Predicting Movie popularity

Setup

Load packages

```r
library(ggplot2)
library(dplyr)
library(statsr)
library(grid)
library(gridExtra)
library(tidyverse)
library(broom)
library(plyr)
library(BAS)
```

```
## Warning: package 'BAS' was built under R version 3.4.4
```

```r
library(pander)
```

Load data

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

```r
min(movies$thtr_rel_year)
```

```
## [1] 1970
```

```r
max(movies$thtr_rel_year)
```

```
## [1] 2014
```

Data

In the `movies` dataset, there is 651 **randomly sampled** movies which were released in United States movie theater in the period of 1970-2014. The data was obtained from Rotten Tomatoes and IMDB. The dataset contains 32 features of each movie, including genre, MPAA rating, production studio, and whether they recieved Oscar nominations.

Even though there is no detailed information about the exact sampling methods used, the movies included in this dataset were randomly sampled from the above two mentioned sources and no bias were created by the sampling method so we can assume that the results obtained can be **generalized to all U.S movies released between 1970 and 2014**. On the other hand, because this is an observational study, **the relationships that could be find from this data indicate association, but *not causation***

Data wrangling

We will first create some new variables that will be used later in the Bayesian modeling making use of the function `mutate` as follows:

- `feature_film`: with two levels:
  - 'yes': if `title_type` is 'Feature Film'
  - 'no': otherwise

```
movies <- movies%>%
        mutate(feature_film = factor(ifelse(title_type == 'Feature Film', 'yes', 'no')))
```

- `drama`: with two levels:
  - 'yes': if `genre` is 'Drama'
  - 'no': otherwise

```
movies <- movies%>%
        mutate(drama = factor(ifelse(genre == 'Drama', 'yes', 'no')))
```

- `mpaa_rating_R`: with two levels:
  - 'yes': if `mpaa_rating` is 'R'
  - 'no': otherwise

```
movies <- movies%>%
        mutate(mpaa_rating_R = factor(ifelse(mpaa_rating == 'R', 'yes', 'no')))
```

- `oscar_season`: with two levels:
  - 'yes': if `thtr_rel_month` is 'November' or 'October' or 'December'
  - 'no': otherwise

```
movies <- movies%>%
        mutate(oscar_season = factor(ifelse(thtr_rel_month == 10|
                                            thtr_rel_month == 11|
                                            thtr_rel_month == 12,
                                                                 'yes', 'no')))
```

- `summer_season`: with two levels:
  - 'yes': if `thtr_rel_month` is 'May' or 'June' or 'July' or 'August'
  - 'no': otherwise

```
movies <- movies%>%
        mutate(summer_season = factor(ifelse(thtr_rel_month == 5|
                                            thtr_rel_month == 6|
                                            thtr_rel_month == 7|
                                            thtr_rel_month == 8,
                                                                 'yes', 'no')))
```

The aim of this project is to assess whether:

> Can the popularity of a movie be predicted by considering certain characteristic information about it such as type, genre, MPAA rating, number of IMDb votes, and whether it has won an award?

- **Why predicting popularity of a movie?**
  Movie popularity can help people to decide which movie to watch, or whether they want to go the cinema to watch it or wait till the DVD is release and watch it at home. Consequently, it could also help theater owner to choose which movies to show or how many times to show it or for how long.

FIRST PART: LINEAR REGRESSION

For this first part of the project, we will considered the following variables:

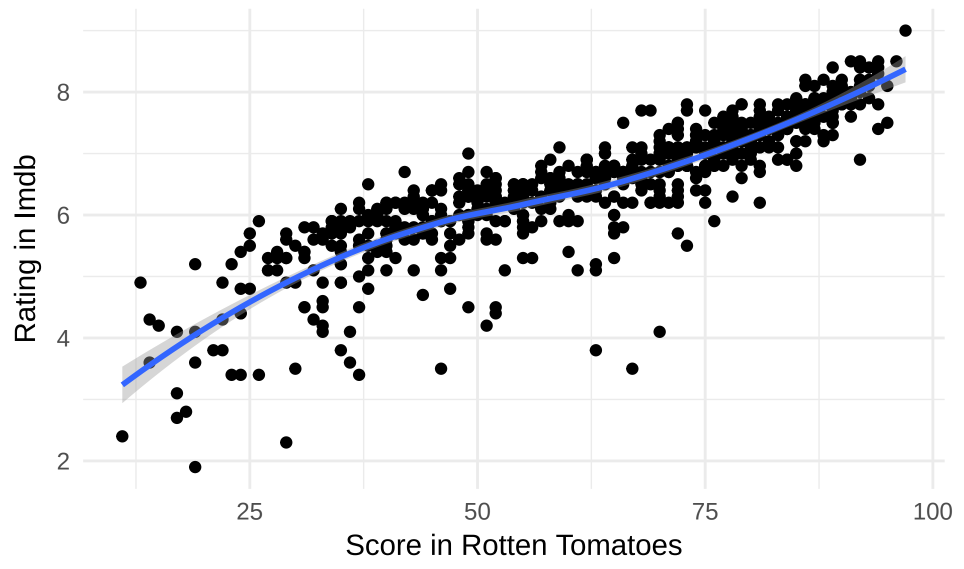
- **Variables to consider:**
  - We will analyze movie popularity by considering the variables:
    - `audience_score`(Audience score on Rotten Tomatoes)
    - `imdb_rating`(Rating on IMDB)
  - For the characteristics of the movie we will considered the following variables:
    - `title_type` (Type of movie)
    - `genre` (Genre of movie)
    - `mpaa_rating` (MPAA rating of the movie)
    - `imdb_num_votes`(Number of votes on IMDB)
    - `best_pic_win` (Whether or not the movie won a best picture Oscar)
    - `best_actor_win` (Whether or not one of the main actors in the movie ever won an Oscar)
    - `best_actress_win` (Whether or not one of the main actress in the movie ever won an Oscar)
    - `best_dir_win` (Whether or not one of the director in the movie ever won an Oscar)

Exploratory data analysis

First of all, we will check whether `audience_score` and `imdb_rating` show a correlation between them. For doing this, we will plot both variables in a scatter plot:

```r
ggplot(movies, aes(x=audience_score, y=imdb_rating))+
  theme_minimal()+
  geom_point()+
  geom_smooth()+
  labs(x = "Score in Rotten Tomatoes", y = "Rating in Imdb")
```

```
## `geom_smooth()` using method = 'loess'
```



We can see that the plot shows a possible positive correlation between the two variables. We will confirm this by using the function `cor` to numerically calculate this correlation:

```
cor(movies$audience_score, movies$imdb_rating)
```

```
## [1] 0.8648652
```

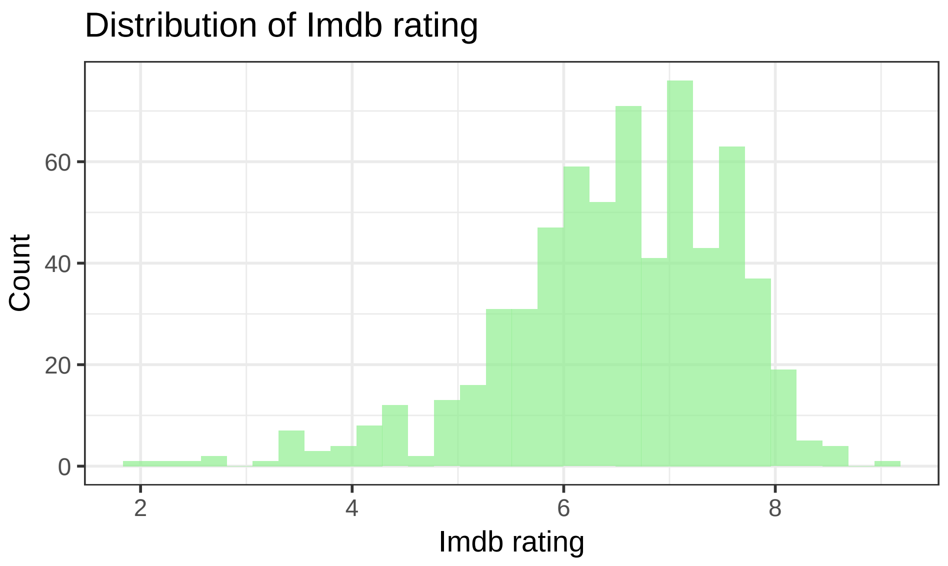
As it can be observed from the plot and the correlation coefficient, there is a high correlation between both variables. So in this case, we can use for the model only one of them.

– **Distribution of `imbd_rating`:**

We will check how our response variable is distributed by making use of `histogram` and summary statistics:

```
ggplot(movies, aes(x=imdb_rating)) +
  geom_histogram(fill="lightgreen", alpha = 0.7)+
  theme_bw()+
  labs(x = "Imdb rating", y= "Count", title = "Distribution of Imdb rating")
```

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



```r
summary(movies$imdb_rating)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.900   5.900   6.600   6.493   7.300   9.000
```

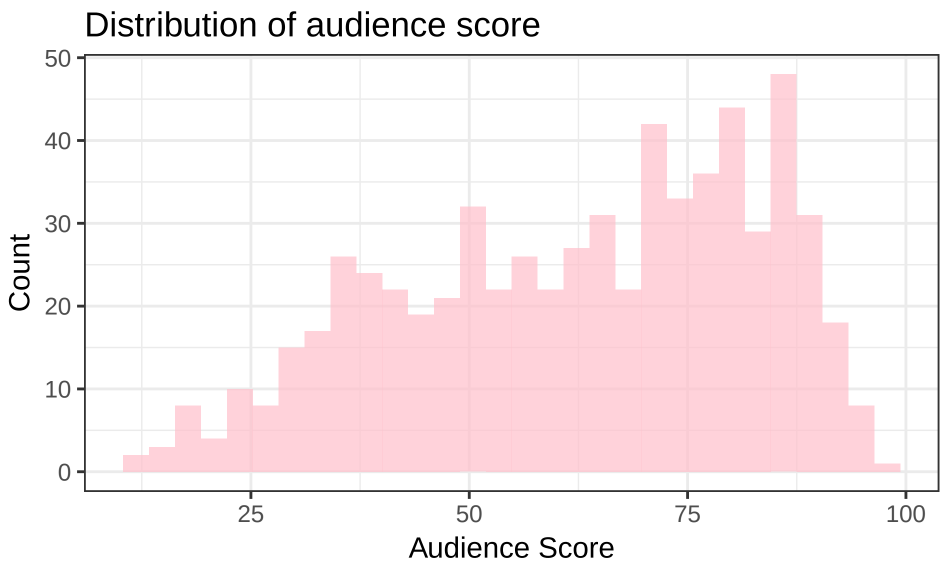
The distribution of **imbd_rating** variable shows a distribution close to normal with a slightly left skew with a mean of 6.493 and a median of 6.00.

– **Distribution of `audience_score`:**

We will check how the `audience_score` variable is distributed by making again use of `histogram` and summary statistics:

```r
ggplot(movies, aes(x=audience_score)) +
  geom_histogram(fill="pink", alpha = 0.7)+
  theme_bw()+
  labs(x = "Audience Score", y= "Count", title = "Distribution of audience score")
```

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



```r
summary(movies$audience_score)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   11.00   46.00   65.00   62.36   80.00   97.00
```

The distribution of **audience_score** variable shows a more uniform distribution with a mean of 62.36 and a median of 65.00.

Because of its distribution, we will choose to considered only `imdb_rating` as the response variable.

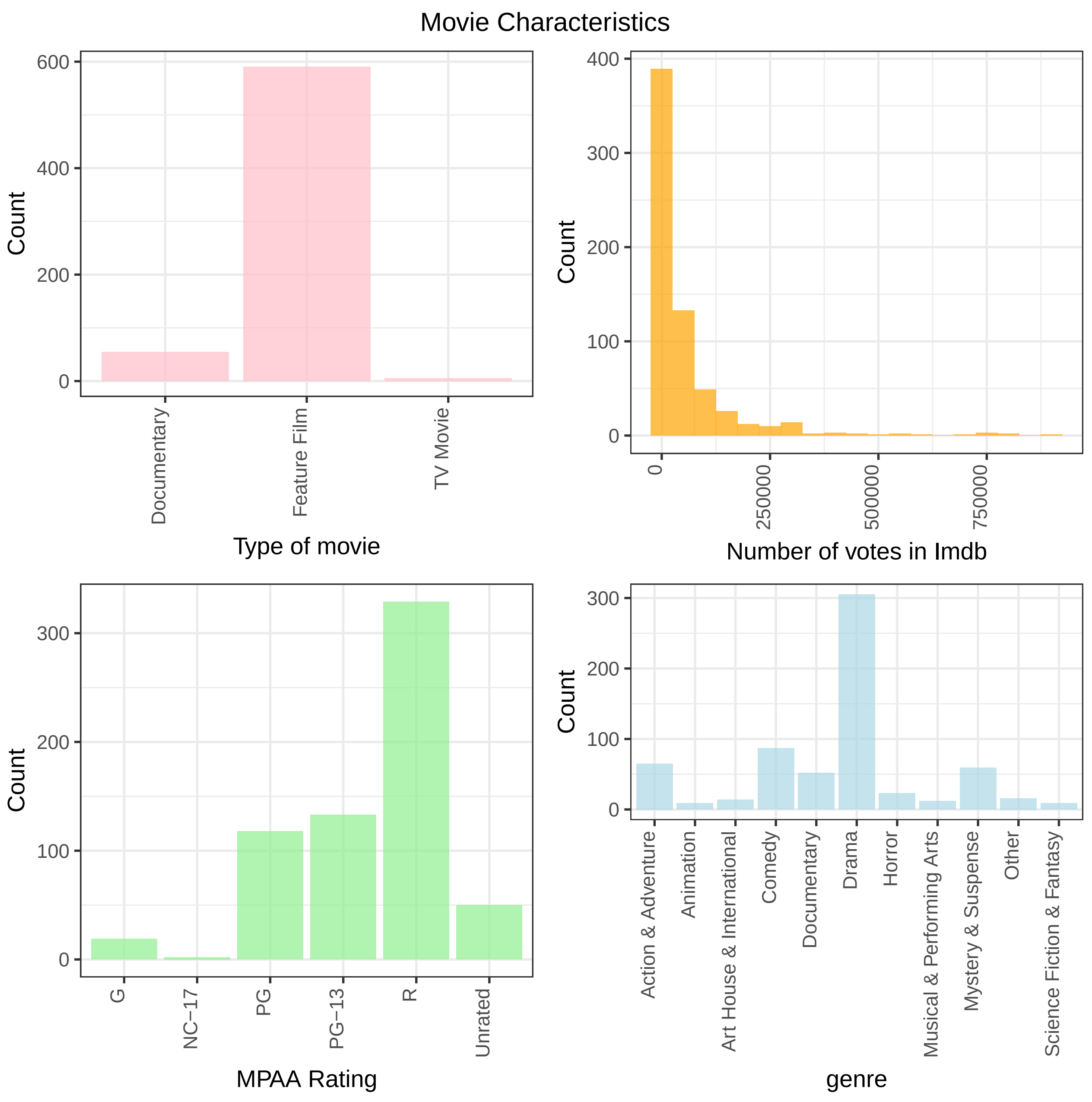
We will subset the dataset to keep only those variables that we are interested in:

```r
movies_interesting <- movies %>%
  select(title_type, genre, mpaa_rating, imdb_num_votes, best_pic_win, best_actor_win,
                                                 best_actress_win, best_dir_win, imdb_rating)
```

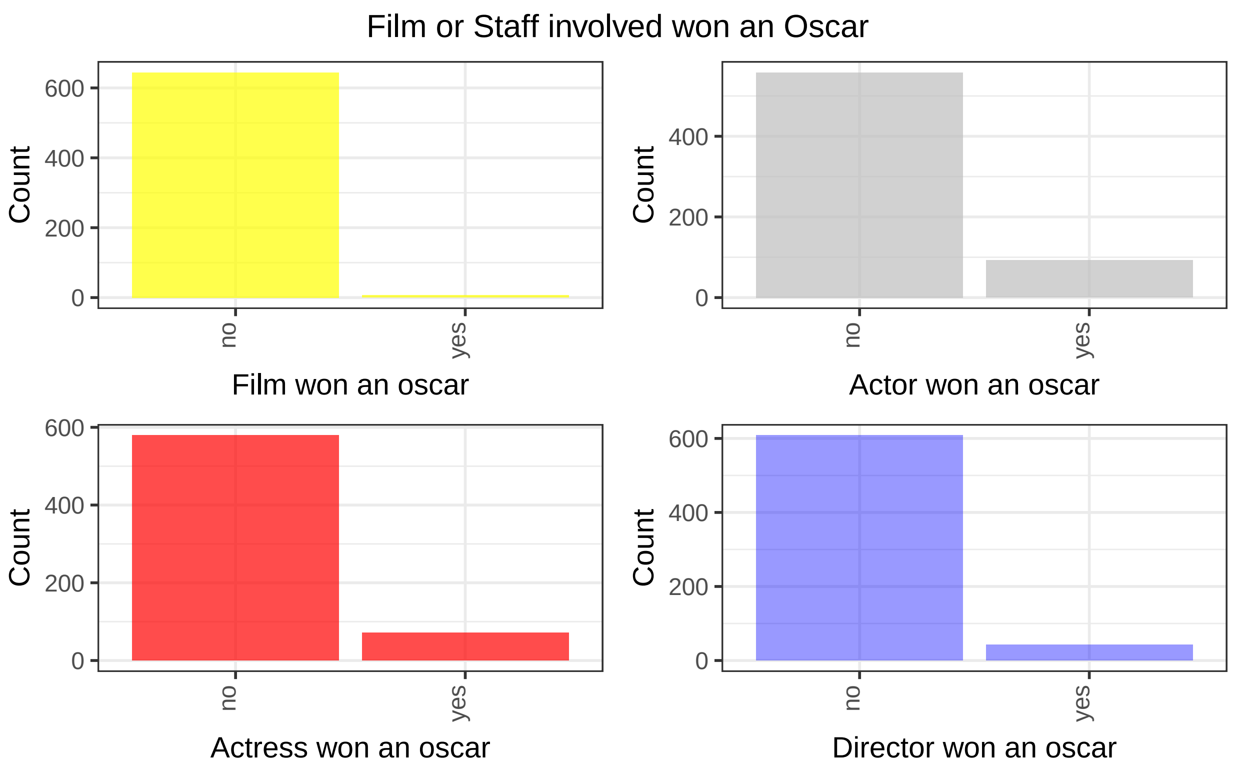
– **Distribution of variables under consideration:**

We will now considered how the variables that we are interested in including in our model are distributed. For this, we will plot a histogram for each of the variables.

```r
g1<- ggplot(movies_interesting, aes(x=genre)) +
  geom_bar(fill="lightblue", alpha = 0.7)+
  theme_bw()+
  labs(x = "genre", y= "Count")+
  theme(axis.text.x=element_text(angle=90, hjust = 1, vjust = 0))

g2 <- ggplot(movies_interesting, aes(x=title_type)) +
  geom_bar(fill="pink", alpha = 0.7)+
  theme_bw()+
  labs(x = "Type of movie", y= "Count")+
  theme(axis.text.x=element_text(angle=90, hjust = 1, vjust = 0))

g3 <- ggplot(movies_interesting, aes(x=mpaa_rating)) +
  geom_bar(fill="lightgreen", alpha = 0.7)+
  theme_bw()+
  labs(x = "MPAA Rating", y= "Count")+
  theme(axis.text.x=element_text(angle=90, hjust = 1, vjust = 0))

g4 <- ggplot(movies_interesting, aes(x=imdb_num_votes)) +
  geom_histogram(binwidth =50000, fill="orange", alpha = 0.7)+
  theme_bw()+
  labs(x = "Number of votes in Imdb", y= "Count")+
  theme(axis.text.x=element_text(angle=90, hjust = 1, vjust = 0))

g5 <- ggplot(movies_interesting, aes(x=best_pic_win)) +
  geom_bar(fill="yellow", alpha = 0.7)+
  theme_bw()+
  labs(x = "Film won an oscar", y= "Count")+
  theme(axis.text.x=element_text(angle=90, hjust = 1, vjust = 0))

g6 <- ggplot(movies_interesting, aes(x=best_actor_win)) +
  geom_bar(fill="grey", alpha = 0.7)+
  theme_bw()+
  labs(x = "Actor won an oscar", y= "Count")+
  theme(axis.text.x=element_text(angle=90, hjust = 1, vjust = 0))

g7 <- ggplot(movies_interesting, aes(x=best_actress_win)) +
  geom_bar(fill="red", alpha = 0.7)+
  theme_bw()+
  labs(x = "Actress won an oscar", y= "Count")+
  theme(axis.text.x=element_text(angle=90, hjust = 1, vjust = 0))

g8 <- ggplot(movies_interesting, aes(x=best_dir_win)) +
  geom_bar(fill="blue", alpha = 0.4)+
  theme_bw()+
  labs(x = "Director won an oscar", y= "Count")+
  theme(axis.text.x=element_text(angle=90, hjust = 1, vjust = 0))


grid.arrange(g2, g4, g3, g1, nrow=2, top = "Movie Characteristics")
```



```r
grid.arrange(g5, g6, g7, g8, nrow =2, top = "Film or Staff involved won an Oscar")
```



We need to obtained summary descriptive tables. For those variables that are categorical, we can use a proportion table in order to summarise them. For this task, the `table` build-in function can be used. On the other hand, we will create a data frame with the two continous variables and apply `summary` to obtained the descriptive statistics:

```
summary(movies_interesting$imdb_num_votes)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##     180    4546   15116   57533   58300  893008
```

```
table(movies_interesting$mpaa_rating)
```

```
## 
##       G   NC-17      PG   PG-13       R Unrated 
##      19       2     118     133     329      50
```

```
table(movies_interesting$title_type)
```

```
## 
##  Documentary Feature Film     TV Movie 
##           55          591            5
```

```
table(movies_interesting$genre)
```

```
## 
##       Action & Adventure                 Animation 
##                       65                         9 
## Art House & International                    Comedy 
##                       14                        87 
##              Documentary                     Drama 
##                       52                       305 
##                   Horror Musical & Performing Arts
##                         23                          12
##         Mystery & Suspense                       Other
##                         59                          16
## Science Fiction & Fantasy
##                          9
```

```
table(movies_interesting$best_pic_win)
```

```
##
##  no yes
## 644   7
```

```
table(movies_interesting$best_actress_win)
```

```
##
##  no yes
## 579  72
```

```
table(movies_interesting$best_actor_win)
```

```
##
##  no yes
## 558  93
```

```
table(movies_interesting$best_dir_win)
```

```
##
##  no yes
## 608  43
```

There are 591 movies in the dataset that are type “Feature Film”. We can observed that there are also 60 movies that belong to the category “Documentary” and “TV movies”. Regarding the MPAA rating, we found 52 movies with MPAA ratings of NC-17 or unrated, 118 of PG, 133 of PG-13, 19 of G, and most of the movies, particularly 329, are rated as R. It’s interesting to see that 305 films are clasified as Drama.
Because “Documentary” and “TV movies” are not likely to be displayed at Cinema theaters, we would exclude those type of movies and only include in the analysis “feature film”

```
movies_interesting <- movies_interesting %>%
            filter(title_type == "Feature Film")
```

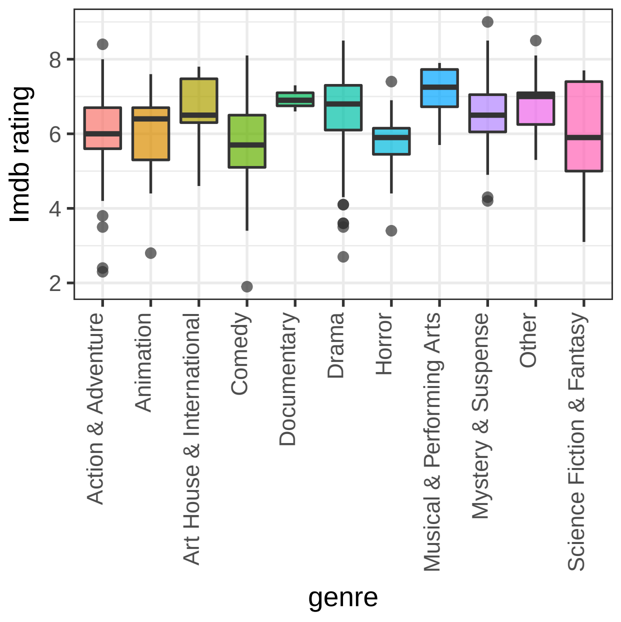
– Interaction between the variables under consideration:

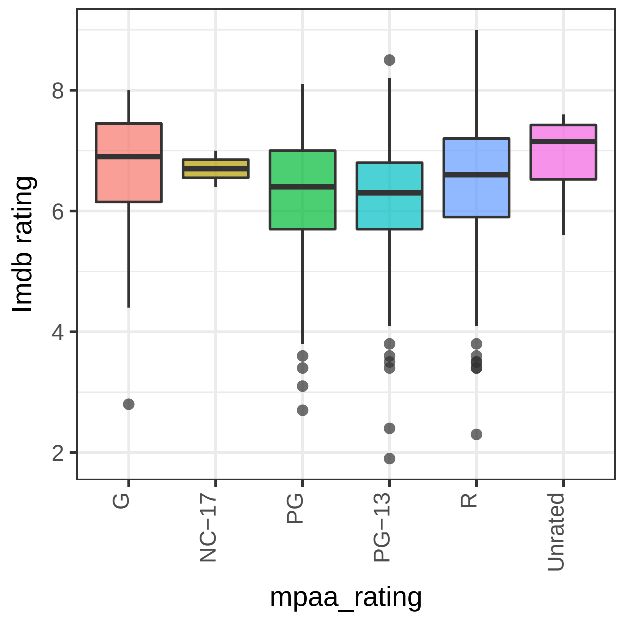
Now, we can analyze the interaction between our exploratory variables and the response variable. For this task, we will plot boxplot or scatter plots according to whether the exploratory variable is numerical o categorical.

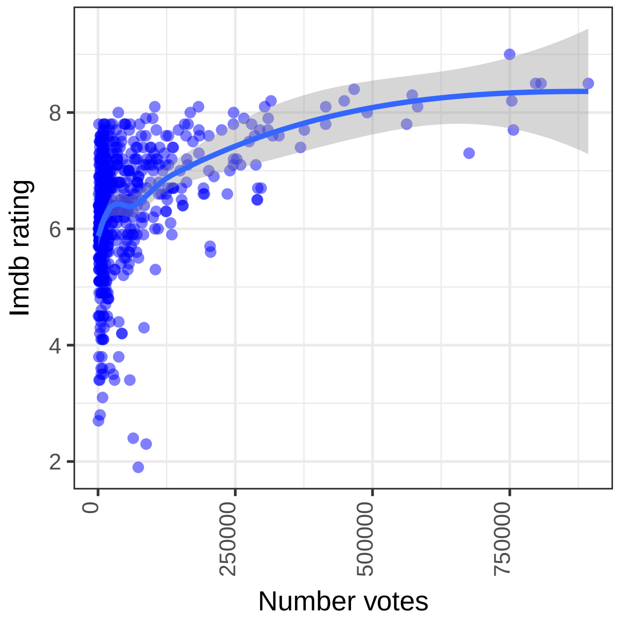
```
p1 <- ggplot(movies_interesting, aes(x=genre, y = imdb_rating, fill=genre))+
  geom_boxplot(alpha = 0.7)+
  theme_bw()+
  labs(x = "genre", y= "Imdb rating")+
  theme(axis.text.x=element_text(angle=90, hjust = 1, vjust = 0))+
  theme(legend.position="none")

p2 <- ggplot(movies_interesting, aes(x=mpaa_rating, y = imdb_rating, fill=mpaa_rating))+
  geom_boxplot(alpha = 0.7)+
  theme_bw()+
  labs(x = "mpaa_rating", y= "Imdb rating")+
  theme(axis.text.x=element_text(angle=90, hjust = 1, vjust = 0))+
```

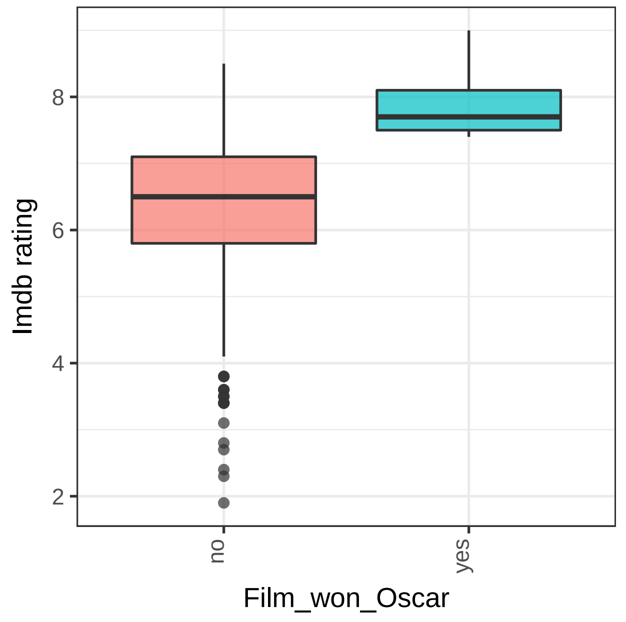
```r
  theme(legend.position="none")

p3 <- ggplot(movies_interesting, aes(x=imdb_num_votes, y = imdb_rating))+
  geom_point(colour = "blue", alpha = 0.5)+
  theme_bw()+
  geom_smooth()+
  labs(x = "Number votes", y= "Imdb rating", fill = "won_oscar")+
  theme(axis.text.x=element_text(angle=90, hjust = 1, vjust = 0))+
  theme(legend.position="none")

p4 <- ggplot(movies_interesting, aes(x=best_pic_win, y = imdb_rating, fill = best_pic_win))+
  geom_boxplot(alpha = 0.7)+
  theme_bw()+
  labs(x = "Film_won_Oscar", y= "Imdb rating", fill = "best_pic_win")+
  theme(axis.text.x=element_text(angle=90, hjust = 1, vjust = 0))+
  theme(legend.position="none")

p5 <- ggplot(movies_interesting, aes(x=best_actress_win, y = imdb_rating, fill = best_actress_win))+
  geom_boxplot(alpha = 0.7)+
  theme_bw()+
  labs(x = "Actress_won_Oscar", y= "Imdb rating", fill = "best_actress_win")+
  theme(axis.text.x=element_text(angle=90, hjust = 1, vjust = 0))+
  theme(legend.position="none")

p6 <- ggplot(movies_interesting, aes(x=best_actor_win, y = imdb_rating, fill = best_actor_win))+
  geom_boxplot(alpha = 0.7)+
  theme_bw()+
  labs(x = "Actor_won_Oscar", y= "Imdb rating", fill = "best_actor_win")+
  theme(axis.text.x=element_text(angle=90, hjust = 1, vjust = 0))+
  theme(legend.position="none")

p7 <- ggplot(movies_interesting, aes(x=best_dir_win, y = imdb_rating, fill = best_dir_win))+
  geom_boxplot(alpha = 0.7)+
  theme_bw()+
  labs(x = "Director_won_Oscar", y= "Imdb rating", fill = "best_dir_win")+
  theme(axis.text.x=element_text(angle=90, hjust = 1, vjust = 0))+
  theme(legend.position="none")


grid.arrange(p1, p2, p3, p4, nrow = 2)
```

```
## `geom_smooth()` using method = 'loess'
```

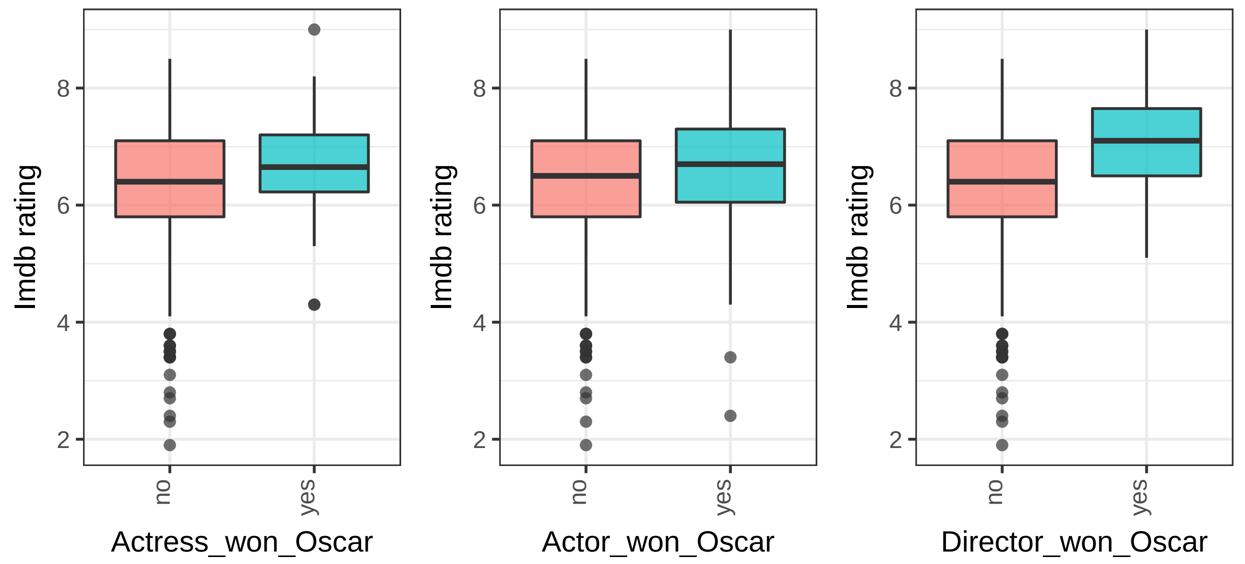








```
grid.arrange(p5, p6, p7, nrow =1)
```



```r
ddply(movies_interesting,~best_dir_win,summarise,mean=mean(imdb_rating),sd=sd(imdb_rating))
```

```
##   best_dir_win     mean        sd
## 1           no 6.336131 1.0549306
## 2          yes 7.041860 0.8272826
```

```r
ddply(movies_interesting,~best_pic_win,summarise,mean=mean(imdb_rating),sd=sd(imdb_rating))
```

```
##   best_pic_win     mean        sd
## 1           no 6.369349 1.0471077
## 2          yes 7.900000 0.5656854
```

From the plots and the summary descriptive, we learn that MPAA rating and genre don't appear to have a clear association with the rating given in IMDB. However, those movies that won an oscar or the director ever won an oscar appear to have a sightly higher rating. Moreover, the number of votes given show a weak positive association with the IMDB rating.

We can see also that the distribution of `imdb_rating` looks rigth skew. In order to adjust this, we can apply a log-transformation to the values:

```r
movies_interesting <- movies_interesting %>% mutate(log_votes = log(imdb_num_votes))
```

Last, the variables `best_actor_win` and `best_actress_win` appear to have the same distribution and a similar association with `imdb_rating`, so we will combine these two variables in a new one called `main_oscar_win`.

```r
movies_interesting <- movies_interesting%>%
                        mutate(main_oscar_win = ifelse(best_actor_win == 'yes' | best_actress_win == 'y
movies_interesting <- movies_interesting%>%
                        select(-c(best_actor_win, best_actress_win))
```

Modeling

We will choose the **backwards elimination** method in order to reach our **parsimonious model** (the simpler model with great explanatory predictive power).

Baseline model

Backwards elimination implies starting with a model comprising all candidates. In our case, our first full model includes all six variables. We will use `lm` for this task and include the variables `genre`, `best_pic_win`, `best_dir_win`, `main_oscar_win`, `log_votes` and `mpaa_rating`

```
fullmodel <- lm(imdb_rating ~ genre+best_pic_win+best_dir_win+main_oscar_win+log_votes+mpaa_rating,
         data = movies_interesting)
summary(fullmodel)
```

```
##
## Call:
## lm(formula = imdb_rating ~ genre + best_pic_win + best_dir_win +
##     main_oscar_win + log_votes + mpaa_rating, data = movies_interesting)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.9785 -0.3924  0.0662  0.5319  1.9703
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                      3.48124    0.35459   9.818  < 2e-16 ***
## genreAnimation                  -0.51095    0.33599  -1.521 0.128881
## genreArt House & International  1.07179    0.26360   4.066 5.45e-05 ***
## genreComedy                      0.05140    0.14230   0.361 0.718090
## genreDocumentary                 1.17262    0.50782   2.309 0.021293 *
## genreDrama                       0.92888    0.12229   7.596 1.26e-13 ***
## genreHorror                      0.01285    0.21195   0.061 0.951675
## genreMusical & Performing Arts  1.35460    0.31867   4.251 2.49e-05 ***
## genreMystery & Suspense          0.58835    0.15825   3.718 0.000221 ***
## genreOther                       0.96670    0.24490   3.947 8.89e-05 ***
## genreScience Fiction & Fantasy -0.16346    0.30147  -0.542 0.587873
## best_pic_winyes                  0.33918    0.34652   0.979 0.328093
## best_dir_winyes                  0.28496    0.14384   1.981 0.048062 *
## main_oscar_winyes               -0.01077    0.08581  -0.125 0.900199
## log_votes                        0.30432    0.02419  12.580  < 2e-16 ***
## mpaa_ratingNC-17                -0.28771    0.65056  -0.442 0.658474
## mpaa_ratingPG                   -0.66992    0.25632  -2.614 0.009194 **
## mpaa_ratingPG-13                -1.06283    0.26018  -4.085 5.04e-05 ***
## mpaa_ratingR                    -0.69771    0.25466  -2.740 0.006340 **
## mpaa_ratingUnrated              -0.21135    0.33682  -0.627 0.530590
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8455 on 571 degrees of freedom
## Multiple R-squared:  0.379,  Adjusted R-squared:  0.3583
## F-statistic: 18.34 on 19 and 571 DF,  p-value: < 2.2e-16
```

Model Selection

After running the full model with all the variables involved, we have obtained an R^2_{adj} of 0.3582, which means that we can still improve the model. In order to do so, we can use the one-by-one remove method and start by removing the variable which has the highest p-value each time, until all the variables remaining in the model are significant. P-value was chosen as elimination criteria due to the fact that in this case, the aim was to create a model that shows the highest predictive value using only variables with sigificance.

So the variable that has the highest p-value in our model is `main_oscar_win`.

```
step1_model <- lm(imdb_rating ~ genre+best_pic_win+best_dir_win+log_votes+mpaa_rating,
          data = movies_interesting)
summary(step1_model)
```

```
##
## Call:
## lm(formula = imdb_rating ~ genre + best_pic_win + best_dir_win +
##     log_votes + mpaa_rating, data = movies_interesting)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.9761 -0.3907  0.0648  0.5328  1.9707
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     3.48439    0.35339   9.860  < 2e-16 ***
## genreAnimation                 -0.51263    0.33544  -1.528 0.127005
## genreArt House & International  1.07082    0.26326   4.068 5.42e-05 ***
## genreComedy                     0.05060    0.14204   0.356 0.721785
## genreDocumentary                1.17242    0.50738   2.311 0.021202 *
## genreDrama                      0.92646    0.12066   7.678 7.01e-14 ***
## genreHorror                     0.01315    0.21176   0.062 0.950522
## genreMusical & Performing Arts  1.35405    0.31837   4.253 2.46e-05 ***
## genreMystery & Suspense         0.58537    0.15632   3.745 0.000199 ***
## genreOther                      0.96410    0.24380   3.954 8.63e-05 ***
## genreScience Fiction & Fantasy -0.16255    0.30112  -0.540 0.589535
## best_pic_winyes                 0.33613    0.34538   0.973 0.330845
## best_dir_winyes                 0.28415    0.14357   1.979 0.048280 *
## log_votes                       0.30398    0.02402  12.654  < 2e-16 ***
## mpaa_ratingNC-17               -0.29099    0.64947  -0.448 0.654298
## mpaa_ratingPG                  -0.67111    0.25592  -2.622 0.008966 **
## mpaa_ratingPG-13               -1.06389    0.25982  -4.095 4.84e-05 ***
## mpaa_ratingR                   -0.69793    0.25444  -2.743 0.006278 **
## mpaa_ratingUnrated             -0.21049    0.33646  -0.626 0.531825
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8448 on 572 degrees of freedom
## Multiple R-squared:  0.3789, Adjusted R-squared:  0.3594
## F-statistic: 19.39 on 18 and 572 DF,  p-value: < 2.2e-16
```

After running again our simpler model, we can see that now our R^2_{adj} is 0.3594. We can try to improve our model more by eliminating again the variable with the highest p-value. In this case, it will be `best_pic_win`.

```
step2_model <- lm(imdb_rating ~ genre+best_dir_win+log_votes+mpaa_rating,
          data = movies_interesting)
summary(step2_model)
```

```
##
## Call:
## lm(formula = imdb_rating ~ genre + best_dir_win + log_votes +
##     mpaa_rating, data = movies_interesting)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.9806 -0.3852  0.0650  0.5319  1.9762
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    3.44528    0.35108   9.813  < 2e-16 ***
## genreAnimation                -0.51184    0.33542  -1.526 0.127571
## genreArt House & International  1.08083    0.26304   4.109 4.55e-05 ***
## genreComedy                    0.05855    0.14180   0.413 0.679816
## genreDocumentary               1.18050    0.50729   2.327 0.020309 *
## genreDrama                     0.93449    0.12037   7.763 3.82e-14 ***
## genreHorror                    0.01806    0.21169   0.085 0.932038
## genreMusical & Performing Arts  1.35430    0.31836   4.254 2.45e-05 ***
## genreMystery & Suspense        0.59136    0.15619   3.786 0.000169 ***
## genreOther                     0.96493    0.24379   3.958 8.50e-05 ***
## genreScience Fiction & Fantasy -0.16337    0.30110  -0.543 0.587630
## best_dir_winyes                0.32650    0.13681   2.386 0.017334 *
## log_votes                      0.30764    0.02373  12.966  < 2e-16 ***
## mpaa_ratingNC-17              -0.29094    0.64944  -0.448 0.654334
## mpaa_ratingPG                 -0.66990    0.25591  -2.618 0.009085 **
## mpaa_ratingPG-13              -1.06911    0.25976  -4.116 4.42e-05 ***
## mpaa_ratingR                  -0.70094    0.25440  -2.755 0.006052 **
## mpaa_ratingUnrated            -0.21287    0.33644  -0.633 0.527165
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8448 on 573 degrees of freedom
## Multiple R-squared:  0.3779, Adjusted R-squared:  0.3595
## F-statistic: 20.48 on 17 and 573 DF,  p-value: < 2.2e-16
```

We now see that the R^2_{adj} is 0.3595, not different from our previous model in step1, but with the difference that this time all variables involved are significant. I will not show it here for practical sake, but removal of any of other variables will decrease R^2_{adj}. So we considered this our final model.

Model Diagnostics

The multiple regression model depends on the following four assumptions:

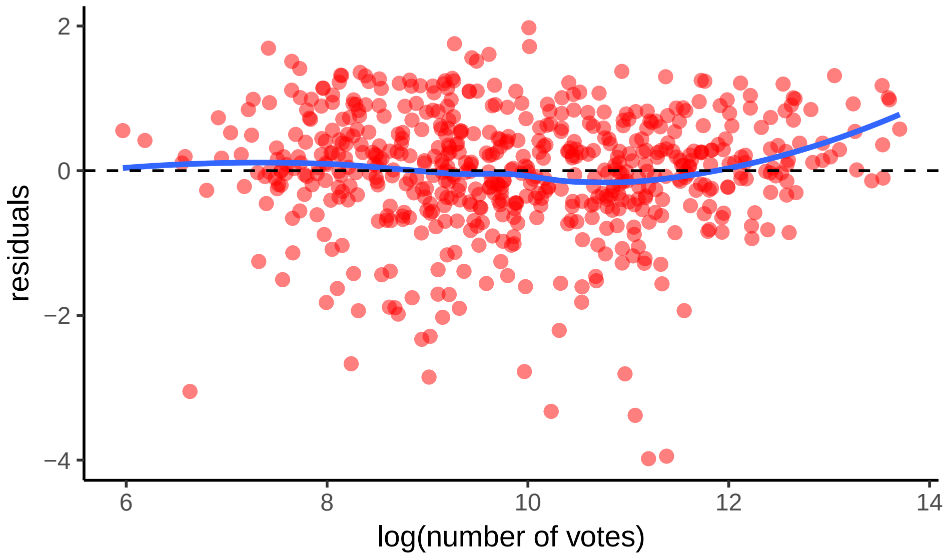
1) Each numerical explanatory variable is linearly related to the response variable
2) Residuals are distributed nearly normal with a mean of 0
3) Variability of residuals is nearly constant
4) The residuals are independant

We will test one-by-one the assumptions in the context of our model:

1) The only numerical variable that we have in our model is `log_values`. So we can explore the first assumption by checking the residual plots (e vs. X).

```r
ggplot(augment(step2_model), aes(x = movies_interesting$log_votes, y = .resid))+
  geom_point(colour = "red", size = 2, alpha = 0.5)+
  theme_classic()+
  geom_smooth(se=FALSE)+
  labs(x = "log(number of votes)", y= "residuals")+
  geom_hline(yintercept = 0, linetype = "dashed")
```

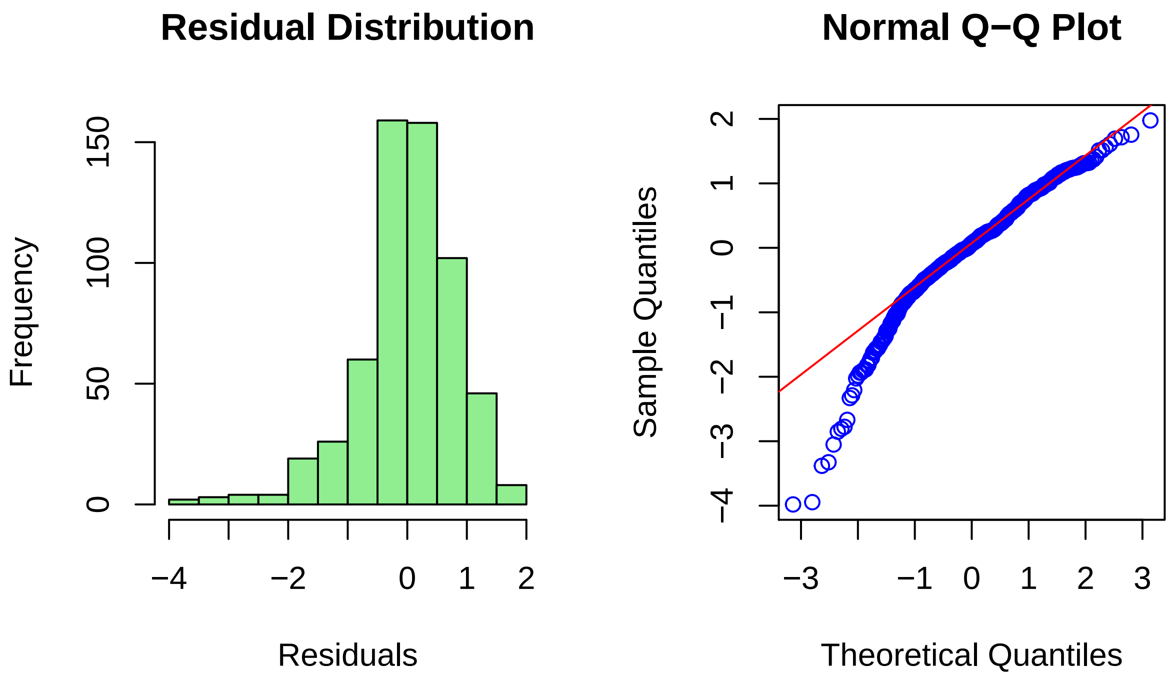
```
## `geom_smooth()` using method = 'loess'
```



The plot shows that the residuals are random scatter around 0, which indicates a linear relationship between the numerical exploratory variable and the response variable.

2) To check this condition, we will perform first the histogram of the residuals and then a residuals Q-Q plot.

```r
par(mfrow=c(1,2))
hist(step2_model$residuals, main = "Residual Distribution",
     xlab = "Residuals", col = "lightgreen")
qqnorm(step2_model$residuals, col = "blue")
qqline(step2_model$residuals, col = "red")
```

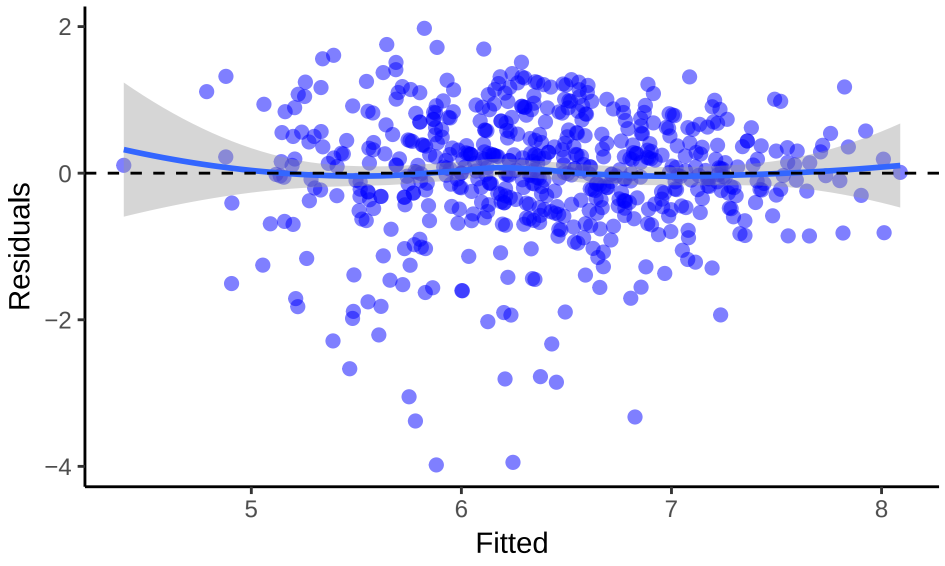


As we can see above, the distribution histogram and the residuals Q-Q plot show a close to normal distribution, and also mimics the left-hand skew that was observed in the original `imdb rating` variable.

3) Now, we need to check that the residuals are equally variable for low and high values of the predicted response variable. Then, we will check the plot of residuals vs. predicted (e vs. $\hat{y}$).

```
ggplot(augment(step2_model), aes(x= .fitted, y= .resid))+
  geom_point(colour = "blue", size = 2, alpha = 0.5)+
  theme_classic()+
  geom_smooth()+
  geom_hline(yintercept = 0, linetype = "dashed")+
  labs(x = "Fitted", y = "Residuals")
```

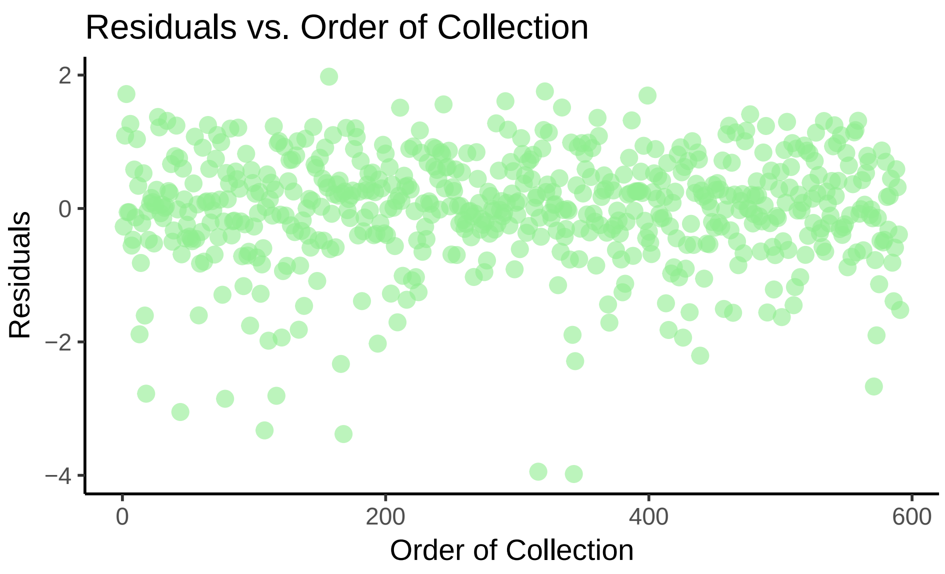
```
## `geom_smooth()` using method = 'loess'
```



The residuals are randomly scattered in a band with a constant width around 0.

4) Lastly, we will check for the independecy of the residuals:

```
ggplot(augment(step2_model), aes(x = seq_along(.resid), y = .resid)) +
  geom_point(colour = "lightgreen", size = 2.5, alpha = 0.6)+
  theme_classic()+
  labs(x = "Order of Collection", y = "Residuals", title = "Residuals vs. Order of Collection")
```



The plot above does not display any particulat pattern, so it is possible to assume that the residuals and as a consequence, the observations are independant.

Prediction

Now, we can test the predictive capability of the developed model using two movies: "Zootropolis" and "Hidden Figures" both released in 2016. The corresponding information was obtained from the IMDB website to be consistent with the analysis data.

```
zoo <- data.frame(genre="Comedy", mpaa_rating="PG", best_dir_win="yes",
                                  log_votes = log(345340))
predict_1 <- predict(step2_model, zoo, interval="predict")

imdb_rating_predictions <- c(8.0, 7.8)

predictions <- data.frame("t" = "Zootropolis",
                          "a" = sprintf("%2.1f", predict_1[1]),
                          "b" = sprintf("%2.1f-%2.1f", predict_1[2], predict_1[3]),
                          "c" = imdb_rating_predictions[1])
colnames(predictions) = c("Movie", "Predicted rating", "95% CI", "IMDb rating")

predictions
```

```
##          Movie Predicted rating  95% CI IMDb rating
## 1 Zootropolis              7.1 5.4-8.8           8
```

First of all, we can say that in this case the 95% confidence interval can be interpreted as the interval around the predicted rating score within which we are 95% confident the real movie rating would fall.

From the table we can observed that the model was close in predicting the rating for Hidden Figures.However, when predicting Zootropolis, the model failed to give the correct prediction but the real rating is inside the 95% confidence prediction interval.

SECOND PART: BAYESIAN MODELING

Another way to predict movie popularity is to use a Bayesian modeling instead of a linear regression model. So, we will start by selecting the variables that are interesting for this part of the project:

```
movies_final <- movies%>%
                  select(feature_film, drama, runtime, mpaa_rating_R,
                         thtr_rel_year, oscar_season, summer_season, imdb_rating,
                         imdb_num_votes, critics_score, best_pic_nom, best_pic_win,
                         best_actor_win, best_actress_win, best_dir_win, top200_box)
```

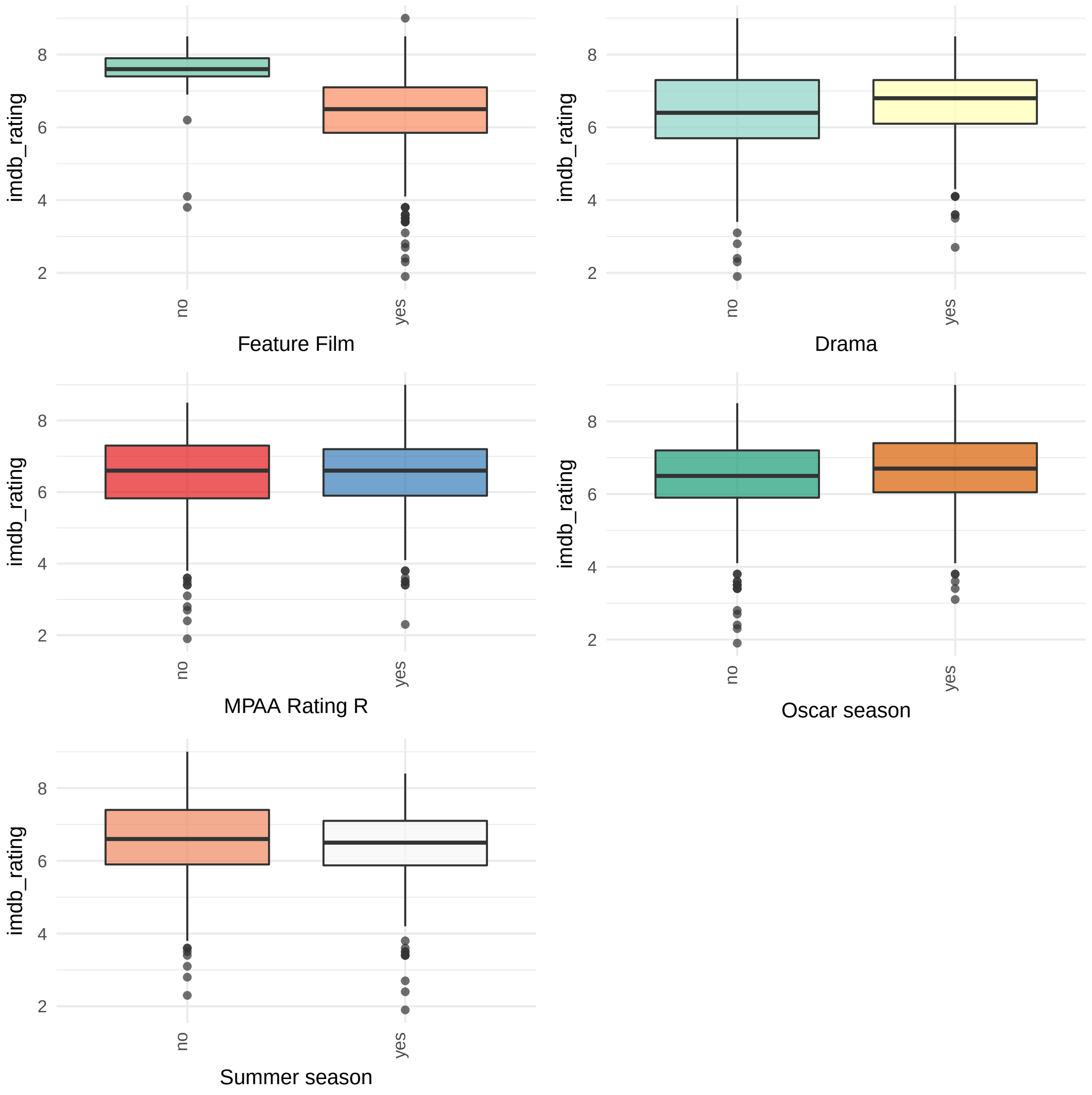
Exploratory data analysis

Relation between new exploratory and response variables:

First of all, we will investigate which is the relationship between the response variable `imdb_rating` and the new exploratory variables created. In doing so, we will create summary statistic tables and side-by-side boxplot:

```
p1 <- ggplot(movies_final, aes(x=feature_film, y = imdb_rating, fill=feature_film))+
  geom_boxplot(alpha = 0.7)+
  theme_minimal()+
  scale_fill_brewer(palette="Set2")+
  labs(x = "Feature Film", y= "imdb_rating")+
  theme(axis.text.x=element_text(angle=90, hjust = 1, vjust = 0))+
  theme(legend.position="none")

p2 <- ggplot(movies_final, aes(x=drama, y = imdb_rating, fill=drama))+
  geom_boxplot(alpha = 0.7)+
  theme_minimal()+
  scale_fill_brewer(palette="Set3")+
  labs(x = "Drama", y= "imdb_rating")+
  theme(axis.text.x=element_text(angle=90, hjust = 1, vjust = 0))+
  theme(legend.position="none")

p3<- ggplot(movies_final, aes(x=mpaa_rating_R, y = imdb_rating, fill=mpaa_rating_R))+
  geom_boxplot(alpha = 0.7)+
  theme_minimal()+
  scale_fill_brewer(palette="Set1")+
  labs(x = "MPAA Rating R", y= "imdb_rating")+
  theme(axis.text.x=element_text(angle=90, hjust = 1, vjust = 0))+
  theme(legend.position="none")

p4 <- ggplot(movies_final, aes(x=oscar_season, y = imdb_rating, fill=oscar_season))+
  geom_boxplot(alpha = 0.7)+
  theme_minimal()+
  scale_fill_brewer(palette="Dark2")+
  labs(x = "Oscar season", y= "imdb_rating")+
  theme(axis.text.x=element_text(angle=90, hjust = 1, vjust = 0))+
  theme(legend.position="none")

p5 <- ggplot(movies_final, aes(x=summer_season, y = imdb_rating, fill=summer_season))+
```

```r
  geom_boxplot(alpha = 0.7)+
  theme_minimal()+
  scale_fill_brewer(palette="RdBu")+
  labs(x = "Summer season", y= "imdb_rating")+
  theme(axis.text.x=element_text(angle=90, hjust = 1, vjust = 0))+
  theme(legend.position="none")


grid.arrange(p1, p2, p3, p4, p5, nrow = 3)
```



- feature film

```r
table_feature <- movies_final %>%
                    tbl_df%>%
                    group_by(feature_film)%>%
                    dplyr::summarize(n = n(), Mean = mean(round(mean(imdb_rating), 2)), Sd = sd(imdb_ra
pandoc.table(table_feature)
```

```
##
## -------------------------------------
##   feature_film    n     Mean     Sd
## -------------- ----- ------ --------
##       no        60    7.53   0.7808
##
##      yes        591   6.39   1.056
## -------------------------------------
```

From the table and the plots, we can observe that:

- Even though there are only 60 non-feature films vs. 591 feature films, a potential relationship between `feature_film` and `imdb_rating` is present in this dataset due to the fact that non feature films appear to have an `imdb_rating` mean of 7.53 point higher than feature films.
- Taking into consideration the variance of both groups, it is neccesary to use inferential tools to distinguish if this difference is statistically significant.

- `drama`:

```r
table_drama <- movies_final %>%
                    tbl_df%>%
                    group_by(drama)%>%
                    dplyr::summarize(n = n(), Mean = mean(round(mean(imdb_rating), 2)), Sd = sd(imdb_ra
pandoc.table(table_drama)
```

```
##
## ------------------------------
##   drama    n     Mean     Sd
## ------- ----- ------ --------
##    no     346   6.33   1.217
##
##   yes     305   6.67   0.8798
## ------------------------------
```

From the plot and the table, we can observe the following:

- There are 346 movies that belong to the category drama and 305 that do not.

- Contrary to what we observed with `feature_film` variable, there is not clear relationship between `drama` and `imdb_rating` as the mean of both groups is similar.

- `mpaa_rating_R`:

```r
table_rating <- movies_final %>%
                    tbl_df%>%
                    group_by(mpaa_rating_R)%>%
                    dplyr::summarize(n = n(), Mean = mean(round(mean(imdb_rating), 2)), Sd = sd(imdb_ra
pandoc.table(table_rating)
```

```
##
```

```
## ------------------------------------
##  mpaa_rating_R    n    Mean    Sd
## ---------------- ----- ------ -------
##       no         322   6.46   1.159
##
##       yes        329   6.52   1.008
## ------------------------------------
```

From the plot and the table, we learnt that:

- The variable `mpaa_rating` shows no relationship with `imdb_rating`.

- Not only we can see that half of the movies (329) belongs to the category 'R' of MPAA Rating, but also the mean of `audience_score` is equal in both groups as well as the variance.

- oscar_season:

```
table_oscar <- movies_final %>%
                    tbl_df%>%
                    group_by(oscar_season)%>%
                    dplyr::summarize(n = n(), Mean = mean(round(mean(imdb_rating), 2)), Sd = sd(imdb_ra
pandoc.table(table_oscar)
```

```
##
## -----------------------------------
##  oscar_season    n    Mean    Sd
## -------------- ----- ------ -------
##       no        460   6.43   1.093
##
##      yes        191   6.64   1.054
## -----------------------------------
```

From the plot and the table, we observe that:

- The variable `oscar_season` do not show a evident relationship with `imdb_rating`.

- There is fewer movies released within Oscar season (191) that outside it (460).
- The mean of `imdb_rating` is similar in both groups as well as the variance.

- summer_season:

```
table_summer <- movies_final %>%
                    tbl_df%>%
                    group_by(summer_season)%>%
                    dplyr::summarize(n = n(), Mean = mean(round(mean(imdb_rating), 2)), Sd = sd(imdb_ra
pandoc.table(table_summer)
```

```
##
## ------------------------------------
##  summer_season    n    Mean    Sd
## ---------------- ----- ------ -------
##       no         443   6.54   1.076
##
##       yes        208   6.4    1.101
## ------------------------------------
```

From the plot and the table, we observe that:

- The variable `summer_season` do not show a evident relationship with `audience_score`.

- There is fewer movies released within summer season (208) that outside it (443).
- The mean of `audience_score` is similar in both groups as well as the variance.

From the exploratory analysis performed, we can speculate that the new created variable `feature_film` would have the strongest relationship with our response variable `imdb_rating` while the other new variables created could have weak or no relationship.

Modeling

We will now proceed to conduct a Bayesian regression using the `BAS` package. We will use Bayesian Model Average (BMA) and Zellner-Siow Cauchy prior along with an uniform model prior to assign equal probabilities to all models. Regarding the option method, we will use "MCMC" (Markov chain Monte Carlo) that improves the model search efficiency.

Fist of all, we will discard any rows with `NAs`:

```
movies_final <- movies_final %>%
      filter(complete.cases(.))
```

The code below creates the full Bayesian model:

```
movies_bas <- bas.lm(imdb_rating ~ .,
                     data = movies_final,
                     method = "MCMC",
                     prior = "ZS-null",
                     modelprior = uniform())
```

We will now print the marginal inclusion probabilities obtained for the model:

```
movies_bas
```

```
##
## Call:
## bas.lm(formula = imdb_rating ~ ., data = movies_final, prior = "ZS-null",
##     modelprior = uniform(), method = "MCMC")
##
##
##  Marginal Posterior Inclusion Probabilities:
##           Intercept      feature_filmyes             dramayes
##             1.00000              0.99991              0.57286
##             runtime      mpaa_rating_Ryes        thtr_rel_year
##             0.98293              0.17561              0.06790
##     oscar_seasonyes      summer_seasonyes        imdb_num_votes
##             0.07691              0.33631              1.00000
##       critics_score       best_pic_nomyes       best_pic_winyes
##             0.99994              0.06004              0.08362
##   best_actor_winyes  best_actress_winyes      best_dir_winyes
##             0.05732              0.05928              0.05788
##       top200_boxyes
##             0.11777
```

After that, we can use the function `summary`

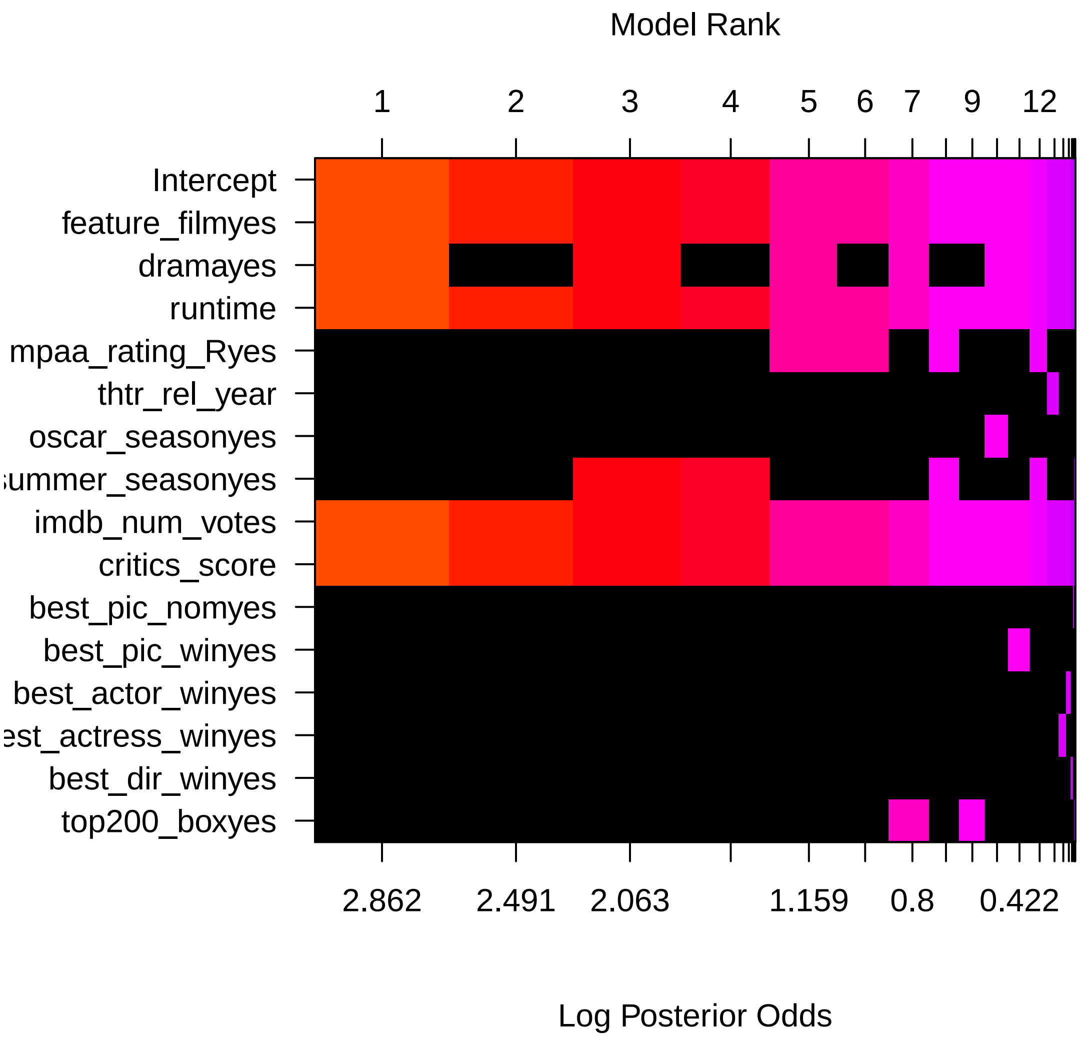
```r
summary(movies_bas)
```

```
##                      P(B != 0 | Y)  model 1     model 2     model 3
## Intercept               1.00000000   1.0000   1.0000000   1.0000000
## feature_filmyes         0.99990997   1.0000   1.0000000   1.0000000
## dramayes                0.57286377   1.0000   0.0000000   1.0000000
## runtime                 0.98292694   1.0000   1.0000000   1.0000000
## mpaa_rating_Ryes        0.17560883   0.0000   0.0000000   0.0000000
## thtr_rel_year           0.06790314   0.0000   0.0000000   0.0000000
## oscar_seasonyes         0.07690887   0.0000   0.0000000   0.0000000
## summer_seasonyes        0.33630981   0.0000   0.0000000   1.0000000
## imdb_num_votes          0.99999695   1.0000   1.0000000   1.0000000
## critics_score           0.99994049   1.0000   1.0000000   1.0000000
## best_pic_nomyes         0.06004028   0.0000   0.0000000   0.0000000
## best_pic_winyes         0.08361969   0.0000   0.0000000   0.0000000
## best_actor_winyes       0.05731659   0.0000   0.0000000   0.0000000
## best_actress_winyes     0.05928345   0.0000   0.0000000   0.0000000
## best_dir_winyes         0.05787811   0.0000   0.0000000   0.0000000
## top200_boxyes           0.11777191   0.0000   0.0000000   0.0000000
## BF                              NA   1.0000   0.6524129   0.4482058
## PostProbs                       NA   0.1766   0.1219000   0.0794000
## R2                              NA   0.6408   0.6371000   0.6431000
## dim                             NA   6.0000   5.0000000   7.0000000
## logmarg                         NA 314.5823 314.1552445 313.7798194
##                          model 4     model 5
## Intercept              1.0000000   1.0000000
## feature_filmyes        1.0000000   1.0000000
## dramayes               0.0000000   1.0000000
## runtime                1.0000000   1.0000000
## mpaa_rating_Ryes       0.0000000   1.0000000
## thtr_rel_year          0.0000000   0.0000000
## oscar_seasonyes        0.0000000   0.0000000
## summer_seasonyes       1.0000000   0.0000000
## imdb_num_votes         1.0000000   1.0000000
## critics_score          1.0000000   1.0000000
## best_pic_nomyes        0.0000000   0.0000000
## best_pic_winyes        0.0000000   0.0000000
## best_actor_winyes      0.0000000   0.0000000
## best_actress_winyes    0.0000000   0.0000000
## best_dir_winyes        0.0000000   0.0000000
## top200_boxyes          0.0000000   0.0000000
## BF                     0.4023594   0.1831721
## PostProbs              0.0719000   0.0322000
## R2                     0.6398000   0.6421000
## dim                    6.0000000   7.0000000
## logmarg              313.6719125 312.8849932
```

to see the top 5 models with the zero-one indicators for variable inclusion. It is also displayed a column with the Bayes factor (BF) for each model to the highest probability model, the posterior probabilities of the models ($PostProbs$), the R^2 of the models, the dimension of the models (dim) and the log marginal likelihood ($logmarg$) under the selected prior distribution.

Last, we can make use of the function `image`

```r
image(movies_bas, rotate=F)
```



to visualize the Log Posterior Odds and Model Rank. In the picture above, each row correspond to each variable included in the full model as well as one extra row for the intercept. In each column, we can see all possible models (2^{16} because we have 16 variables included) sorted by their posterior probability from the best to worst rank on the top (from left to right).

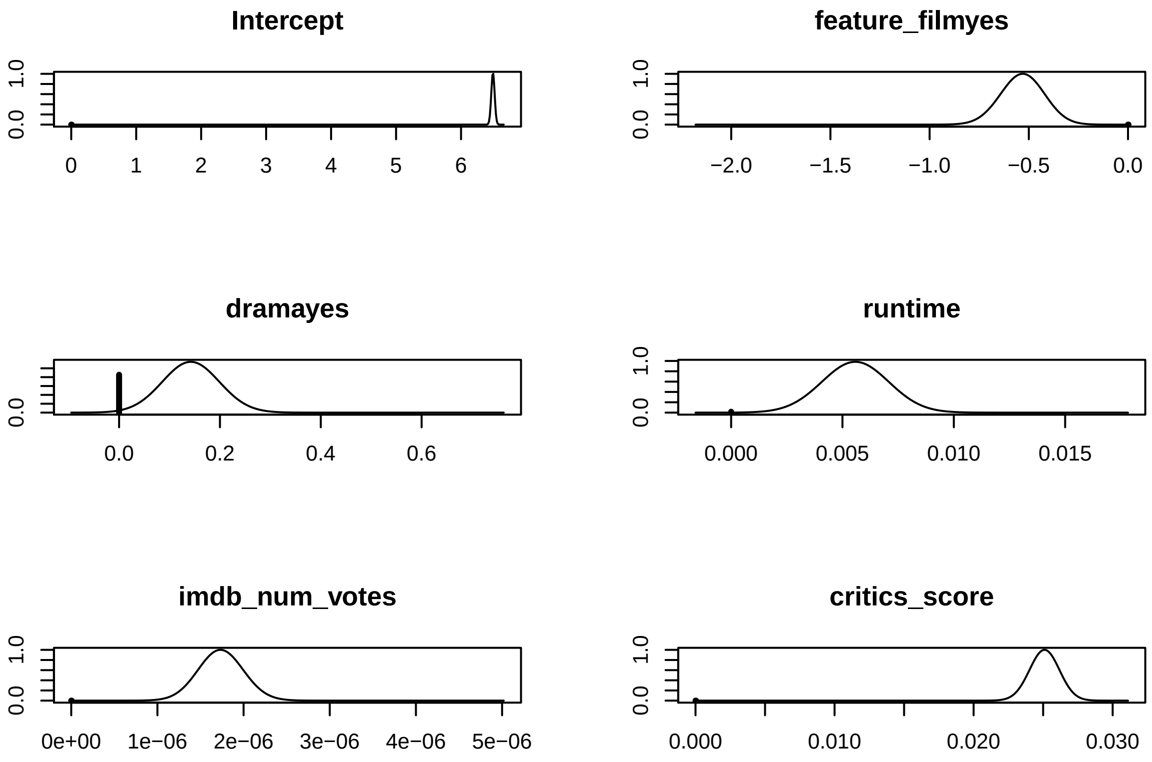
> From the model and the image above, we can see that:
> * `feature_film` has a marginal probability of 0.999, and appears in all five top models
> * `critics_score` has a marginal probability of 0.999 and also appears in all five top models * `runtime` has a marginal probability of 0.98 and appears in all five top models * `drama` has a marginal probability of 0.57 and appears in three of the five top models * `imbd_num_votes` has a marginal probability of 0.99 and appears in three of the five top models * the intercept also has a marginal probability of 1, and appears in all five top models

According to this, the best model includes the intercept, `feature_film`, `critics_score`, `drama`, `imbd_num_votes` and `runtime`

Posterior Distributions of Coefficients

We can now obtain the coefficients estimates and standard deviations under BMA in order to be able to examine the marginal distributions for the important variables coefficients. To do so, we will use the function `coef` and plot them using `plot`:

```
coef_movies <- coef(movies_bas)

par(mfrow=c(3,2))
plot(coef_movies, subset = c(1, 2, 3, 4, 9, 10), ask=F)
```



The vertical line corresponds to the posterior probability that the coefficient equals to 0. On the other hand, the shaped curve shows the density of posiible values where the coefficient is non-zero. It is worthy to mention that the height of the line is scaled to its probability. This implies that intercept and `feature_film`, `critics_score`, `imbd_num_votes` and `runtime` show no line denoting non-zero probability.

Last, we can obtain credible intervals for coefficients using `confint` method:

```
confint(coef_movies)
```

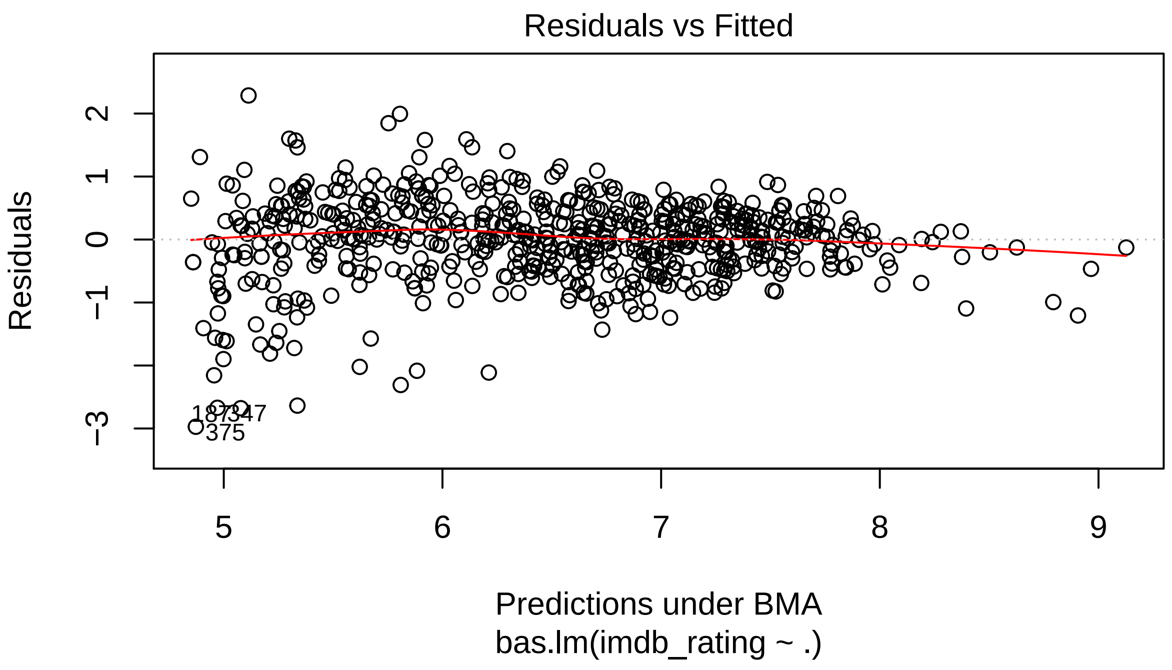
```
##                            2.5%         97.5%          beta
## Intercept           6.442754e+00  6.541224e+00  6.491538e+00
## feature_filmyes    -7.447819e-01 -3.293580e-01 -5.323484e-01
## dramayes            0.000000e+00  2.201282e-01  8.154224e-02
## runtime             2.503860e-03  8.625527e-03  5.476860e-03
## mpaa_rating_Ryes    0.000000e+00  1.258972e-01  1.484579e-02
## thtr_rel_year      -1.734267e-03  8.223891e-06 -1.027020e-04
## oscar_seasonyes    -6.731061e-04  5.564267e-02  3.328630e-03
## summer_seasonyes   -1.786014e-01  0.000000e+00 -3.904294e-02
## imdb_num_votes      1.250136e-06  2.266964e-06  1.736220e-06
## critics_score       2.296850e-02  2.705028e-02  2.508912e-02
## best_pic_nomyes    -5.424952e-02  1.705218e-02  2.802495e-03
## best_pic_winyes    -2.748420e-01  0.000000e+00 -2.054875e-02
## best_actor_winyes  -1.530452e-02  9.261951e-03  5.235979e-04
## best_actress_winyes -1.107928e-02  2.615183e-02 -1.166546e-03
## best_dir_winyes    -4.868322e-03  1.964635e-02 -7.318050e-04
## top200_boxyes       -3.086945e-01  1.216180e-03 -2.674579e-02
## attr(,"Probability")
## [1] 0.95
## attr(,"class")
## [1] "confint.bas"
```

Graphical Summaries

`BAS` package provides us with an easy way to get graphical summaries for our model just using the function `plot` and the `which` option

Residual vs. fitted plot

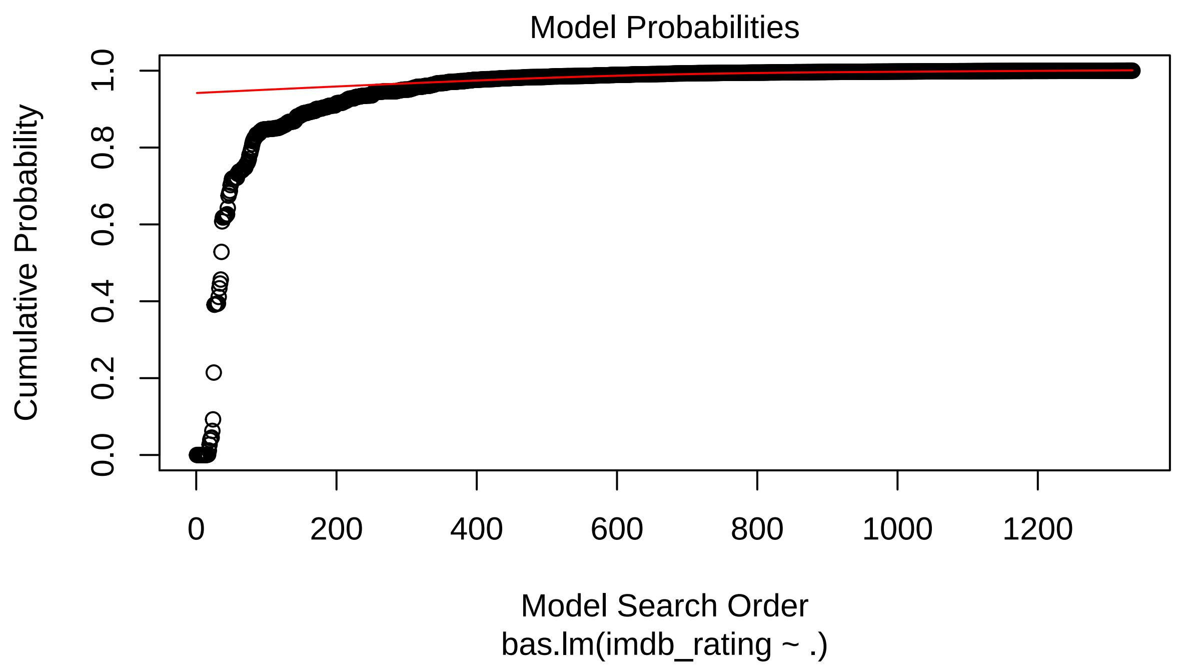
```
plot(movies_bas, which = 1, ask=F)
```



We can observe here a plot of the “Residuals vs. Fitted values”" under BMA. Ideally, we will expect to not see outliers or non-constant variance. However, in this case we can see that there is a constant spead over the prediction but there are two outliers.

Model probabilities

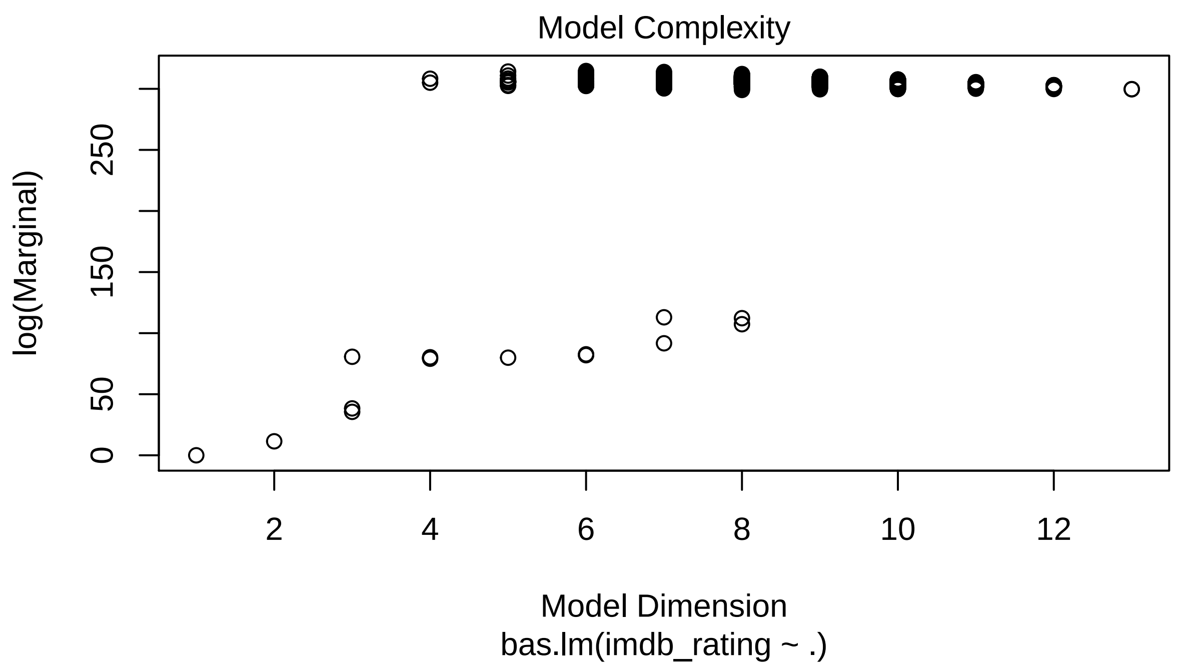
```
plot(movies_bas, which = 2, ask=F)
```



This plot displays the cumulative probability of the models in the order that they are sampled. This plot shows that the cumulative probability starts to level off after 300 model trials as each additional model adds only a small increment to the cumulative probability. The model search stops at ~1400 instead of enumerations of 2^15 combinations.

Model complexity

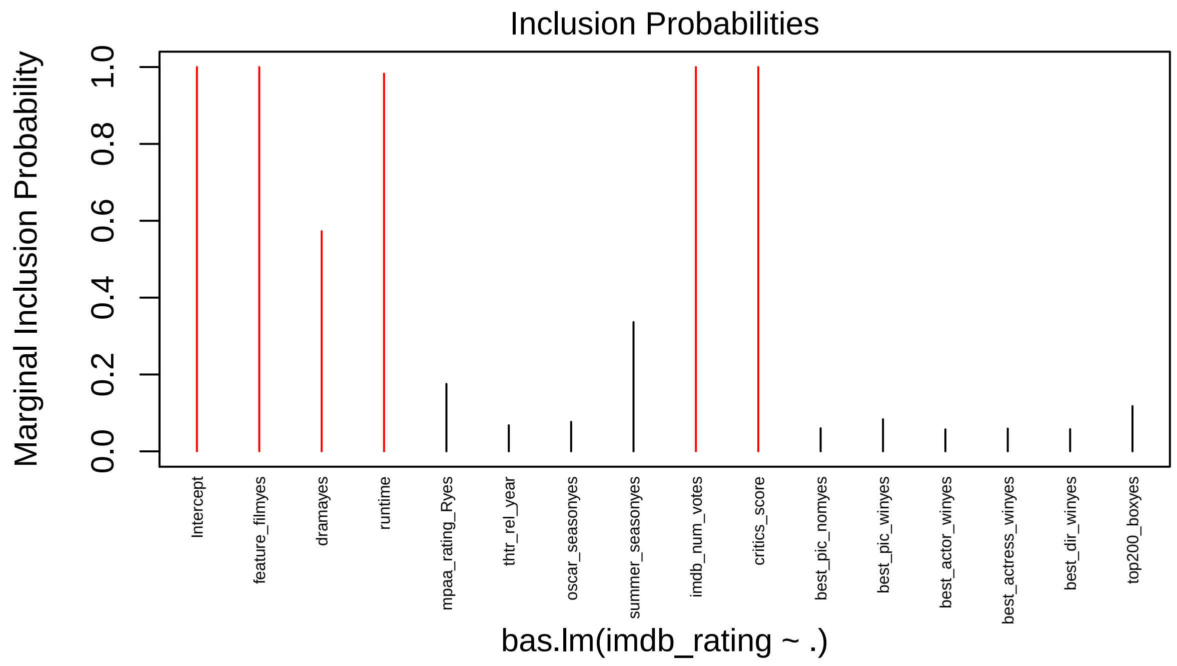
```
plot(movies_bas, which = 3, ask=F)
```



This plot shows the dimension of each model, that is the number of regression coefficients including the intercept versus the log of the marginal likelihood of the model. In this case, we can see that highest log marginal can be reached from 5 to 12 dimensions.

Marginal inclusion probabilities

```r
plot(movies_bas, which = 4, ask=F, cex.lab=0.5)
```



In this case, we can observe the marginal posterior inclusion probabilities for each of the covariates, with marginal posterior inclusion probabilities that are greater than 0.5 shown in red (important variables for explaining the data and prediction). In the graph, we can see what it was show already before about which variables contribute to the final scores.

Prediction

Now, we can test the predictive capability of the developed model using two movies: "Zootropolis" released in 2016. The corresponding information was obtained from the IMDB website and RottenTomatoes to be consistent with the analysis data.

```r
zootropolis <- data.frame(feature_film = "yes", drama="no",
                          runtime=108, mpaa_rating_R = "no",
                          thtr_rel_year = 2016, oscar_season = "no",
                          summer_season = "no",
                          imdb_num_votes = 345433, critics_score=98,
                          best_pic_nom = "yes", best_pic_win = "yes",
                          best_actor_win = "no", best_actress_win = "no",
                          best_dir_win = "yes", top200_box = "no")

predict_1 <- predict(movies_bas, zootropolis, estimator="BMA", interval = "predict", se.fit=TRUE)

predict_1$Ybma
```

```
##          [,1]
## [1,] 7.913361
```

The true `imdb_rating` is 8, which is pretty close to what our model predicted.

From the linear regression and the Bayesian model we learnt that in fact the popularity of a movie can be predicted by considering characteristic data of each movie.

In the linear regression analysis, it was possible to build a parsimonious, multivariable, linear model that is able to some extend to predict the movie popularity, understood as $IMDb\ rating$, with the four statistically significant predictors chosen. However, it is important to remember that the R^2_{adj} of our final model is only 0.3595, so this means that 35.95% of the variability is explained by the model. In the Bayesian model, we finally got a parsimonious model that also fullfilled the Bayesian assumptions.
From both models, we can see that the Bayesian model is the one which prediction was close to the real `imdb_rating`.