Modeling and prediction for movies

Setup

Load packages

```
library(ggplot2)
library(dplyr)
library(statsr)
library(GGally)
library(knitr)
```

Load data

Make sure your data and R Markdown files are in the same directory. When loaded your data file will be called `movies`. Delete this note when before you submit your work.

```
load("movies.Rdata")
head(movies)
```

```
## # A tibble: 6 x 32
##   title title_type genre runtime mpaa_rating studio thtr_rel_year thtr_rel_month
##   <chr> <fct>      <fct>   <dbl> <fct>       <fct>          <dbl>          <dbl>
## 1 Fill~ Feature F~ Drama      80 R           Indom~          2013              4
## 2 The ~ Feature F~ Drama     101 PG-13       Warne~          2001              3
## 3 Wait~ Feature F~ Come~      84 R           Sony ~          1996              8
## 4 The ~ Feature F~ Drama     139 PG          Colum~          1993             10
## 5 Male~ Feature F~ Horr~      90 R           Ancho~          2004              9
## 6 Old ~ Documenta~ Docu~      78 Unrated     Shcal~          2009              1
## # ... with 24 more variables: thtr_rel_day <dbl>, dvd_rel_year <dbl>,
## #   dvd_rel_month <dbl>, dvd_rel_day <dbl>, imdb_rating <dbl>,
## #   imdb_num_votes <int>, critics_rating <fct>, critics_score <dbl>,
## #   audience_rating <fct>, audience_score <dbl>, best_pic_nom <fct>,
## #   best_pic_win <fct>, best_actor_win <fct>, best_actress_win <fct>,
## #   best_dir_win <fct>, top200_box <fct>, director <chr>, actor1 <chr>,
## #   actor2 <chr>, actor3 <chr>, actor4 <chr>, actor5 <chr>, imdb_url <chr>,
## #   rt_url <chr>
```

Part 1: Data

Context of the study

This project is based on a fictitious scenario - one where I've been hired as a data scientist at Paramount pictures. The data, sourced by Paramount, presents numerous variables on movies such as audience and critic ratings. Paramount endeavors to gather insights into determining the acclaim of a film and other novel patterns or ideas. The data set is comprised of 651 randomly sampled movies produced and released before 2016.

Sampling Design

```
movies$studio %>%
  unique() %>%
  head(10)
```

```
##  [1] Indomina Media Inc.      Warner Bros. Pictures    Sony Pictures Classics
##  [4] Columbia Pictures        Anchor Bay Entertainment Shcalo Media Group
##  [7] Paramount Home Video     MGM/United Artists       Independent Pictures
## [10] IFC Films
## 211 Levels: 20th Century Fox ... Zeitgeist Films
```

```
movies %>%
  summarise(max_year = max(thtr_rel_year), min_year = min(thtr_rel_year))
```

```
## # A tibble: 1 x 2
##   max_year min_year
##      <dbl>    <dbl>
## 1     2014     1970
```

Some of the film studios in the dataset are Indomina Media Inc., Warner Bros. Pictures, and Columbia Pictures. The majority of these studios are based in the US, and since the date ranges between the years 2014 and 1970. Most of these film studios, including ones by Sony Corporation, headquartered in Tokyo, are shot in California [1]. Furthermore, US-based film studios were inexperienced in film productions abroad and often sold the rights to their films to foreign parties [2]. It is thereby safe to assume such studios targeted a US-based audience. However, it should be noted that ratings produced by rotten tomatoes and IMDb tend to be based on a global audience.

Scope of Inference

The data collected is observational and captures all the significant factors of a movie such as its genre, runtime, and whether or not the movie is in the Top 200 Box Office list on BoxOfficeMojo. Furthermore, The data is also a random sample of the movies. Since no strata, blocks, quotas, nor clusters are defined, a simple random sample can be assumed. Since this data does not result from an experimentation study, nor are there any variables that are blocked nor controlled, we cannot infer causation from the data.

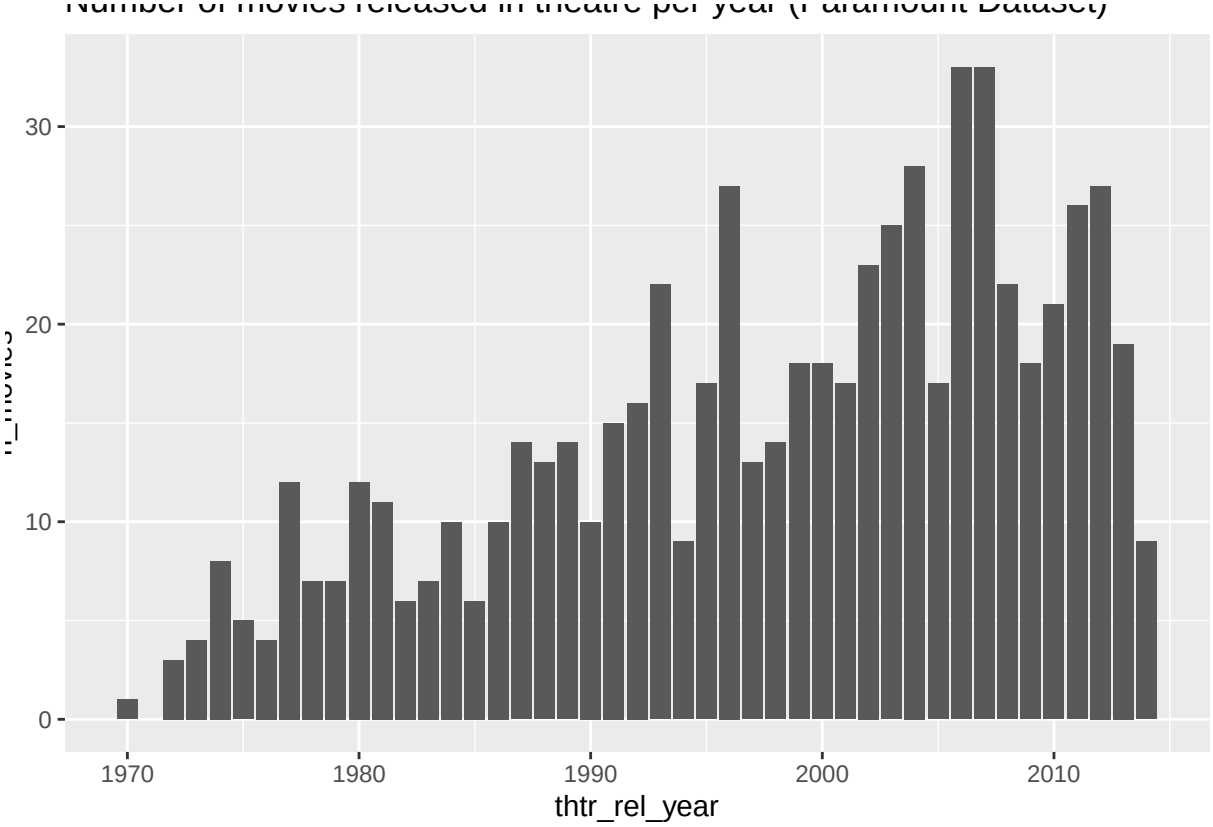
Generalizability

```
year_n_movies <- movies %>%
  group_by(thtr_rel_year) %>%
  summarize(n_movies = n())
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
ggplot(data = year_n_movies, aes(x = thtr_rel_year, y = n_movies)) +
  geom_bar(stat='identity') +
  ggtitle("Number of movies released in theatre per year (Paramount Dataset)")
```

[1] Wikipedia - Film industry
[2] Wikipedia - Cinema of the United States



As according to the bar graph above, over 50% of the years have below 20 movies. Therefore, we cannot generalize these results to all U.S. movies released in each year. However, we can generalize this to a random sample of all U.S. movies as the dataset consists of 651. This is well below 10% of all movies released in the U.S. and it is a random sample, so we can safely assume each movie is independent of one another.

Part 2: Research question

Since Paramount is likely to intend to capitalize on popular movies, it leads us to the following research question:

> What are the driving factors for determining the popularity of a movie?

A popular movie implies higher box office sales resulting in more revenue for the company. Paramount is likely to benefit in the form of ticket sales, DVD sales, and licensing the movie to third party vendors. Though it would be ideal to capture the popularity of a movie in terms of its financial gain, due to the nature of this dataset, we cannot do so.

Such a prediction would also allow Paramount to decide which movie ideas to invest in based on its `genre`, `runtime`, `mpaa_rating` and other such factors available prior to the release of a movie. Furthermore, this project will be using the `imdb_rating` as our response variable. The IMDb is a Amazon owned online movie database that provides information related to films and television programs. As of January 2020, 83 million registered users [3]. Since customers are likely to refer to IMDb ratings as a credible source for movie recommendation, a high rating is important.

The IMDb originally used the following formula to calculate their weighted rating [4]:

[3] Press Room - IMDb
[4] Wikipedia - IMDb

$$W = \frac{R \times v + C \times m}{v + m}$$

Where,

- W = weighted rating
- R = average for the movie as a number from 1 to 10 (mean) = (Rating)
- v = number of votes for the movie = (votes)
- m = minimum votes required to be listed in the Top 250 (currently 25,000)
- C = the mean vote across the whole report (currently 7.0)

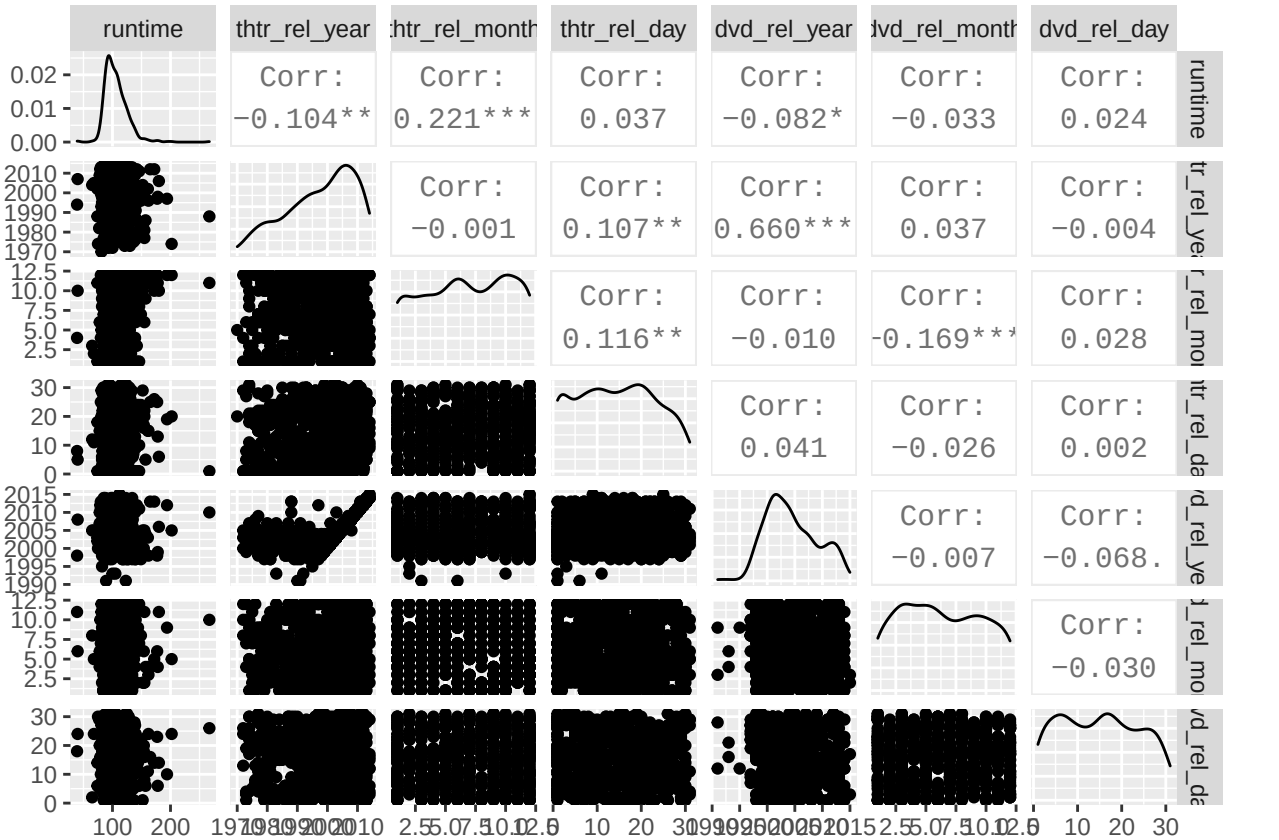
This means that an influential driver for higher ratings is a large number of top votes (preferably above 8). Furthermore, since this project is primarily concerned with significant variables, the p-value will be considered.

Part 3: Exploratory data analysis

Collinearity and Parsimony

Two predictor variables are said to be collinear when they are correlated with each other. In order to test for collinearity, the R^2 coefficient of correlation can be compared between any two numerical variables (or categorical, in the case of day, month or year) we select for this analysis. Inclusion of collinear predictors are likely to bias model estimators and complicate the model estimation process.

```
ggpairs(movies, columns = c('runtime', 'thtr_rel_year', 'thtr_rel_month',
                            'thtr_rel_day', 'dvd_rel_year', 'dvd_rel_month', 'dvd_rel_day'))
```

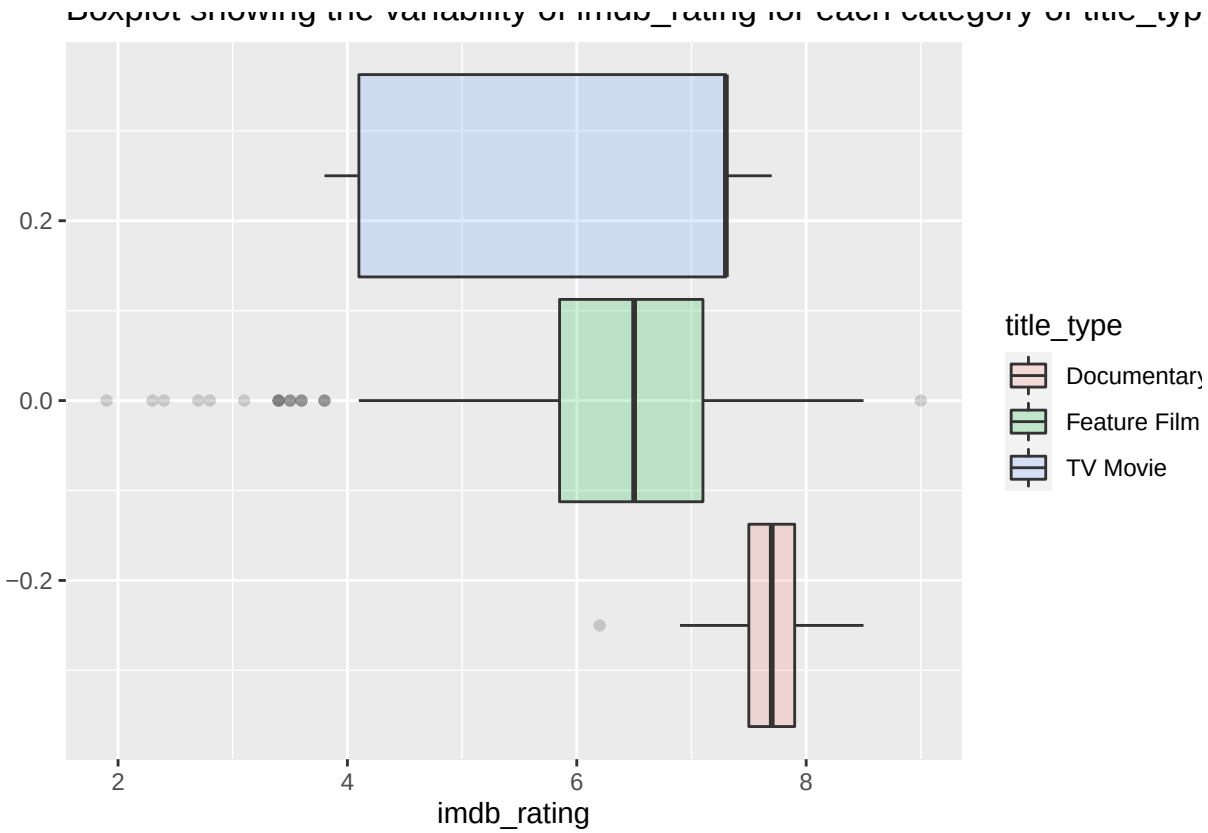


Since none of the correlation coefficients are high, a model comprising of the above numerical variables can be deemed as parsimonious, since it would contain the fewest assumptions.

Categorical Variables

We can draw a boxplot for each categorical variable, of interest, to capture the distribution of the with respect to the response variable, i.e., `imdb_rating`. Furthermore, we'll also use a barplot to show comparisons among the count of each category.

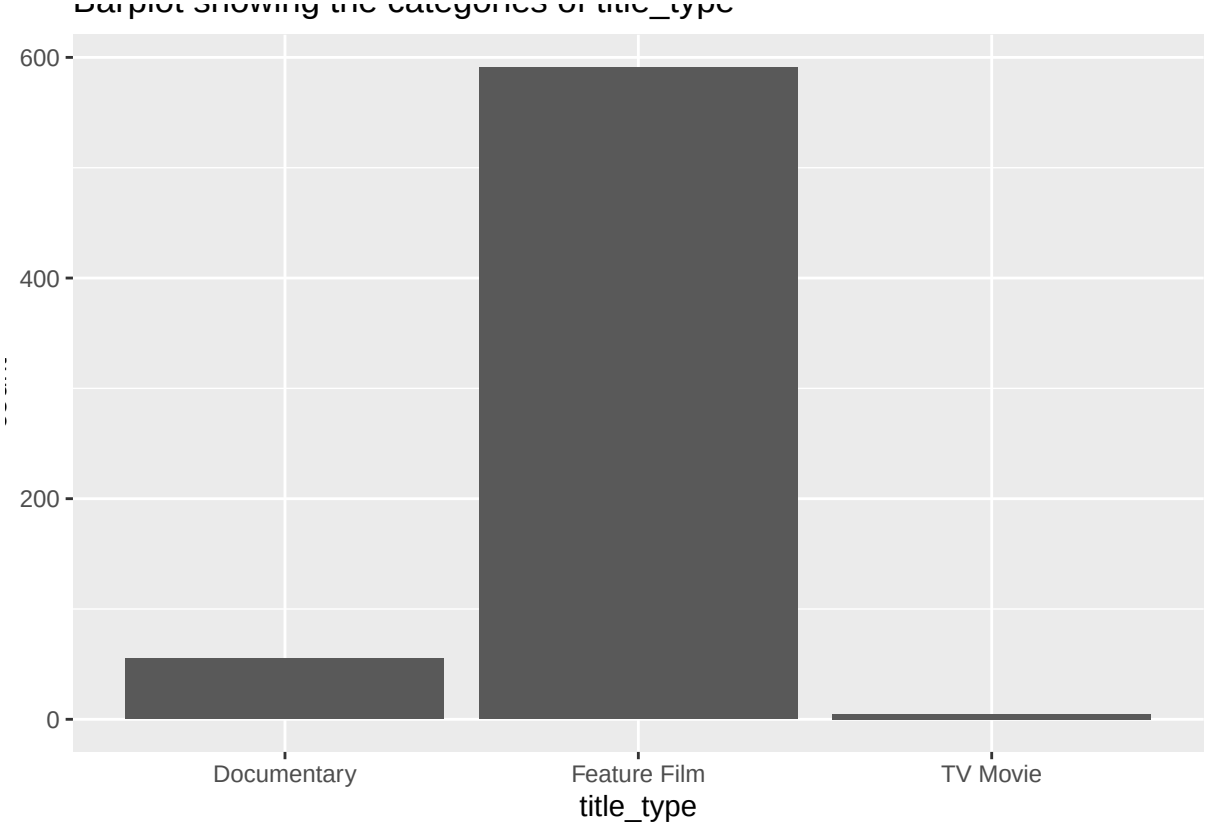
```
movies %>%
    group_by(title_type) %>%
    ggplot(aes(x = imdb_rating, fill = title_type)) +
    geom_boxplot(alpha=.2) +
    ggtitle("Boxplot showing the variability of imdb_rating for each category of title_type")
```



title_type

Based on the boxplot above, the ratings are least variable for the Documentary category and most variable for the TV Movie category.

```
ggplot(data = movies, aes(x = title_type)) +
  geom_bar() +
  ggtitle("Barplot showing the categories of title_type")
```




```
movies %>%
  group_by(title_type) %>%
  summarize(n())
```

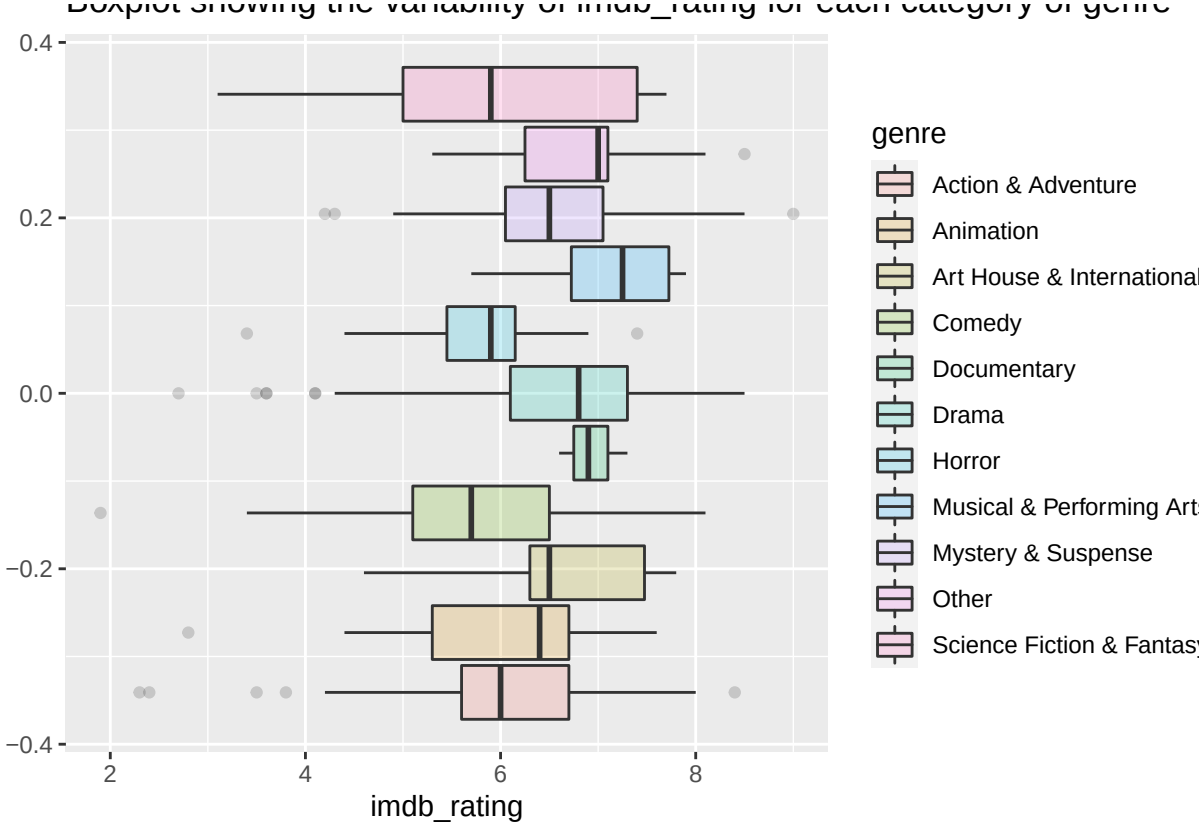
```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 3 x 2
##   title_type   `n()`
##   <fct>        <int>
## 1 Documentary     55
## 2 Feature Film   591
## 3 TV Movie         5
```

The majority of the films in the dataset are of the type "Feature Film." Furthermore, "TV Movie" is a tiny segment of the dataset comprising of only 5 movies, while there are only 55 Documentaries. Additionally, since this analysis is meant for Paramount pictures, a company that specializes in making movies, we'll only be using movies of the Feature Film type.

```
# subsetting to include only Feature Film
movies <- movies[movies$title_type == 'Feature Film', ]
```

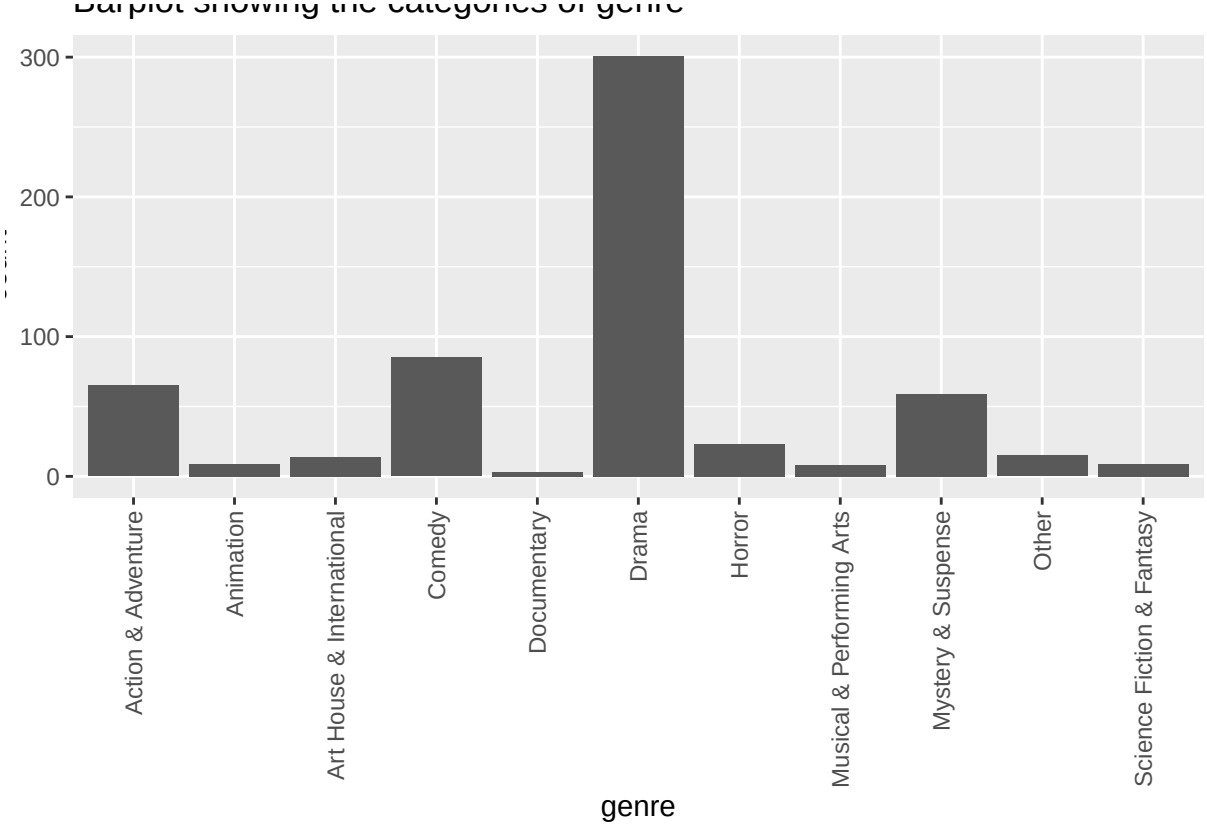
```
movies %>%
    group_by(genre) %>%
    ggplot(aes(x = imdb_rating, fill = genre)) +
    geom_boxplot(alpha=.2) +
    ggtitle("Boxplot showing the variability of imdb_rating for each category of genre")
```



genre

The variability isn't constant among the different genres as displayed in the boxplot above. The variability is the highest for the "Action & Adventure" genre and the lowest for the "Drama" genre.

```
ggplot(data = movies, aes(x = genre)) +
  geom_bar() +
  ggtitle("Barplot showing the categories of genre") +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1))
```




```
movies %>%
  group_by(genre) %>%
  summarize(n())
```

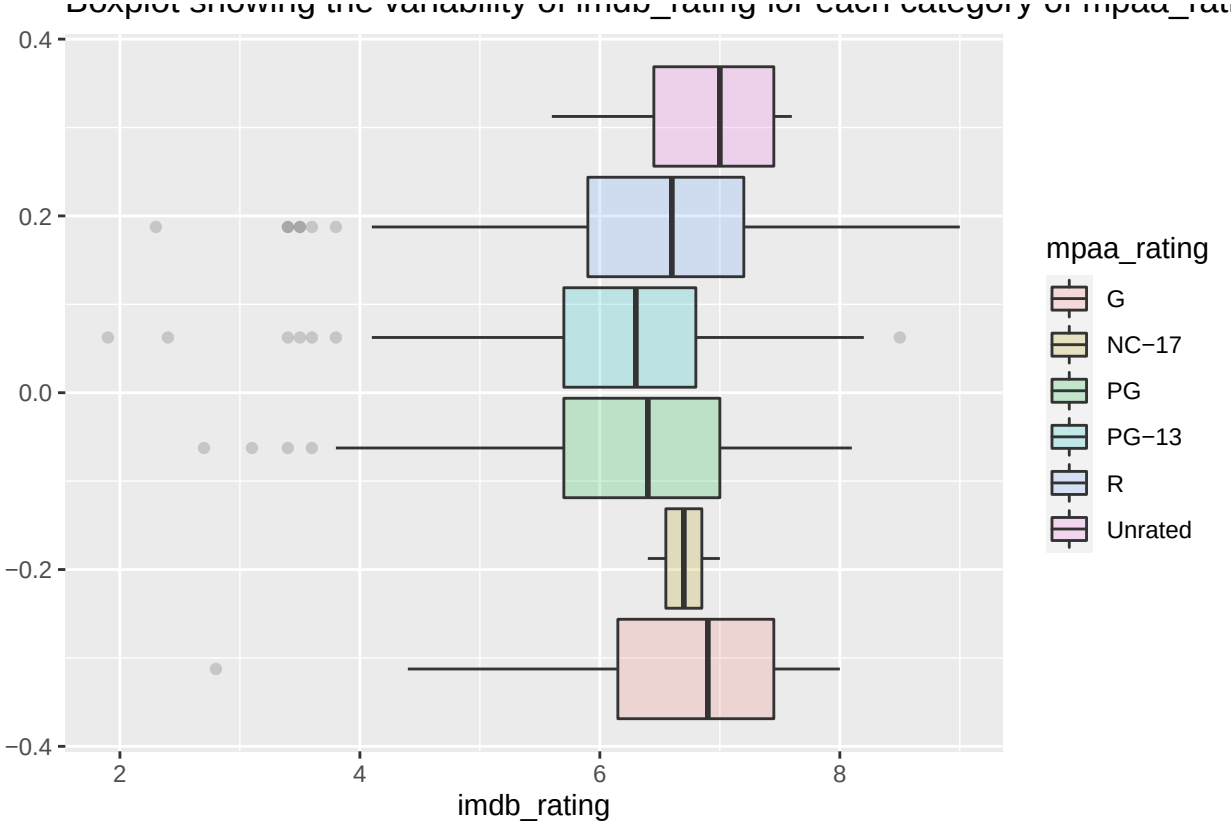
```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 11 x 2
##    genre                     `n()`
##    <fct>                     <int>
##  1 Action & Adventure           65
##  2 Animation                     9
##  3 Art House & International    14
##  4 Comedy                       85
##  5 Documentary                   3
##  6 Drama                       301
##  7 Horror                       23
##  8 Musical & Performing Arts     8
##  9 Mystery & Suspense           59
## 10 Other                        15
## 11 Science Fiction & Fantasy     9
```

Since we removed the "Documentary" title type in the previous section, we'll remove all the titles from the "Documentary" genre. It would be counter-intuitive to remove all Documentaries but still have a segment of films that are of the genre "Documentary". Furthermore, there are only 3 documentaries in this dataset which further supports our decision of removing this genre.

```
movies <- movies[movies$genre != "Documentary", ]
```

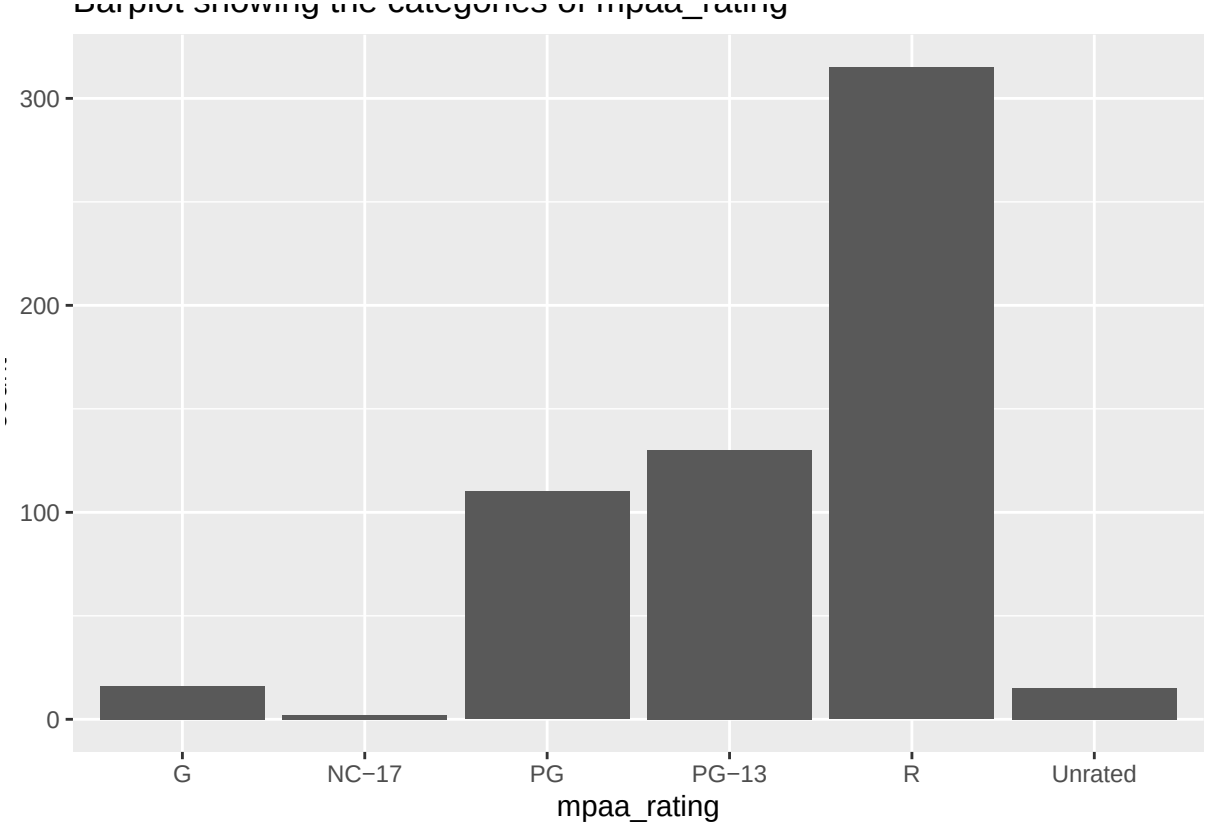
```
movies %>%
    group_by(mpaa_rating) %>%
    ggplot(aes(x = imdb_rating, fill = mpaa_rating)) +
    geom_boxplot(alpha=.2) +
    ggtitle("Boxplot showing the variability of imdb_rating for each category of mpaa_rating")
```



mpaa_rating

The variability of the imdb_rating of the different mpaa_rating categories seems almost consistent for R, Unrated, and PG. The lowest variability is in the NC-17 rating.

```
ggplot(data = movies, aes(x = mpaa_rating)) +
  geom_bar() +
  ggtitle("Barplot showing the categories of mpaa_rating")
```



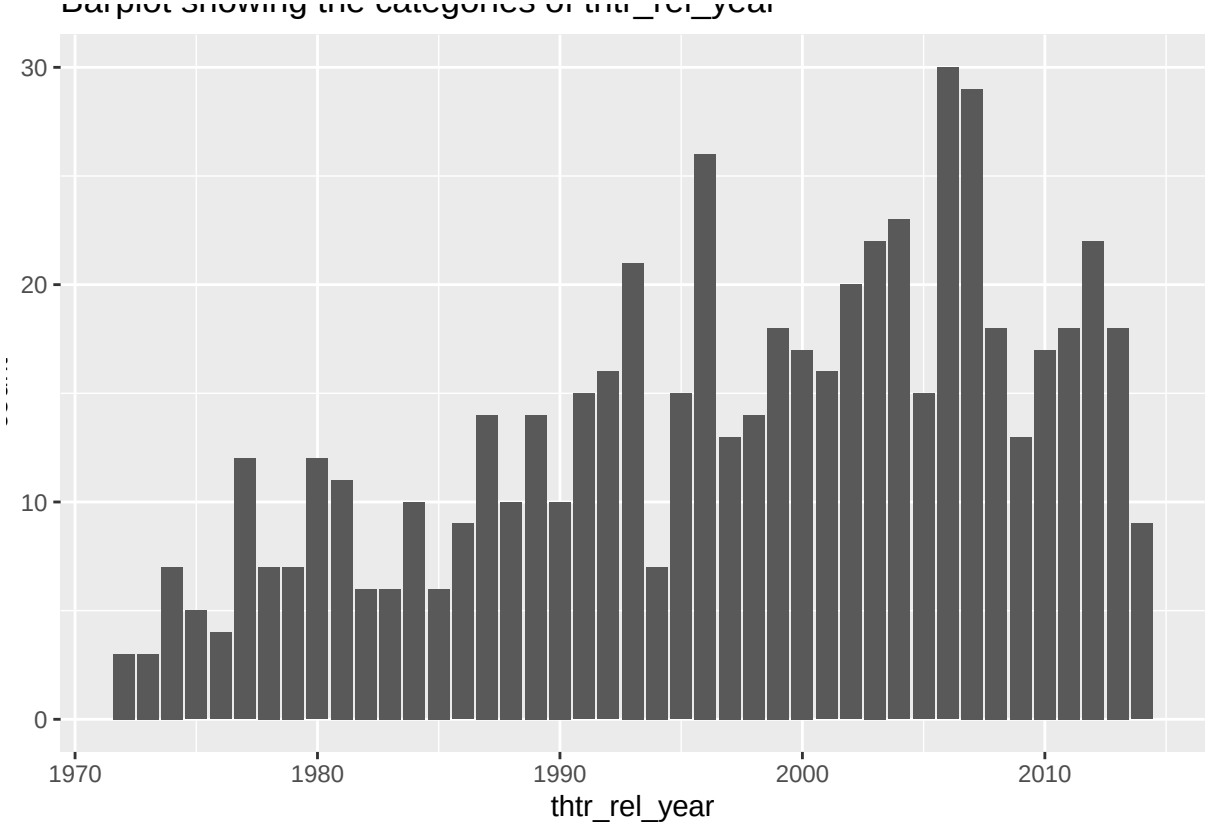

```
movies %>%
  group_by(mpaa_rating) %>%
  summarize(n())
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 6 x 2
##   mpaa_rating `n()`
##   <fct>       <int>
## 1 G              16
## 2 NC-17           2
## 3 PG            110
## 4 PG-13         130
## 5 R             315
## 6 Unrated        15
```

There are only 2 movies from the NC-17 rated category in this dataset. However, since we're interested in how movies of all ratings perform, we won't be removing movies from this category.

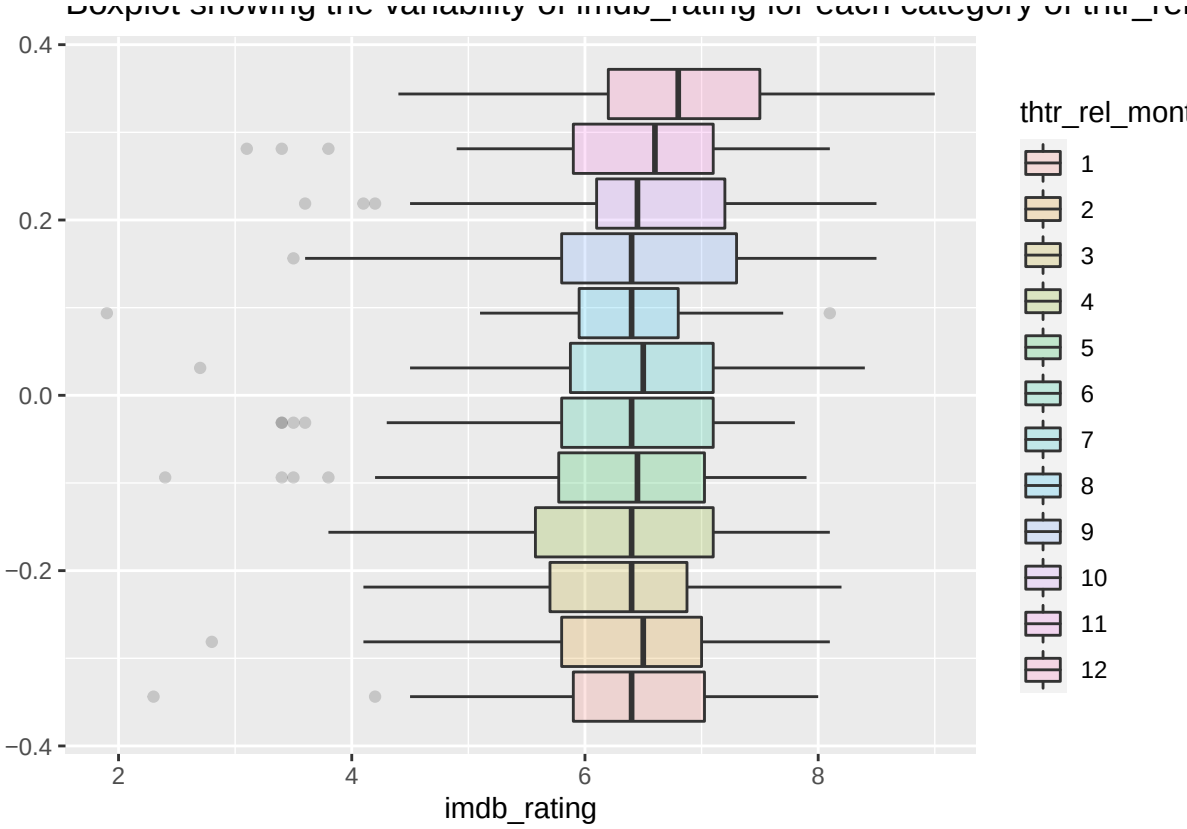
```
ggplot(data = movies, aes(x = thtr_rel_year)) +
  geom_bar() +
  ggtitle("Barplot showing the categories of thtr_rel_year")
```



thtr_rel_year

As mentioned earlier, the movies aren't not stratified by thtr_rel_year. Each thtr_rel_year contains a maximum of 30 movies and a minimum of 2.

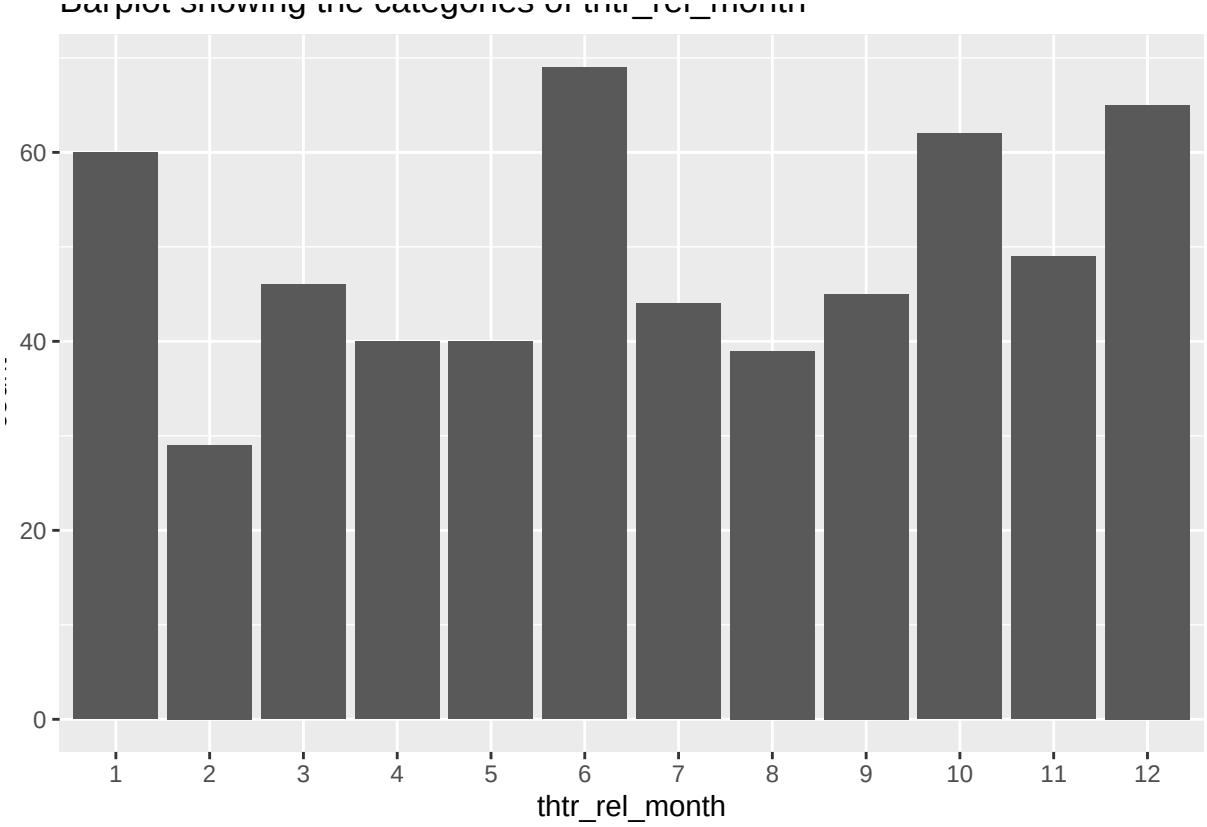
```
# converting thtr_rel_month to a factor
movies$thtr_rel_month <- as.factor(movies$thtr_rel_month)

movies %>%
    group_by(thtr_rel_month) %>%
    ggplot(aes(x = imdb_rating, fill = thtr_rel_month)) +
    geom_boxplot(alpha=.2) +
    ggtitle("Boxplot showing the variability of imdb_rating for each category of thtr_rel_month")
```



thtr_rel_month

The variability imdb_rating of movies among months looks consistent for most of the months as displayed in the boxplot above.

```
ggplot(data = movies, aes(x = thtr_rel_month)) +
  geom_bar() +
  ggtitle("Barplot showing the categories of thtr_rel_month")
```




```
movies %>%
  group_by(thtr_rel_month) %>%
  summarize(n())
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 12 x 2
##    thtr_rel_month `n()`
##    <fct>          <int>
##  1 1                 60
##  2 2                 29
##  3 3                 46
##  4 4                 40
##  5 5                 40
##  6 6                 69
##  7 7                 44
##  8 8                 39
##  9 9                 45
## 10 10                62
## 11 11                49
## 12 12                65
```

```
# converting thtr_rel_month to a numerical variable
movies$thtr_rel_month <- as.numeric(movies$thtr_rel_month)
```

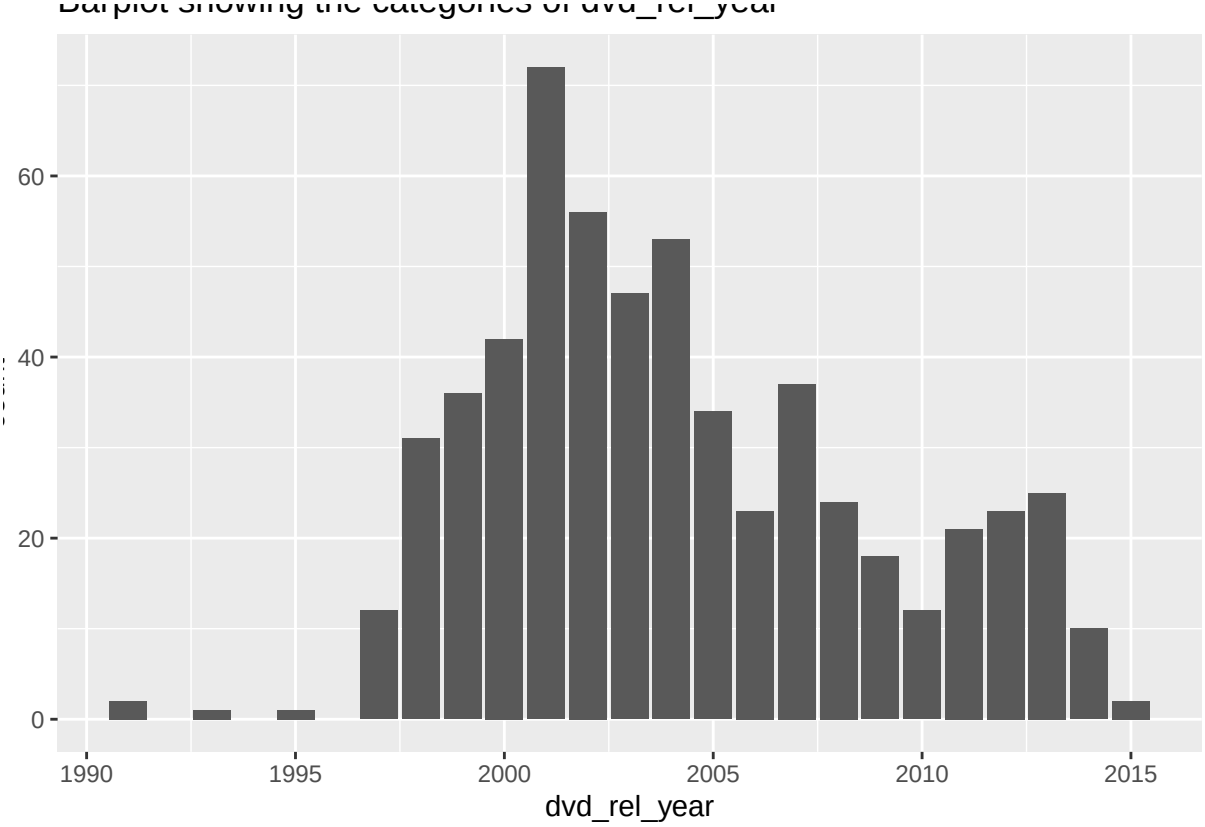
The are a maximum number of 69 movies in each month and a minimum of 29.

```
ggplot(data = movies, aes(x = dvd_rel_year)) +
```

```
  geom_bar() +
  ggtitle("Barplot showing the categories of dvd_rel_year")
```

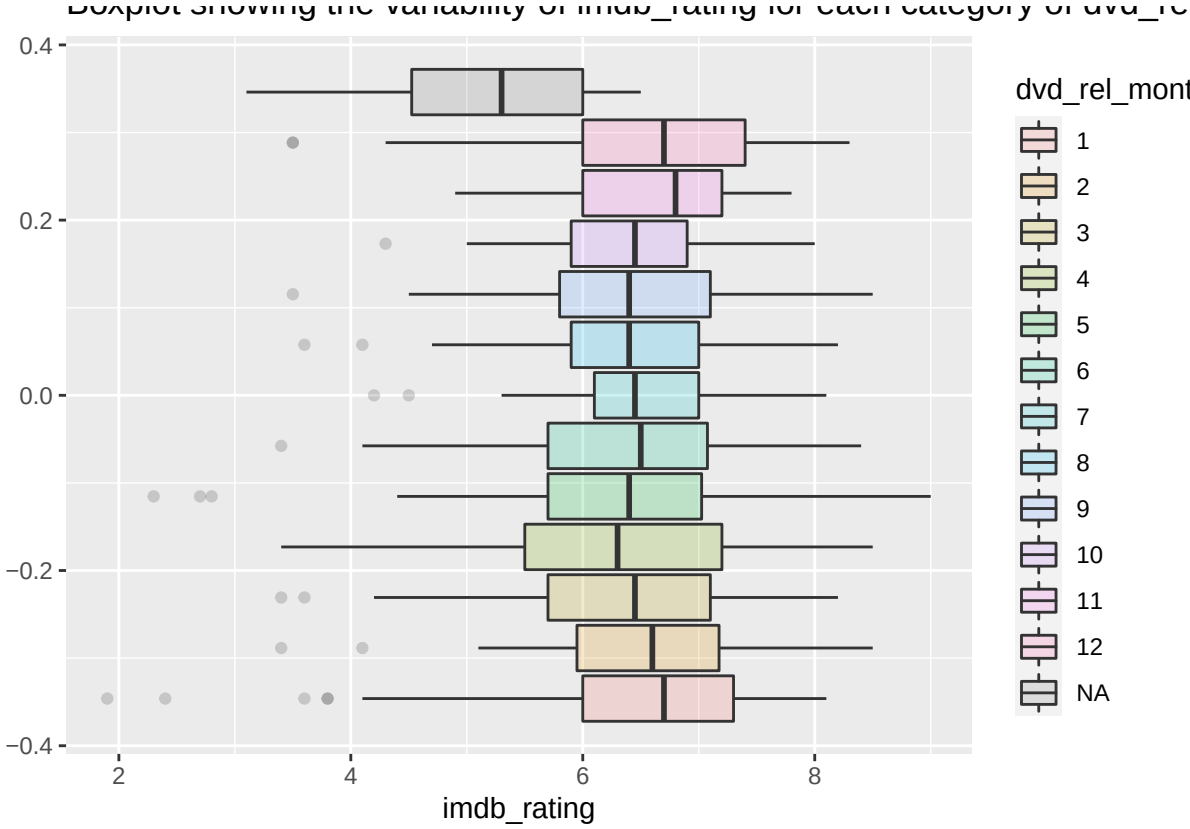
dvd_rel_year

```
## Warning: Removed 6 rows containing non-finite values (stat_count).
```



The dvd_rel_year has some years where no movies were realeased. This makes sense as DVDs were prevelant during the early and mid 2000s after which movies began moving to online mediums such as Netflix and Hulu.

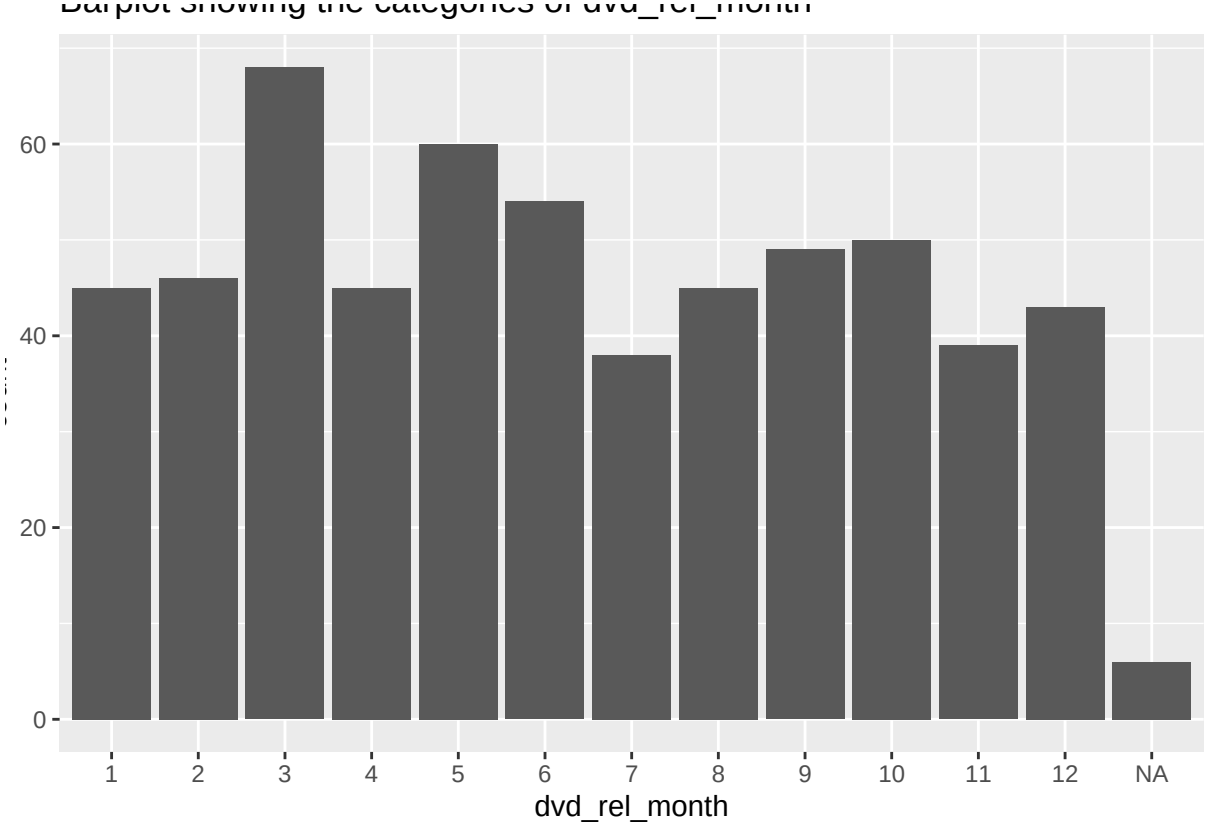
```
# converting dvd_rel_month to a factor
movies$dvd_rel_month <- as.factor(movies$dvd_rel_month)

movies %>%
    group_by(dvd_rel_month) %>%
    ggplot(aes(x = imdb_rating, fill = dvd_rel_month)) +
    geom_boxplot(alpha=.2) +
    ggtitle("Boxplot showing the variability of imdb_rating for each category of dvd_rel_month")
```



dvd_rel_month

The variability imdb_rating of movies among months looks consistent for most of the months as displayed in the boxplot above.

```
ggplot(data = movies, aes(x = dvd_rel_month)) +
  geom_bar() +
  ggtitle("Barplot showing the categories of dvd_rel_month")
```




```
movies %>%
  group_by(dvd_rel_month) %>%
  summarize(n())
```

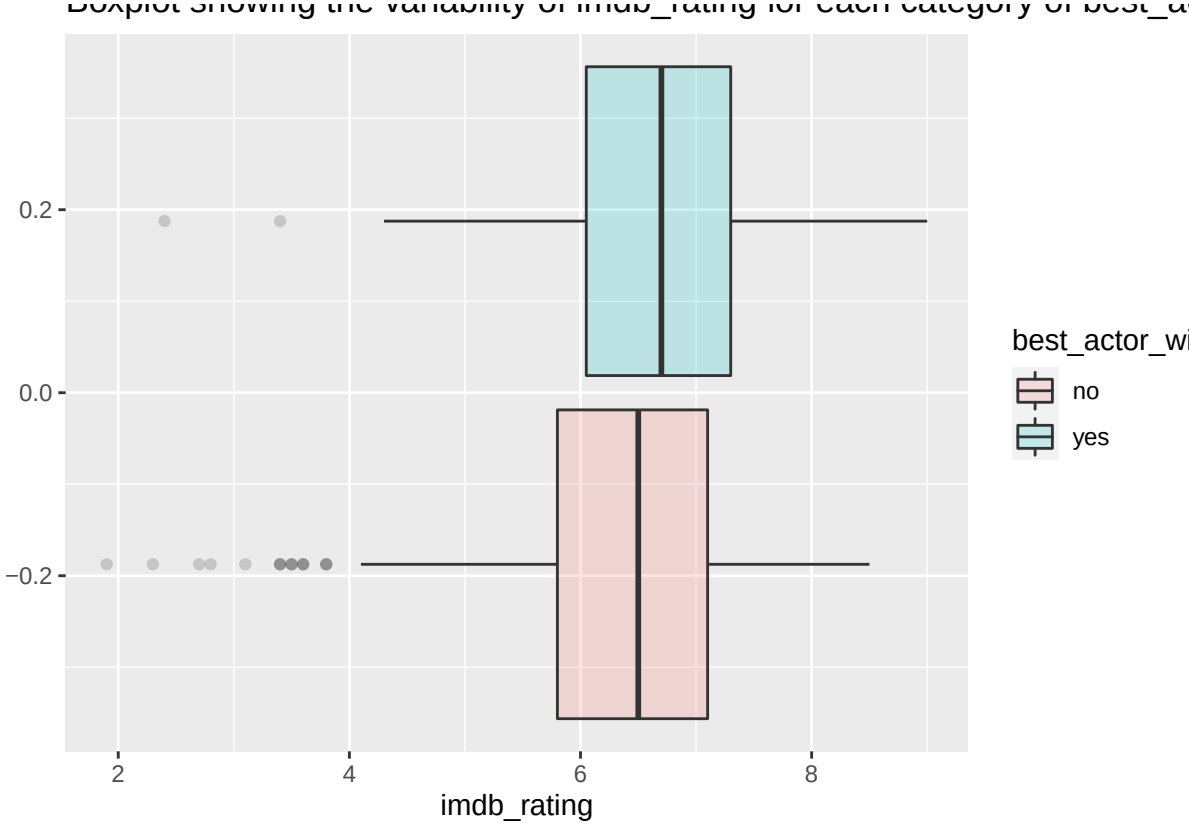
```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 13 x 2
##    dvd_rel_month `n()`
##    <fct>         <int>
##  1 1                45
##  2 2                46
##  3 3                68
##  4 4                45
##  5 5                60
##  6 6                54
##  7 7                38
##  8 8                45
##  9 9                49
## 10 10               50
## 11 11               39
## 12 12               43
## 13 <NA>              6
```

```
# converting dvd_rel_month to a numerical variable
movies$dvd_rel_month <- as.numeric(movies$dvd_rel_month)
```

The are a maximum number of 68 movies in each month and a minimum of 6 for dvd releases.

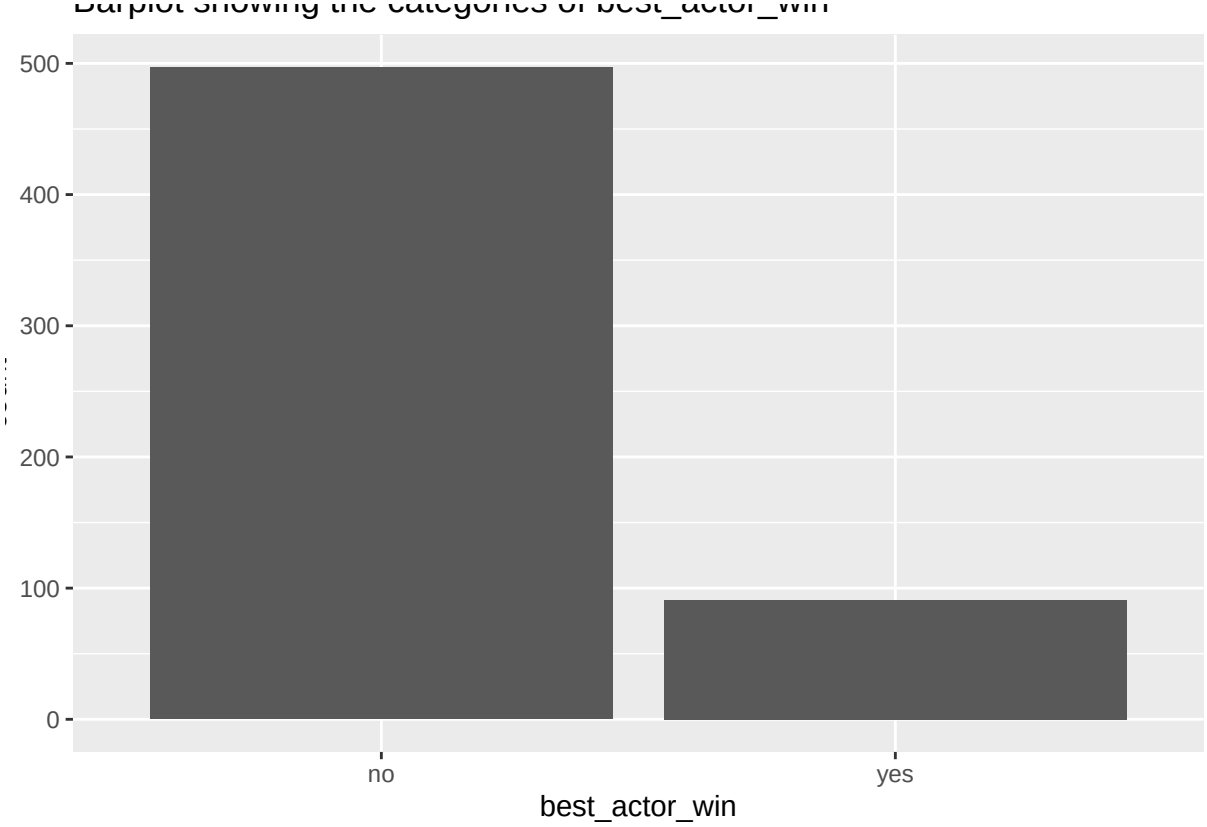
```
movies %>%
    group_by(best_actor_win) %>%
    ggplot(aes(x = imdb_rating, fill = best_actor_win)) +
    geom_boxplot(alpha=.2) +
    ggtitle("Boxplot showing the variability of imdb_rating for each category of best_actor_win")
```



best_actor_win

The variability for imdb_rating is consistent between the two categories.

```
ggplot(data = movies, aes(x = best_actor_win)) +
  geom_bar() +
  ggtitle("Barplot showing the categories of best_actor_win")
```



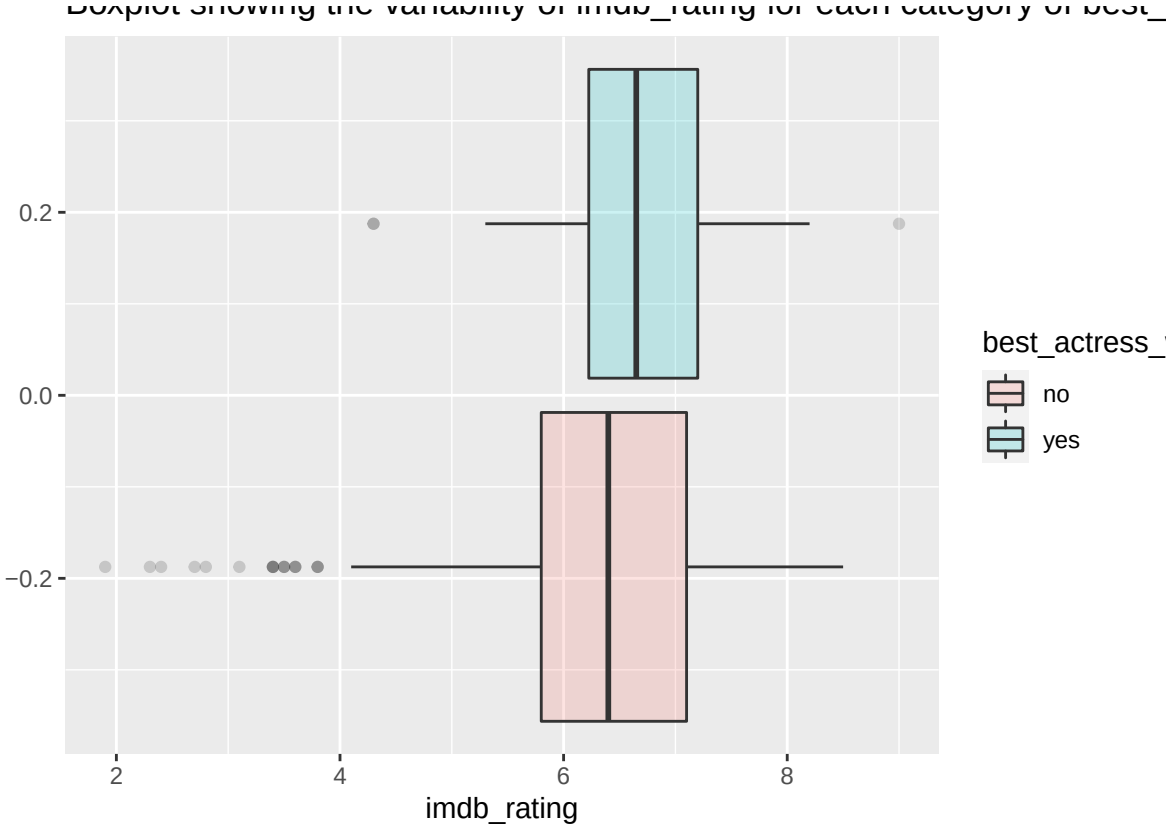

```
movies %>%
  group_by(best_actor_win) %>%
  summarize(n())
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 2 x 2
##   best_actor_win `n()`
##   <fct>          <int>
## 1 no               497
## 2 yes               91
```

There are significantly more movies with a single lead actor that hasn't won an oscar than those movies with a single lead actor that has won an oscar.

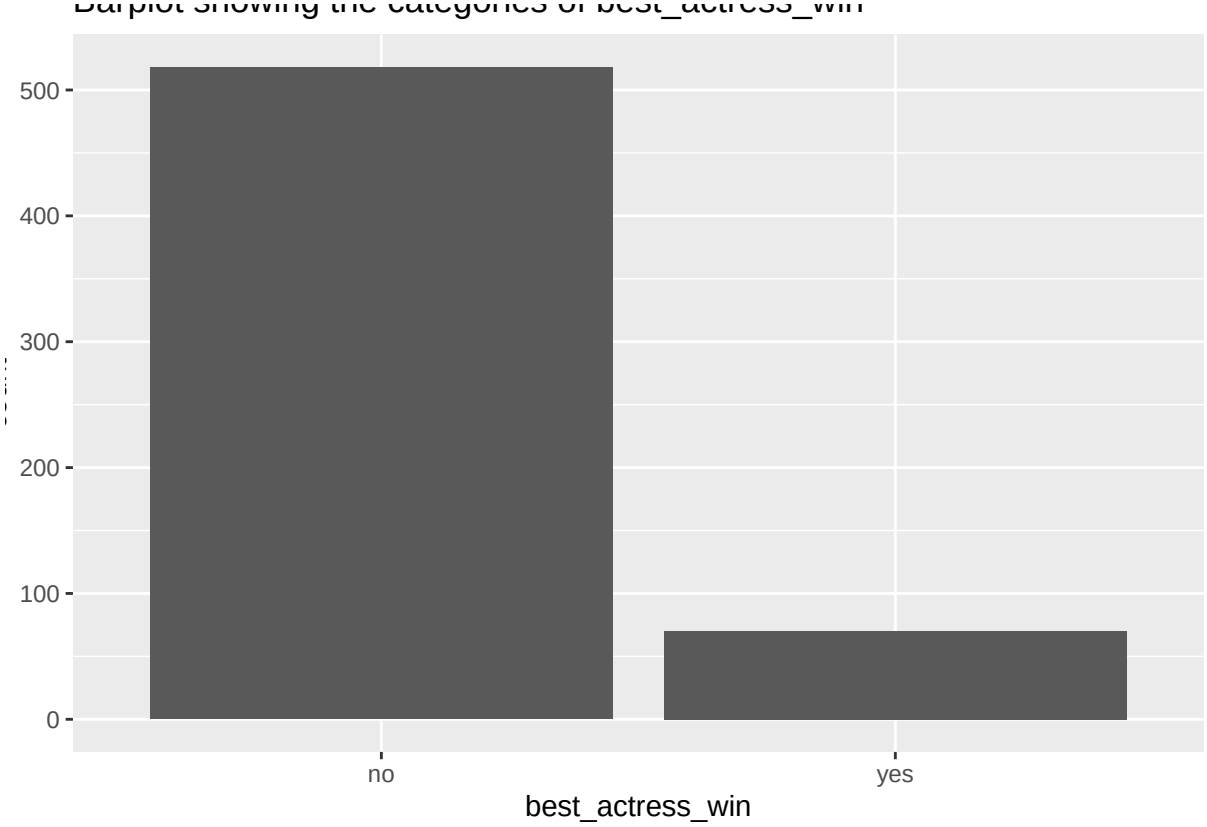
```
movies %>%
    group_by(best_actress_win) %>%
    ggplot(aes(x = imdb_rating, fill = best_actress_win)) +
    geom_boxplot(alpha=.2) +
    ggtitle("Boxplot showing the variability of imdb_rating for each category of best_actress_win")
```



best_actress_win

The variability for imdb_rating is consistent between the two categories.

```
ggplot(data = movies, aes(x = best_actress_win)) +
  geom_bar() +
  ggtitle("Barplot showing the categories of best_actress_win")
```



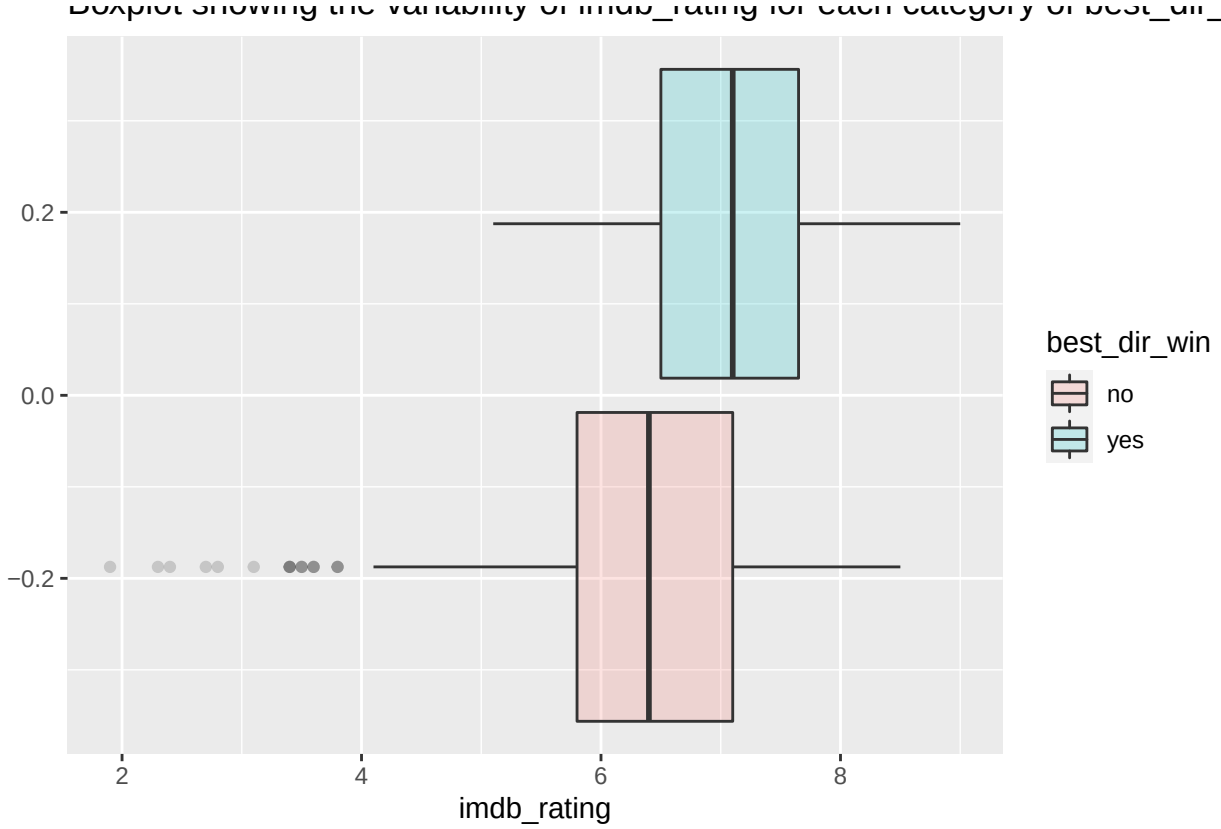

```
movies %>%
  group_by(best_actress_win) %>%
  summarize(n())
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 2 x 2
##   best_actress_win `n()`
##   <fct>            <int>
## 1 no                 518
## 2 yes                 70
```

There are significantly more movies with a single lead actress that hasn't won an oscar than those movies with a single lead actress that has won an oscar.

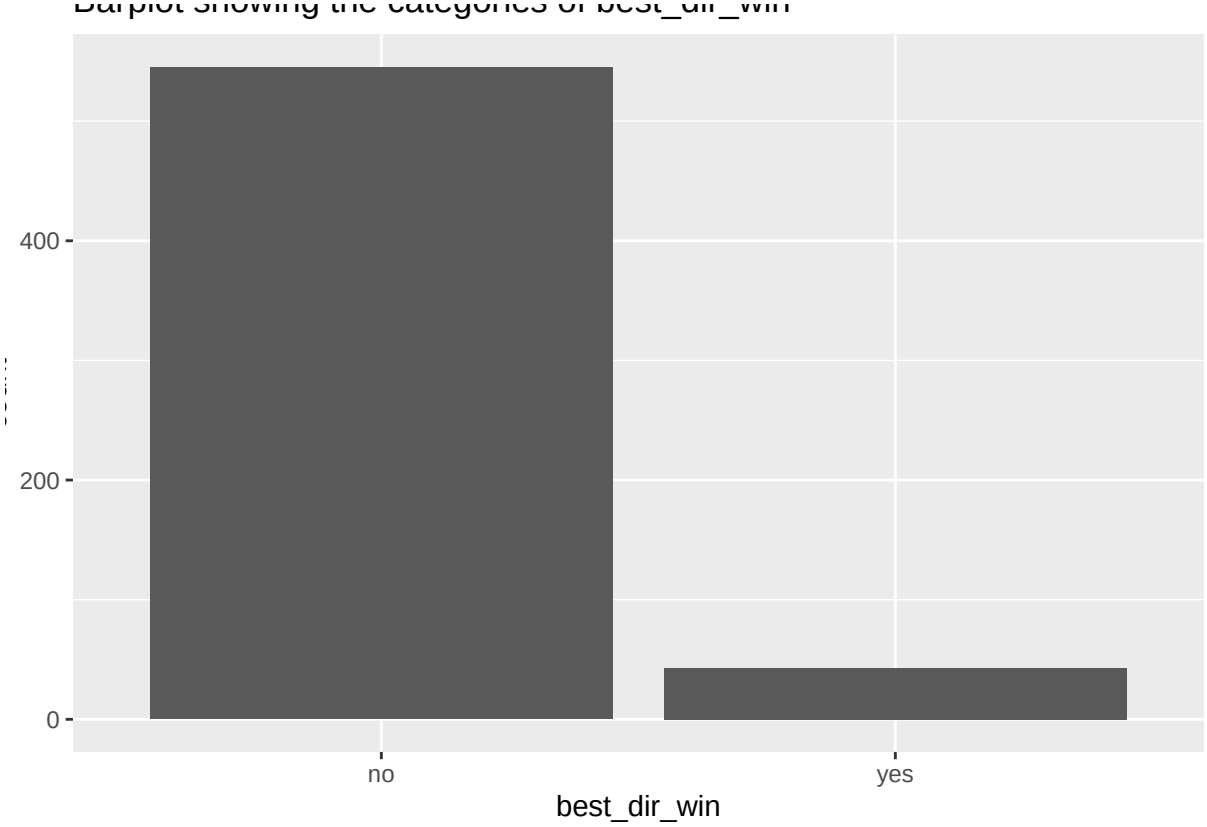
```
movies %>%
    group_by(best_dir_win) %>%
    ggplot(aes(x = imdb_rating, fill = best_dir_win)) +
    geom_boxplot(alpha=.2) +
    ggtitle("Boxplot showing the variability of imdb_rating for each category of best_dir_win")
```



best_dir_win

The variability for imdb_rating is consistent between the two categories. But movies with a best_dir_win being yes seem to have a higher rating.

```
ggplot(data = movies, aes(x = best_dir_win)) +
  geom_bar() +
  ggtitle("Barplot showing the categories of best_dir_win")
```




```
movies %>%
  group_by(best_dir_win) %>%
  summarize(n())
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

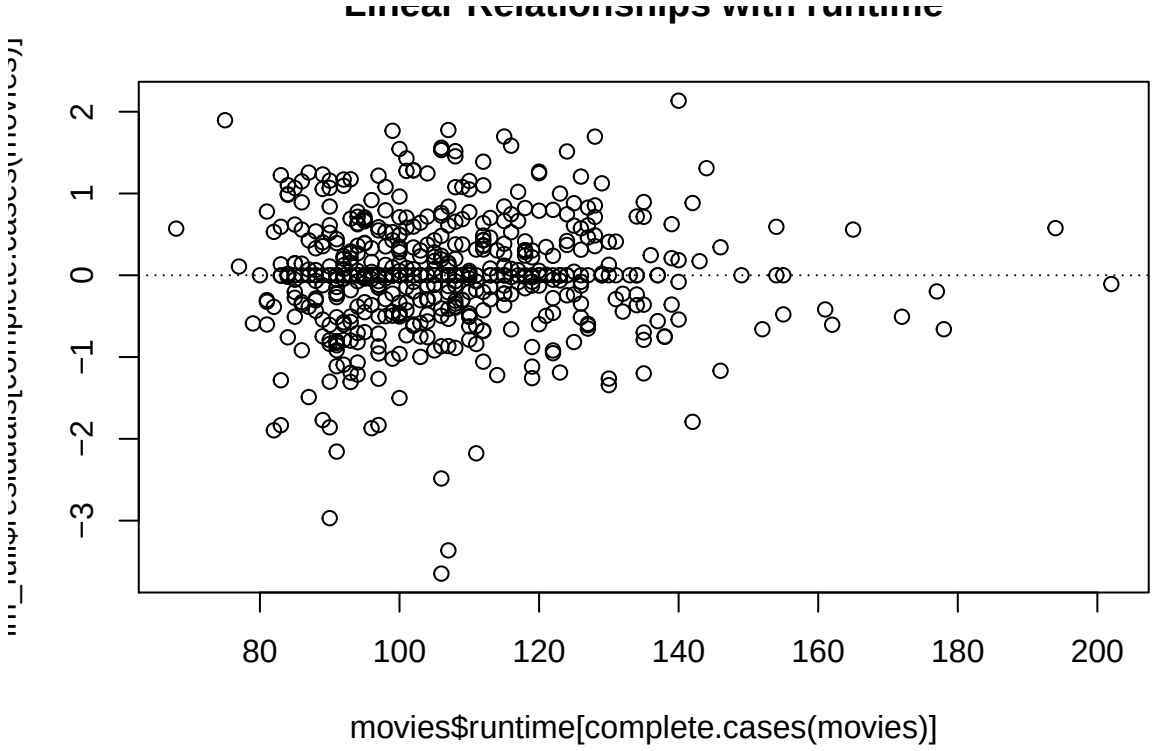
```
## # A tibble: 2 x 2
##   best_dir_win `n()`
##   <fct>        <int>
## 1 no             545
## 2 yes             43
```

There are significantly more movies with a director that hasn't won an oscar than those movies with a director that has won an oscar.

Conditions for Multiple Linear Regression (MLR)

1. Linear Relationships between numerical variables and response variable

```
lm_full <- lm(imdb_rating ~ genre + runtime + mpaa_rating + studio
              + thtr_rel_year + thtr_rel_month + thtr_rel_day + dvd_rel_year
              + dvd_rel_month + dvd_rel_day + best_actor_win + best_actress_win
              + best_dir_win, data = movies)

plot(lm_full$residuals[complete.cases(movies)] ~ movies$runtime[complete.cases(movies)])
  abline(0, 0, lty = 3) +
  title("Linear Relationships with runtime")
```



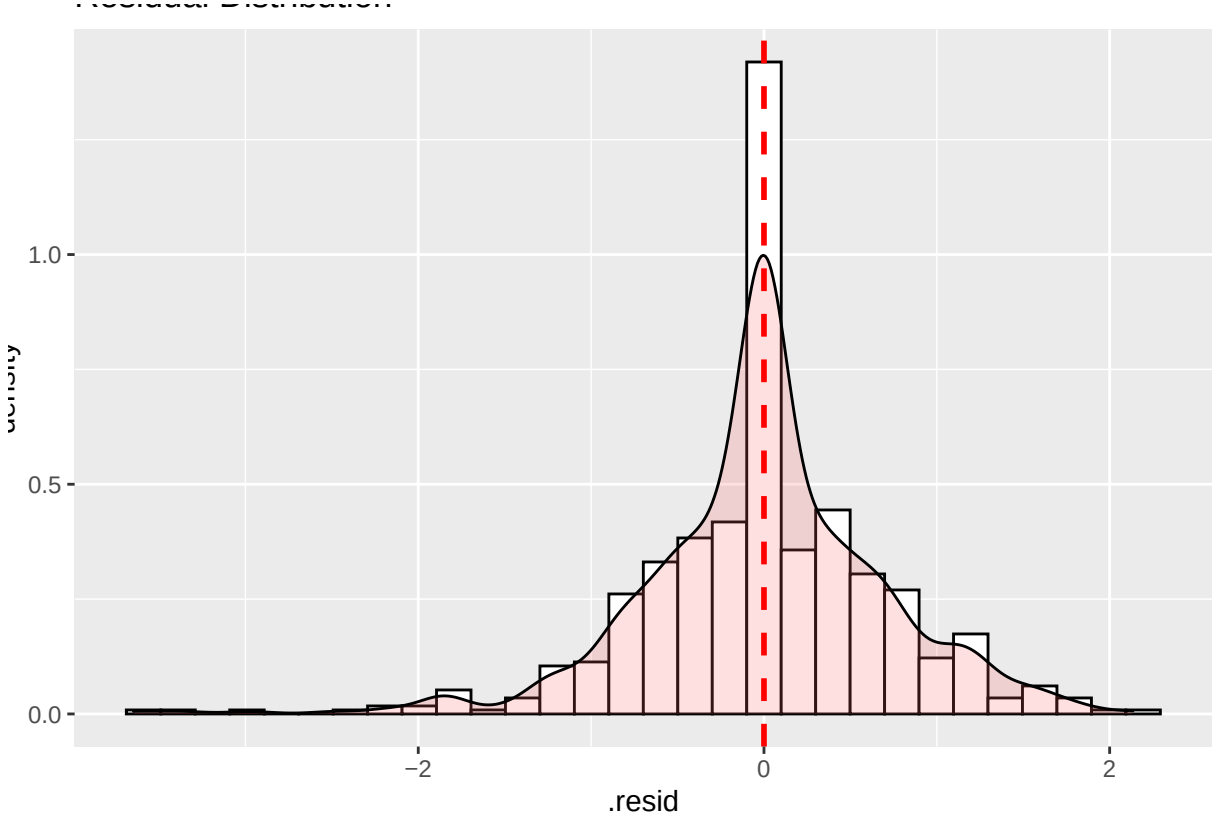

```
## integer(0)
```

There seems to be no relationships between the numerical variable, `runtime`, and the response variable, `imdb_rating`. There is a random scatter around 0 for every plot above. Hence, we can consider `runtime` in our analysis.

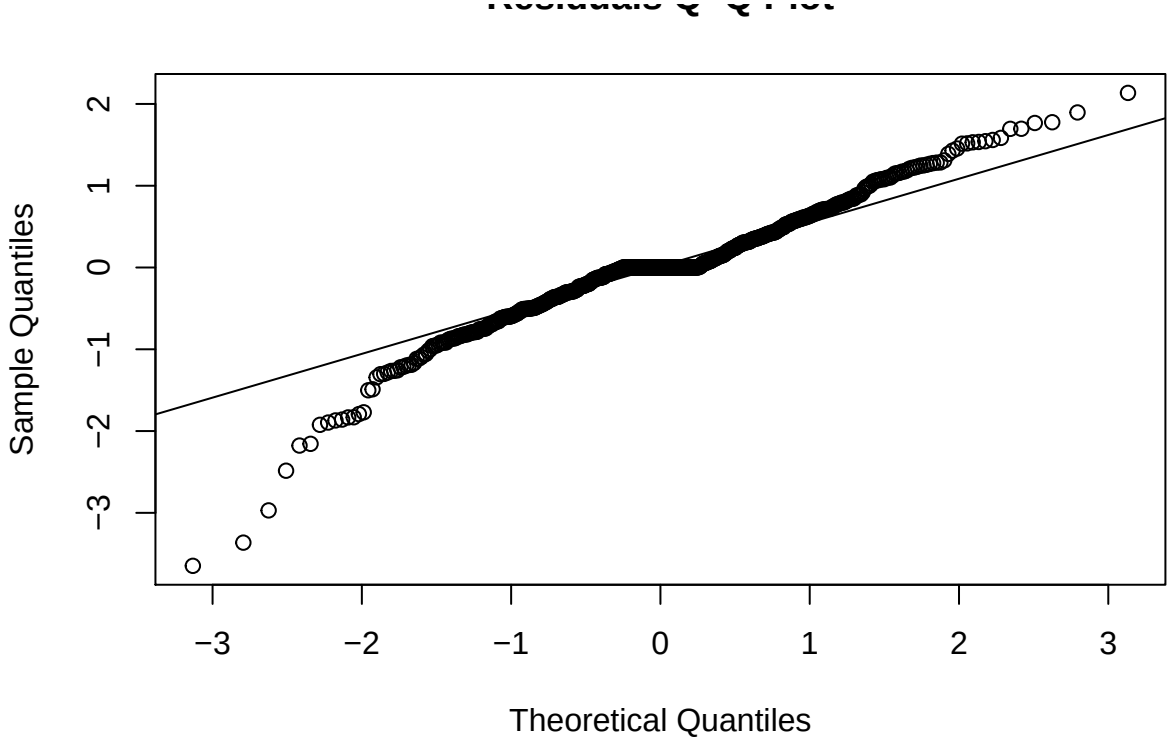
```
ggplot(lm_full, aes(x=.resid)) +
  geom_histogram(aes(y=..density..), color="black", fill="white") +
  geom_density(alpha=0.2, fill="#FF6666") +
  geom_vline(aes(xintercept=mean(lm_full$residuals)), col = 'red', lwd = 1, lty = 2) +
  ggtitle("Residual Distribution")
```

2. Nearly Normal Residuals

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



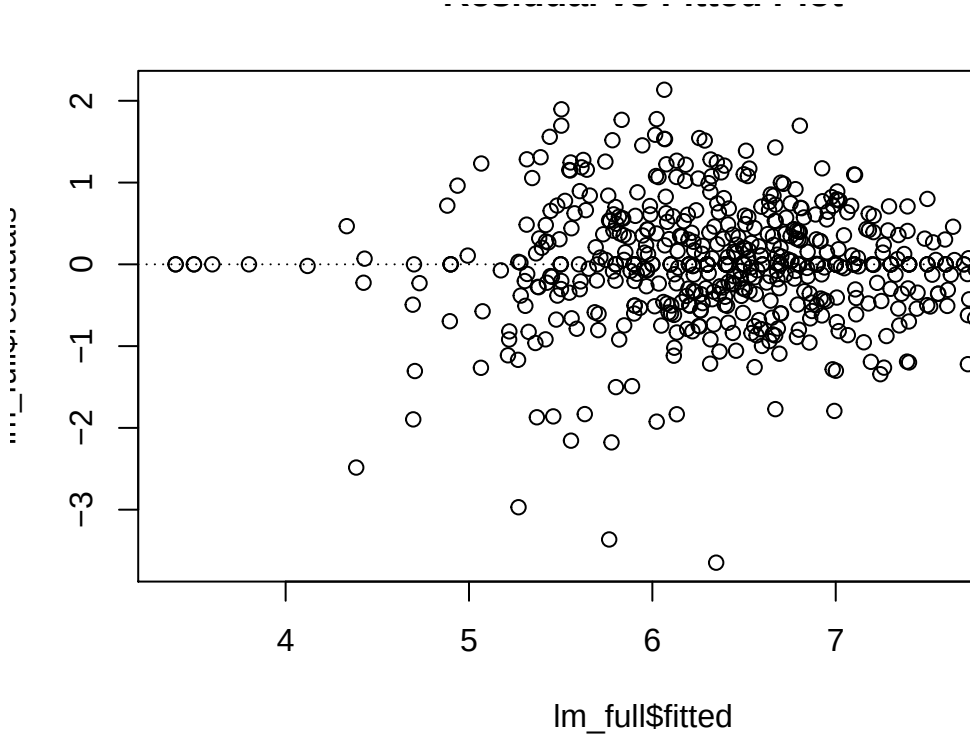

```
qqnorm(lm_full$residuals, main = "Residuals Q-Q Plot")
qqline(lm_full$residuals)
```



The residuals are fairly symmetric, with only a slightly longer tail on the left, hence it would be appropriate

to deem the normal distribution of residuals condition met.

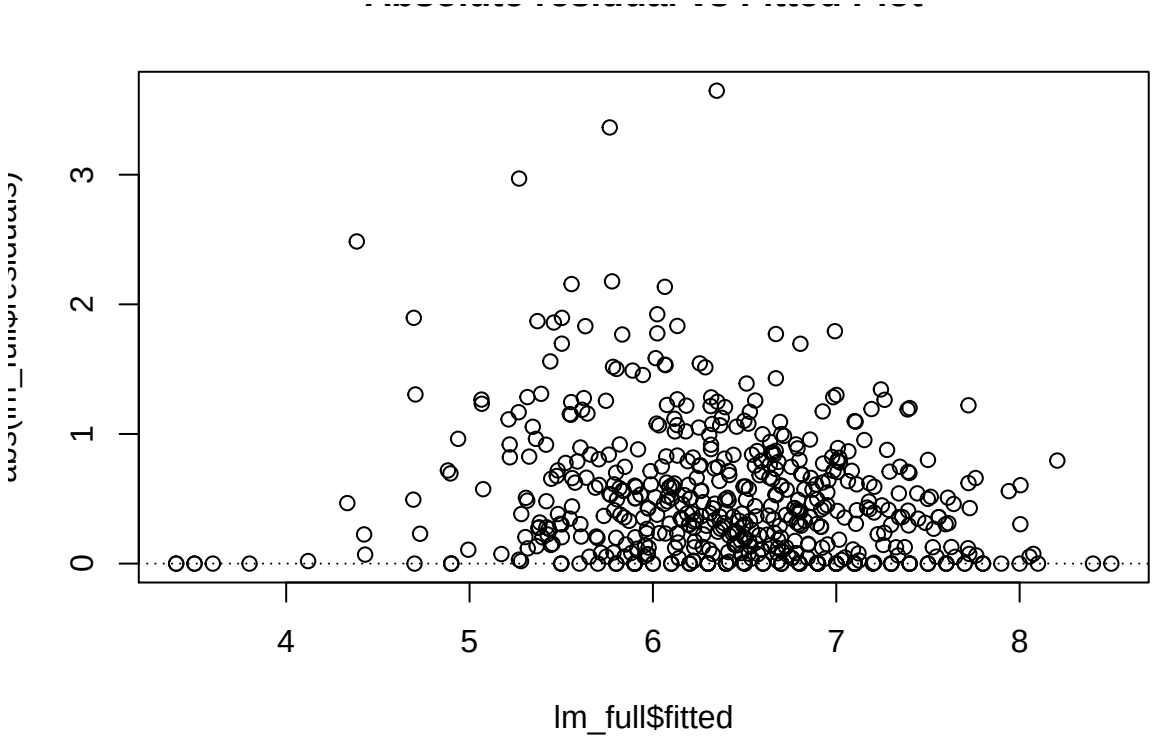
```
plot(lm_full$residuals ~ lm_full$fitted) +
  abline(0, 0, lty = 3) +
  title("Residual vs Fitted Plot")
```



3. Constant variability of residuals

```
## integer(0)
```

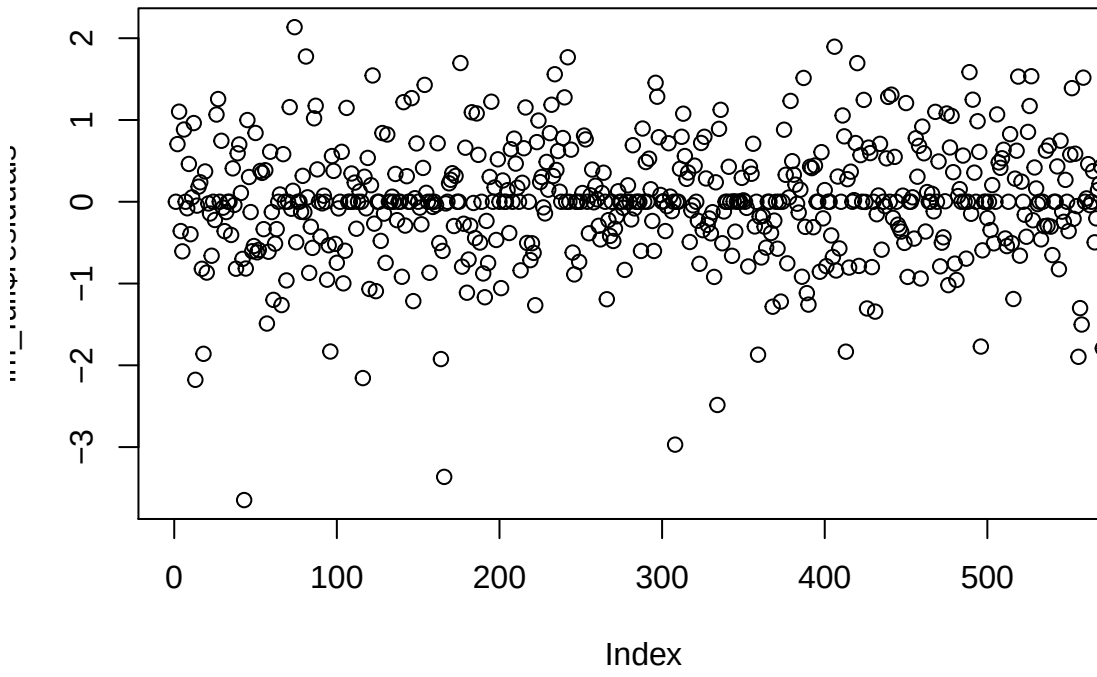
```
plot(abs(lm_full$residuals) ~ lm_full$fitted) +
  abline(0, 0, lty = 3) +
  title("Absolute residual vs Fitted Plot")
```




```
## integer(0)
```

The absolute value of the residuals don't follow a triangle and the scatter is spread above and below the mean. It's safe to say homoscedasticity is present and the variability of the residuals is constant.

```
plot(lm_full$residuals)
```



4. Independent residuals

There is no intrinsic ordering present in the residuals and no relation present in the graph above, hence we

can say that the residuals are independent of each other.

Part 4: Modeling

Variable Selection

The explanatory variables of interest in our analysis are as follows:

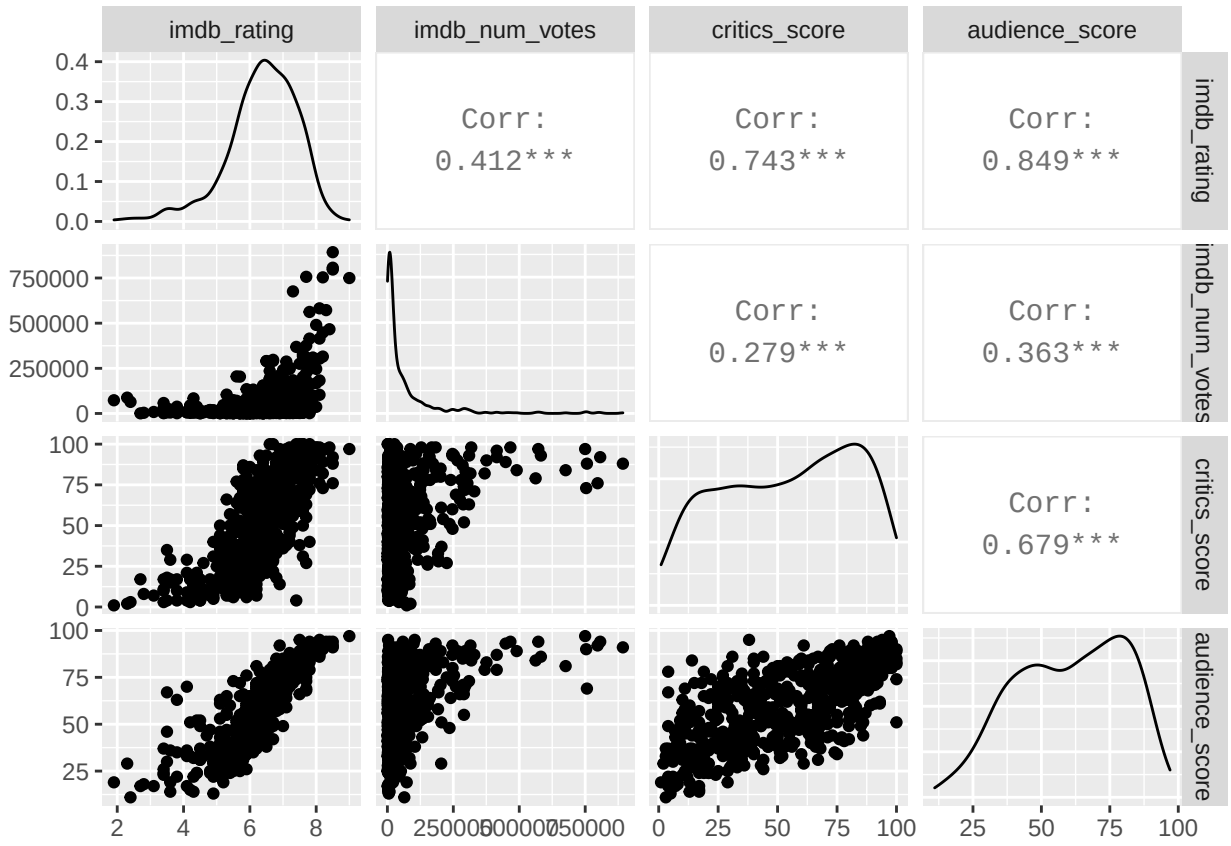
Column	Description
title_type	Type of movie (Documentary, Feature Film, TV Movie)
genre	Genre of movie (Action & Adventure, Comedy, Documentary, Drama, Horror, Mystery & Suspense, Other)
runtime	Runtime of movie (in minutes)
mpaa_rating	MPAA rating of the movie (G, PG, PG-13, R, Unrated)
studio	Studio that produced the movie
thtr_rel_year	Year the movie is released in theaters
thtr_rel_month	Month the movie is released in theaters
thtr_rel_day	Day of the month the movie is released in theaters
dvd_rel_year	Year the movie is released on DVD
dvd_rel_month	Month the movie is released on DVD
dvd_rel_day	Day of the month the movie is released on DVD
best_actor_win	Whether or not one of the main actors in the movie ever won an Oscar (no, yes) – note that this is not necessarily whether the actor won an Oscar for their role in the given movie
best_actress_win	Whether or not one of the main actresses in the movie ever won an Oscar (no, yes) – not that this is not necessarily whether the actresses won an Oscar for their role in the given movie
best_dir_win	Whether or not the director of the movie ever won an Oscar (no, yes) – not that this is not necessarily whether the director won an Oscar for the given movie

This analysis will not consider the following as explanatory variables. These correspond to so some form of ratings or ranking on Imdb, rotten tomatoes or BoxOfficeMojo:

- `imdb_num_votes`: Number of votes on IMDB
- `critics_rating`: Categorical variable for critics rating on Rotten Tomatoes (Certified Fresh, Fresh, Rotten)
- `critics_score`: Critics score on Rotten Tomatoes
- `audience_rating`: Categorical variable for audience rating on Rotten Tomatoes (Spilled, Upright)
- `audience_score`: Audience score on Rotten Tomatoes
- `top200_box`: Whether or not the movie is in the Top 200 Box Office list on BoxOfficeMojo (no, yes)
- `best_pic_nom`: Whether or not the movie was nominated for a best picture Oscar (no, yes)
- `best_pic_win`: Whether or not the movie won a best picture Oscar (no, yes)

Based on domain knowledge, ratings on other websites wouldn't be available prior to the release of the movie and such variables are likely to have high collinearity with the response variable which will result in incorrect estimators for other factors in this analysis. We can use the a pairplot to verify our assumption:

```
# numerical variables
ggpairs(movies, columns = c('imdb_rating','imdb_num_votes','critics_score','audience_score'))
```



As mentioned earlier, some numerical rating variables such as `audience_score` and `critics_score` on rotten tomatoes tend to result in a high correlation coefficient. Thus, they have a strong linear association with the response variable, which could result in biased estimators for other variables. Furthermore, these variables won't be available to us prior to the release of a movie.

In terms of categorical variables, we can use ANOVA to verify if the variability in the response variable, `imbd_rating`, can be explained by each of the categorical rating variables.

In order to verify ANOVA conditions:

```
movies %>%
  group_by(critics_rating) %>%
  summarize(n())
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 3 x 2
##   critics_rating  `n()`
##   <fct>           <int>
## 1 Certified Fresh   116
## 2 Fresh             171
## 3 Rotten            301
```

```
movies %>%
  group_by(audience_rating) %>%
  summarize(n())
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 2 x 2
```

```
##   audience_rating `n()`
##   <fct>           <int>
## 1 Spilled           272
## 2 Upright           316
```

```
movies %>%
  group_by(top200_box) %>%
  summarize(n())
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

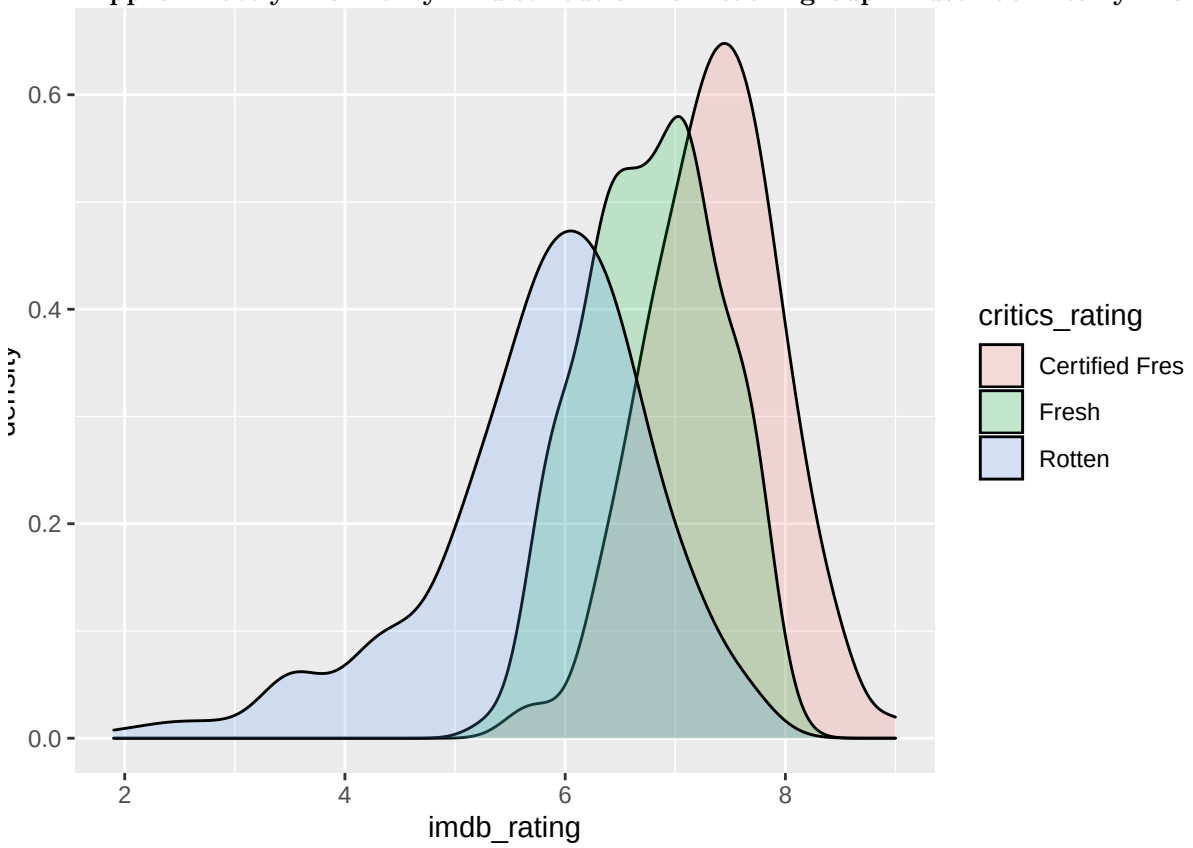
```
## # A tibble: 2 x 2
##   top200_box `n()`
##   <fct>      <int>
## 1 no           573
## 2 yes           15
```

1. Independence:

- within groups: Sampled observations in each group can be assumed independent of every other group. Furthermore, each customer can only give one rating. Each group has less than 10% of the population based on the results in the tables above.
- between groups: one must be vary of audience and critic ratings since a critic could rate a movie as an audience and vice versa.

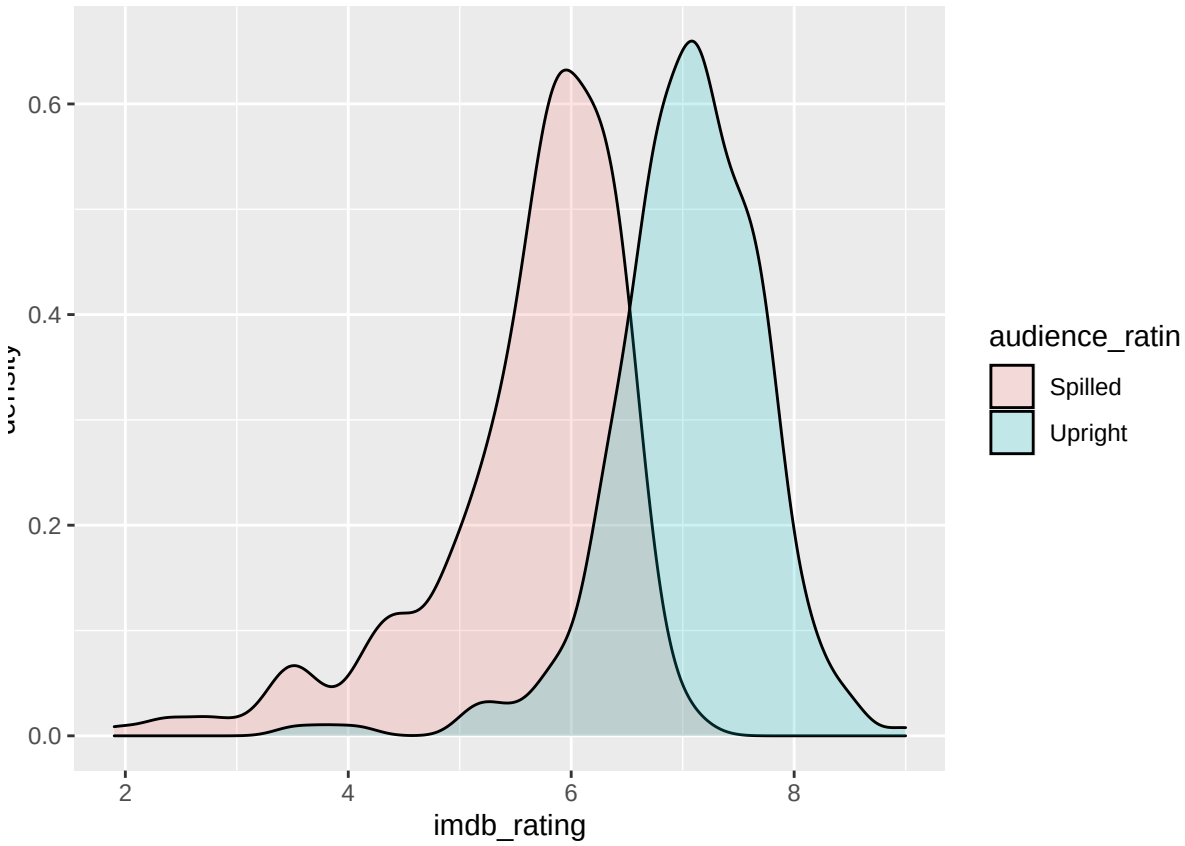
```
movies %>%
  group_by(critics_rating) %>%
  ggplot(aes(x = imdb_rating, fill=critics_rating)) +
  geom_density(alpha=.2)
```

2. Approximately normality: distribution of each group must be nearly normal.



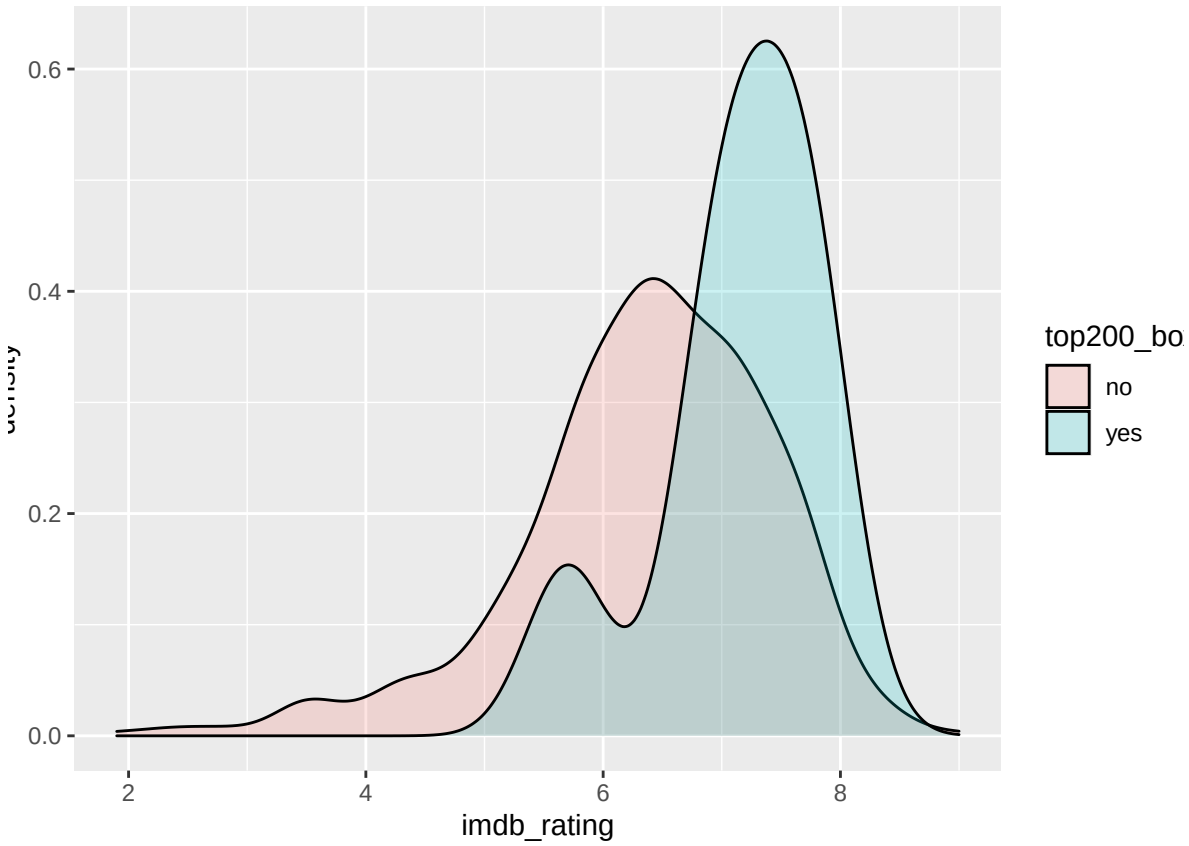
The distributions of the response variables for each `critics_rating` are nearly normal, even though there's a left skew.

```
movies %>%
  group_by(audience_rating) %>%
  ggplot(aes(x = imdb_rating, fill=audience_rating)) +
  geom_density(alpha=.2)
```



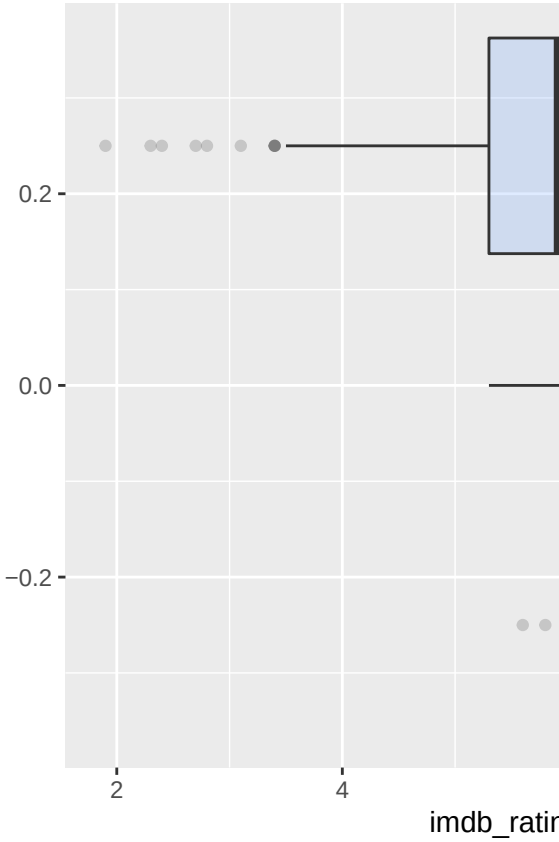
The distributions of the response variables for each `audience_rating` are nearly normal, even though there's a left skew.

```
movies %>%
  group_by(top200_box) %>%
  ggplot(aes(x = imdb_rating, fill=top200_box)) +
  geom_density(alpha=.2)
```



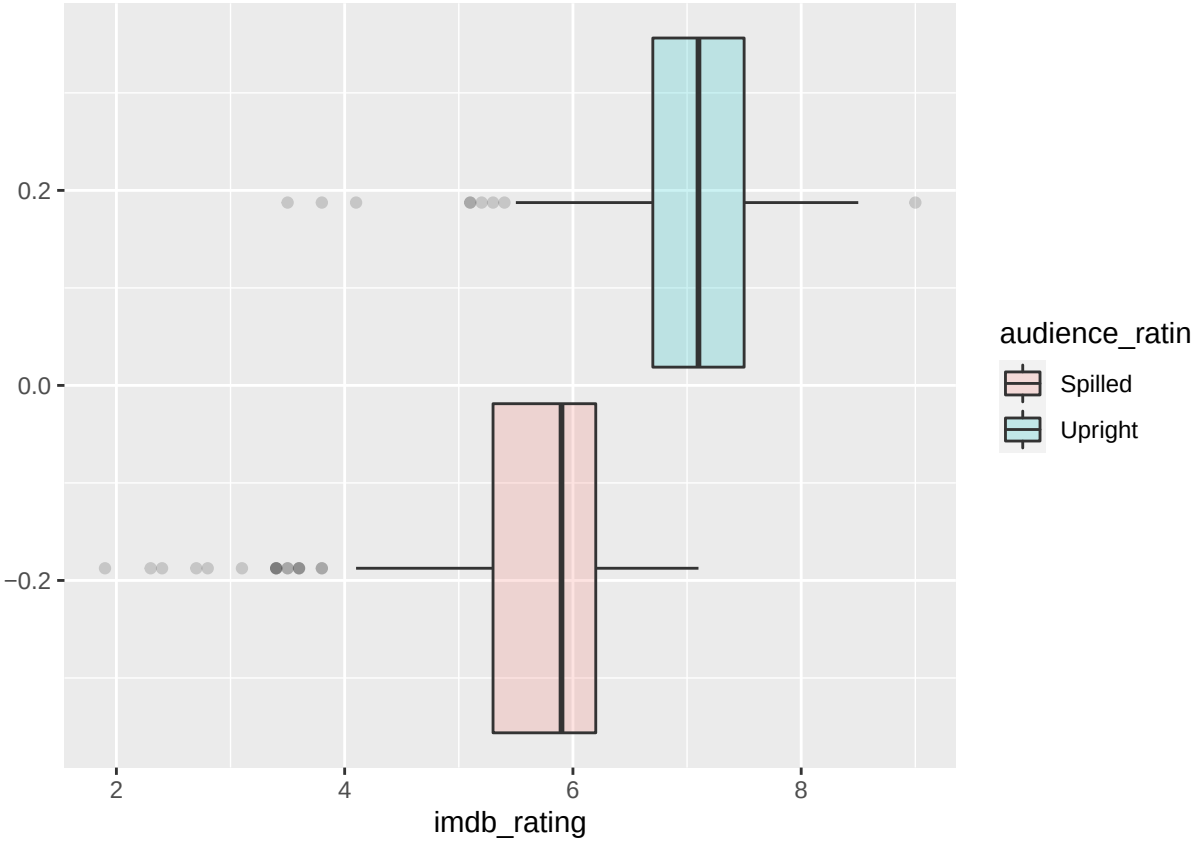
The distributions of the response variables for each `top200_box` are nearly normal, even though there's a left skew.

```
movies %>%
  group_by(critics_rating) %>%
  ggplot(aes(x = imdb_rating, fill=critics_rating)) +
  geom_boxplot(alpha=.2)
```

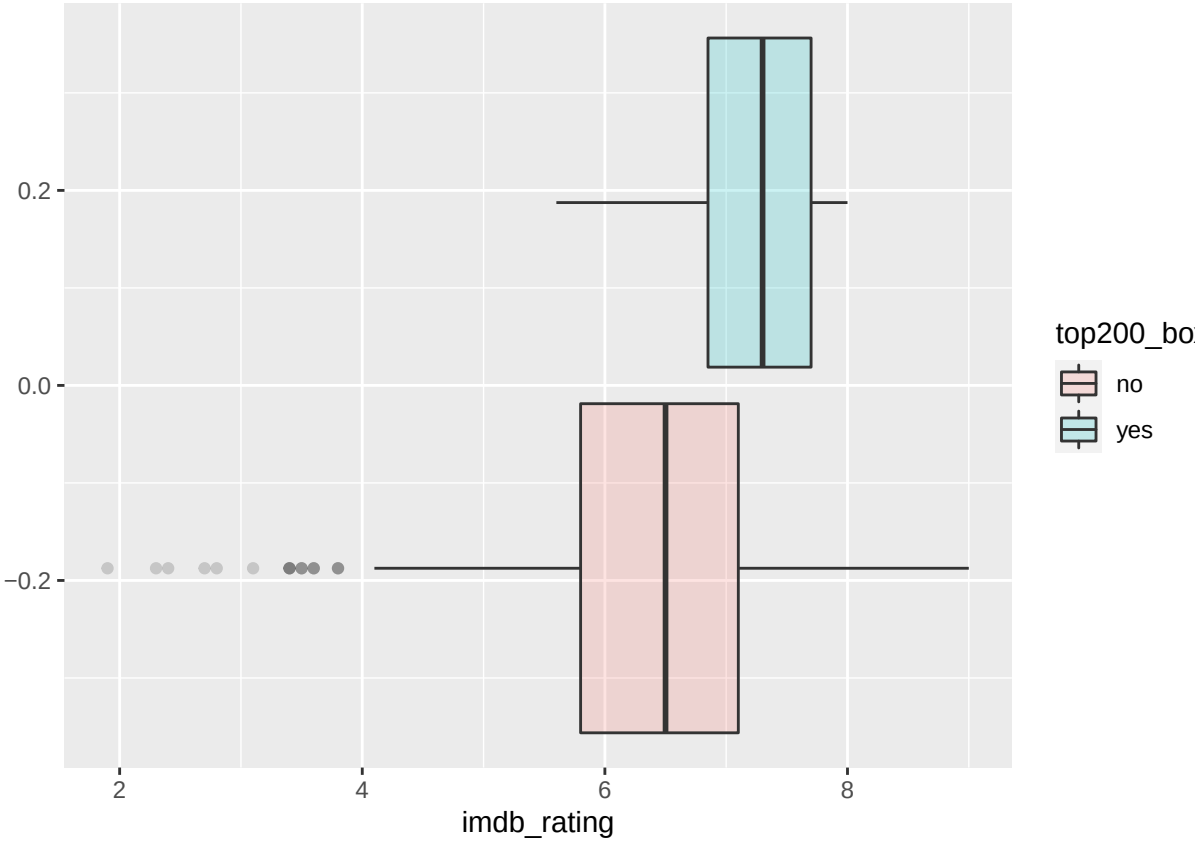


3. Equal Variance: Groups should have roughly equal variability.

```
movies %>%
  group_by(audience_rating) %>%
  ggplot(aes(x = imdb_rating, fill=audience_rating)) +
  geom_boxplot(alpha=.2)
```




```
movies %>%
  group_by(top200_box) %>%
  ggplot(aes(x = imdb_rating, fill=top200_box)) +
  geom_boxplot(alpha=.2)
```



Based on the boxplots above, the variability between groups is roughly equal.

We can hence run ANOVA:

```
cat_aov <- aov(imdb_rating ~ critics_rating + audience_rating + top200_box, data = movies)
summary(cat_aov)
```

```
##                   Df Sum Sq Mean Sq F value Pr(>F)
## critics_rating     2 245.58  122.79 242.669 <2e-16 ***
## audience_rating    1 115.48  115.48 228.227 <2e-16 ***
## top200_box         1   0.13    0.13   0.261   0.61
## Residuals        583 295.00    0.51
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Since the p-value for two out of the three categories are high, there are statistically significant differences between group means as determined. These variables would thus result in biased estimators. Furthermore, since they're not known prior to the release of a movie, it wouldn't contribute to the analysis.

Model Selection

Since we're interested in predicting the factors of significance that constitute a move with a high rating, we'll be using the p-value for model selection. These factors would be influential drivers in determining what the next "best" movie should consist of.

We'll be using backward selection since we'll have to refit fewer models to identify the variables with significant p-values.

```
lm_full <- lm(imdb_rating ~ genre + runtime + mpaa_rating + studio
              + thtr_rel_year + thtr_rel_month + thtr_rel_day + dvd_rel_year
              + dvd_rel_month + dvd_rel_day + best_actor_win + best_actress_win
              + best_dir_win, data = movies)
sum_full <- summary(lm_full)
tail(sum_full$coefficients[,4])
```

1. Starting with the complete model

```
##        dvd_rel_year       dvd_rel_month         dvd_rel_day   best_actor_winyes
##          0.60478458          0.02220094          0.43961329          0.50362941
## best_actress_winyes     best_dir_winyes
##          0.59679391          0.07302599
```

2. Drop the variable with the highest p-value and refit the model Since majority of the above variables are categorical variables, elimination of a single variable would mean dropping multiple estimators in the above output. However, since `dvd_rel_year` has the highest p-value, we'll eliminate it and reconstruct the multiple linear regression model.

```
lm1 <- lm(imdb_rating ~ genre + runtime + mpaa_rating + studio
          + dvd_rel_month + dvd_rel_day + best_actor_win + best_actress_win
          + thtr_rel_year + thtr_rel_month + thtr_rel_day + best_dir_win, data = movies)
sum_lm1 <- summary(lm1)
tail(sum_lm1$coefficients[,4], 10)
```

```
##                   studioWinstar studioYari Film Group Releasing
##                      0.30723252                       0.46194530
##                   dvd_rel_month                     dvd_rel_day
##                      0.01976738                       0.38114633
##               best_actor_winyes             best_actress_winyes
##                      0.54740013                       0.61161047
##                   thtr_rel_year                  thtr_rel_month
##                      0.22906467                       0.08259358
##                    thtr_rel_day                 best_dir_winyes
##                      0.44079316                       0.07224884
```

3. Repeat until all remaining variables are significant The next highest p-value is that of `best_actress_winyes`, so we'll be eliminating it and re-fitting our model.

```
lm2 <- lm(imdb_rating ~ genre + runtime + mpaa_rating + studio
          + dvd_rel_month + dvd_rel_day +  best_actor_win
          + thtr_rel_year + thtr_rel_month + thtr_rel_day + best_dir_win, data = movies)

tail(summary(lm2)$coefficients[,4], 10)
```

```
##         studioWeinstein Company                   studioWinstar
##                      0.12223772                       0.30248289
## studioYari Film Group Releasing                   dvd_rel_month
##                      0.45395677                       0.01966244
##                     dvd_rel_day               best_actor_winyes
##                      0.37907552                       0.52448122
##                   thtr_rel_year                  thtr_rel_month
##                      0.23779798                       0.08205479
##                    thtr_rel_day                 best_dir_winyes
##                      0.44520851                       0.06966679
```

Rather than displaying each summary, this project will be running through each iteration in the code below.

```
# removing best_actor_win
lm3 <- lm(imdb_rating ~ genre + runtime + mpaa_rating + studio
          + dvd_rel_month + dvd_rel_day + thtr_rel_year + thtr_rel_month
          + thtr_rel_day + best_dir_win, data = movies)
tail(summary(lm3)$coefficients[,4])
```

```
##   dvd_rel_month     dvd_rel_day   thtr_rel_year  thtr_rel_month    thtr_rel_day
##      0.01622719      0.38961000      0.23515472      0.08613942      0.44862899
## best_dir_winyes
##      0.06847065
```

```
# removing dvd_rel_day
lm4 <- lm(imdb_rating ~ genre + runtime + mpaa_rating + studio
          + dvd_rel_month + thtr_rel_year + thtr_rel_month + thtr_rel_day
          + best_dir_win, data = movies)
tail(summary(lm4)$coefficients[,4])
```

```
## studioYari Film Group Releasing                   dvd_rel_month
##                      0.40935158                      0.01631044
##                   thtr_rel_year                  thtr_rel_month
##                      0.24617920                      0.08134644
##                    thtr_rel_day                 best_dir_winyes
##                      0.42424316                      0.06106955
```

```
# removing thtr_rel_day
lm5 <- lm(imdb_rating ~ genre + runtime + mpaa_rating + studio
          + dvd_rel_month + thtr_rel_year + thtr_rel_month
          + best_dir_win, data = movies)
tail(summary(lm5)$coefficients[,4])
```

```
##                   studioWinstar studioYari Film Group Releasing
##                      0.27721272                      0.39370491
##                   dvd_rel_month                   thtr_rel_year
##                      0.01703754                      0.23517938
##                  thtr_rel_month                 best_dir_winyes
##                      0.06868953                      0.06335936
```

```
# removing thtr_rel_year
lm6 <- lm(imdb_rating ~ genre + runtime
          + mpaa_rating + studio + dvd_rel_month + thtr_rel_month
          + best_dir_win, data = movies)
tail(summary(lm6)$coefficients[,4])
```

```
##         studioWeinstein Company                    studioWinstar
##                      0.12850138                      0.24786175
## studioYari Film Group Releasing                   dvd_rel_month
##                      0.39584130                      0.01678922
##                  thtr_rel_month                 best_dir_winyes
##                      0.07270783                      0.06004186
```

```
# removing thtr_rel_month
lm7 <- lm(imdb_rating ~ genre + runtime
          + mpaa_rating + studio + dvd_rel_month
          + best_dir_win, data = movies)
tail(summary(lm7)$coefficients[,4])
```

```
##    studioWarners Bros. Pictures         studioWeinstein Company
##                       0.65239049                       0.12384259
##                    studioWinstar studioYari Film Group Releasing
##                       0.19872018                       0.38536455
##                    dvd_rel_month                 best_dir_winyes
##                       0.03377084                       0.06421385
```

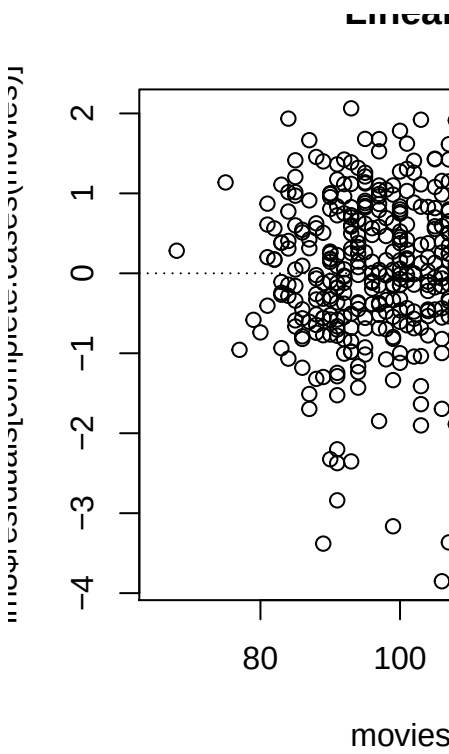
```
# removing studio
lm8 <- lm(imdb_rating ~ genre + runtime
          + mpaa_rating + dvd_rel_month
          + best_dir_win, data = movies)
summary(lm8)
```

```
##
## Call:
## lm(formula = imdb_rating ~ genre + runtime + mpaa_rating + dvd_rel_month +
##     best_dir_win, data = movies)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.8511 -0.5363  0.0455  0.6024  2.0635
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     4.895694   0.370117  13.227  < 2e-16 ***
## genreAnimation                 -0.259133   0.366483  -0.707 0.479809
## genreArt House & International  0.703054   0.293003   2.399 0.016742 *
## genreComedy                    -0.056625   0.155904  -0.363 0.716588
## genreDrama                      0.605792   0.132156   4.584 5.62e-06 ***
## genreHorror                    -0.097850   0.231485  -0.423 0.672671
## genreMusical & Performing Arts  0.956580   0.347749   2.751 0.006136 **
## genreMystery & Suspense         0.391019   0.171386   2.282 0.022890 *
## genreOther                      0.709728   0.266783   2.660 0.008029 **
## genreScience Fiction & Fantasy  0.001489   0.345898   0.004 0.996566
## runtime                         0.015941   0.002432   6.555 1.25e-10 ***
## mpaa_ratingNC-17               -0.626455   0.707552  -0.885 0.376327
## mpaa_ratingPG                  -0.761632   0.279377  -2.726 0.006606 **
## mpaa_ratingPG-13               -1.035342   0.284002  -3.646 0.000292 ***
## mpaa_ratingR                   -0.707375   0.277399  -2.550 0.011035 *
## mpaa_ratingUnrated             -0.406057   0.371597  -1.093 0.274977
## dvd_rel_month                   0.024303   0.011404   2.131 0.033504 *
## best_dir_winyes                 0.379352   0.152606   2.486 0.013213 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.919 on 564 degrees of freedom
##   (6 observations deleted due to missingness)
## Multiple R-squared:  0.2537, Adjusted R-squared:  0.2312
## F-statistic: 11.28 on 17 and 564 DF,  p-value: < 2.2e-16
```

Since the p-value for the entire value is below 0.05, the model as a whole is significant. This also means that at least one of the predictors are significant (or not 0), conditional on the other variables included in the model.

Model Diagnostics

```
plot(lm8$residuals[complete.cases(movies)] ~ movies$runtime[complete.cases(movies)])
  abline(0, 0, lty = 3) +
  title("Linear Relationships with runtime")
```



1. Linear Relationships between numerical variables and response variable

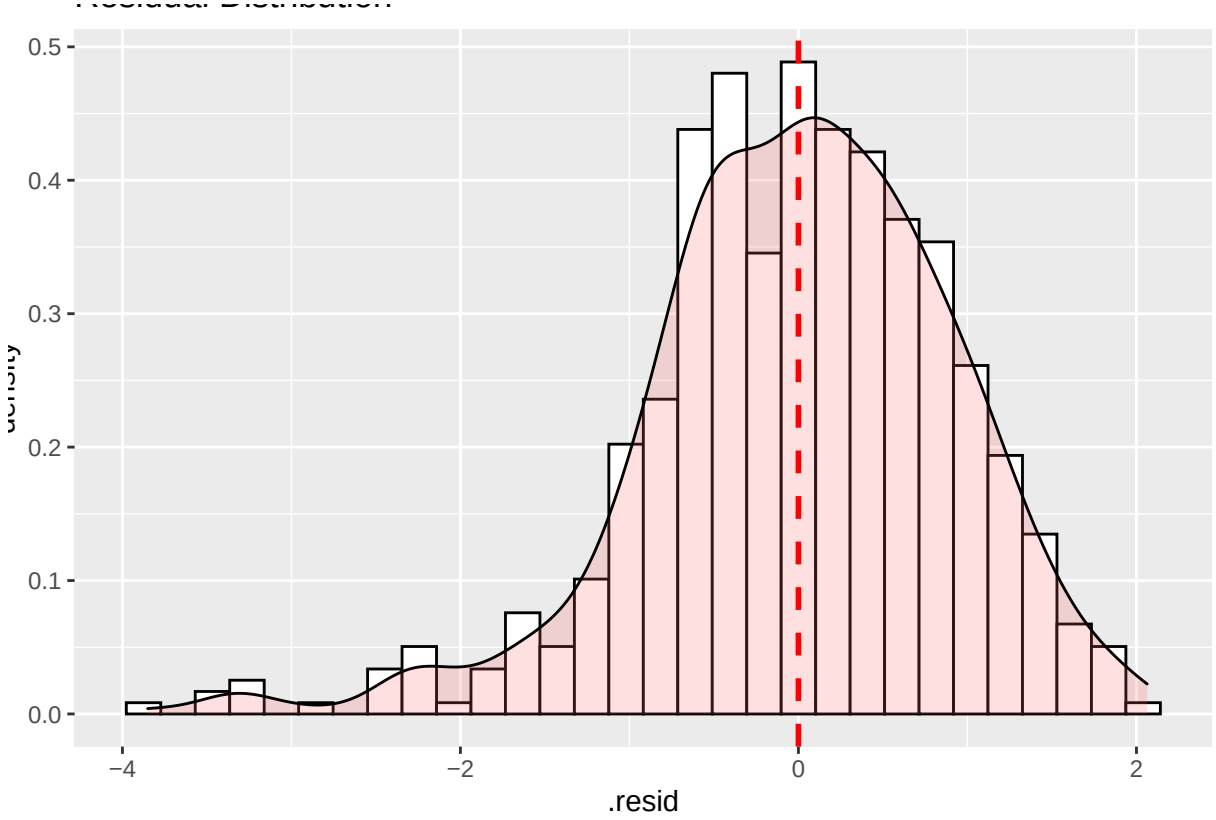
```
## integer(0)
```

There seems to be no relationships between the numerical variable, `runtime` and the response variable. There is a random scatter around 0 for every plot above with the exception of one leverage point (above 250). Hence there is a linear relationship between `runtime` and `imdb_rating`.

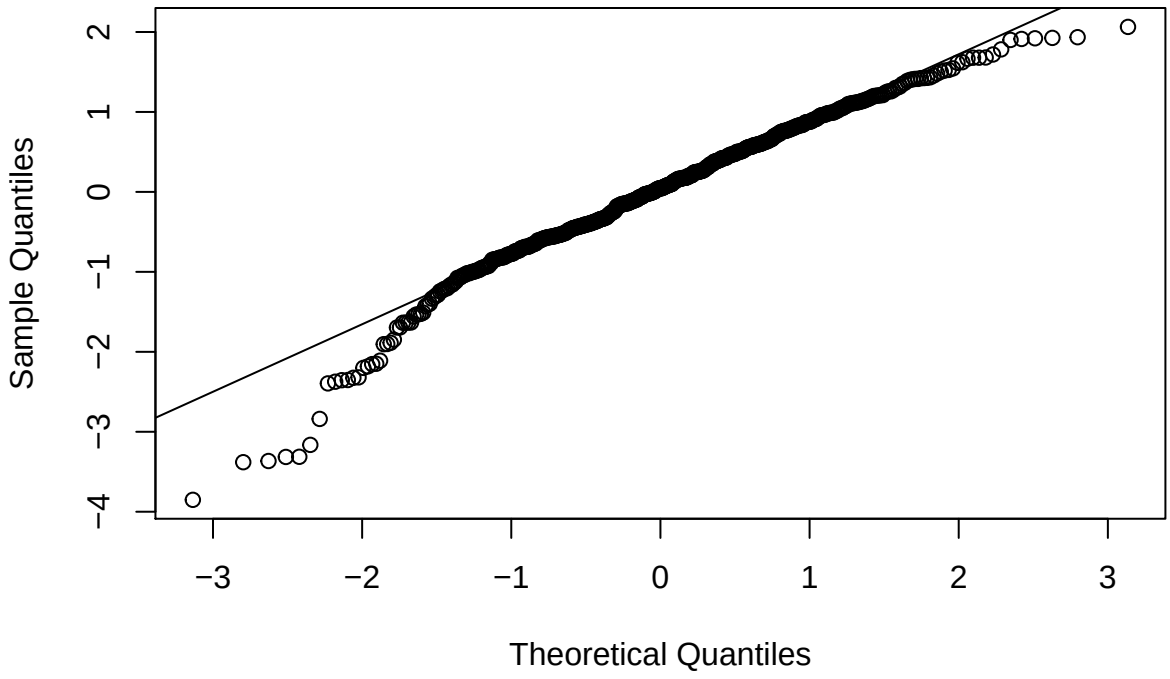
```
ggplot(lm8, aes(x=.resid)) +
  geom_histogram(aes(y=..density..), color="black", fill="white") +
  geom_density(alpha=0.2, fill="#FF6666") +
  geom_vline(aes(xintercept=mean(lm8$residuals)), col = 'red', lwd = 1, lty = 2) +
  ggtitle("Residual Distribution")
```

2. Nearly Normal Residuals

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



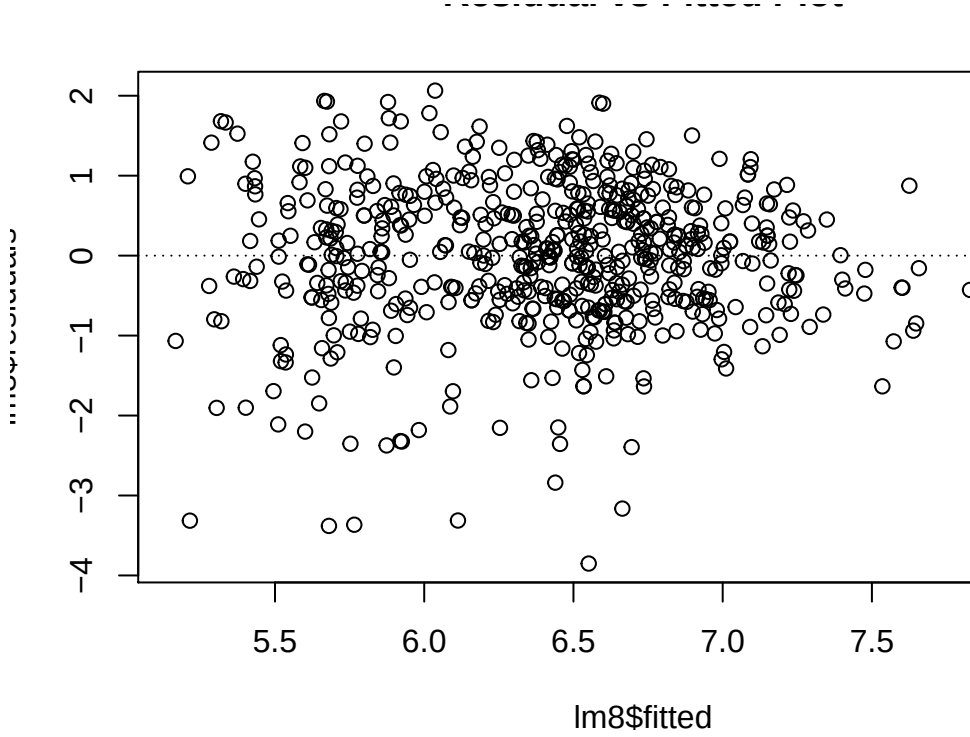

```
qqnorm(lm8$residuals, main = "Residuals Q-Q Plot")
qqline(lm8$residuals)
```



The residuals are reasonably symmetric, with only a slightly longer tail on the left. The q-q plot also shows

that a significant number of residuals lie on the standard normal line. Hence it would be appropriate to consider the normality condition to be met.

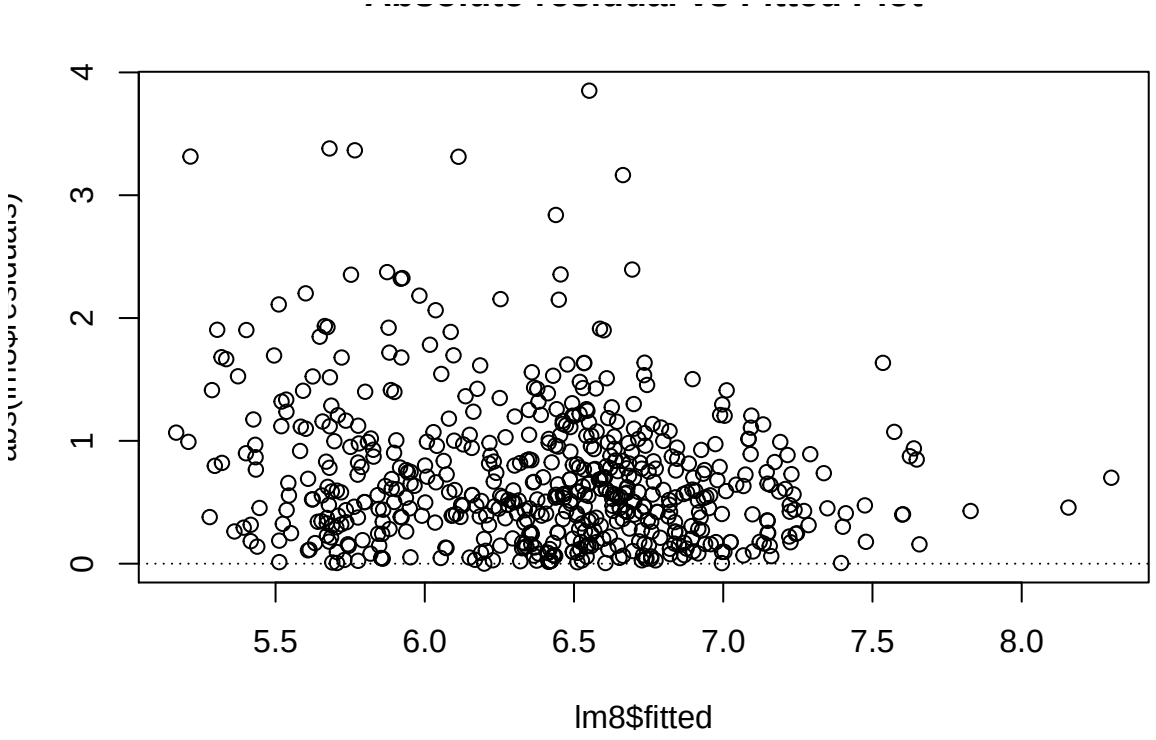
```
plot(lm8$residuals ~ lm8$fitted) +
  abline(0, 0, lty = 3) +
  title("Residual vs Fitted Plot")
```



3. Constant variability of residuals

```
## integer(0)
```

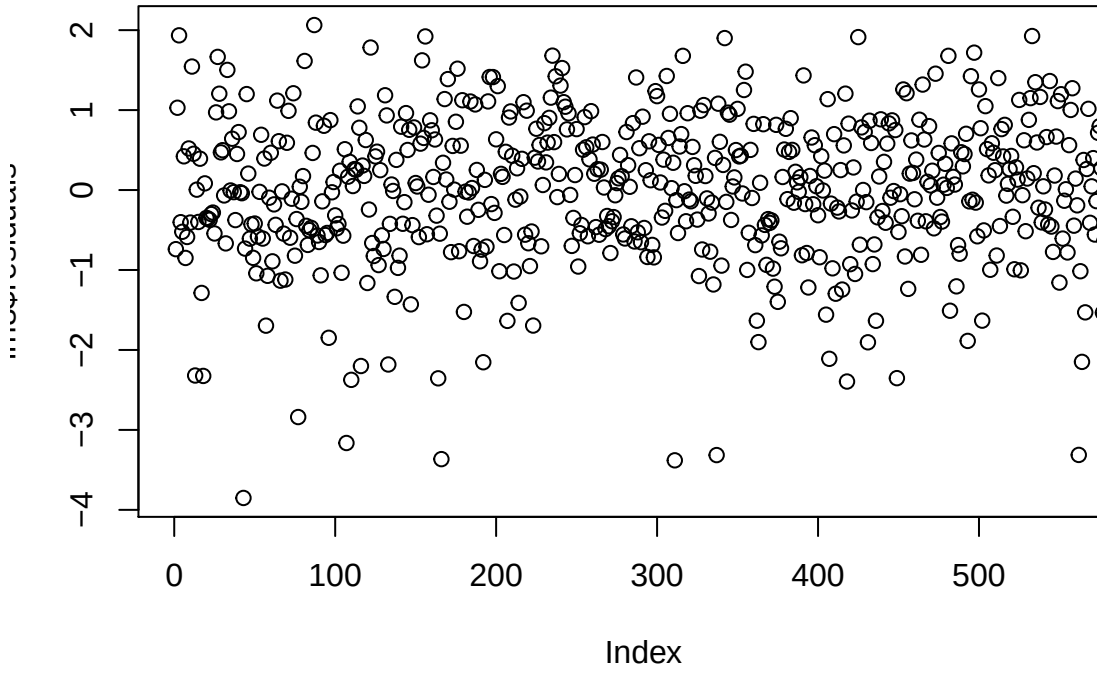
```
plot(abs(lm8$residuals) ~ lm8$fitted) +
  abline(0, 0, lty = 3) +
  title("Absolute residual vs Fitted Plot")
```




```
## integer(0)
```

Since most of the ratings are below 9, the scatter is random (above and below the zero residual line) around a score of 6. The absolute value of the residuals doesn't follow a triangle, and the scatter is spread above and below the mean. It's safe to say homoscedasticity is present, and the variability of the residuals is constant.

```
plot(lm8$residuals)
```



4. Independent residuals

There is no intrinsic ordering present in the residuals and no relation present in the graph above. Hence we can say that the residuals are independent of each other.

Interpretation of model coefficients

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	4.8956939	0.3701165	13.2274397	0.0000000
genreAnimation	-0.2591327	0.3664827	-0.7070804	0.4798086
genreArt House & International	0.7030540	0.2930034	2.3994736	0.0167421
genreComedy	-0.0566252	0.1559045	-0.3632046	0.7165881
genreDrama	0.6057924	0.1321564	4.5839043	0.0000056
genreHorror	-0.0978499	0.2314845	-0.4227060	0.6726709
genreMusical & Performing Arts	0.9565802	0.3477488	2.7507797	0.0061363
genreMystery & Suspense	0.3910193	0.1713863	2.2815084	0.0228902
genreOther	0.7097283	0.2667830	2.6603203	0.0080288
genreScience Fiction & Fantasy	0.0014893	0.3458982	0.0043055	0.9965662
runtime	0.0159409	0.0024317	6.5554032	0.0000000
mpaa_ratingNC-17	-0.6264554	0.7075522	-0.8853840	0.3763269
mpaa_ratingPG	-0.7616317	0.2793769	-2.7261800	0.0066064
mpaa_ratingPG-13	-1.0353416	0.2840024	-3.6455385	0.0002915
mpaa_ratingR	-0.7073747	0.2773989	-2.5500271	0.0110347
mpaa_ratingUnrated	-0.4060570	0.3715970	-1.0927350	0.2749767
dvd_rel_month	0.0243033	0.0114035	2.1312093	0.0335037
best_dir_winyes	0.3793524	0.1526058	2.4858324	0.0132134

genre All else held constant, the model predicts that:

- the "Animation" genre gets a rating which is 0.25 points lower than the reference level, on average.
- the "Art House & International" genre gets a rating which is 0.7 points higher than the reference level, on average.
- the "Comedy" genre gets a rating which is 0.05 points lower than the reference level, on average.
- the "Drama" genre gets a rating which is 0.61 points higher than the reference level, on average.
- the "Horror" genre gets a rating which is 0.09 points lower than the reference level, on average.
- the "Musical & Performing Arts" genre gets a rating which is 0.95 points higher than the reference level, on average.
- the "Mystery & Suspense" genre gets a rating which is 0.39 points higher than the reference level, on average.
- the "Other" genre gets a rating which is 0.71 points higher than the reference level, on average.
- the "Science Fiction & Fantasy" genre gets a rating which is 0.001 points higher than the reference level, on average.

Now that we know how each genre scores higher or lower than its reference level, an extension of this analysis could identify whether specific genres tend to score significantly more than other genres. In other words, we can test whether the genre is associated with an IMDb rating using a chi-square independence test.

runtime The model predicts, given that all else is held constant, for each 1-minute increase in runtime, the imdb_rating increases by 0.01 points.

mpaa_rating All else held constant, the model predicts that a movie with a:

- "NC-17" rating scores 0.63 points lower than the reference level, on average.
- "PG" rating scores 0.76 points lower than the reference level, on average.
- "R" rating scores 0.71 points lowest than the reference level, on average.

- "Unrated" rating scores 0.41 points lower than the reference level, on average.

Since the reference level, in this case, is a movie with a "G" rating, it can be said that G rated movies tend to score better on average compared to all other ratings. However, to verify if these results are significant, a chi-square independent test can be conducted along with a Bonferroni correction to check whether any specific rating tends to score statistically better than the other.

dvd_rel_month The model predicts that with each additional month in DVD release month, the rating increase by 0.02 on average, given all else is held constant.

Intercept or Reference level The intercept in this model represents a film of the "Action & Adventure" genre, with a runtime of 0 minutes, a mpaa_rating of "G," a dvd_rel_month of 0, and the director hasn't won an Oscar. A movie with these parameters would score a rating of 4.89. However, given a runtime of 0 minutes and a DVD release month of 0, the intercept is meaningless in this case and only serves to adjust the line's height. Since the reference level, in this case, is a movie with a "G" rating, it can be said that G rated movies tend to score better on average compared to all other ratings. However, to verify if these results are significant, a chi-square independent test can be conducted along with a Bonferroni correction to check whether any specific rating tends to score statistically better than the other.

Part 5: Prediction

In order to test our model, this project will use a movie from 2016 and verify whether the model predicts the correct imdb_rating. The movie we'll use is "Deadpool", a superhero film based on a Marvel Comics character. The movie was produced by a conglomerate of production studios, namely:

- 20th Century Fox
- Marvel Entertainment
- Kinberg Genre
- The Donners' Company
- TSG Entertainment

First, we'll create a dataframe for this movie. Since the movie consists of three genres, we'll be creating two dataframes, one for "Action & Adventure" and the other for "Comedy". The data is obtained from the IMDb page for this movie. [5]

```
deadpool <- data.frame("genre" = c("Action & Adventure", "Comedy"),
                       "runtime" = c(108,108),
                       "mpaa_rating" = c("R","R"),
                       "dvd_rel_month" = c(5, 5),
                       "best_dir_win" = c("no", "no"))
deadpool
```

```
##                genre runtime mpaa_rating dvd_rel_month best_dir_win
## 1 Action & Adventure     108           R             5           no
## 2             Comedy     108           R             5           no
```

Using the predict function in R:

```
deadpool_imdb_pred <- predict(lm8, newdata = deadpool, interval = "confidence")
deadpool_imdb_pred
```

```
##        fit      lwr      upr
## 1 6.031451 5.772127 6.290774
## 2 5.974825 5.741849 6.207802
```

[5] Deadpool (2016) - IMDb

The imdb_ratings in this case are 6.0 for both genres. In formal terms, we are 95% confident that the movie "Deadpool" scored an IMDb rating between 4 and 8. The actual imdb_rating for Deadpool is 8.0 as of August 2020. This means the model was close to the actual score. It should be noted that the margin of error in this case is also very large.

Part 6: Conclusion

The R^2 of the model is 0.253 and the R^2_{adj} is 0.231. This means that the model doesn't perform well while predicting `imdb_rating` values. Based on the p-values, the model as a whole is significant and individual predictors in the model are also significant. However, significant in this case only refers to the fact that at least one of the predictors are non-zero and not to the accuracy of the model's predictions.

Some of the reasons why the model didn't perform well can be listed as:

- Even though the predictor is significant according to its p-value, it doesn't necessarily mean that a longer movie is better. The model also predicts that every single minute only contributes to a minor increase in rating. Furthermore, in reality, a longer or shorter movie doesn't result in a popular film, it's how "engaging" or "engrossing" a film is, a highly subjective variable to measure. Moreover, it's difficult to determine the "engaging-ness" of a title prior to its release.
- Based on the coefficient for the DVD release year parameter, the DVD month doesn't significantly impact the ratings as the maximum month, in this case, would be 12, i.e., December. Additionally, it wouldn't counter-intuitive that movies with DVD releases in certain months tend to score better than those released in others.
- Although winning an Oscar is a prestigious milestone for any film, a movie that hasn't won an Oscar isn't necessarily poor. Furthermore, Oscars have been criticized as being financially-driven rather than talent-driven. [6]

References

[6] New York TImes: There's more to winning an oscar than meets the eye