Code

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 Username: clightsey3

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 IMDb's 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 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 (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

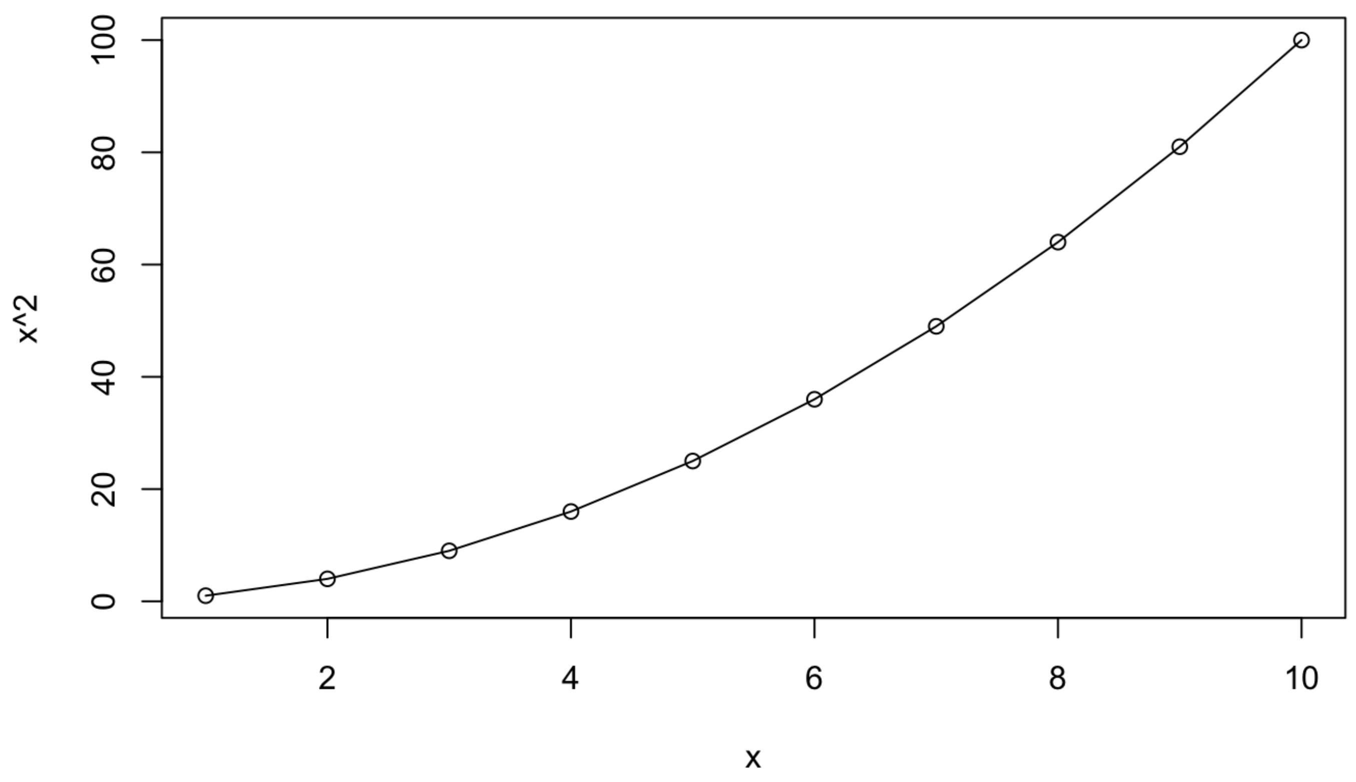
```
x = 1:10
print(x^2)
```

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

```
Data set has 40789 rows and 39 columns
```

Hide

```
colnames(df)
```

```
 [1] "Title"            "Year"             "Rated"            "Released"         "Ru
ntime"
 [6] "Genre"            "Director"         "Writer"           "Actors"           "Pl
ot"
[11] "Language"         "Country"          "Awards"           "Poster"           "Me
tascore"
[16] "imdbRating"       "imdbVotes"        "imdbID"           "Type"             "to
matoMeter"
[21] "tomatoImage"      "tomatoRating"     "tomatoReviews"    "tomatoFresh"      "to
matoRotten"
[26] "tomatoConsensus"  "tomatoUserMeter"  "tomatoUserRating" "tomatoUserReviews" "to
matoURL"
[31] "DVD"              "BoxOffice"        "Production"       "Website"          "Re
sponse"
[36] "Budget"           "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.

Hide

```
library(ggplot2)
library(GGally)
library(plyr)
library(tibble)
library(tidyr)
library(NLP)
library(tm)
library(stringi)
```

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

```
# Remove all rows from df that do not correspond to movies
cat("Number of Movies, Series, and Games:", dim(df)[1], "\n")
```

```
Number of Movies, Series, and Games: 40789
```

Hide

```
df = df[df$Type == "movie",]
cat("Number of Movies:", dim(df)[1])
```

```
Number of Movies: 40000
```

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

A: 40,000 rows are left after removal. 789 rows were removed due to having a "Type" not equal to "movie".

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.

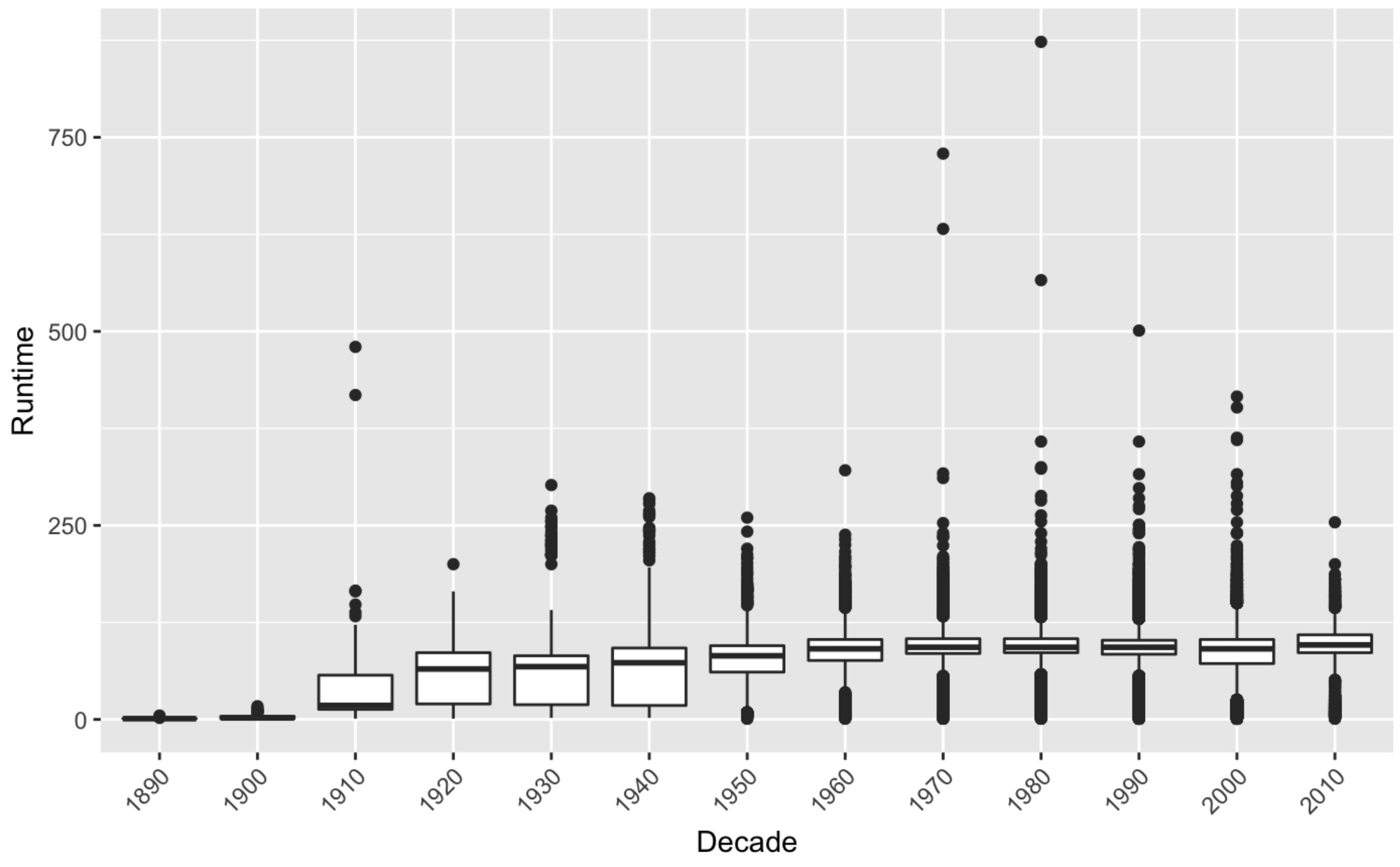
Hide

```
# Replace df$Runtime with a numeric column containing the runtime in minutes
ri = gsub(" min", "", df$Runtime)
ri = gsub("1 h", "60", ri)
ri = gsub("2 h", "120", ri)
ri = gsub("3 h", "180", ri)
ri = gsub("4 h", "240", ri)
ri = gsub("N/A", NA, ri)
rf = strsplit(ri, " ")
rtf = c()
co = 1
for (i in rf){
  if (length(i) > 1){
    rtf[co] = as.numeric(i[[1]]) + as.numeric(i[[2]])
    co = co + 1
  } else {
    rtf[co] = as.numeric(i[[1]])
    co = co + 1
    }
}
df$Runtime = rtf
```

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.

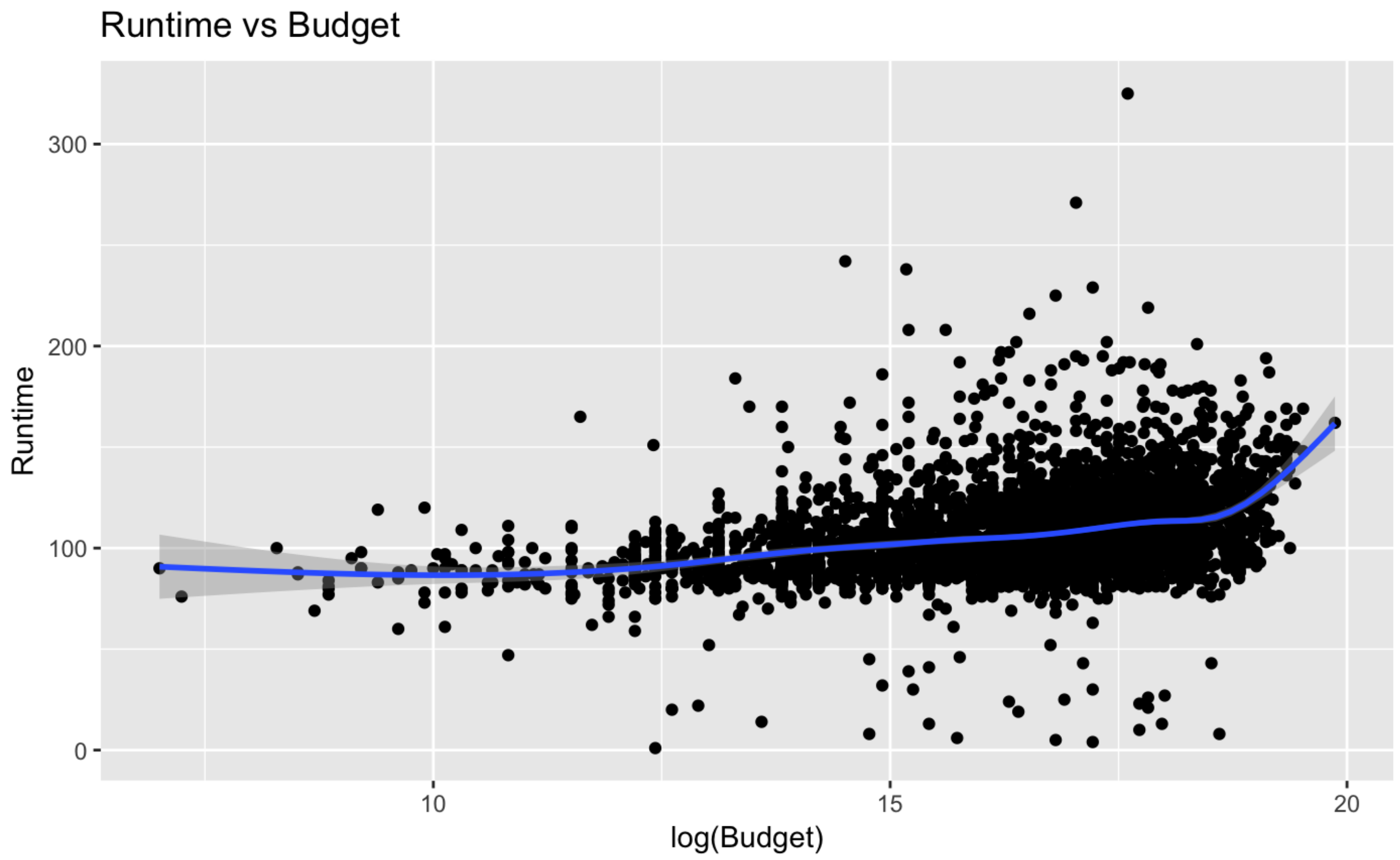
Hide

```
# Investigate the distribution of Runtime values and how it varies by Year and Budget
# Create a thinner dataframe with only Runtime and Year
df2y = data.frame(Runtime = df$Runtime, Year = df$Year)
# Transform df2y to create groups by decade
df2yt = transform(df2y, Decade = cut(Year, 13, seq(1890, 2010, by=10)))
# Fix ---9 year error
df2yt[df2yt[,"Year"] == 1899, "Decade"] = 1890
df2yt[df2yt[,"Year"] == 1909, "Decade"] = 1900
df2yt[df2yt[,"Year"] == 1919, "Decade"] = 1910
df2yt[df2yt[,"Year"] == 1929, "Decade"] = 1920
df2yt[df2yt[,"Year"] == 1939, "Decade"] = 1930
df2yt[df2yt[,"Year"] == 1949, "Decade"] = 1940
df2yt[df2yt[,"Year"] == 1959, "Decade"] = 1950
df2yt[df2yt[,"Year"] == 1969, "Decade"] = 1960
df2yt[df2yt[,"Year"] == 1979, "Decade"] = 1970
df2yt[df2yt[,"Year"] == 1989, "Decade"] = 1980
df2yt[df2yt[,"Year"] == 1999, "Decade"] = 1990
df2yt[df2yt[,"Year"] == 2009, "Decade"] = 2000
df2yt[df2yt[,"Year"] == 2019, "Decade"] = 2010
# Plot Runtime by Decade using ggplot2
p1 = ggplot(df2yt, aes(x=Decade, y=Runtime )) + geom_boxplot(na.rm = TRUE) + theme(axis.
text.x = element_text(angle = 45, hjust = 1))
print(p1)
```



Hide

```
# Create a thinner dataframe with only Runtime and Budget
df2b = data.frame(Runtime=df$Runtime, Budget=df$Budget)
# Omit NA values in df2b for warning/error free plotting with qplot
df2b_nona = na.omit(df2b)
# Plot with qplot
ggplot(df2b_nona, aes(y=Runtime, x=log(Budget))) + labs(title="Runtime vs Budget") + geo
m_point() + stat_smooth(method="auto") #+ scale_y_continuous(limits = c(-20, 1.5e+09))
```



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: Looking at the boxplot of Runtime by Decade, there was a sharp increase in median runtime between the 1910's and 1920's, but there was a very minimal increase in Runtime after the 1920's. This could be explained by the introduction of movies with sound (i.e. talkies) in the 1920's. The distribution of Runtimes by Decade before the 1950's was mostly non-Gaussian, evident by the off-center median lines. From the 1950's and beyond, the Runtime distributions are mostly Gaussian. When the Runtime is plotted against the log of Budget, a relatively flat, linear relationship is observed. In other words, higher Budget films have mostly the same Runtime as lower budget films, but there is more deviation amongst the higher Budget films than among the lower Budget films.

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).

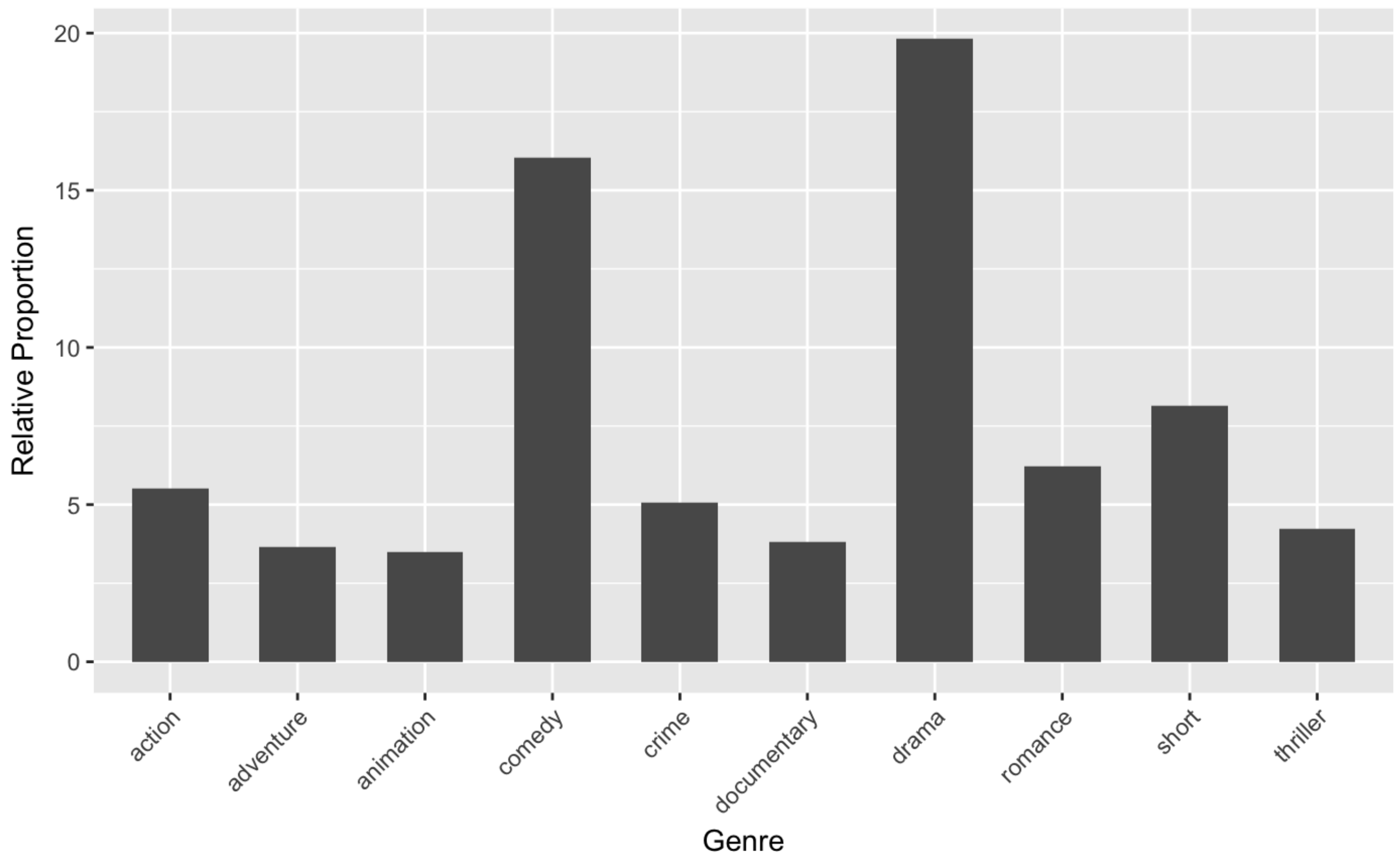
Hide

```
# Replace Genre with a collection of binary columns
# Create a vector of the df$Genre column
genre = df$Genre
# Create a dictionary of all possible genres
genres = unlist(strsplit(genre, ", "))
genres_dict = as.list(unique(genres))
# Create a list of genres for each movie
genres_l = strsplit(genre, ", ")
# Build the Corpus
genre_corp = Corpus(VectorSource(genres_l))
# Build the DocumentTermMatrix
genre_dtm = DocumentTermMatrix(genre_corp, genres_dict)
# Convert DocumentTermMatrix to a DataFrame
genre_dtm_matrix = as.matrix(genre_dtm)
genre_dtm_df = as.data.frame(genre_dtm_matrix)
# Add a join key to both DataFrames
genre_dtm_df$JoinKey = seq(1, dim(genre_dtm_df)[1], by=1)
df$JoinKey = seq(1, dim(df)[1], by=1)
# Merge the two DataFrames
df = merge(x=df, y=genre_dtm_df, by.x="JoinKey", by.y= "JoinKey")
# Drop JoinKey and Genre
df$JoinKey = NULL
df$Genre = NULL
```

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

Hide

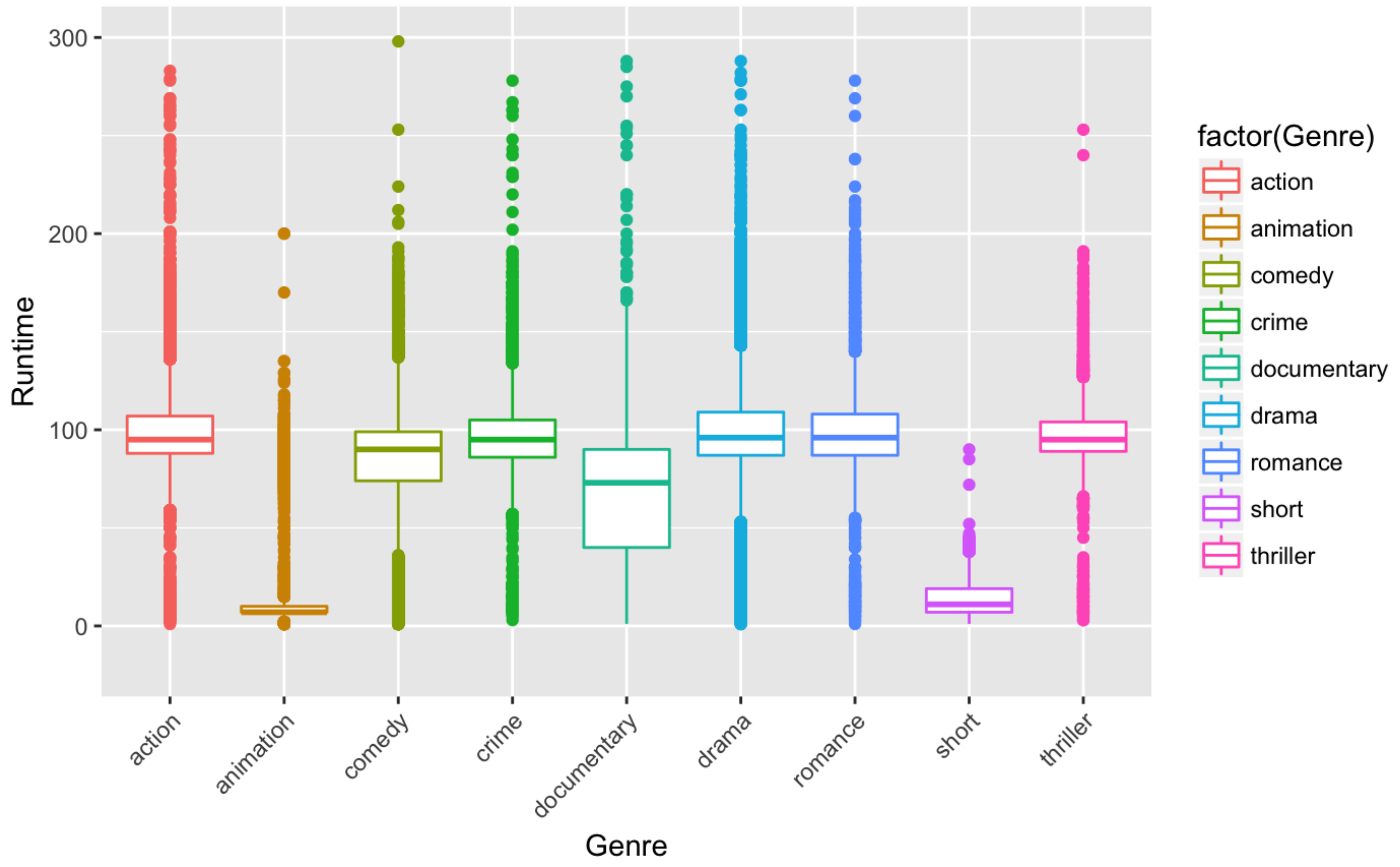
```
# Select movies from top 10 most common genres and plot their relative proportions
df3_rp = NULL
df3_rp_o = NULL
# Sum for each genre
df3_rp = data.frame(Genre_Sum=colSums(df[39:67], na.rm = TRUE))
# Calculate relative proportion for each genre, where total count of genres with a 1 use
d as denominator
df3_rp$Genre_RP = 100 * df3_rp$Genre_Sum / sum(df3_rp$Genre_Sum)
# Sort Genre_RP from largest to smallest value and keep the top 10 rows in a new DataFra
me
df3_rp_o = head(df3_rp[order(df3_rp$Genre_RP, decreasing = TRUE),], 10)
# Set row name as column for plotting
df3_rp_o = rownames_to_column(df3_rp_o, "Genre")
# Plot
ggplot(df3_rp_o, aes(x=Genre, y=Genre_RP)) + geom_bar(width=0.6, stat="identity") + scal
e_y_continuous() + labs(x="Genre", y="Relative Proportion") + theme(axis.text.x = elemen
t_text(angle = 45, hjust = 1))
```



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

Hide

```
# Plot Runtime distribution for top 10 most common genres
# Create a new DataFrame with Runtimes for each of the top 10 Genres and Merge
df3_drama = data.frame(Runtime = df[df[, "drama"] == 1, "Runtime"])
#df3_drama$JoinKey = seq(1, dim(df3_drama)[1])
df3_drama$Genre = "drama"
df3_action = data.frame(Runtime = df[df[, "action"] == 1, "Runtime"])
df3_action[dim(df3_action)[1]:dim(df3_drama)[1],] = NA
df3_action$Genre = "action"
#df3_action$JoinKey = seq(1, dim(df3_drama)[1])
df3_adventure = data.frame(Runtime = df[df[, "adventure"] == 1, "Runtime"])
df3_adventure[dim(df3_adventure)[1]:dim(df3_drama)[1],] = NA
df3_adventure$Genre = "adventure"
#df3_adventure$JoinKey = seq(1, dim(df3_drama)[1])
df3_animation = data.frame(Runtime = df[df[, "animation"] == 1, "Runtime"])
df3_animation[dim(df3_animation)[1]:dim(df3_drama)[1],] = NA
df3_animation$Genre = "animation"
#df3_animation$JoinKey = seq(1, dim(df3_drama)[1])
df3_comedy = data.frame(Runtime = df[df[, "comedy"] == 1, "Runtime"])
df3_comedy[dim(df3_comedy)[1]:dim(df3_drama)[1],] = NA
df3_comedy$Genre = "comedy"
#df3_comedy$JoinKey = seq(1, dim(df3_drama)[1])
df3_crime = data.frame(Runtime = df[df[, "crime"] == 1, "Runtime"])
df3_crime[dim(df3_crime)[1]:dim(df3_drama)[1],] = NA
df3_crime$Genre = "crime"
#df3_crime$JoinKey = seq(1, dim(df3_drama)[1])
df3_documentary = data.frame(Runtime = df[df[, "documentary"] == 1, "Runtime"])
df3_documentary[dim(df3_documentary)[1]:dim(df3_drama)[1],] = NA
df3_documentary$Genre = "documentary"
#df3_documentary$JoinKey = seq(1, dim(df3_drama)[1])
df3_romance = data.frame(Runtime = df[df[, "romance"] == 1, "Runtime"])
df3_romance[dim(df3_romance)[1]:dim(df3_drama)[1],] = NA
df3_romance$Genre = "romance"
#df3_romance$JoinKey = seq(1, dim(df3_drama)[1])
df3_short = data.frame(Runtime = df[df[, "short"] == 1, "Runtime"])
df3_short[dim(df3_short)[1]:dim(df3_drama)[1],] = NA
df3_short$Genre = "short"
#df3_short$JoinKey = seq(1, dim(df3_drama)[1])
df3_thriller = data.frame(Runtime = df[df[, "thriller"] == 1, "Runtime"])
df3_thriller[dim(df3_thriller)[1]:dim(df3_drama)[1],] = NA
df3_thriller$Genre = "thriller"
#df3_thriller$JoinKey = seq(1, dim(df3_drama)[1])
genrebind = rbind(df3_drama, df3_action, df3_animation, df3_comedy, df3_crime, df3_docum
entary, df3_romance, df3_short, df3_thriller)
ggplot(genrebind, aes(y=Runtime, x=Genre, colour=factor(Genre))) + geom_boxplot(na.rm =
TRUE) + theme(axis.text.x = element_text(angle = 45, hjust = 1)) + scale_y_continuous(li
mits = c(-20, 300))
```



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

A: Please reference the boxplot above of Runtime by Genre for the following discussion. As expected, Animation and Shorts tend to have low Runtimes. This is expected because a Short by nature has a low Runtime, and many Animation movies are geared towards children who tend to have lower attention spans than adults. Documentaries are also a little shorter than all other groups, possibly due to the nature of the content being usually non-fiction. However, I am a little suprprised that all other Top Genres have almost the exact same median Runtime. I would have expected Actions and Thrillers to be a bit shorter judging from my experiential knowledge.

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.

Hide

```
# Remove rows with Released-Year mismatch
# Create a new DataFrame to work with
df4 = df
# Create a Release_Month Column for use in Part 5
df4$Release_Month = months.Date(as.Date(df4$Released, "%y-%m-%d"))
# Convert Released to numeric year
df4$Released = as.numeric(gsub("-.*$", "", df$Released))
# Compare Year and Released columns in a new column named "Date_Compare"
df4$Date_Compare = df4$Year >= (df4$Released - 1) & df4$Year <= (df4$Released + 1)
# Subset the DataFrame to keep Matches or rows with NA
df4 = subset(df4, df4$Date_Compare == TRUE | is.na(df4$Date_Compare) == TRUE)
# Count and print pre/pst year/released compare
cat("Total movies before Eliminating mismatched rows: ", dim(df)[1], "\n",
    "Total movies after Eliminating mismatched rows: ", dim(df4)[1], "\n",
    "Percent of movies removed with a Gross value present: ", 100 * (dim(df[is.na(df$Gro
ss) == FALSE,])[1] - dim(df4[is.na(df4$Gross) == FALSE,])[1]) / dim(df[is.na(df$Gross) =
= FALSE,])[1], "%")
```

```
Total movies before Eliminating mismatched rows:  40000
 Total movies after Eliminating mismatched rows:  38181
 Percent of movies removed with a Gross value present:  1.99649 %
```

Hide

```
# Dump the filtered df4 back into df and get rid of Date_Compare
df4$Date_Compare = NULL
df = df4
```

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

A: First, I removed all rows where Year and Released did not match years exactly. This method gave me a removal percentage of ~16% for rows with values in the Gross column. I decided this was too far above the 10% suggested in the instructions, so I decided to loosen my Year/Released matching criteria a bit. I assumed that some movies are released at the end or beginning of a year. In these cases, one or the other of the databases may get the release date/year incorrect. So, I matched Year to a range (+-1) around the Release Date. This methoed gave me a removal percentage of ~2% (91 movies) for rows with values in the Gross column. The total amount of movies removed using this second method was 91.

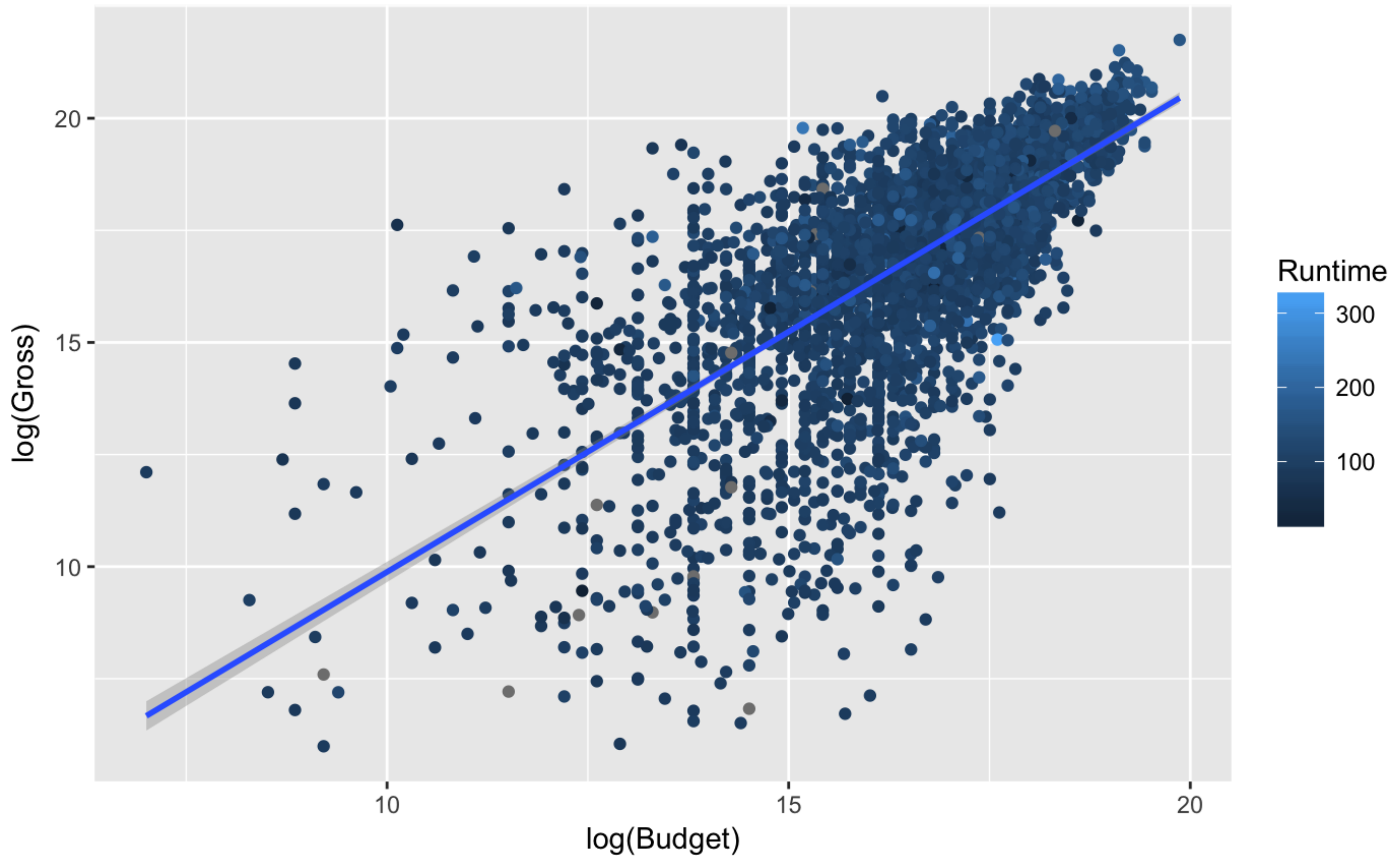
5. Explore `Gross` revenue

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.

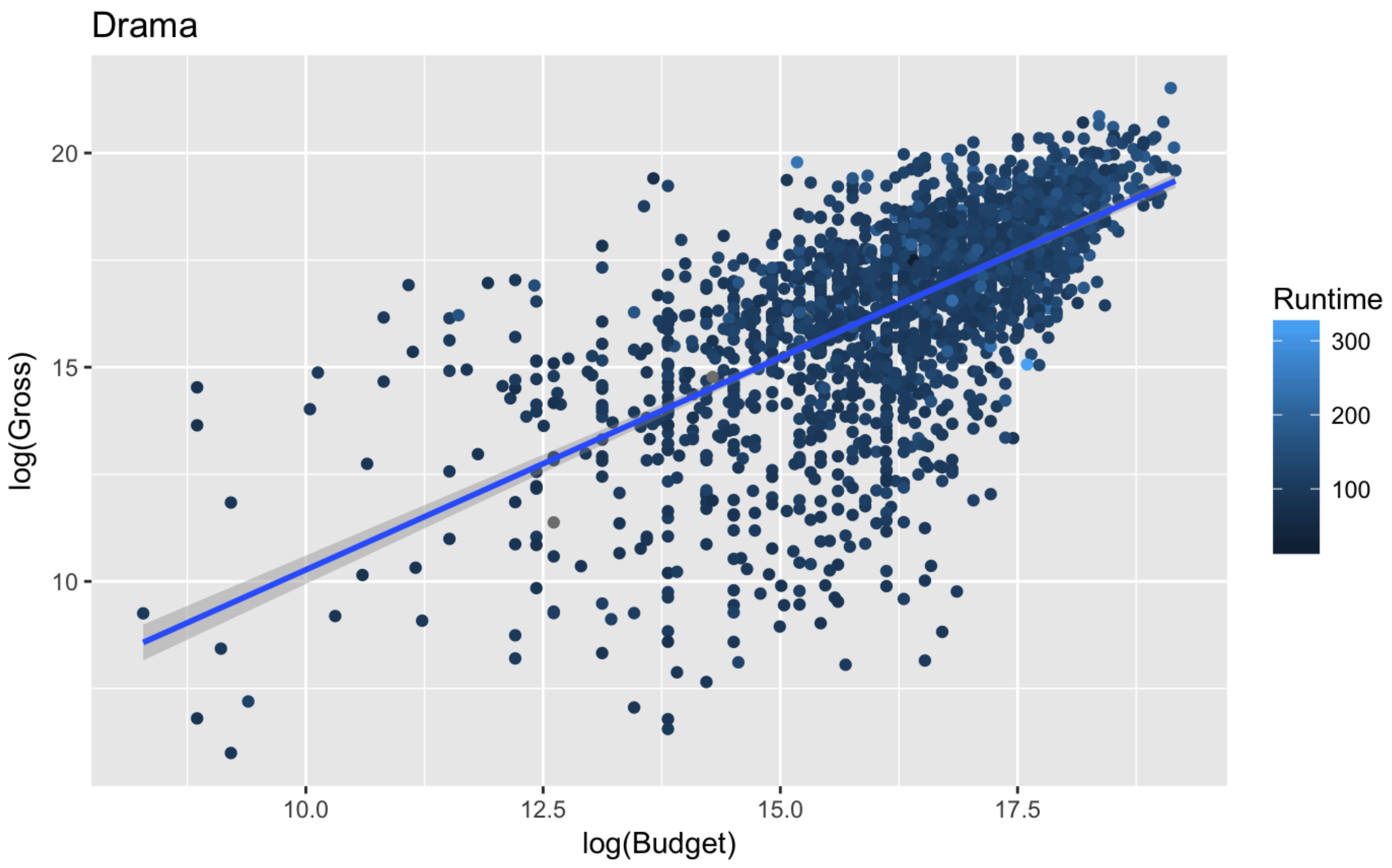
Hide

```
# Investigate if Gross Revenue is related to Budget, Runtime or Genre
# Work with a subset of df where there are values for Gross
df5 = df[is.na(df$Gross) == FALSE,]
df5 = df5[,c(1, 5, 35, 37, 39:68)]
# Separate outlier films who Grossed less than $100
df5 = df5[df5$Gross > 100,]
# Subset DataFrames by Runtime
df5s = df5[df5$Runtime < 80,]
df5m = df5[df5$Runtime >= 80 & df5$Runtime <= 120,]
df5l = df5[df5$Runtime > 120,]
# Build a new DataFrame based on df5m DataFrame with Runtimes for each Genre
df5d = df5[df5$drama == 1,]
df5ld = df5l[df5l$drama == 1,]
df5mac = df5m[df5m$action == 1,]
df5man = df5m[df5m$animation == 1,]
df5mc = df5m[df5m$comedy == 1,]
df5ho = df5[df5$horror == 1,]
df5sh = df5[df5$short == 1,]
df5th = df5[df5$thriller == 1,]
# Plot
ggplot(df5, aes(x=log(Budget), y=log(Gross), colour=Runtime)) + geom_point(na.rm=TRUE) +
 stat_smooth(method="lm")
```



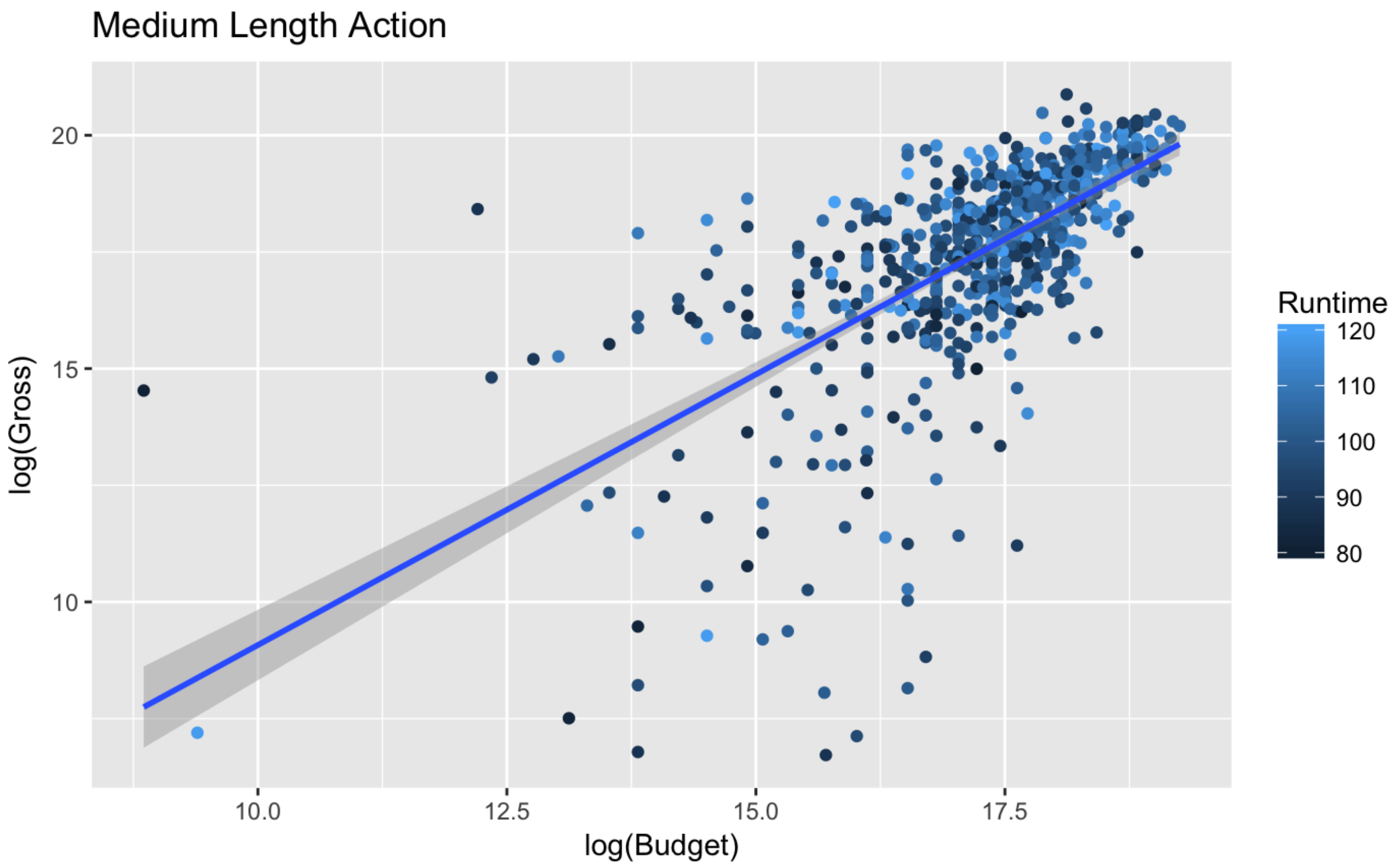
Hide

```
#ggplot(df5s, aes(x=log(Budget), y=log(Gross), colour=Runtime)) + geom_point(na.rm=TRUE)
#ggplot(df5m, aes(x=log(Budget), y=log(Gross), colour=Runtime)) + geom_point(na.rm=TRUE)
#ggplot(df5l, aes(x=log(Budget), y=log(Gross), colour=Runtime)) + geom_point(na.rm=TRUE)
ggplot(df5d, aes(x=log(Budget), y=log(Gross), colour=Runtime)) + geom_point(na.rm=TRUE)
+ labs(title="Drama") + stat_smooth(method="lm")
```



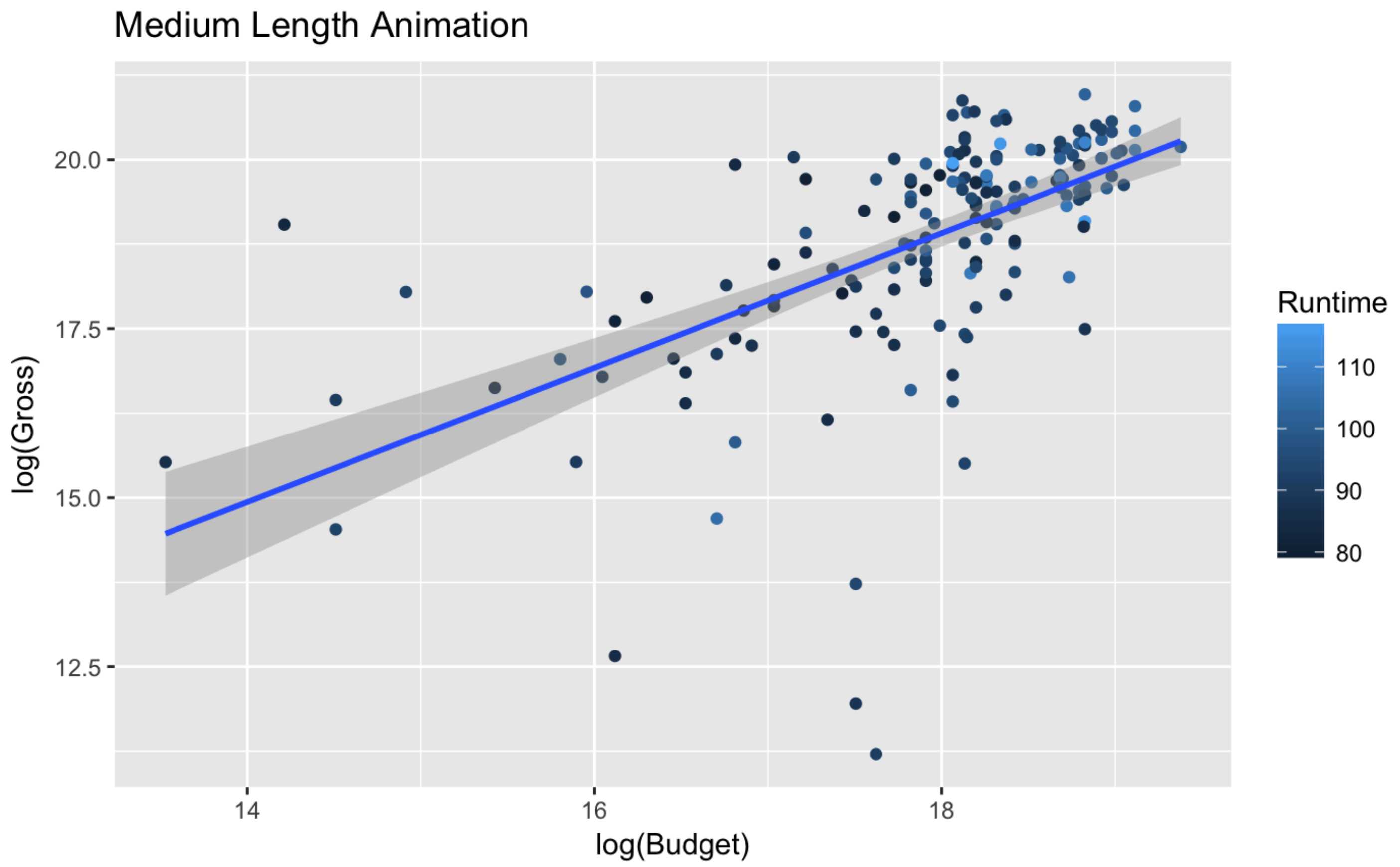
Hide

```
ggplot(df5mac, aes(x=log(Budget), y=log(Gross), colour=Runtime)) + geom_point(na.rm=TRUE
) + labs(title="Medium Length Action") + stat_smooth(method="lm")
```



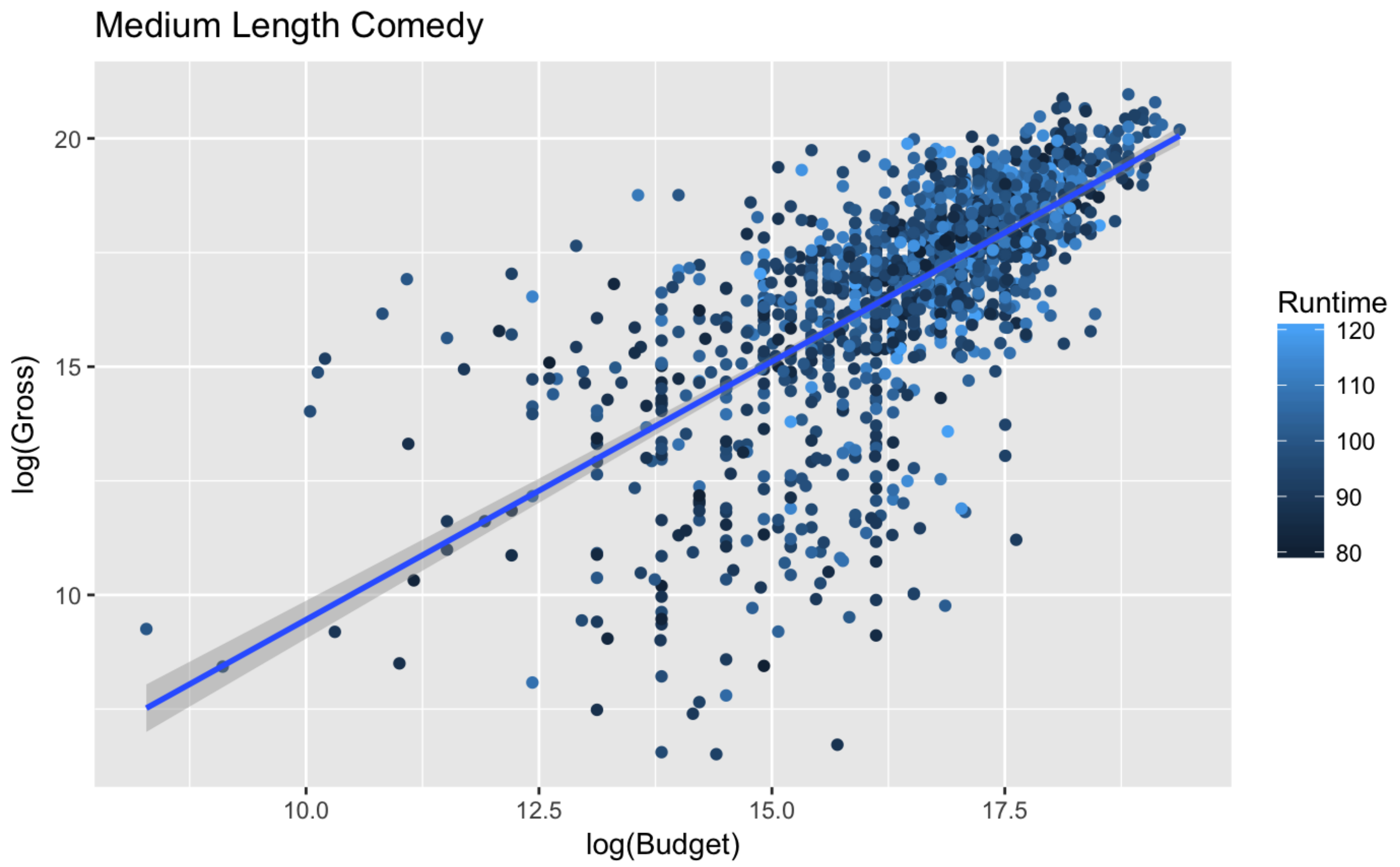
Hide

```
ggplot(df5man, aes(x=log(Budget), y=log(Gross), colour=Runtime)) + geom_point(na.rm=TRUE
) + labs(title="Medium Length Animation") + stat_smooth(method="lm")
```



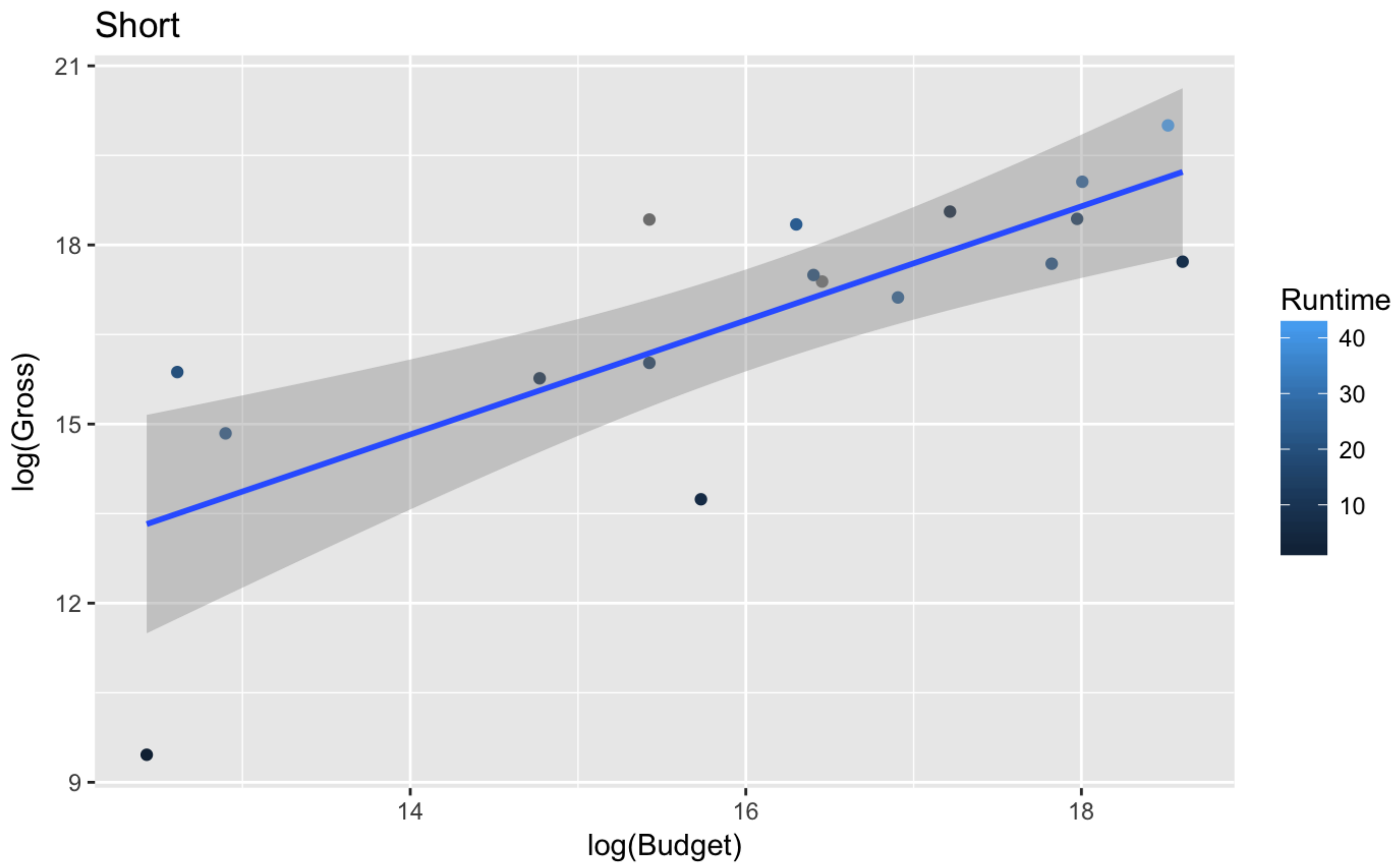
Hide

```
ggplot(df5mc, aes(x=log(Budget), y=log(Gross), colour=Runtime)) + geom_point(na.rm=TRUE)
 + labs(title="Medium Length Comedy") + stat_smooth(method="lm")
```



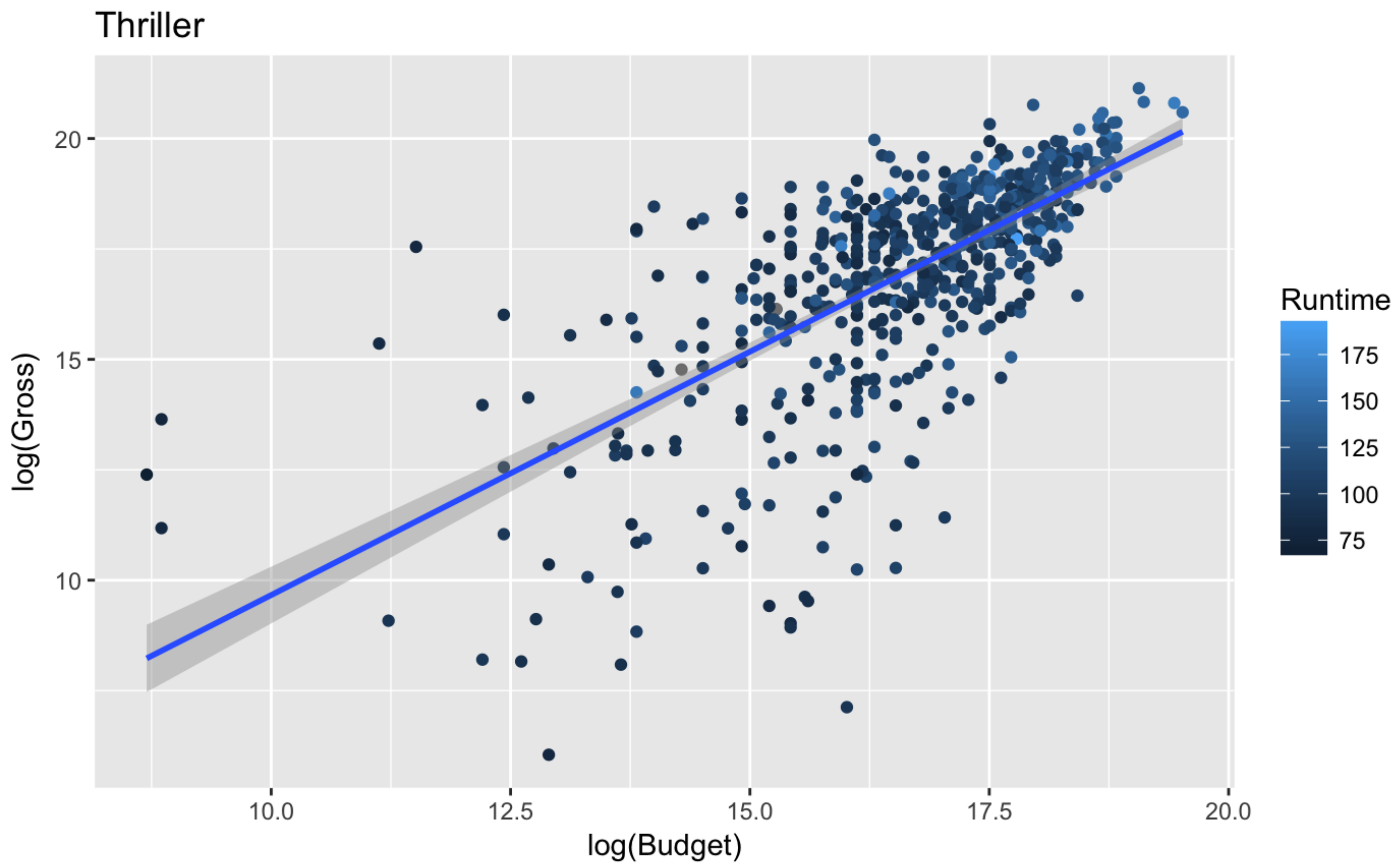
Hide

```
ggplot(df5sh, aes(x=log(Budget), y=log(Gross), colour=Runtime)) + geom_point(na.rm=TRUE)
 + labs(title="Short") + stat_smooth(method="lm")
```



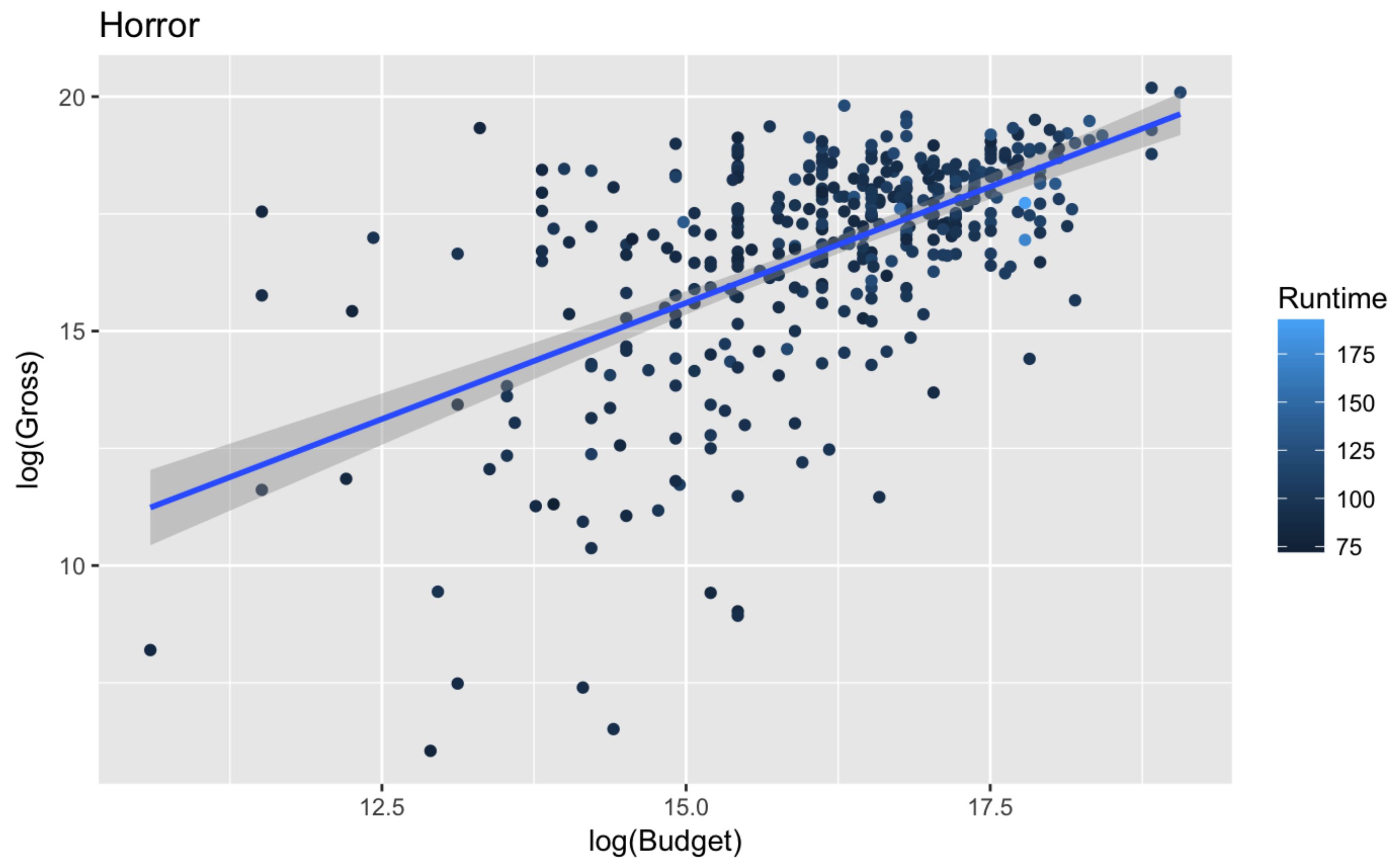
Hide

```
ggplot(df5th, aes(x=log(Budget), y=log(Gross), colour=Runtime)) + geom_point(na.rm=TRUE)
 + labs(title="Thriller") + stat_smooth(method="lm")
```



Hide

```
ggplot(df5ho, aes(x=log(Budget), y=log(Gross), colour=Runtime)) + geom_point(na.rm=TRUE)
 + labs(title="Horror") + stat_smooth(method="lm")
```



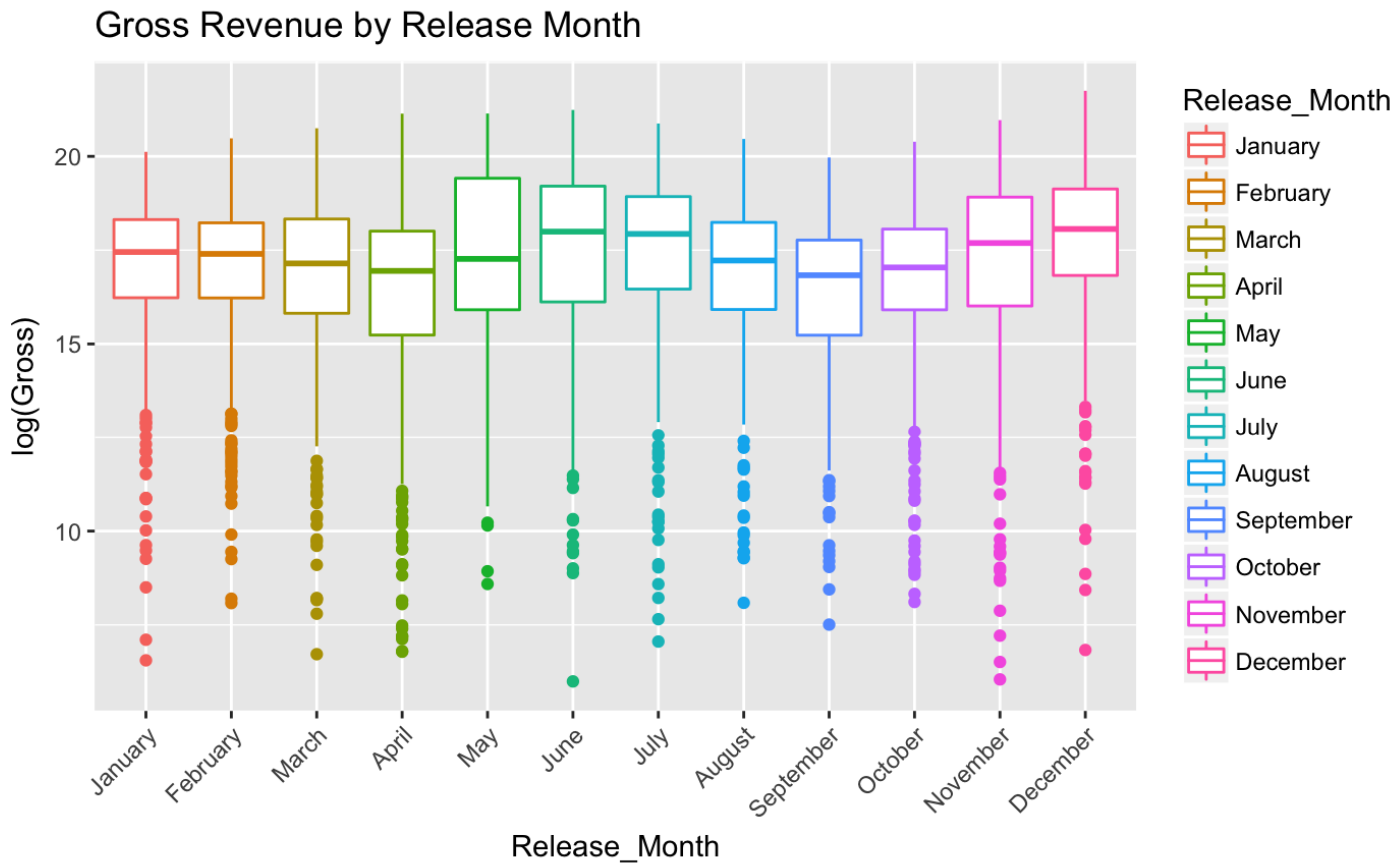
Q: Did you find any observable relationships or combinations of Budget/Runtime/Genre that result in high Gross revenue? If you divided the movies into different subsets, you may get different answers for them - point out interesting ones.

A: Looking at the log transforms of Gross Revenue vs Budget color coded by Runtime, I see a low positive correlation, where the variation reduces at higher Gross Revenue and Budget. This seems to track with the financial ideology of needing to “spend money to make money”. However, when you look at subsets of this plot by Genre some interesting information emerges. First, Medium Length Animation movies (medium length is between 80 and 120 minutes) tend to earn about the same Gross Revenue at all budgets ($500k to over $10m). Horror movie Gross Revenues aren’t as resilient to Budget as Medium Length Animation movies, but horror movies seem to be also a safe bet for production houses to invest in at lower budget.

In Addition, It appears that movies earn more gross revenue during the summer months of June and July and during the winter months of November and December, as shown in the boxplot below.

Hide

```
# Investigate if Gross Revenue is related to Release Month
# Add an order to df5%Release_Month
df5$Release_Month = factor( df5$Release_Month, levels = c( "January", "February", "Marc
h", "April", "May", "June", "July", "August", "September", "October", "November", "Decem
ber" ) )
# Do not include NA in analysis of Gross Revenue and Release_Month
df5rm_nona = df5[is.na(df5$Release_Month) == FALSE, ]
ggplot(df5rm_nona, aes(y=log(Gross), x=Release_Month, colour=Release_Month)) + geom_boxp
lot(na.rm=TRUE) + labs(title="Gross Revenue by Release Month") + theme(axis.text.x = ele
ment_text(angle = 45, hjust = 1))
```



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

```
# Convert Awards to 2 numeric columns: wins and nominations
df$Awards_Wins = regmatches(df$Awards, gregexpr("\\d+(?= (win))", df$Awards, perl=TRUE))
df$Awards_Wins = as.numeric(df$Awards_Wins)
df$Awards_Wins[is.na(df$Awards_Wins)] = 0
df$Awards_Wins1 = regmatches(df$Awards, gregexpr("(?<=\\b(Won)\\s)[0-9]", df$Awards, per
l=TRUE))
df$Awards_Wins1 = as.numeric(df$Awards_Wins1)
df$Awards_Wins1[is.na(df$Awards_Wins1)] = 0
df$Awards_Nomination = regmatches(df$Awards, gregexpr("\\d+(?= (nomination))", df$Awards
, perl=TRUE))
df$Awards_Nomination = as.numeric(df$Awards_Nomination)
df$Awards_Nomination[is.na(df$Awards_Nomination)] = 0
df$Awards_Nomination1 = regmatches(df$Awards, gregexpr("(?<=\\b(for)\\s)[0-9]", df$Award
s, perl=TRUE))
df$Awards_Nomination1 = as.numeric(df$Awards_Nomination1)
df$Awards_Nomination1[is.na(df$Awards_Nomination1)] = 0
df$Awards_NA = ifelse(df$Awards == "N/A", NA, 0)
df$Award_Wins = df$Awards_Wins + df$Awards_Wins1 + df$Awards_NA
df$Award_Nominations = df$Awards_Nomination + df$Awards_Nomination1 + df$Awards_NA
df[,c(12, 69, 70, 71, 72, 73)] = NULL
```

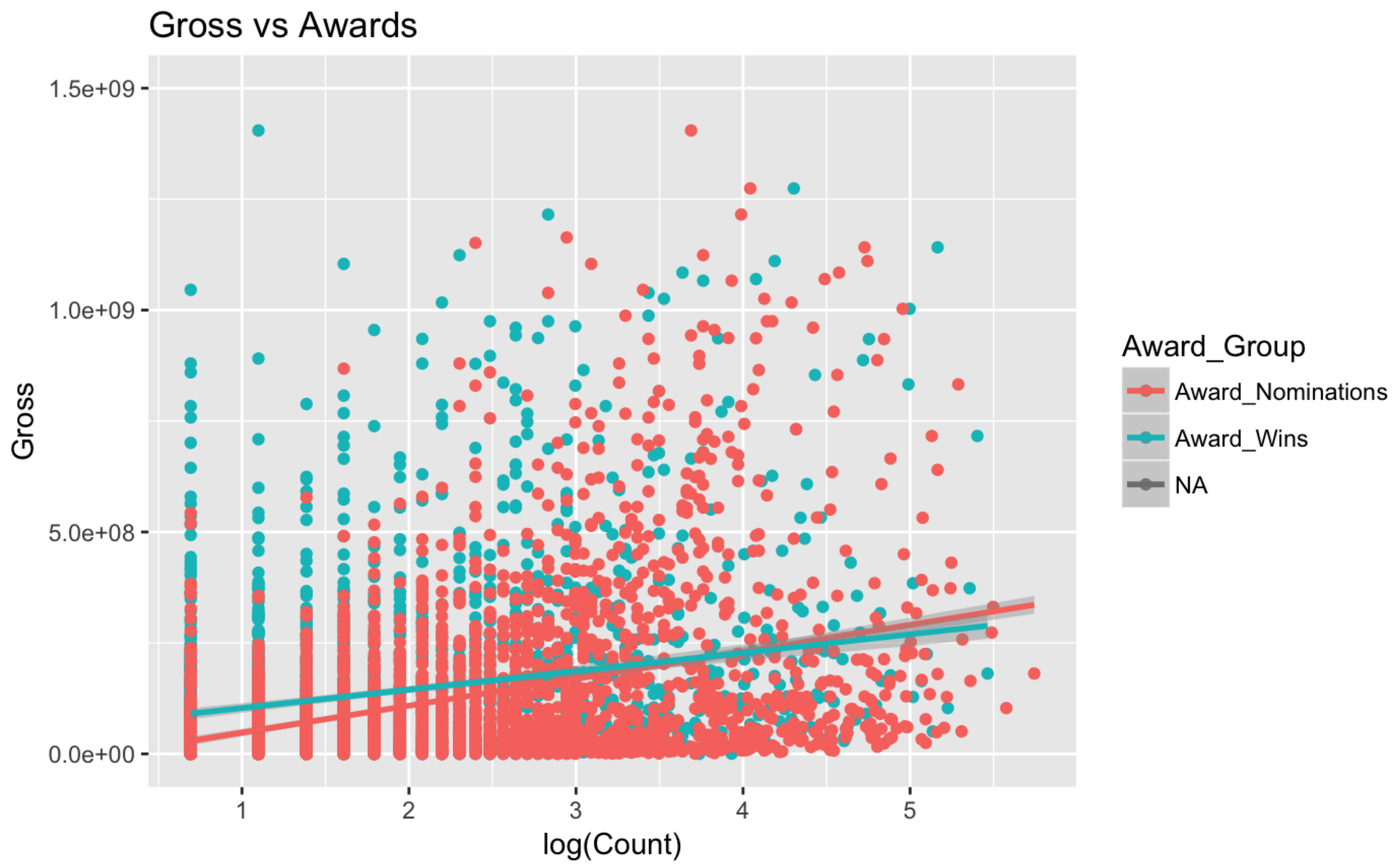
Q: How did you construct your conversion mechanism? How many rows had valid/non-zero wins or nominations?

A: I searched for and removed numbers preceeding “win” or “nomination” and numbers after “nominations for” (actually just “for”) and “Won”. I put all of these numbers into 4 columns (2 columns for nominations and 2 columns for wins). I also created a column for rows that had N/A. After creating these 5 new rows, I added the win columns and NA column and created a new column for wins, and I added the nomination columns and NA column and created a new column for nominations. I spot checked my two new columns for Wins (Award_Wins) and Nominations (Award_Nominations) vs the old Awards column for errors in groups of 50 rows across the entire length of the data frame.

Also, I found 13,958 rows with a win or a nomination (rows without N/A in the Awards column).

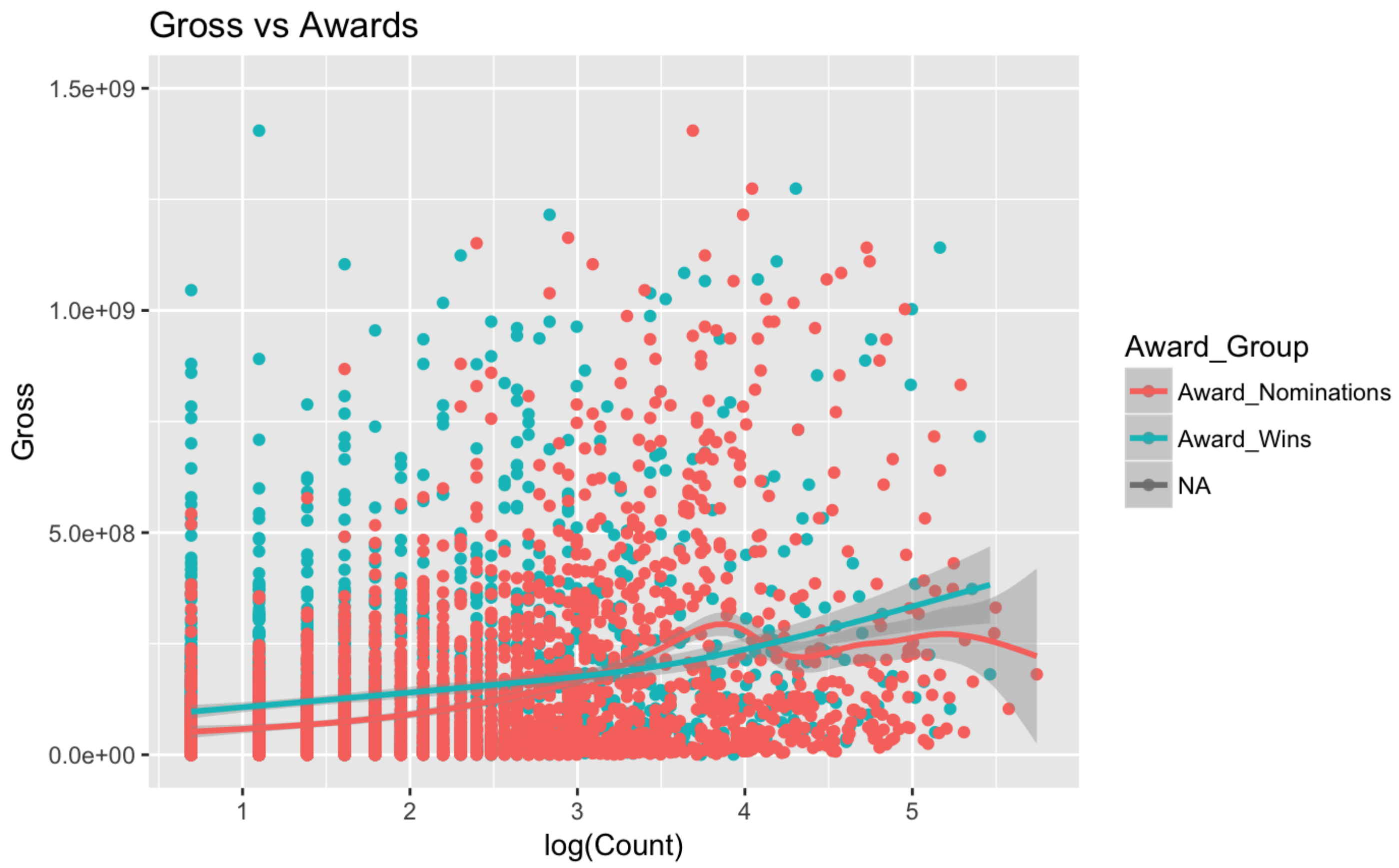
Hide

```
# Plot Gross revenue against wins and nominations
df6 = df[, c(36, 68, 69)]
df6 = gather(df6, Award_Group, Count, Award_Wins:Award_Nominations)
df6 = df6[df6$Gross > 100,]
df6 = df6[df6$Count > 1,]
ggplot(df6, aes(y=Gross, x=log(Count), colour=Award_Group)) + geom_point(na.rm=TRUE) + l
abs(title="Gross vs Awards") + stat_smooth(method="lm", na.rm=TRUE) + scale_y_continuous
(limits = c(-20, 1.5e+09))
```



Hide

```
ggplot(df6, aes(y=Gross, x=log(Count), colour=Award_Group)) + geom_point(na.rm=TRUE) + l
abs(title="Gross vs Awards") + stat_smooth(method="auto", na.rm=TRUE) + scale_y_continuo
us(limits = c(-20, 1.5e+09))
```



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

A: Looking at the log(Gross) vs the Log(Counts) (where Counts are the counts of Nominations or Awards) scatter plot, there is a low, and slightly positive correlation between the Gross Revenue of a film and the count of awards. There is certainly much more variation in the Gross Revenue at low counts of awards and less variation in Gross Revenue at high counts of awards.

7. Movie ratings from IMDb and Rotten Tomatoes

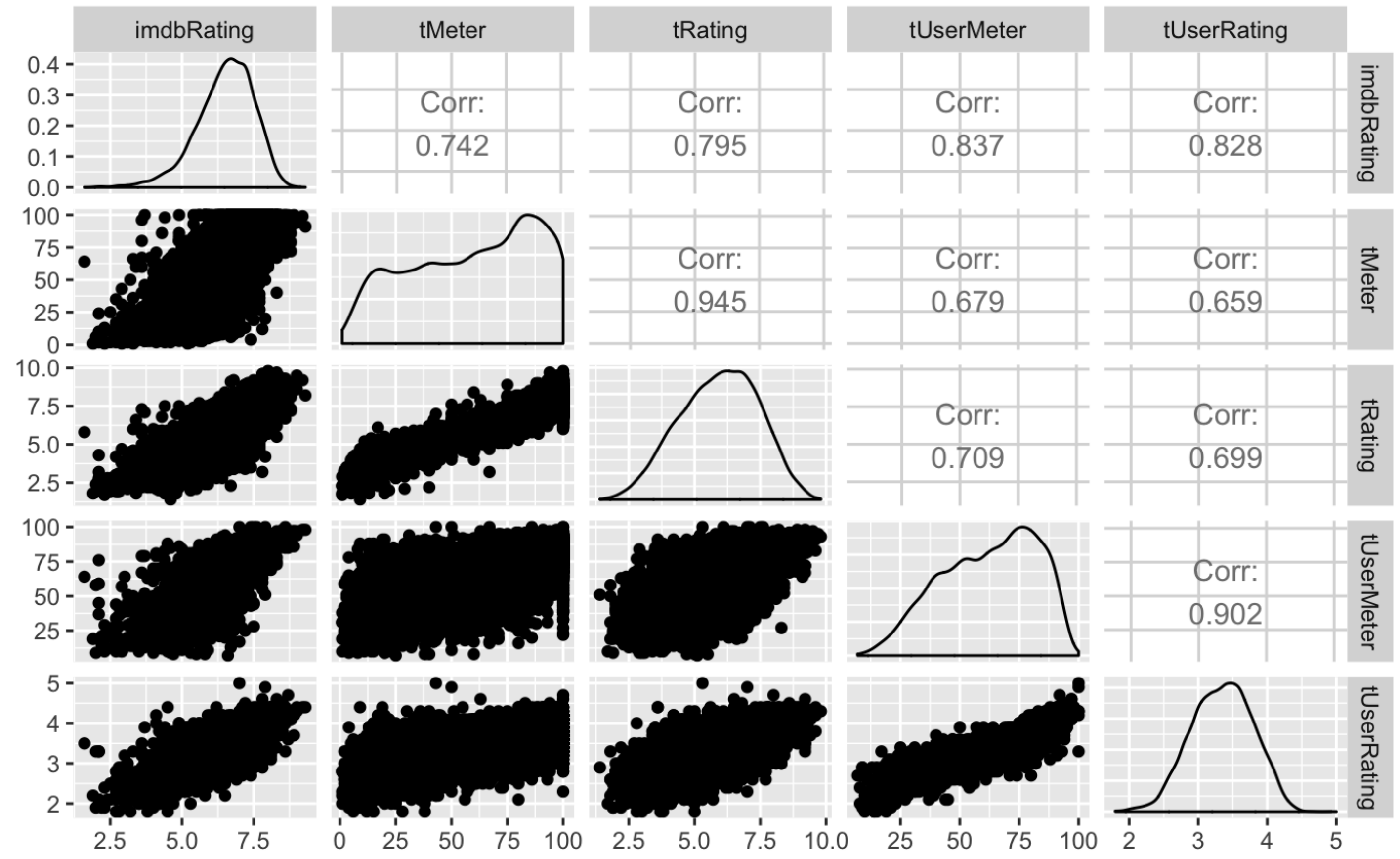
There are several variables that describe ratings, including IMDb ratings (`imdbRating` represents average user ratings and `imdbVotes` represents the number of user ratings), and multiple Rotten Tomatoes ratings (represented by several variables pre-fixed by `tomato`). Read up on such ratings on the 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

```
# Illustrate how ratings from IMDb and Rotten Tomatoes are related
# Work with a data frame specifically for this question to mask changes that are not des
ired in df
df7 = df[, c(14, 15, 18, 19, 20, 25, 26)]  # 22, 23 are counts of rotten or fresh that a
re used to calc tomatoMeter
# Remove NA and 0's as they tend to skew the data
df7 = df7[df7$imdbRating > 0,]
df7 = df7[is.na(df7$imdbRating) == FALSE,]
df7 = df7[df7$tomatoMeter > 0,]
df7 = df7[is.na(df7$tomatoMeter) == FALSE,]
df7 = df7[df7$tomatoRating > 0,]
df7 = df7[is.na(df7$tomatoRating) == FALSE,]
df7 = df7[df7$tomatoUserMeter > 0,]
df7 = df7[is.na(df7$tomatoUserMeter) == FALSE,]
df7 = df7[df7$tomatoUserRating > 0,]
df7 = df7[is.na(df7$tomatoUserRating) == FALSE,]
ggpairs(df7, columns = c("imdbRating", "tomatoMeter", "tomatoRating", "tomatoUserMeter",
 "tomatoUserRating"), columnLabels = c("imdbRating", "tMeter", "tRating", "tUserMeter",
"tUserRating"))
```

```
 plot: [1,1] [===-----------------------------------------------------------------------
----]  4% est: 0s
 plot: [1,2] [======--------------------------------------------------------------------
----]  8% est: 1s
 plot: [1,3] [=========-----------------------------------------------------------------
----] 12% est: 1s
 plot: [1,4] [============--------------------------------------------------------------
----] 16% est: 1s
 plot: [1,5] [================----------------------------------------------------------
----] 20% est: 2s
 plot: [2,1] [===================-------------------------------------------------------
----] 24% est: 1s
 plot: [2,2] [======================----------------------------------------------------
----] 28% est: 1s
 plot: [2,3] [=========================-------------------------------------------------
----] 32% est: 1s
 plot: [2,4] [============================----------------------------------------------
----] 36% est: 1s
 plot: [2,5] [===============================-------------------------------------------
----] 40% est: 1s
 plot: [3,1] [==================================----------------------------------------
----] 44% est: 1s
 plot: [3,2] [=====================================-------------------------------------
----] 48% est: 1s
 plot: [3,3] [=========================================---------------------------------
----] 52% est: 1s
 plot: [3,4] [============================================------------------------------
----] 56% est: 1s
 plot: [3,5] [===============================================---------------------------
----] 60% est: 1s
 plot: [4,1] [==================================================------------------------
----] 64% est: 1s
 plot: [4,2] [=====================================================---------------------
----] 68% est: 1s
 plot: [4,3] [========================================================------------------
----] 72% est: 1s
 plot: [4,4] [===========================================================---------------
----] 76% est: 1s
 plot: [4,5] [==============================================================------------
----] 80% est: 0s
 plot: [5,1] [==================================================================--------
----] 84% est: 0s
 plot: [5,2] [=====================================================================-----
----] 88% est: 0s
 plot: [5,3] [========================================================================--
----] 92% est: 0s
 plot: [5,4] [==========================================================================
=---] 96% est: 0s
 plot: [5,5] [==========================================================================
====]100% est: 0s
```



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

A: I have plotted imdbRating vs a variety of Tomato ratings. All pairings have a positive correlation and a pretty high correlation coefficient. imdbRating vs tomatoUserRating has one of the highest correlation coefficients at 0.828. Furthermore, users rated each movie at about the same value (should they both be on the same scale, 0-100 or 0-10). Intuitively, it makes sense that two similar user ratings on two similar, popular sites (IMDB - imdbRating and Rotten Tomato - tomatoUserRating) gives roughly the same score on each site for each movie. On the other hand, when the tomatoMeter (a score obtained from movie critics only) is compared to the imdbRating, a slightly more positive (i.e. larger slope of a fit line) correlation is obtained. This means that Tomato Critics tend to give movies a higher rating than IMDB users do.

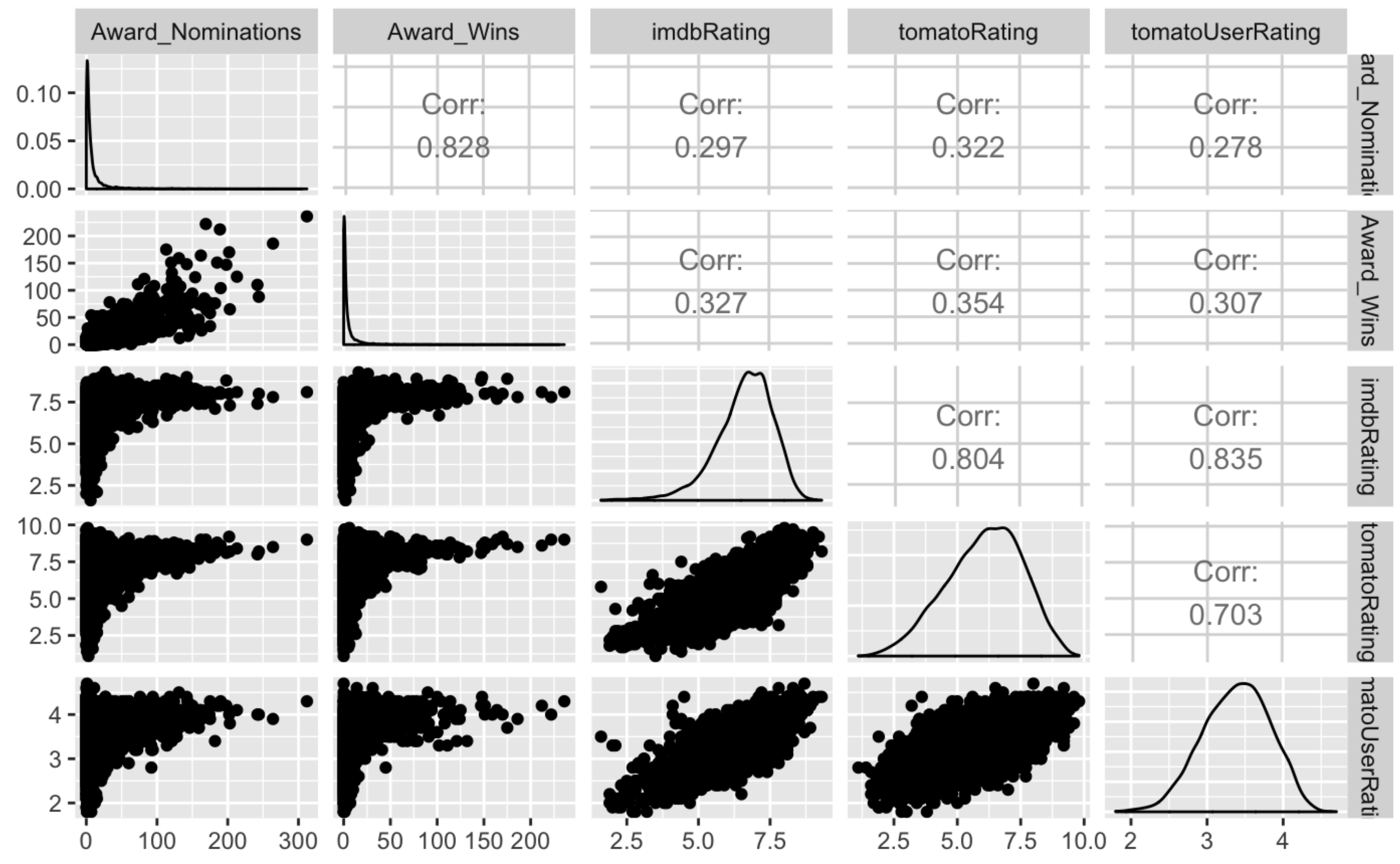
8. Ratings and awards

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

```
# Show how ratings and awards are related
# Work with a data frame specifically for this question to mask changes that are not des
ired in df
df8 = df[, c(14, 20, 26, 68, 69)]
# Remove NA and 0's as they tend to skew the data
df8 = df8[df8$imdbRating > 0,]
df8 = df8[is.na(df8$imdbRating) == FALSE,]
df8 = df8[df8$tomatoRating > 0,]
df8 = df8[is.na(df8$tomatoRating) == FALSE,]
df8 = df8[df8$tomatoUserRating > 0,]
df8 = df8[is.na(df8$tomatoUserRating) == FALSE,]
df8 = df8[is.na(df8$Award_Nominations) == FALSE,]
df8 = df8[is.na(df8$Award_Wins) == FALSE,]
#df8$Award_Wins_trans = (df8$Award_Wins)^0.5
#df8$Award_Nominations_trans = (df8$Award_Nominations)^0.5
#df8$Award_Wins_trans = log(df8$Award_Wins + 1)
#df8$Award_Nominations_trans = log(df8$Award_Nominations + 1)
#ggpairs(df8, columns = c("Award_Nominations_trans", "Award_Wins_trans", "imdbRating",
 "tomatoRating", "tomatoUserRating"), columnLabels = c("Award_Nominations_trans", "Award
_Wins_trans", "imdbRating", "tomatoRating", "tomatoUserRating"), )
ggpairs(df8, columns = c("Award_Nominations", "Award_Wins", "imdbRating", "tomatoRating"
, "tomatoUserRating"), columnLabels = c("Award_Nominations", "Award_Wins", "imdbRating",
 "tomatoRating", "tomatoUserRating"),)
 plot: [1,1] [===-----------------------------------------------------------------------
----]  4% est: 0s
 plot: [1,2] [======--------------------------------------------------------------------
----]  8% est: 1s
 plot: [1,3] [=========-----------------------------------------------------------------
----] 12% est: 2s
 plot: [1,4] [============--------------------------------------------------------------
----] 16% est: 2s
 plot: [1,5] [================----------------------------------------------------------
----] 20% est: 2s
 plot: [2,1] [===================-------------------------------------------------------
----] 24% est: 1s
 plot: [2,2] [======================----------------------------------------------------
----] 28% est: 1s
 plot: [2,3] [=========================-------------------------------------------------
----] 32% est: 1s
 plot: [2,4] [============================----------------------------------------------
----] 36% est: 1s
 plot: [2,5] [===============================-------------------------------------------
----] 40% est: 1s
 plot: [3,1] [==================================----------------------------------------
----] 44% est: 1s
 plot: [3,2] [=====================================-------------------------------------
----] 48% est: 1s
 plot: [3,3] [=========================================---------------------------------
----] 52% est: 1s
 plot: [3,4] [============================================------------------------------
----] 56% est: 1s
 plot: [3,5] [===============================================---------------------------
----] 60% est: 1s
 plot: [4,1] [==================================================------------------------
----] 64% est: 1s
 plot: [4,2] [=====================================================---------------------
----] 68% est: 1s
 plot: [4,3] [========================================================------------------
----] 72% est: 1s
 plot: [4,4] [===========================================================---------------
----] 76% est: 1s
 plot: [4,5] [==============================================================------------
----] 80% est: 0s
 plot: [5,1] [==================================================================--------
----] 84% est: 0s
 plot: [5,2] [=====================================================================-----
----] 88% est: 0s
 plot: [5,3] [========================================================================--
----] 92% est: 0s
 plot: [5,4] [==========================================================================
=---] 96% est: 0s
 plot: [5,5] [==========================================================================
====]100% est: 0s
```



Hide

```
#ggpairs(df8, columns = c("Award_Nominations_recip", "Award_Wins_recip", "imdbRating",
 "tomatoRating", "tomatoUserRating"), columnLabels = c("Award_Nominations_recip", "Award
_Wins_recip", "imdbRating", "tomatoRating", "tomatoUserRating"), )
```

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: Note: The distributions of the two awards columns are very strongly right-skewed. I used a log(x+1) transformation (the “+ 1” portion is used to handle) to obtain a more Gaussian-like distribution for these variables before using ggpairs to look at correlations. This transformation is not perfect, but it works better than the other transformations for right-skewed distributions (e.g. sqrt(x+0.5), reciprocal).

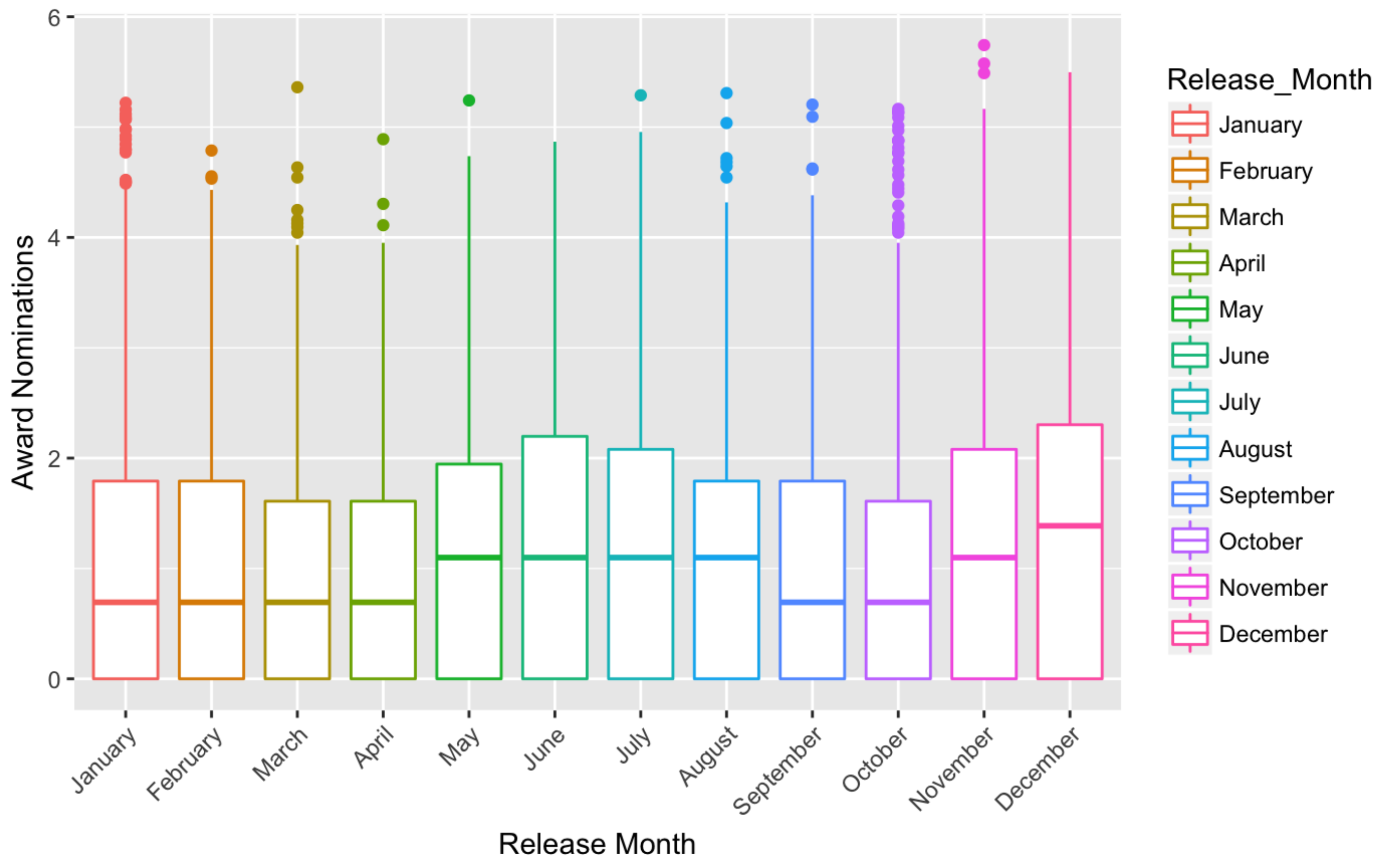
All correlations between the Awards columns and the various Ratings columns give a pretty low, positive correlation. The highest correlation coefficient is obtained for tomatoRating and log(Award_Wins +1). This is probably what one would expect as the tomatoRating is obtained from a relatively small group of professional movie critics and Award_Wins are given by an even smaller group of professional movie critics. In the case of all pairings between Awards and Ratings, there is a lot of variability on the low end of Awards/Ratings.

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.

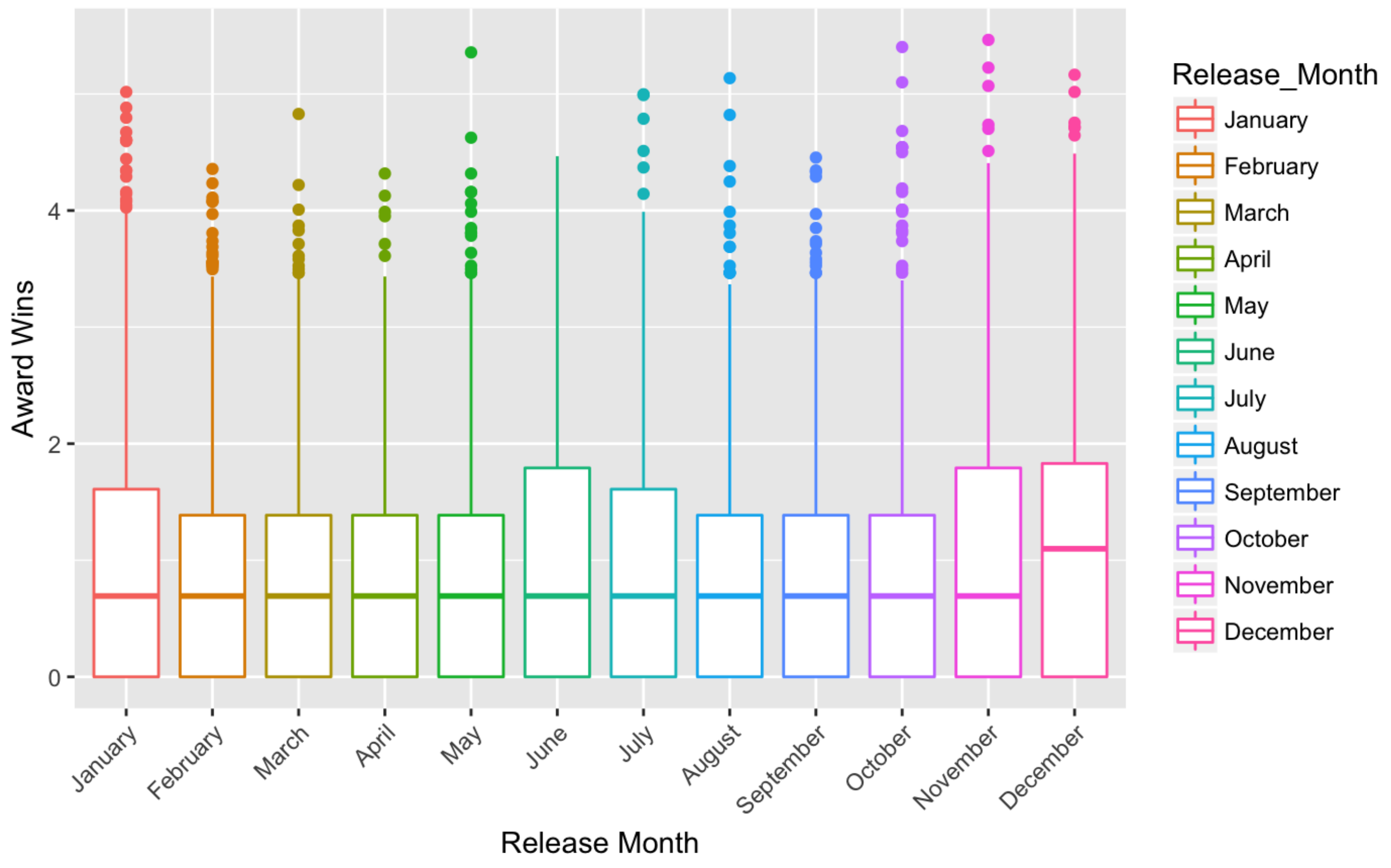
Hide

```
# Find and illustrate two expected insights
# Insight 1
df9 = df
# Add an order to df5%Release_Month
df9$Release_Month = factor( df9$Release_Month, levels = c( "January", "February", "Marc
h", "April", "May", "June", "July", "August", "September", "October", "November", "Decem
ber" ) )
# Do not include NA in analysis of Gross Revenue and Release_Month
df9 = df9[is.na(df9$Release_Month) == FALSE, ]
#df9 = df9[df9$Award_Wins > 20, ]
# Plot
ggplot(df9, aes(x=Release_Month, y=log(Award_Nominations), colour=Release_Month)) + geom
_boxplot(na.rm = TRUE) + scale_y_continuous() + labs(x="Release Month", y="Award Nominat
ions") + theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



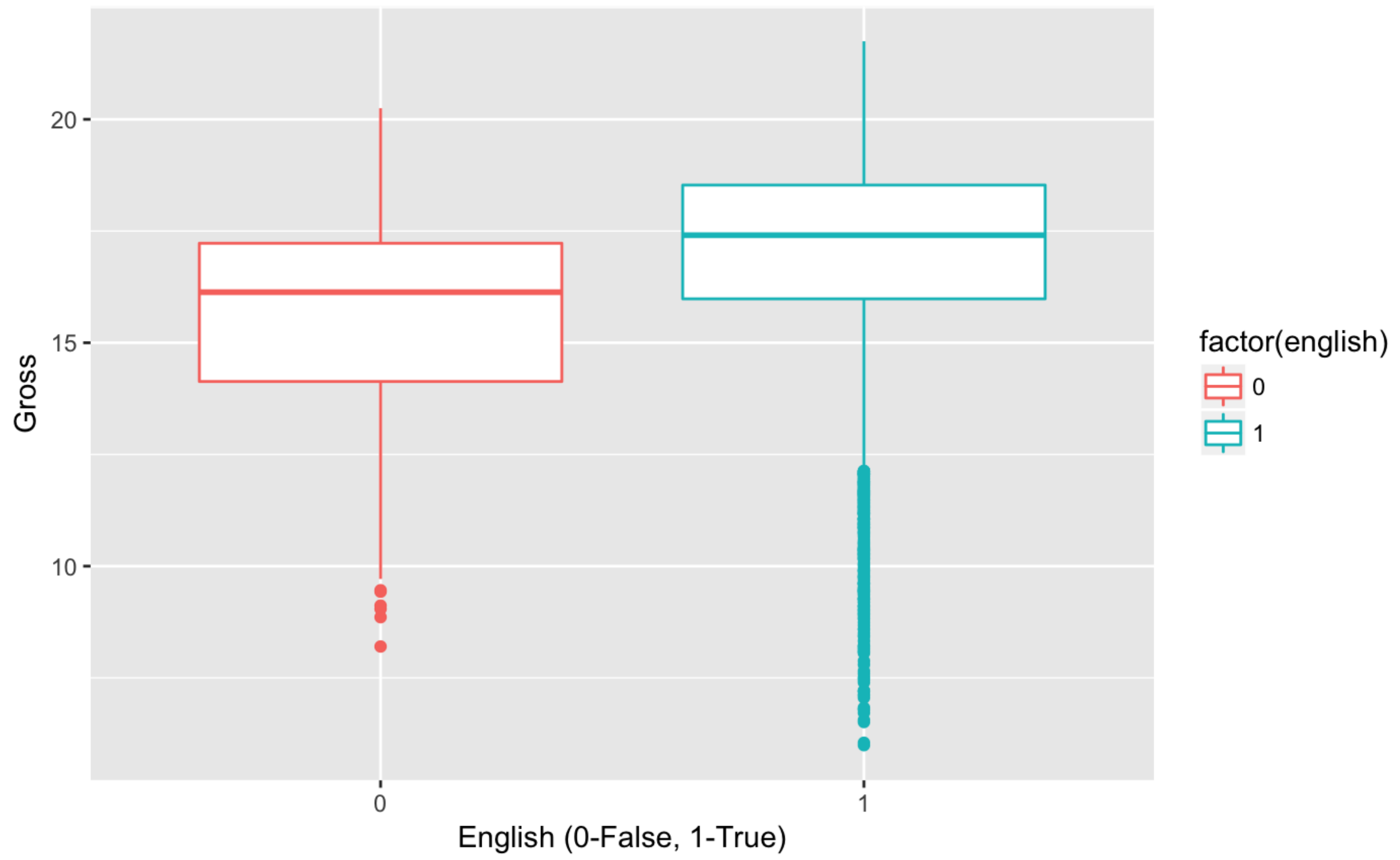
Hide

```
ggplot(df9, aes(x=Release_Month, y=log(Award_Wins), colour=Release_Month)) + geom_boxplo
t(na.rm = TRUE) + scale_y_continuous() + labs(x="Release Month", y="Award Wins") + theme
(axis.text.x = element_text(angle = 45, hjust = 1))
```


Release_Month
January
February
March
April
May
June
July
August
September
October
November
December
Award Wins
0
2
4
Release Month

Hide

```
# Insight 2
df9b = df
# Create a vector of the df$Language column
language = df$Language
# Create a dictionary of all possible languages
languages = unlist(strsplit(language, ", "))
languages_dict = as.list(unique(languages))
# Create a list of languages for each movie
languages_l = strsplit(language, ", ")
# Build the Corpus
language_corp = Corpus(VectorSource(languages_l))
# Build the DocumentTermMatrix
language_dtm = DocumentTermMatrix(language_corp, languages_dict)
# Convert DocumentTermMatrix to a DataFrame
language_dtm_matrix = as.matrix(language_dtm)
language_dtm_df = as.data.frame(language_dtm_matrix)
# Add a join key to both DataFrames
language_dtm_df$JoinKey = seq(1, dim(language_dtm_df)[1], by=1)
df9b$JoinKey = seq(1, dim(df9b)[1], by=1)
# Merge the two DataFrames
df9b = merge(x=df9b, y=language_dtm_df, by.x="JoinKey", by.y= "JoinKey")
# Drop JoinKey and Genre
df9b$JoinKey = NULL
df9b[,c(71, 72, 81, 193)] = NULL
df9bs = df9b[,c(10, 36, 120)]
df9bs$english = gsub(2, 1, df9bs$english)
df9bs$english = as.numeric(as.character(df9bs$english))
ggplot(df9bs, aes(x=factor(english), y=log(Gross), color=factor(english))) + geom_boxplo
t(na.rm = TRUE) + scale_y_continuous() + labs(x="English (0-False, 1-True)", y="Gross")
```



Q: Expected insight #1.

A: Looking at the two boxplots above for Award Wins by Release Month and Award Nominations by Release Month, movies released in December have a higher median than other months. I did not realize this fact, but some online research informed me that “Oscar Season” is actually from the beginning of November to the end of December every year. “Oscar Season” is apparently the time of year that Hollywood considers to be the optimal Release Date for movies with Oscar aspirations.

Q: Expected insight #2.

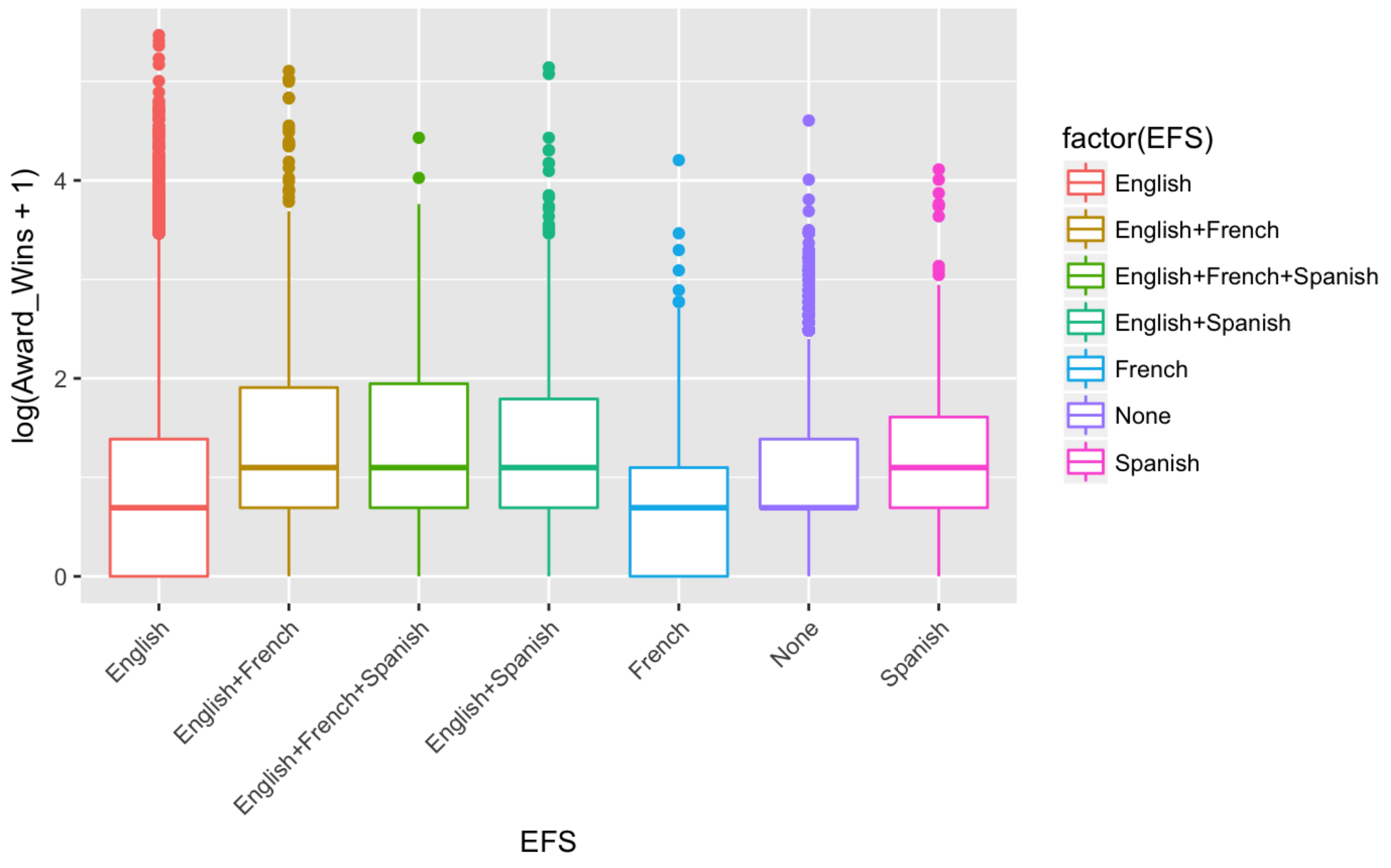
A: The boxplot which shows Gross Revenue by English (True or False) shows clearly that movies earn a higher Gross Revenue when also released in English. Releasing a movie in English does appear to have a considerable effect on Gross Revenue which is expected because the English langauge is the second most spoken language around the world (and more secondary speakers than any other language). Furthermore, many countries, where english is spoken as a primary or secondary language, also have very high GDP per capita. These two facts lead me to believe that this result is expected.

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.

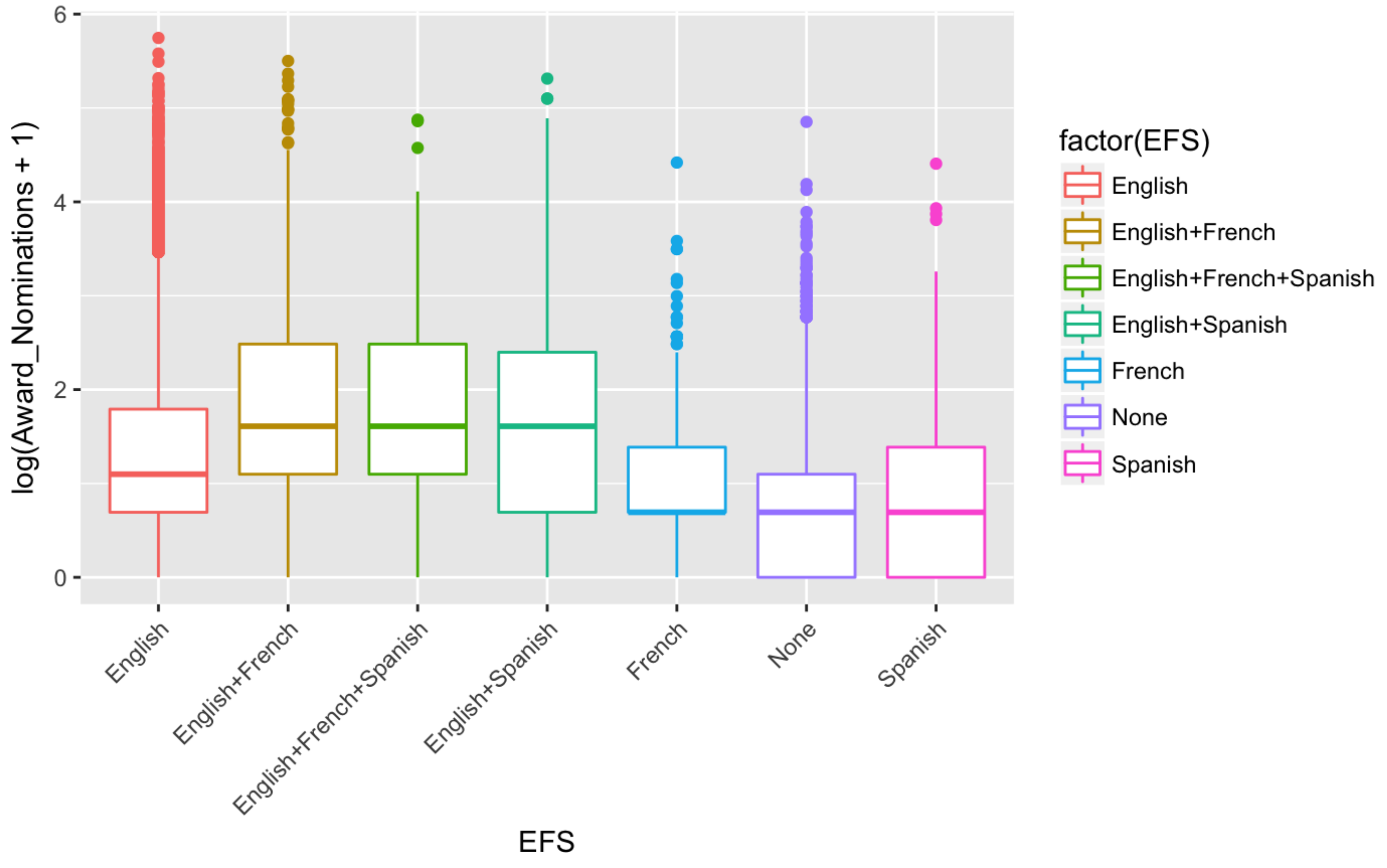
Hide

```
# Find and illustrate one unexpected insight
#df3_rp = data.frame(Genre_Sum=colSums(df[39:67], na.rm = TRUE))
df10 = df9b[, c(1:3, 70:274)]
# Find top languages
df10_ls = data.frame(Language_Sum=colSums(df10[4:208], na.rm = TRUE))
df10_ls = rownames_to_column(df10_ls, var = "Language")
df10_ls = df10_ls[order(df10_ls$Language_Sum, decreasing = TRUE),]
# Build a column with English/French/Spanish Language information
df10f = df9b[, c(2, 10, 34, 36, 68, 69, 120, 128, 245)]
df10f$english = gsub(2, 1, df10f$english)
df10f$english = as.numeric(as.character(df10f$english))
df10f$french = as.numeric(as.character(df10f$french))
df10f$french = ifelse(df10f$french == 1, 5, 0)
df10f$spanish = ifelse(df10f$spanish == 1, 10, 0)
df10f$EFS = df10f$english + df10f$french + df10f$spanish
# Filter out low count combinations
df10f = df10f[df10f$EFS != 15,]
df10f$EFS = gsub(10, "Spanish", df10f$EFS)
df10f$EFS = gsub("11", "English+Spanish", df10f$EFS)
df10f$EFS = gsub("16", "English+French+Spanish", df10f$EFS)
df10f$EFS = gsub("0", "None", df10f$EFS)
df10f$EFS = gsub("1", "English", df10f$EFS)
df10f$EFS = gsub("5", "French", df10f$EFS)
df10f$EFS = gsub("6", "English+French", df10f$EFS)
df10fw = df10f[is.na(df10f$Award_Wins) == FALSE,]
ggplot(df10fw, aes(x=EFS, y=log(Award_Wins + 1), colour=factor(EFS))) + geom_boxplot() +
 theme(axis.text.x = element_text(angle = 45, hjust = 1))#scale_y_continuous() + scale_x
_continuous() + labs(title="Award Wins over Time by Language") + stat_smooth(method="aut
o", na.rm=TRUE, se=FALSE) #+ stat_summary(fun.y=mean, geom="line", size = 2)
```



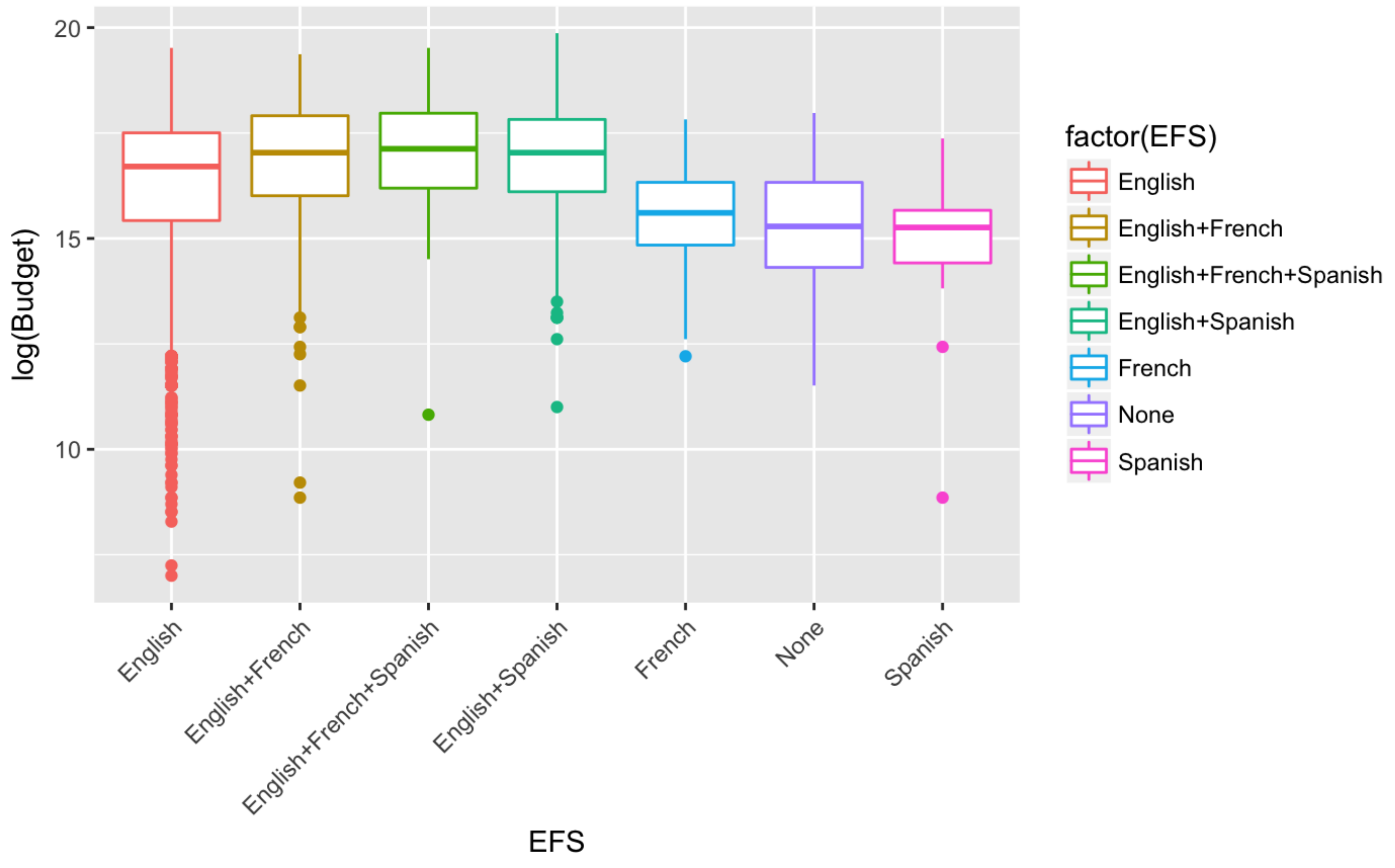
Hide

```
df10fn = df10f[is.na(df10f$Award_Nominations) == FALSE,]
ggplot(df10fn, aes(x=EFS, y=log(Award_Nominations + 1), colour=factor(EFS))) + geom_boxp
lot() + theme(axis.text.x = element_text(angle = 45, hjust = 1))#scale_y_continuous() +
 scale_x_continuous() + labs(title="Award Wins over Time by Language") + stat_smooth(met
hod="auto", na.rm=TRUE, se=FALSE) #+ stat_summary(fun.y=mean, geom="line", size = 2)
```



Hide

```
df10fb = df10f[is.na(df10f$Budget) == FALSE,]
ggplot(df10fb, aes(x=EFS, y=log(Budget), colour=factor(EFS))) + geom_boxplot(na.rm=TRUE)
 + theme(axis.text.x = element_text(angle = 45, hjust = 1))#scale_y_continuous() + scale
_x_continuous() + labs(title="Award Wins over Time by Language") + stat_smooth(method="a
uto", na.rm=TRUE, se=FALSE) #+ stat_summary(fun.y=mean, geom="line", size = 2)
```



Q: Unexpected insight.

A: Please refer to the box plots for Award Wins and Award Nominations and Budget by a few of the most common release languages, where Award Wins and Award Nomination were transformed by log(x+1) and Budget was transformed by log(x). In the Award Nominations boxplot, a movie released in english or any combination of english and the other two languages tends to be nominated more often than when the language is only French or Spanish, and this result is expected as many major film Awards committees are based in English speaking countries (e.g. United States). However, I did not expect the result of the Award Wins plots, which shows that a Spanish speaking movie is more likely to actually win more awards when compared to an English speaking movie. Note: The sample size of French and Spanish speaking movies is a lot smaller than the other groups, but the sample size is still statistically valid.