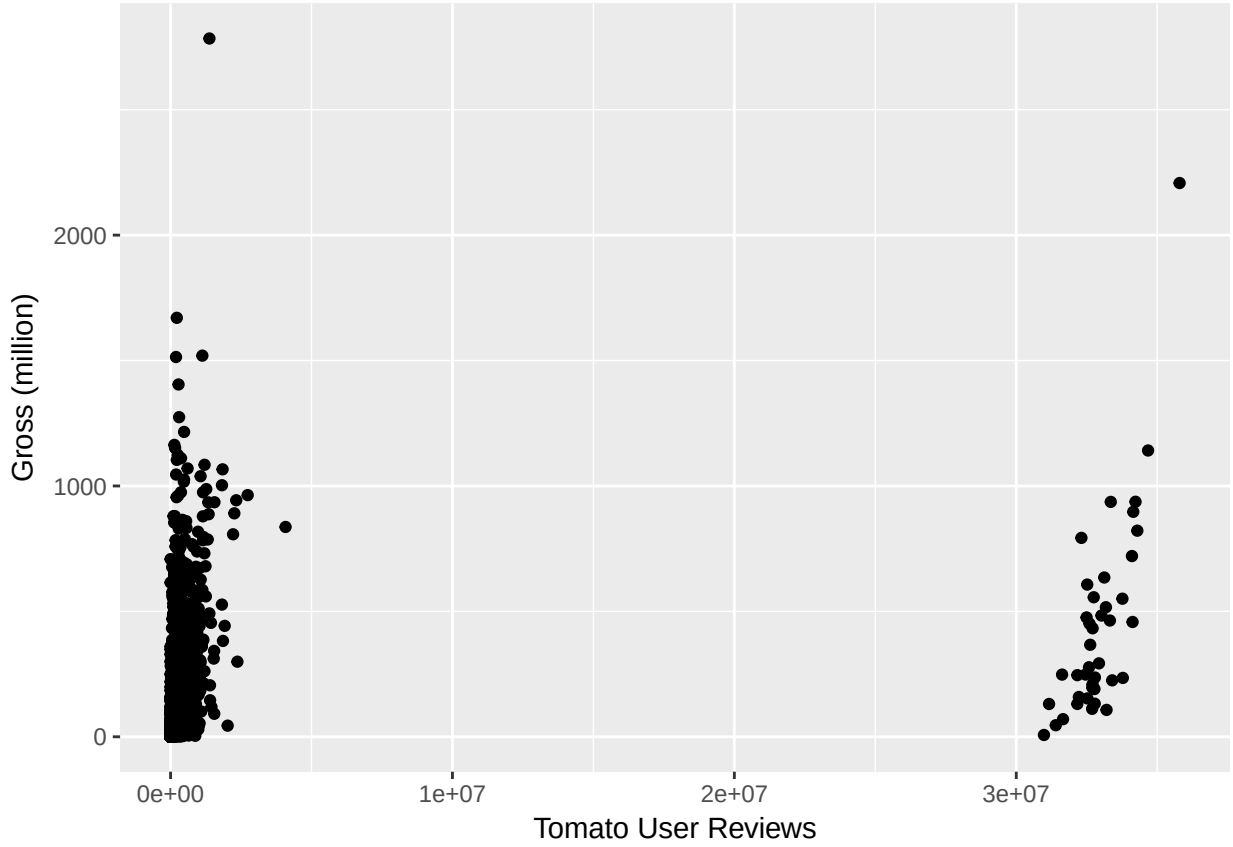
Q: Expected insight #2.

A: I expect to see that the top 100 movies (in terms of the gross revenue) shares about 40% of total market gross revenue.

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
# Unexpected insight: Tomato User reviews and Gross
ggplot(budget_gross, aes(tomatoUserReviews,Gross)) + geom_point(na.rm=TRUE) + xlab("Tomato User Reviews
```




```
cor_reviews_gross = cor(budget_gross$tomatoUserReviews, budget_gross$Gross, use = "pairwise.complete.obs
cor_reviews_gross
```

```
## [1] 0.2460778
```

Q: Unexpected insight.

A: At first I thought more reviews means more people have watched the movie, so that more reviews should relate to the higher gross revenue. The data does not support this assumption.