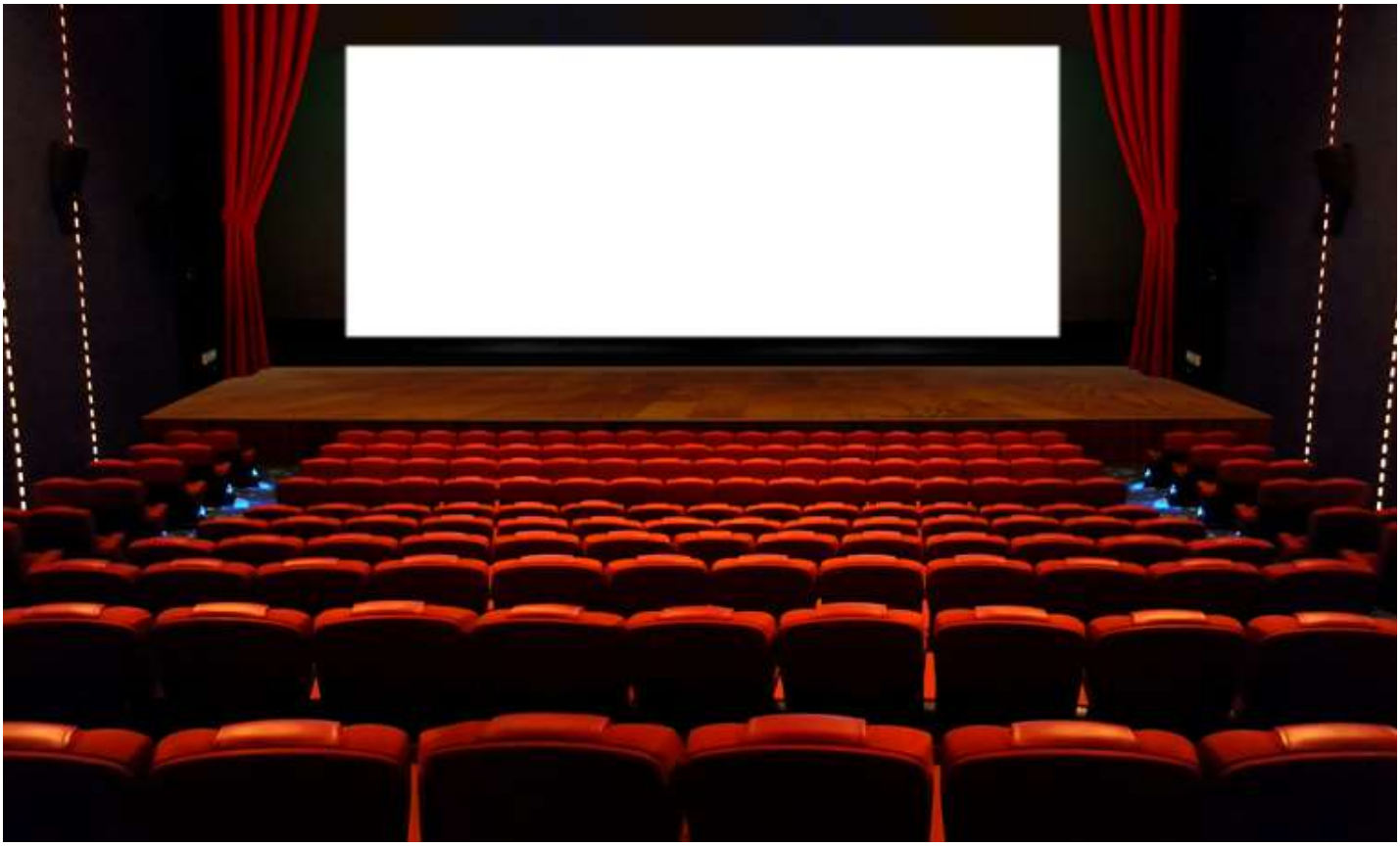
IMDB- Data Analysis

Kirill Bulgakov, Michal Cohen and Shachar Schneider

June 2021



Introduction

In our project we will examine this (https://raw.githubusercontent.com/itsmichalc/IMDB_movies_project/main/imdb_top_1000.csv) data set describing 1000 movies. This data set comes from the IMDB website and it contains a great deal of intriguing information about movies such as: popularity of genre, gross, IMDB rating, MetaScore etc.

Goals

In today's world, we know that a successful movie not only entertains the audience but also enables film companies to make a profit. Many factors can affect the gross such as directors, the year of making, ratings, and the genre.

Our research will examine how different parameters affect the gross of a movie, specifically popular genres. In addition, we will learn how the Meta Score rating is compared to the IMDB rating. Finally, we will explore the correlation between the number of votes to the IMDB rating and the revenue.

Tidying

Let's get to know our data set by looking at it briefly.

```
raw_data<-read.csv(file='C:/Users/bulga/Downloads/imdb_top_1000.csv')
glimpse(raw_data)
```

```
## Rows: 1,000
## Columns: 16
## $ Poster_Link   <chr> "https://m.media-amazon.com/images/M/MV5BMDFkYTc0MGEtZmN~
## $ Series_Title  <chr> "The Shawshank Redemption", "The Godfather", "The Dark K~
## $ Released_Year <chr> "1994", "1972", "2008", "1974", "1957", "2003", "1994", ~
## $ Certificate   <chr> "A", "A", "UA", "A", "U", "U", "A", "A", "UA", "A", "U",~
## $ Runtime       <chr> "142 min", "175 min", "152 min", "202 min", "96 min", "2~
## $ Genre         <chr> "Drama", "Crime, Drama", "Action, Crime, Drama", "Crime,~
## $ IMDB_Rating   <dbl> 9.3, 9.2, 9.0, 9.0, 9.0, 8.9, 8.9, 8.9, 8.8, 8.8, 8.8, 8~
## $ Overview      <chr> "Two imprisoned men bond over a number of years, finding~
## $ Meta_score    <int> 80, 100, 84, 90, 96, 94, 94, 94, 74, 66, 92, 82, 90, 87,~
## $ Director      <chr> "Frank Darabont", "Francis Ford Coppola", "Christopher N~
## $ Star1         <chr> "Tim Robbins", "Marlon Brando", "Christian Bale", "Al Pa~
## $ Star2         <chr> "Morgan Freeman", "Al Pacino", "Heath Ledger", "Robert D~
## $ Star3         <chr> "Bob Gunton", "James Caan", "Aaron Eckhart", "Robert Duv~
## $ Star4         <chr> "William Sadler", "Diane Keaton", "Michael Caine", "Dian~
## $ No_of_Votes   <int> 2343110, 1620367, 2303232, 1129952, 689845, 1642758, 182~
## $ Gross         <chr> "28,341,469", "134,966,411", "534,858,444", "57,300,000"~
```

So we need to make some changes. First, we want to tidy the database by dropping NA's and irrelevant categories such as "Poster_Link" , "Certificate" and "Overview".

```
clean_data = subset(raw_data, select = -c(Poster_Link, Certificate, Overview))%>% na.omit(clean_
data$Gross,clean_data$No_of_Votes,clean_data$Meta_score,clean_data$IMDB_Rating,clean_data$Runtim
e)
view(clean_data)
```

Secondly, we want to make some more adjustments for us to be able to work properly: we need to remove the word "min" from the Runtime category, remove commas from the Gross and change the numeric information from character to integer. In addition, we want to convert the Meta Score number from dozens to units, in order to compare it with the IMDB rating later.

```
clean_data$Runtime <-gsub(" min","",as.character(clean_data$Runtime))

clean_data$Gross <-gsub(",","",as.character(clean_data$Gross))

clean_data <- transform(clean_data, Runtime = as.integer(Runtime),
                        Released_Year = as.integer(Released_Year), Gross = as.integer(Gross),No_
of_Votes = as.integer(No_of_Votes), IMDB_Rating=as.double(IMDB_Rating), Meta_score=as.double(Met
a_score))

clean_data <- mutate(clean_data,Meta_score = Meta_score/10,Gross_in_millions = Gross/1000000,vot
es_in_thousands= No_of_Votes/1000)
```

Now let's look at the clean data.

```
glimpse(clean_data)
```

```
## Rows: 843
## Columns: 15
## $ Series_Title       <chr> "The Shawshank Redemption", "The Godfather", "The D~
## $ Released_Year      <int> 1994, 1972, 2008, 1974, 1957, 2003, 1994, 1993, 201~
## $ Runtime            <int> 142, 175, 152, 202, 96, 201, 154, 195, 148, 139, 17~
## $ Genre              <chr> "Drama", "Crime, Drama", "Action, Crime, Drama", "C~
## $ IMDB_Rating        <dbl> 9.3, 9.2, 9.0, 9.0, 9.0, 8.9, 8.9, 8.9, 8.8, 8.8, 8~
## $ Meta_score         <dbl> 8.0, 10.0, 8.4, 9.0, 9.6, 9.4, 9.4, 9.4, 7.4, 6.6, ~
## $ Director           <chr> "Frank Darabont", "Francis Ford Coppola", "Christop~
## $ Star1              <chr> "Tim Robbins", "Marlon Brando", "Christian Bale", "~
## $ Star2              <chr> "Morgan Freeman", "Al Pacino", "Heath Ledger", "Rob~
## $ Star3              <chr> "Bob Gunton", "James Caan", "Aaron Eckhart", "Rober~
## $ Star4              <chr> "William Sadler", "Diane Keaton", "Michael Caine", ~
## $ No_of_Votes        <int> 2343110, 1620367, 2303232, 1129952, 689845, 1642758~
## $ Gross              <int> 28341469, 134966411, 534858444, 57300000, 4360000, ~
## $ Gross_in_millions  <dbl> 28.341469, 134.966411, 534.858444, 57.300000, 4.360~
## $ votes_in_thousands <dbl> 2343.110, 1620.367, 2303.232, 1129.952, 689.845, 16~
```

Much better.

We can see there are a total of 12 genres in the data. But some of the movies contain more than one genre:

```
clean_data %>%
  group_by(Genre) %>%
  summarise()
```

```
## # A tibble: 184 x 1
##    Genre
##    <chr>
##  1 Action, Adventure
##  2 Action, Adventure, Comedy
##  3 Action, Adventure, Drama
##  4 Action, Adventure, Family
##  5 Action, Adventure, Fantasy
##  6 Action, Adventure, History
##  7 Action, Adventure, Horror
##  8 Action, Adventure, Mystery
##  9 Action, Adventure, Romance
## 10 Action, Adventure, Sci-Fi
## # ... with 174 more rows
```

In order to get a better view of the recurrence, we will extract and organize each of the genres separately, present in a table, and conclude which are the most popular genres.

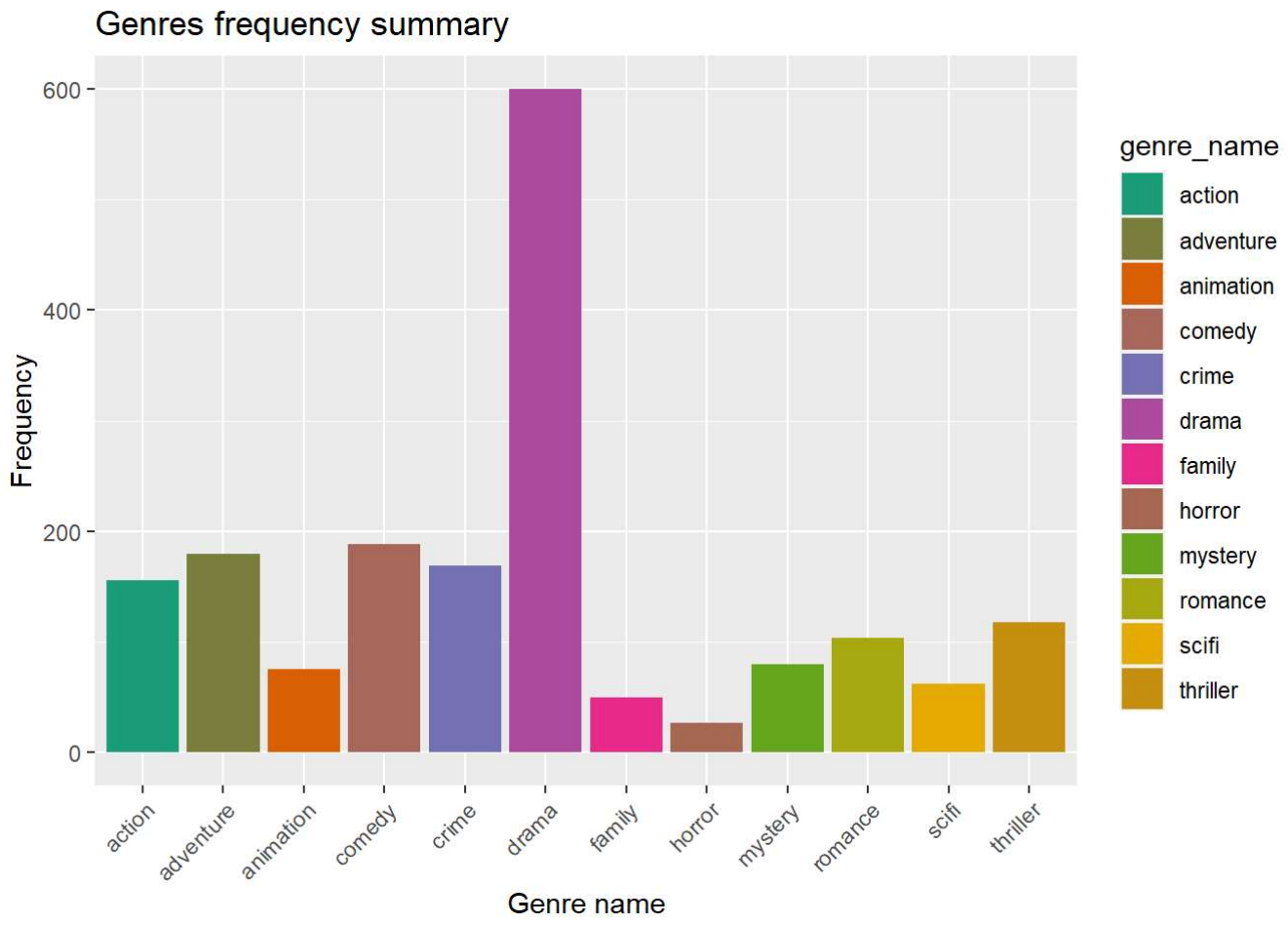
```
Table_Comedy<-grepl("Comedy",clean_data$Genre)
Num_Comedy<-sum(Table_Comedy)

Table_Adventure<-grepl("Adventure",clean_data$Genre)
Num_Adventure<-sum(Table_Adventure)

Table_Drama<-grepl("Drama",clean_data$Genre)
Num_Drama<-sum(Table_Drama)

Table_Romance<-grepl("Romance",clean_data$Genre)
Num_Romance<-sum(Table_Romance)

Table_Crime<-grepl("Crime",clean_data$Genre)
Num_Crime<-sum(Table_Crime)

Table_Thriller<-grepl("Thriller",clean_data$Genre)
Num_Thriller<-sum(Table_Thriller)

Table_Horror<-grepl("Horror",clean_data$Genre)
Num_Horror<-sum(Table_Horror)

Table_Action<-grepl("Action",clean_data$Genre)
Num_Action<-sum(Table_Action)

Table_Mystery<-grepl("Mystery",clean_data$Genre)
Num_Mystery<-sum(Table_Mystery)

Table_Scifi<-grepl("Sci-Fi",clean_data$Genre)
Num_Scifi<-sum(Table_Scifi)

Table_Animation<-grepl("Animation",clean_data$Genre)
Num_Animation<-sum(Table_Animation)

Table_Family<-grepl("Family",clean_data$Genre)
Num_Family<-sum(Table_Family)

tab <- matrix(c(Num_Action,Num_Adventure, Num_Animation, Num_Comedy, Num_Crime, Num_Drama, Num_F
amily, Num_Horror
                , Num_Mystery, Num_Romance, Num_Scifi, Num_Thriller), ncol=1, byrow=TRUE)
colnames(tab) <- c('Frequency')
rownames(tab) <- c('action','adventure','animation', 'comedy', 'crime', 'drama', 'family', 'horr
or', 'mystery',
                   'romance', 'scifi', 'thriller')
tab <- as.table(tab)
genres_table<-as.data.frame(tab)
genres_table = subset(genres_table, select = -c(Var2))
genres_table <- mutate(genres_table, genre_name=Var1)
genres_table = subset(genres_table, select = -c(Var1))
genres_table
```

```
##    Freq genre_name
## 1   156     action
## 2   180  adventure
## 3    75  animation
## 4   188     comedy
## 5   169      crime
## 6   600      drama
## 7    50     family
## 8    27     horror
## 9    80    mystery
## 10  104    romance
## 11   62      scifi
## 12  118   thriller
```

Visualizations

Now we can take a look at the popularity of the genres.

```
true_genres <- clean_data %>%
 mutate(is_drama = str_detect(clean_data$Genre, "Drama", negate = FALSE),
 is_action = str_detect(clean_data$Genre, "Action", negate = FALSE),
 is_comedy = str_detect(clean_data$Genre, "Comedy", negate = FALSE),
 is_adventure = str_detect(clean_data$Genre, "Adventure", negate = FALSE),
 is_crime = str_detect(clean_data$Genre, "Crime", negate = FALSE))
only_comedy <- true_genres%>%filter(is_comedy==TRUE)
only_drama <- true_genres%>%filter(is_drama==TRUE)
only_crime <- true_genres%>%filter(is_crime==TRUE)
only_adventure <- true_genres%>%filter(is_adventure==TRUE)
only_action <- true_genres%>%filter(is_action==TRUE)

palette_Dark2 <- colorRampPalette(brewer.pal(8, "Dark2"))
ggplot(data=genres_table, aes(x=genre_name, y=Freq, fill=genre_name)) +
  geom_bar(stat="identity") +
  theme(axis.text.x=element_text(angle=45, hjust=1))+
  labs(x="Genre name", y="Frequency",
       title="Genres frequency summary")+scale_fill_manual(values = palette_Dark2(15))
```



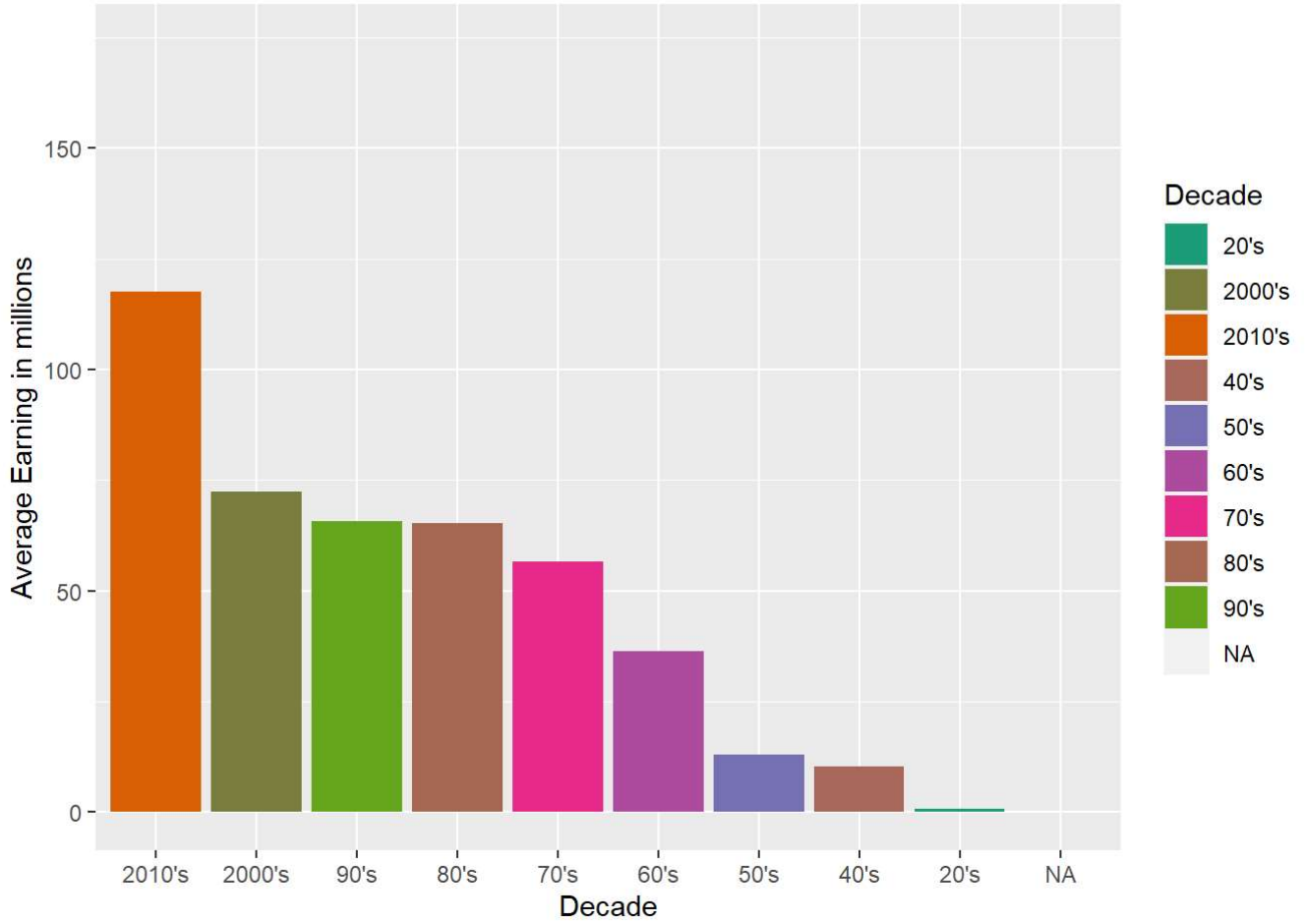
As we can see, Drama movies are much more popular than the other genres.

What about the highest-earning decade?

```
clean_data <- clean_data %>%
  mutate(Decade = if_else(Released_Year >= 2000,
                          paste0(Released_Year  %/% 10 * 10, "'s"),
                          paste0((Released_Year - 1900) %/% 10 * 10, "'s")))
decade_avg<- clean_data %>%
  group_by(Decade) %>%
  summarize(average_in_millions = mean(Gross_in_millions, na.rm = TRUE))%>%
  filter(!is.na(average_in_millions))

decade_avg
```

```
## # A tibble: 11 x 2
##    Decade average_in_millions
##    <chr>                <dbl>
##  1 20's                 0.644
##  2 2000's              72.3
##  3 2010's             118.
##  4 30's                23.7
##  5 40's                10.3
##  6 50's                13.0
##  7 60's                36.3
##  8 70's                56.6
##  9 80's                65.4
## 10 90's                65.8
## 11 <NA>               174.
```

```
ggplot(data=decade_avg[-4,],aes(x=reorder(Decade, -average_in_millions), y=average_in_millions,
 fill=Decade)) +
  geom_bar(stat = 'identity') +
  xlab('Decade')+
  ylab('Average Earning in millions')+
  scale_fill_manual(values = palette_Dark2(15))
```



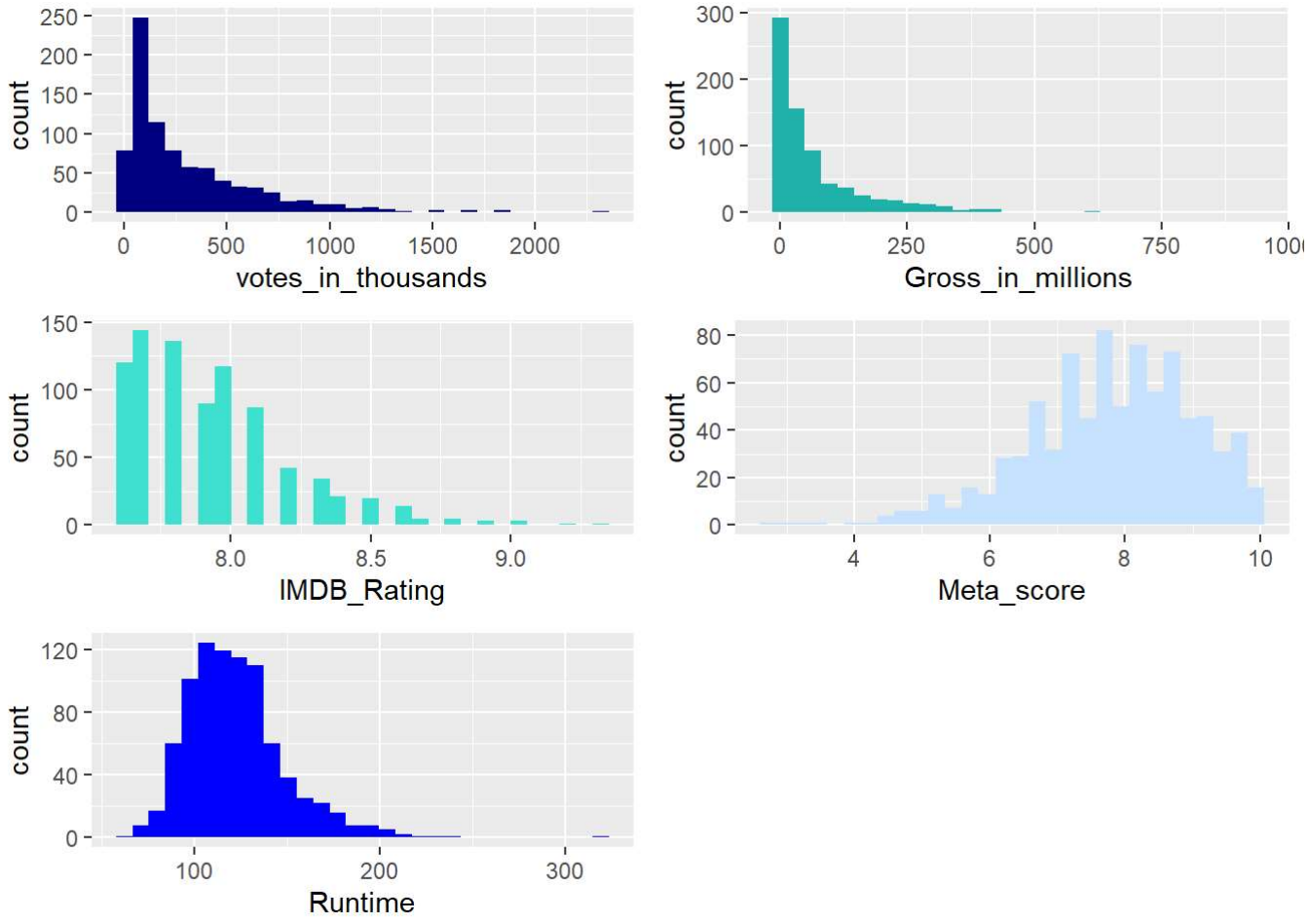
So we can see that the latest the movie the higher the revenue. It will be also interesting to check who are the directors with highest gross earned by their movie. We will narrow it to the top five directors:

```
DirectorByGross <- clean_data %>% group_by(Director) %>% mutate(is.na(Gross_in_millions) <- medi
an(Gross_in_millions)) %>%
                        summarise(totalGross = sum(Gross_in_millions), count= n()) %>%
                                arrange(desc(totalGross))  %>%  head(n = 5L)%>%
    ggplot(aes(x = reorder(Director, totalGross), y = totalGross, fill = Director)) +
    geom_bar(stat = "identity", show.legend = FALSE) +
    coord_flip() +
   geom_text(aes(label = paste0("$" ,round(totalGross)), hjust=1), size = 4) +
   labs(x = "Directors", y = "Gross amount earned by their movies in million dollars",
        title = "Top 5 directors by Gross of movies")+
    theme(plot.title = element_text(size=12, hjust = 1),axis.text.y = element_text(size=10),
         axis.ticks.x = element_blank(),axis.text.x = element_blank())

DirectorByGross
```

So it actually makes sense; most of the names rings a bell.

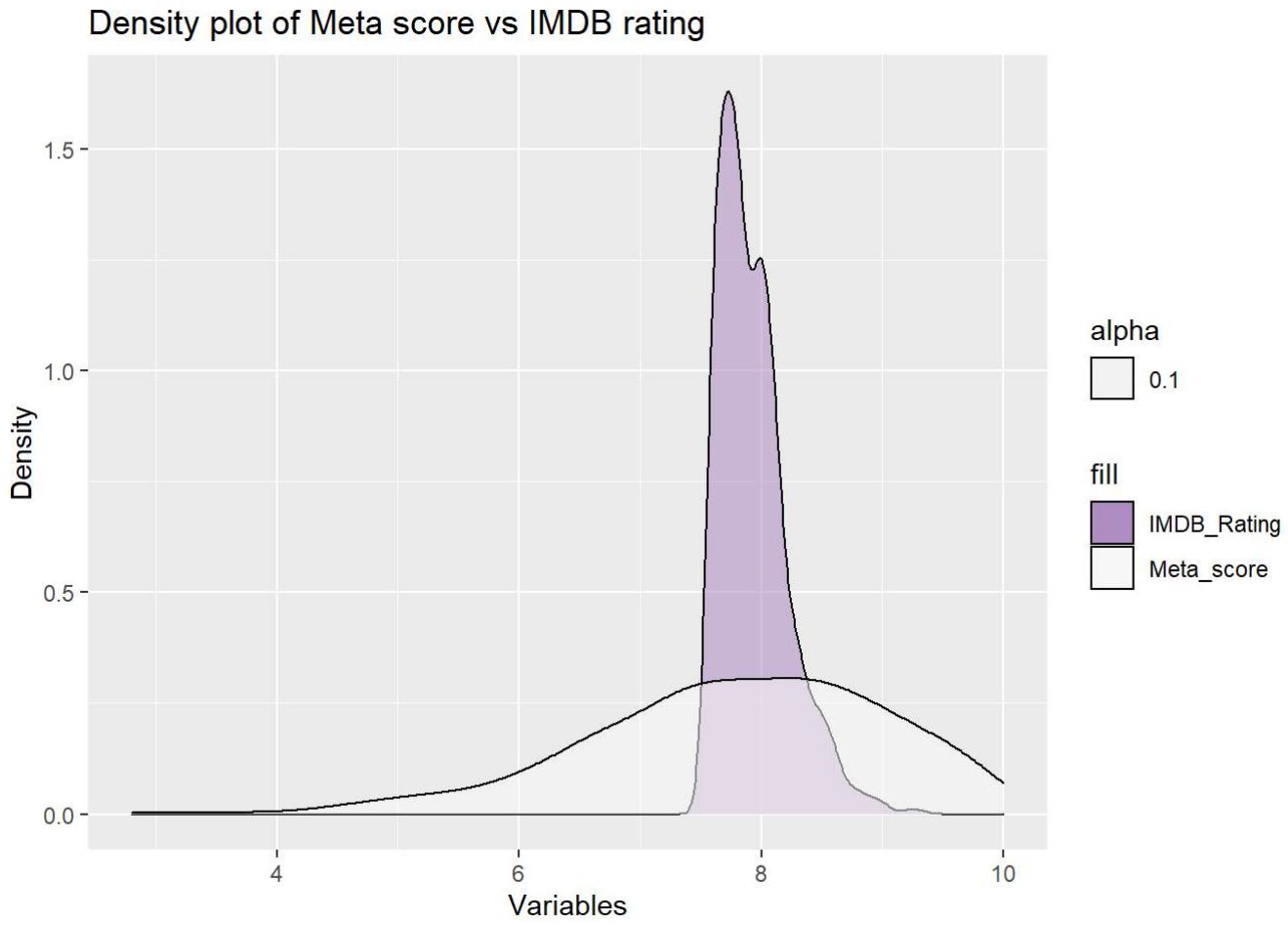
After we examined the different variables and their connection to the gross, it will be interesting to see the distribution of different variables compared to the gross:

```
p1 <-ggplot(clean_data, aes(x= votes_in_thousands)) +
  geom_histogram(bins=30, fill="Navy")
p2 <-ggplot(clean_data, aes(x= Gross_in_millions)) +
  geom_histogram(bins=30, fill="lightseagreen")
p3 <-ggplot(clean_data, aes(x=IMDB_Rating )) +
  geom_histogram(bins=30, fill="turquoise")
p4 <-ggplot(clean_data, aes(x=Meta_score )) +
  geom_histogram(bins=30, fill="slategray1")
p5 <-ggplot(clean_data, aes(x=Runtime )) +
  geom_histogram(bins=30, fill="blue")

grid.arrange(grobs= list(p1,p2,p3,p4,p5),
    fontface = 3, fontsize = 3,hjust = 1,x = 1)
```



We know that IMDB and MetaScore are both different types of ratings. Let's see their distributions, compared to one another:

```
Meta_IMDB_density <- ggplot(clean_data) +
  geom_density(aes(IMDB_Rating, fill ="IMDB_Rating", alpha = 0.1)) +
  geom_density(aes(Meta_score, fill ="Meta_score", alpha = 0.1)) +
  scale_x_continuous(name = "Variables") +
  scale_y_continuous(name = "Density") +
  ggtitle("Density plot of Meta score vs IMDB rating") +
  scale_fill_brewer(palette="PRGn")

Meta_IMDB_density
```



We can see that the mean of both ratings looks about the same (with the exception of a little “hump” in the IMDB rating graph).

So by looking at the visualizations, it seems like we can explore further three main graphical assumptions:

1. the mean of the MetaScore and the IMDB score look similar.

2. Drama is the most popular genre, suggesting it’s also the most profitable one.

3. The distributions of the Number of Votes, Gross, and IMDB rating look similar.

Now it’s time to test our assumptions.

Modeling

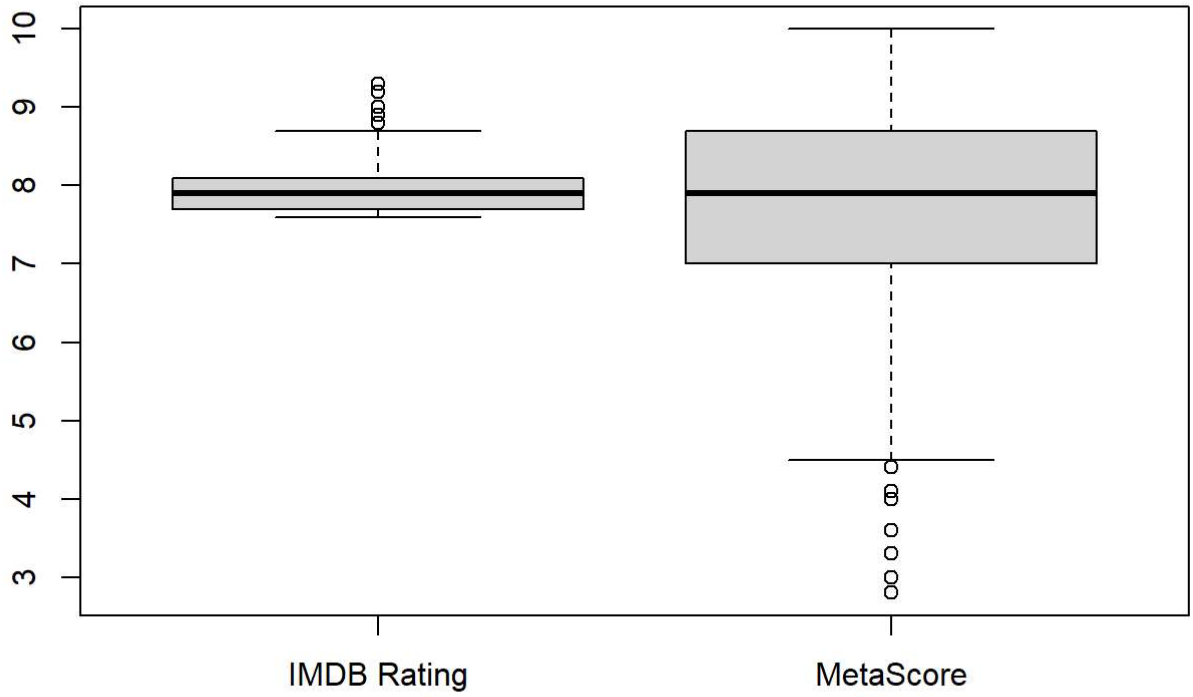
IMDB rating is the score which is given to a movie by visitors in the IMDB website. Metascore is the rating a movie received from expert film critics. We would like to examine if they align. Every IMDB rating and Metascore is compared to the same movie, therefore we will use paired t.test for the comparison.

Paired t-test for IMDB & Meta Score

H0: The average IMDB rating and the average MetaScore are equal.

H1: The average IMDB rating and the average MetaScore are not equal.

```
boxplot(clean_data$IMDB_Rating, clean_data$Meta_score,
        names = c("IMDB Rating", "MetaScore"))
```




```
t.test(x = clean_data$IMDB_Rating,
       y = clean_data$Meta_score,
       paired = TRUE, alternative = "two.sided")
```

```
##
##  Paired t-test
##
## data:  clean_data$IMDB_Rating and clean_data$Meta_score
## t = 3.2735, df = 842, p-value = 0.001106
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.05386073 0.21517842
## sample estimates:
## mean of the differences
##               0.1345196
```

Conclusion

As we can see from the results, although it was possible to believe that the wisdom of the crowds would align with the opinion of the experts, from the results we found that in a confidence level of 95%, we reject the null hypothesis. IMDB rating and MetaScore means are not equal.

Paired t-test for Gross of movies

As we saw earlier in the genre's graph bar, Drama is the most common genre by a significant margin. We know the film industry is a business that aims to maximize profit, so it makes sense to assume that the most common genre will be the most profitable. Having said that, we expect the top 100 drama movies gross mean to be higher than the top 100 not drama movies gross mean.

Let's have a quick look:

```
drama_v <- true_genres%>%filter(is_drama=="TRUE")%>%na.omit()%>%arrange(desc(Gross_in_millions))
%>% slice(1:100)
top_100_v <- true_genres%>%filter(is_drama=="FALSE")%>%na.omit()%>%arrange(desc(Gross_in_million
s)) %>% slice(1:100)
drama_v_mu <- mean(drama_v$Gross_in_millions)
top_100_v_mu <- mean(top_100_v$Gross_in_millions)
drama_v_mu
```

```
## [1] 175.8771
```

```
top_100_v_mu
```

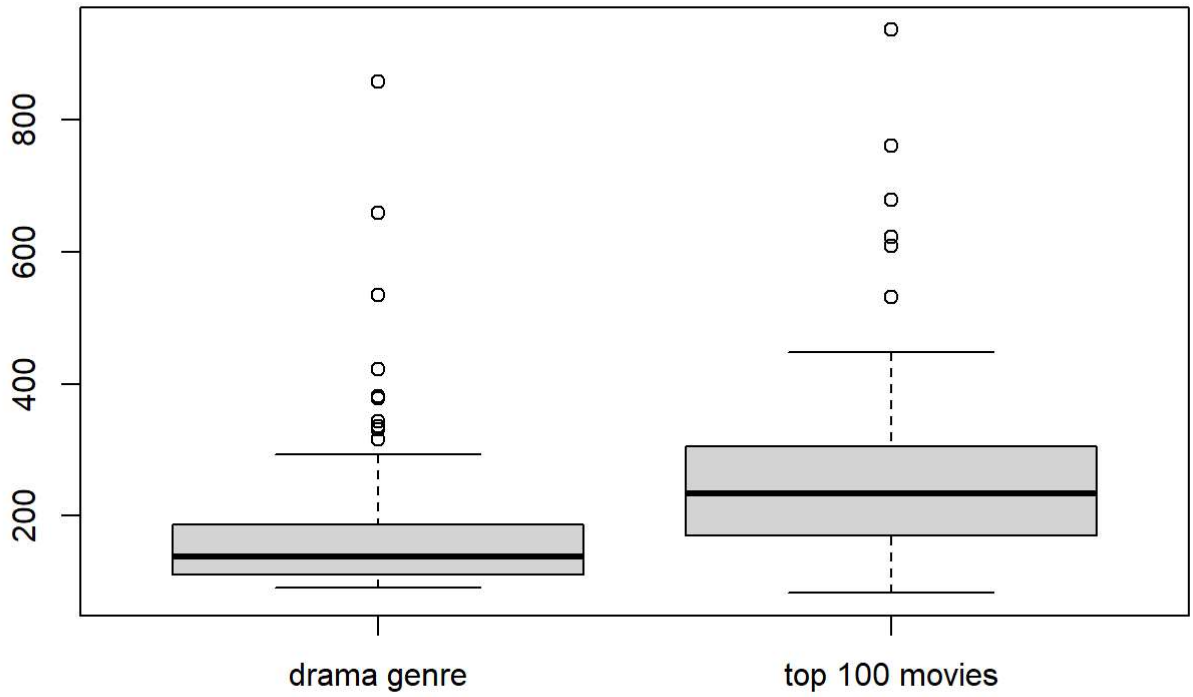
```
## [1] 259.1349
```

Surprisingly, we see that the average gross income of drama movies is much lower than the average of rest of the movies. This leads us to believe that the true gross income for drama movies is lower than rest of the movies average gross income. We will take the 100 top-earning drama movies and the 100 top-earning movies in general ,and test our new assumption. The 100 most earning movies are excluding drama genre movies, so we will use an unpaired t.test.

H0: The average gross income of drama movies is equal to the average income of the rest of movies.

H1: Drama is less profitable genre than the average the rest of the movies.

```
boxplot(drama_v$Gross_in_millions, top_100_v$Gross_in_millions,
        names = c("drama genre", "top 100 movies"))
```




```
t.test(x = drama_v$Gross_in_millions,
       y = top_100_v$Gross_in_millions,
       paired = FALSE, alternative = "less")
```

```
##
##  Welch Two Sample t-test
##
## data:  drama_v$Gross_in_millions and top_100_v$Gross_in_millions
## t = -4.4917, df = 189.12, p-value = 6.137e-06
## alternative hypothesis: true difference in means is less than 0
## 95 percent confidence interval:
##       -Inf -52.61915
## sample estimates:
## mean of x mean of y
##  175.8771  259.1349
```

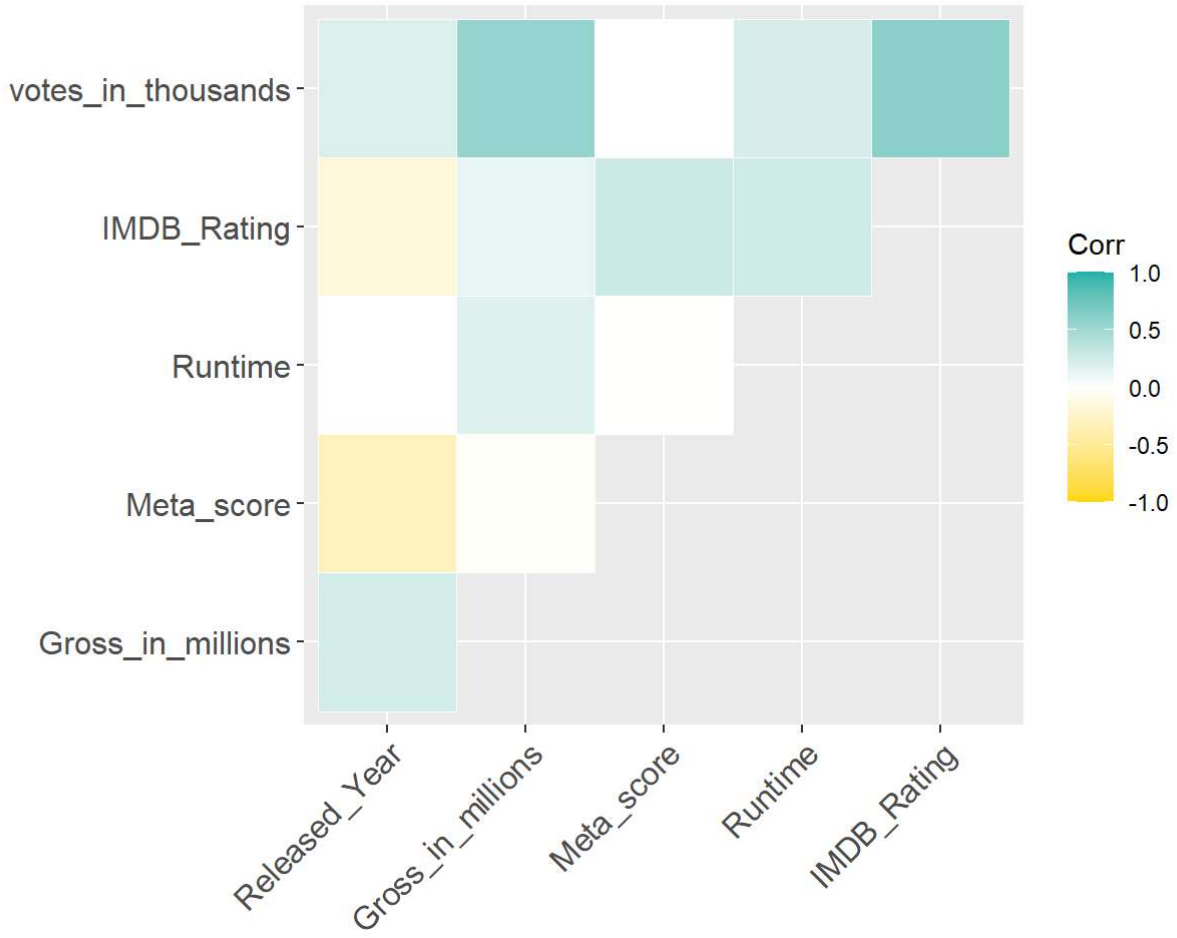
Conclusion

As we can see from the results, our assumption was confirmed by the paired t.test that we have conducted, and in a confidence level of 95% we reject the null hypothesis- the average gross income for drama genre movies is in fact lower than the average gross income of the rest of the genres.

Linear regression model

We will start with correlation check in order to see which variables are associated to each other.

```
num_data = subset(clean_data, select = -c(Series_Title, Star1, Star2, Star3, Star4, Director, Gr
oss, Genre, No_of_Votes,Decade ))%>%na.omit()
cor_graph =  cor(num_data)
ggcorrplot(cor_graph, type = "upper", hc.order = TRUE, outline.col = "white",ggtheme = ggplot2::
theme_gray,
           colors = c("gold","white","lightseagreen"))
```



We can see a strong correlation between the Number of Votes and Gross (which makes sense because the more people watched the movie, the more people ranked it). We have a strong correlation between IMDB rating and Number of Votes as well. In addition as we saw in the visualizations, the distribution of those variables reminds each other. We will examine those connections separately in a linear regression models:

Number of votes & Gross

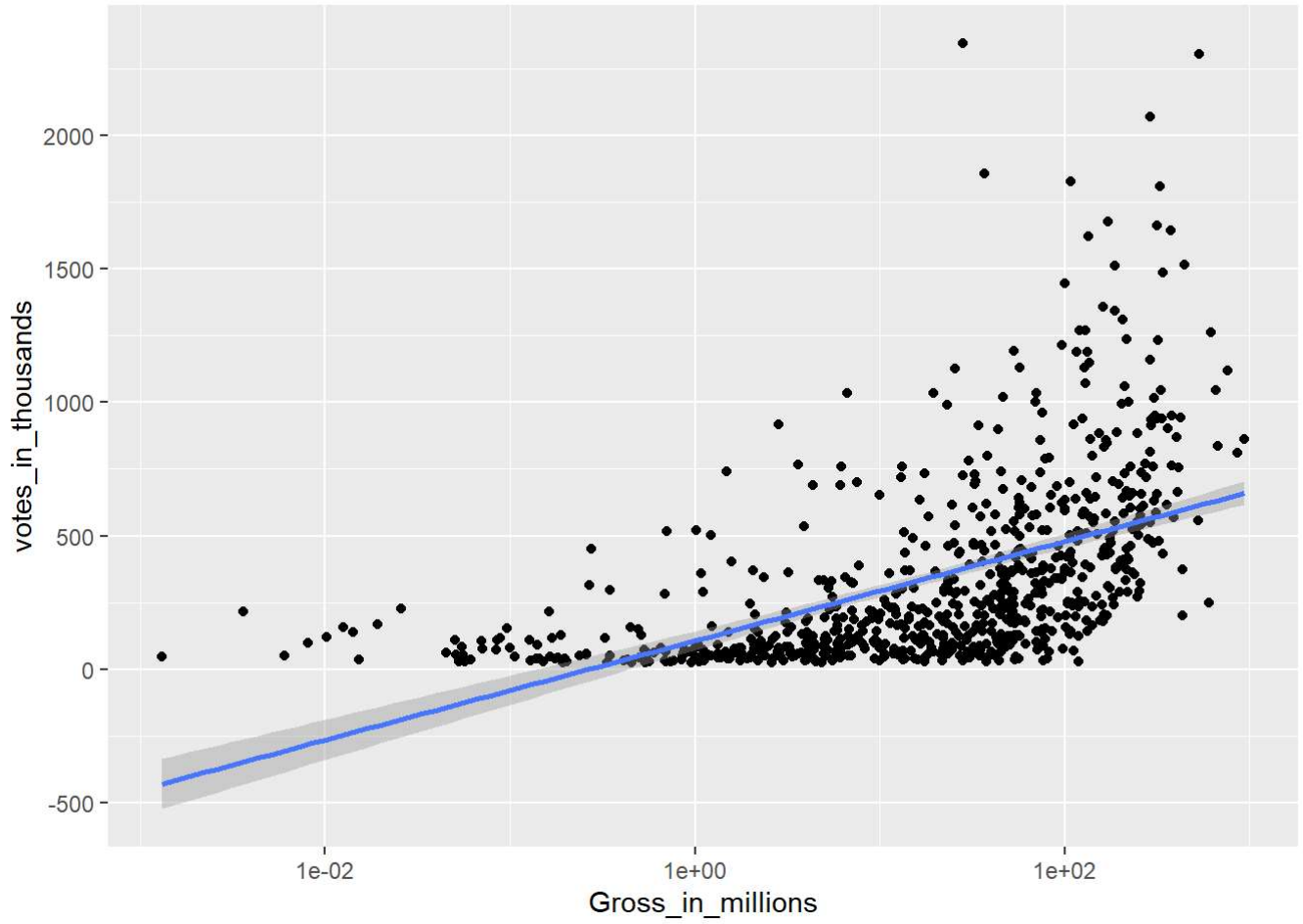
```
linear_model <- lm((votes_in_thousands) ~ log10(Gross_in_millions), data=num_data)
summary(linear_model)
```

```
##
## Call:
## lm(formula = (votes_in_thousands) ~ log10(Gross_in_millions),
##     data = num_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -465.89 -187.64  -74.12  111.89 1965.54
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 107.71      17.98   5.992 3.22e-09 ***
## log10(Gross_in_millions)    185.80      11.27  16.492  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 300.9 on 747 degrees of freedom
## Multiple R-squared:  0.2669, Adjusted R-squared:  0.2659
## F-statistic:   272 on 1 and 747 DF,  p-value: < 2.2e-16
```

We can see we have around 26% of Adjusted R-squared. The adjusted R-squared compares the explanatory power of regression models that contain different numbers of predictors. To make it simple, **26% R squared** indicates that the model does not imply on strong connection between the variables. We also know that a **p.value less than 0.05 is statistically significant.** It indicates strong evidence against the null hypothesis.

Now we will take a look at the visualization of the linear model between number of votes & Gross:

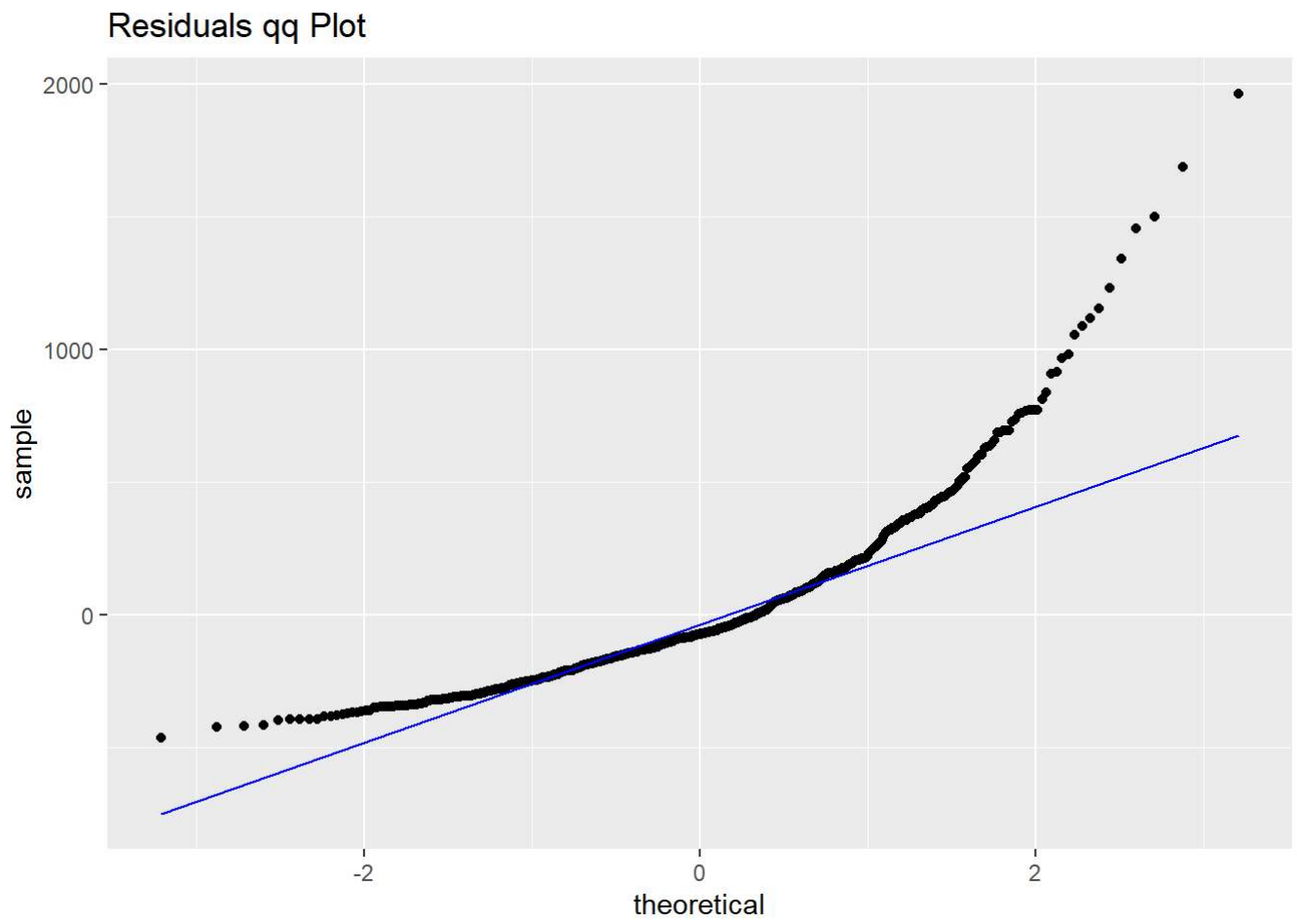
```
votes_gross <- ggplot(num_data, aes(x=Gross_in_millions, y=votes_in_thousands))+scale_x_log10()
+geom_point() +
  geom_smooth(col="royalblue1", method = "lm")

votes_gross
```



We can see that there is a pattern of more number of votes the higher the gross is, but it is not significant (the 26% R squared fits pretty well here). Moreover we see that we do not have homoscedacity here.

Now we will take a look at the residuals plot to examine whether they are normally distributed:

```
residuals_qq<- linear_model %>% ggplot(aes(sample=.resid)) +
  geom_qq() + geom_qq_line(col="blue") +
  labs(title="Residuals qq Plot")
residuals_qq
```



We can see from the qq plot that our residuals **do not distribute normally**.

Now let's make Breusch-Pagan Test for Homoscedasticity of the data:

```
bptest(linear_model)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  linear_model
## BP = 19.53, df = 1, p-value = 9.903e-06
```

Result of p.value < 0.05 suggesting that our data **is not homoscedastic**.

Conclusions

In order to use our model for predicting model, two of conditions must exist: **homoscedasticity** of the data and **normally distributed** residuals. Our data does not answer this conditions, so our model can not be used as a predicting model.

Number of votes & IMDB Rating

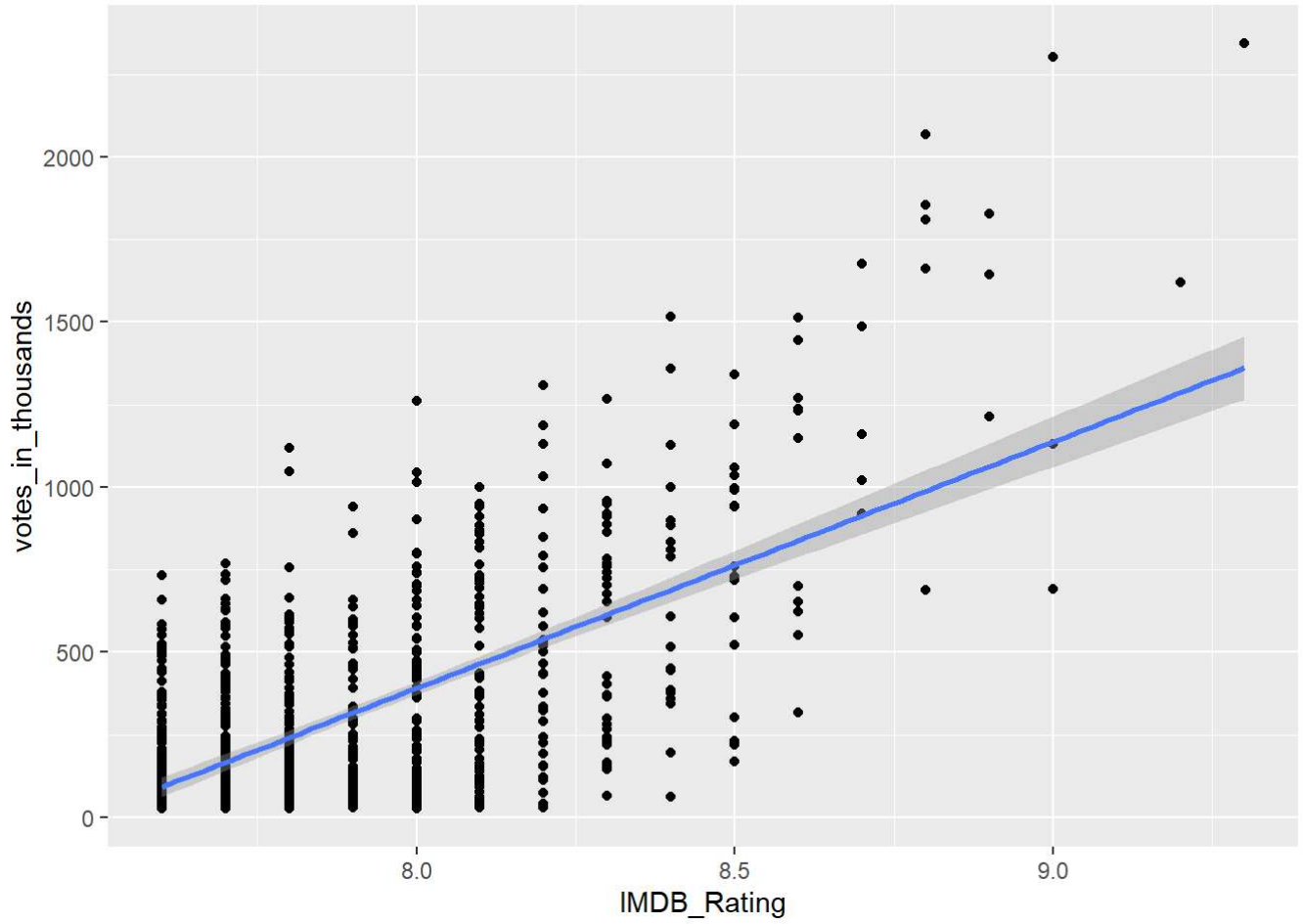
```
linear_model2 <- lm((votes_in_thousands) ~ (IMDB_Rating), data=num_data)
summary(linear_model2)
```

```
##
## Call:
## lm(formula = (votes_in_thousands) ~ (IMDB_Rating), data = num_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -626.53 -195.55  -38.99  169.71 1166.17
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -5581.41     278.54  -20.04   <2e-16 ***
## IMDB_Rating   746.50      35.08   21.28   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 277.3 on 747 degrees of freedom
## Multiple R-squared:  0.3774, Adjusted R-squared:  0.3766
## F-statistic: 452.9 on 1 and 747 DF,  p-value: < 2.2e-16
```

We can see we have around 37% of Adjusted R-squared. **37% R squared** indicates that the model does not imply on strong connection between the variables. We also know that a **p.value less than 0.05 is statistically significant.** It indicates strong evidence against the null hypothesis.

Now we will take a look at the visualization of the linear model between number of votes & IMDB Rating:

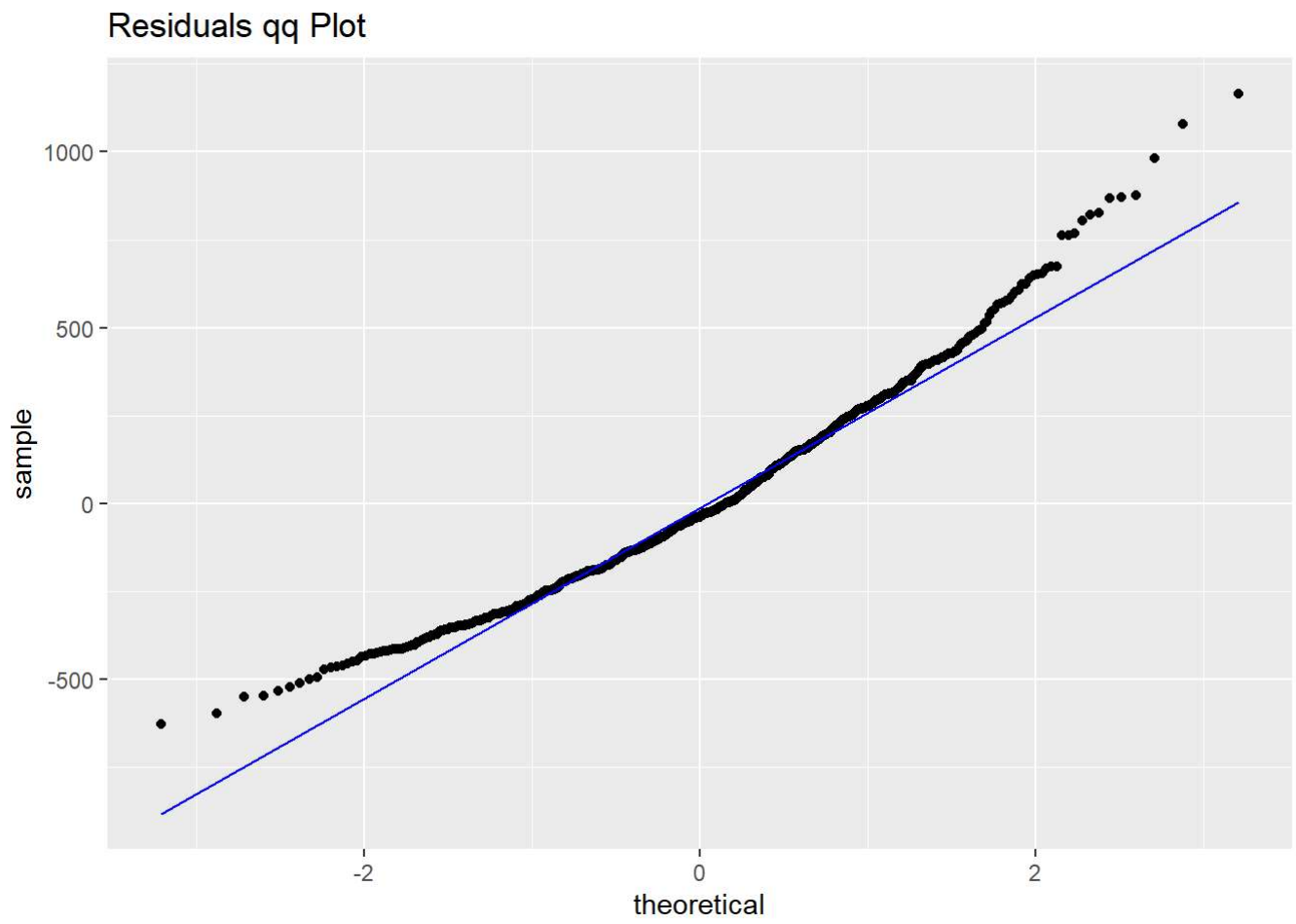
```
votes_imdb <- ggplot(num_data, aes(x=IMDB_Rating, y=votes_in_thousands))+geom_point() +
  geom_smooth(col="royalblue1", method = "lm")

votes_imdb
```



We can learn from the visualization that from the lowest IMDB ratings to around 8.1 we almost do not have a difference in the number of votes, but as we go higher, specially higher than around 8.6 most of the movies receive a significantly high number of votes. But in this model as well, we can see that we do not have homoscedacity.

Now we will take a look at the residuals plot to examine whether they are normally distributed:

```
residuals_qq<- linear_model2 %>% ggplot(aes(sample=.resid)) +
  geom_qq() + geom_qq_line(col="blue") +
  labs(title="Residuals qq Plot")
residuals_qq
```



We can see from the qq plot that our residuals **do not distribute normally**.

Now let's make Breusch-Pagan Test for Homoscedasticity of the data:

```
bptest(linear_model2)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  linear_model2
## BP = 148.08, df = 1, p-value < 2.2e-16
```

Result of p.value < 0.05 suggesting that our data **is not homoscedastic**.

Conclusions

In this model as well, we could not meet the requirements of **homoscedasticity** and **normally distributed residuals**, so we can not except the model, but we have got some more insights about the data.

Discussion and Summary

In our report, we explored the many ways the gross is affected.

We learned who directs the highest-earning movies and also what are the top-earning decades.

We found that the average rating of the MetaScore and IMDB is not the same, and also learned that the profits of Drama films are quite low compared to other films, even though it is the most popular genre. Finally, in the Multiple Linear Regression Test we received a low p.value, which is statistically significant, indicating strong evidence against the null hypothesis. The model assumptions aren't met, so we can confirm that the model's outcome cannot be fully trusted.

Thank you for reading!