Modeling and prediction for movies

Mohamed ESDAIRI

Setup

Load packages

```
library(ggplot2)
library(dplyr)
library(statsr)
library(GGally)
library(gridExtra)
library(knitr)
library(plyr)
library(kableExtra)
library(broom)
```

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

Part 1: Data

The data set is comprised of 651 **randomly** sampled movies produced and released before 2016, as there is No mention to any kind of experimentation protocol or decision into a treatment and control group, so this is an **observational study** so all the results in this project are **associations** by nature and we can't draw any **causation** conclusions. Since random sampling was used, we can **generalize** the association results drawn from this analysis to all the movies produced and released before 2016, this actually the reservation that we have, since all the movies in the data-set are Before 2016, can we generalize to all movies, or are me just limited to movies before 2016, since we want to be very cautious about theses things, we will just say that we can generalize to Before 2016 movies.

```
dim(movies)
```

```
## [1] 651  32
```

Part 2: Research question

How can we describe the **association** of following variables:

- critics_score,
- genre,
- best_actor_win,
- best_actress_win,
- best_dir_win,
- run-time,

with how the audience will react to a movie?

can we fit a regression model with all or a subset of those variables to accurately describe this association?

this is a very interesting question to answer since we can predict how good is a movie well ahead of time.

Part 3: Exploratory data analysis

First lets extract all the variables of interest from the data-set: how well the audience is reacting to a movie will be calculate as the average between IMDB ratting and Rotten Tomatoes score after re-scaling the variables.

```
movies %>%
select(genre, runtime, critics_score,
       imdb_rating,  audience_score,
       best_actor_win, best_actress_win, best_dir_win) %>%
na.omit() %>%
mutate(movie_rating= (audience_score/20)+(imdb_rating/2)) %>%
select(-c(audience_score, imdb_rating)) ->
refined_data_set

str(refined_data_set)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    650 obs. of  7 variables:
##  $ genre           : Factor w/ 11 levels "Action & Adventure",..: 6 6 4 6 7 5 6 6 5 6 ...
##  $ runtime         : num  80 101 84 139 90 78 142 93 88 119 ...
##  $ critics_score   : num  45 96 91 80 33 91 57 17 90 83 ...
##  $ best_actor_win  : Factor w/ 2 levels "no","yes": 1 1 1 2 1 1 1 2 1 1 ...
##  $ best_actress_win: Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ best_dir_win    : Factor w/ 2 levels "no","yes": 1 1 1 2 1 1 1 1 1 1 ...
##  $ movie_rating    : num  6.4 7.7 8.35 7.4 3.9 8.2 7.4 5.1 8.2 6.6 ...
##  - attr(*, "na.action")= 'omit' Named int 334
##    ..- attr(*, "names")= chr "334"
```

3.1: EDA for categroical variables:

Lets look at the distribution of observations across categorical variables: we will begin by looking at counts:

```
count(refined_data_set, 'genre') %>%
  kable() %>%
  kable_styling(bootstrap_options = "striped", full_width = F, position = "left" )
```

genre	freq
Action & Adventure	65
Animation	9
Art House & International	14
Comedy	87
Documentary	51
Drama	305
Horror	23
Musical & Performing Arts	12
Mystery & Suspense	59
Other	16
Science Fiction & Fantasy	9

we can see here that the highest counts are for Drama, followed by comedy

```
format_summary_stats <- function(.data){
   kable(.data) %>%
   kable_styling(bootstrap_options = "striped", full_width = F, position = "left" )
}
```

```
count(refined_data_set, 'best_actor_win')%>% format_summary_stats
```

best_actor_win	freq
no	557
yes	93

```
count(refined_data_set, 'best_actress_win')%>% format_summary_stats
```

best_actress_win	freq
no	578
yes	72

```
count(refined_data_set, 'best_dir_win')%>% format_summary_stats
```

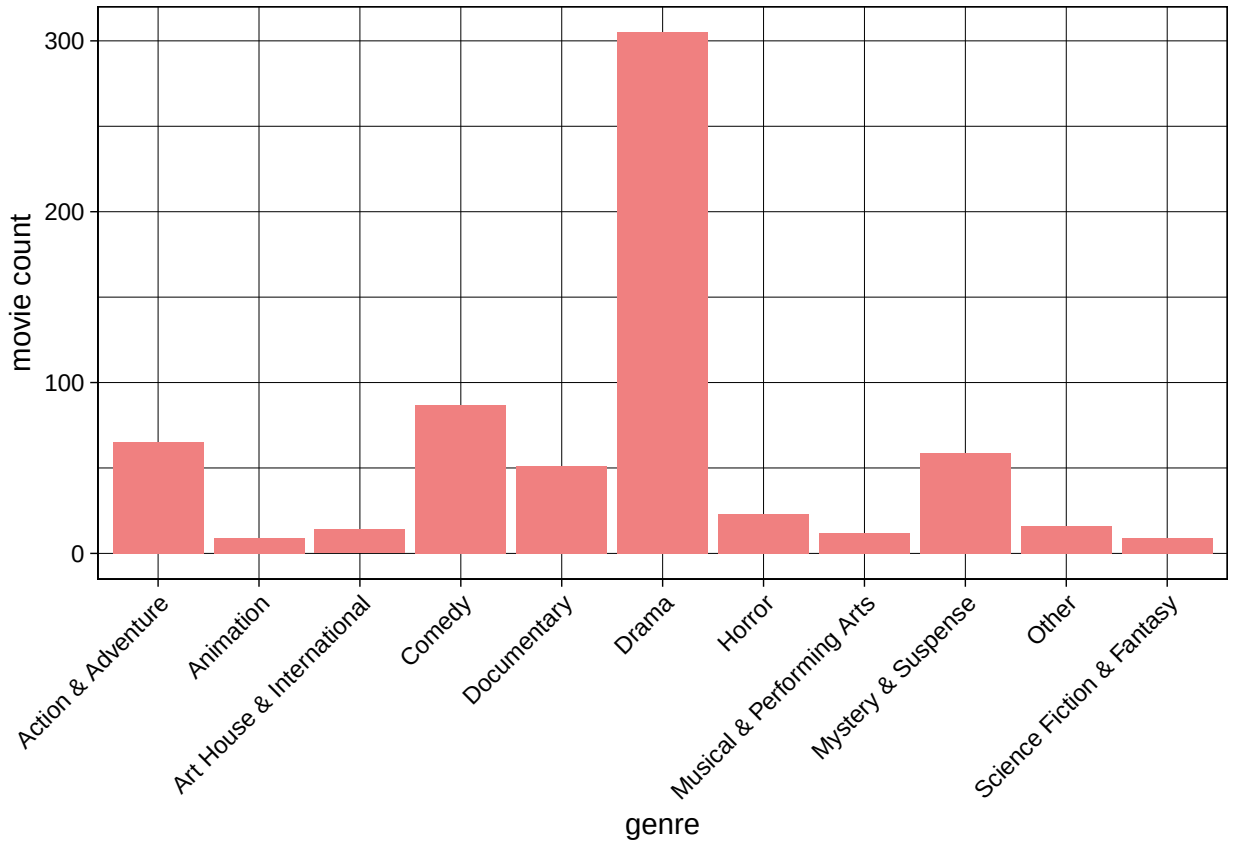
best_dir_win	freq
no	607
yes	43

and we can clearly see as expected that the count of wines of Oscar awards is very low compared to non winners.

Now lets plot those counts to get a visual perspective on the distributions:

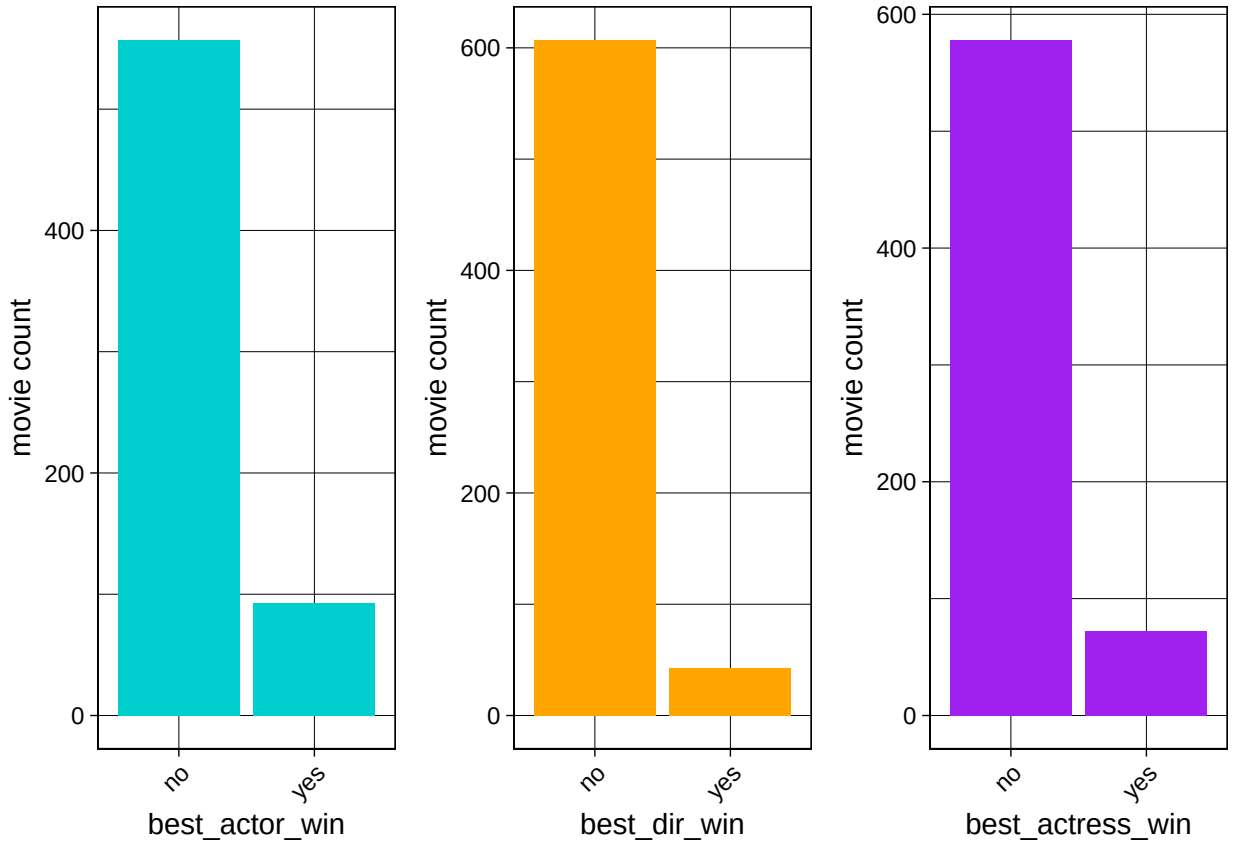
```
plot_categorical_distribution <- function(data, x_axis_var, fill_color, angle){
  ggplot(data) +
  geom_bar( aes(x=get(x_axis_var), y=..count..), fill=fill_color) +
  labs(x=x_axis_var, y='movie count') +
  theme_linedraw() +
  theme(axis.text.x = element_text(angle = angle, hjust = 1))
}
```

```
plot_categorical_distribution(refined_data_set, 'genre', 'lightcoral', 45)
```




```
layout <- rbind(c(1,2,3))

grid.arrange(plot_categorical_distribution(refined_data_set, 'best_actor_win', 'cyan3', 45),
             plot_categorical_distribution(refined_data_set, 'best_dir_win', 'orange', 45),
             plot_categorical_distribution(refined_data_set, 'best_actress_win', 'purple', 45),
             layout_matrix= layout)
```



3.2: EDA for continous variables:

lets see some summary statistics for the continuous variables

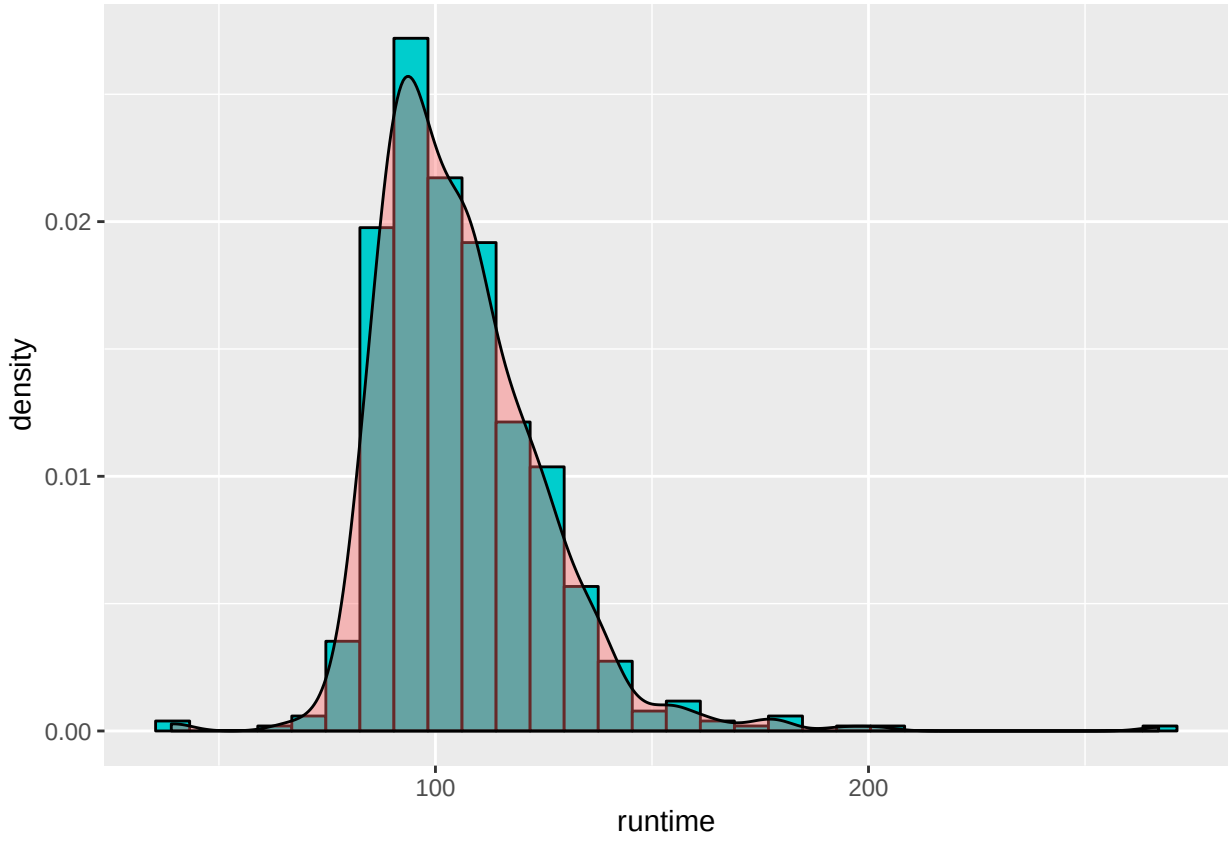
```
refined_data_set %>%
  select(runtime, critics_score, movie_rating) ->
  continous_vars
summary_res <- t(do.call(cbind, lapply(continous_vars, summary)))
```

```
summary_res %>% format_summary_stats
```

	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
runtime	39.00	92.00	103.00	105.821539	115.75	267.00
critics_score	1.00	33.00	61.00	57.653846	83.00	100.00
movie_rating	1.75	5.25	6.55	6.363154	7.60	9.35

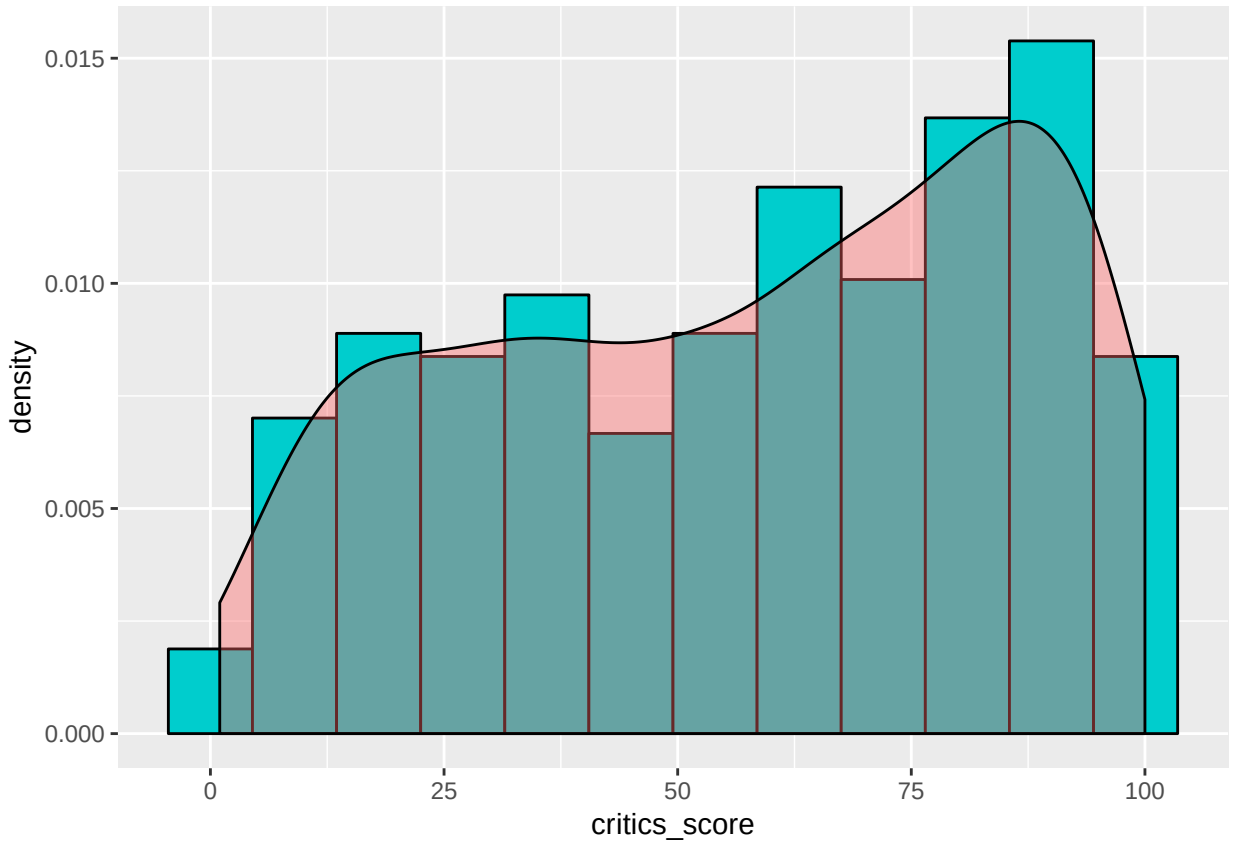
```
plot_continous_distributions <- function(data, x_axis, bins){
 ggplot(data, aes(x=get(x_axis))) +
    geom_histogram(aes(y=..density..),colour="black", fill="cyan3", bins = bins) +
    geom_density(alpha=.4, fill="#FF6666") +
    labs(x=x_axis)
}
```

```
plot_continous_distributions(refined_data_set, "runtime", 30)
```



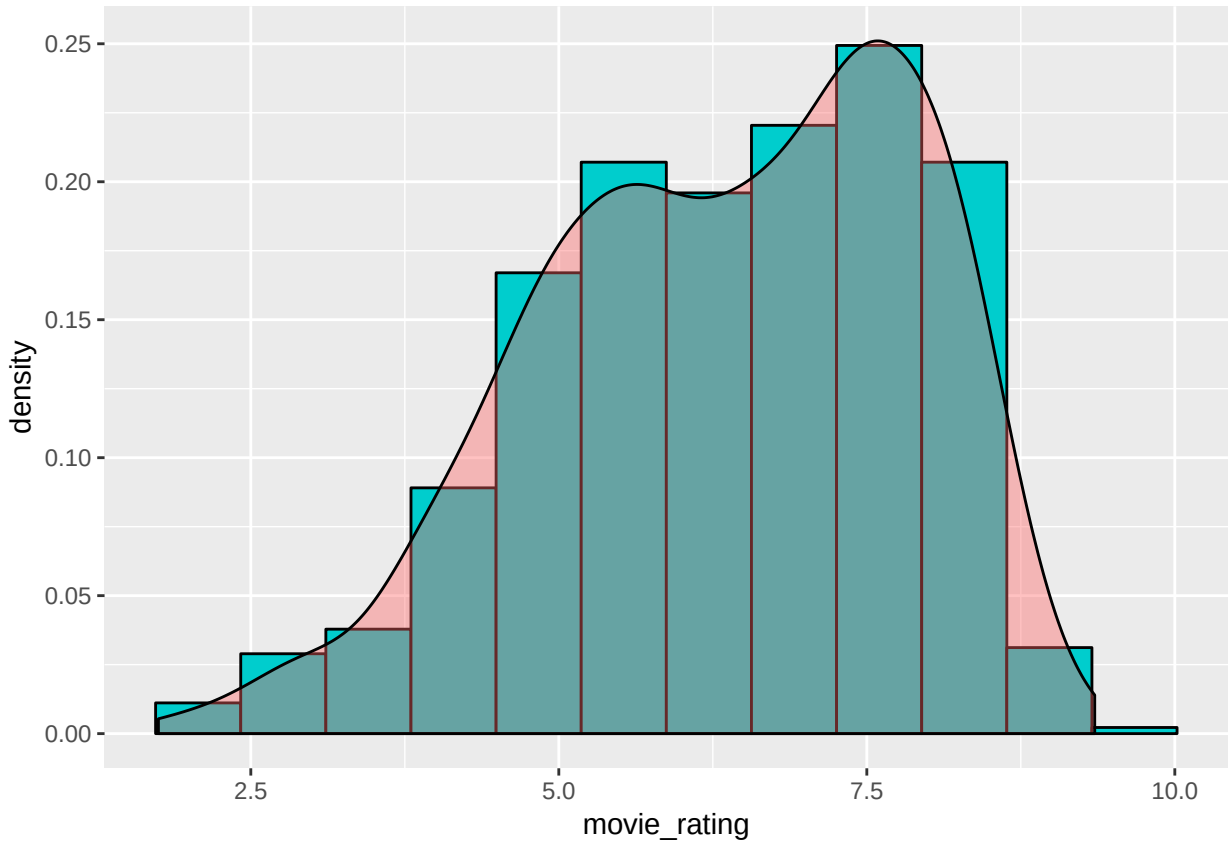
we can see that the distribution of run-time across movies in the data-set is fairly normal with a little right skew, the peak is at nearly 100 which is expect since most movies produced in Hollywood are around 100 to 120 minutes long.

```
plot_continous_distributions(refined_data_set, "critics_score", 12)
```



The distribution of critics score is not normal it has two peaks one near 100 and the other near 40 min which suggests that most of the movies in the data-set are rated somewhere around 100 if they're they receive good critics, and around 40 if they're not received as well.

```
plot_continous_distributions(refined_data_set, "movie_rating", 12)
```



for the movie rating we calculated we can see that it too is not following a normal distribution, also it has to prominent peaks one near 7.5 and the other 5.4 which suggests that users on IMDb and rotten tomatoes tend to give movies 7.5 for movies that they think are good and tend to rate movies that they not perceive as so good with 5.4, if we take into account that the scale is 10 then we can say that users on both sites gave our movies above average rating for the most part.

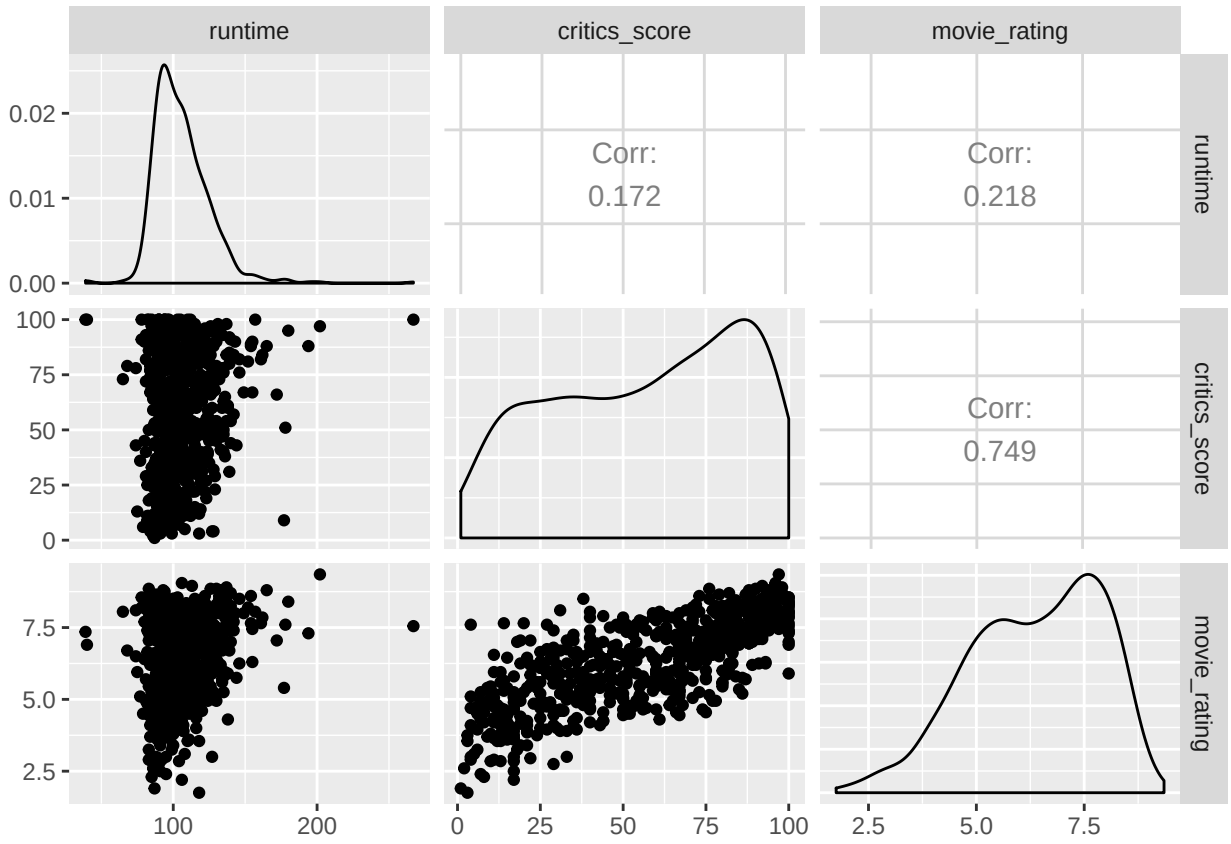
Part 4: Modeling

First let's check for the conditions for MLR:

4.1 Model diagnostics:

4.1.1 Linear relashionship between explanatory and response variables:

```
ggpairs(continous_vars)
```



we can see that there is **NO linear relashionship between critics_score and runtime** which are our continuous explanatory variables here also the correlation coefficient is 0.17 which is low, so there is **NO colinearity between explanatory variables**.

Lets build the linear model:

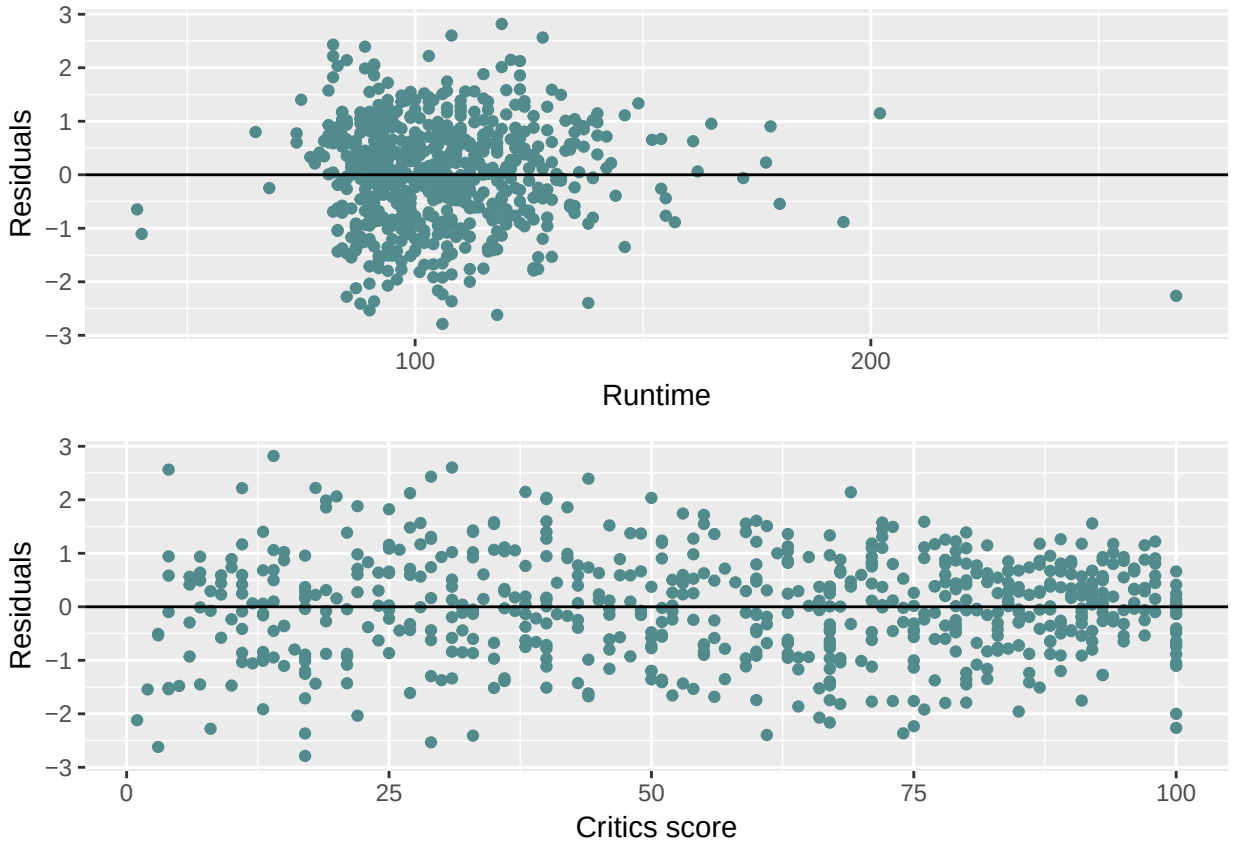
```
full_mlr_model <- lm(movie_rating~., data=refined_data_set)
```

Let's look at the residuals in respect to each numeric variable:

```
runtime_resid_plot<-
  ggplot(data=full_mlr_model, aes(x=refined_data_set$runtime, y=.resid))+
  geom_point(color = "darkslategray4") +
  geom_hline(yintercept = 0) +
  labs(x="Runtime", y="Residuals")

critics_score_resid_plot <-
  ggplot(data=full_mlr_model, aes(x=refined_data_set$critics_score, y=.resid))+
  geom_point(color = "darkslategray4") +
  geom_hline(yintercept = 0) +
  labs(x="Critics score", y="Residuals")

grid.arrange(runtime_resid_plot, critics_score_resid_plot, ncol=1)
```



both residuals plots show random scatter around zero so we can say that there is a linear relationship between our numeric explanatory variables and the response variable.

4.1.2 Nearly normal residuals with mean 0:

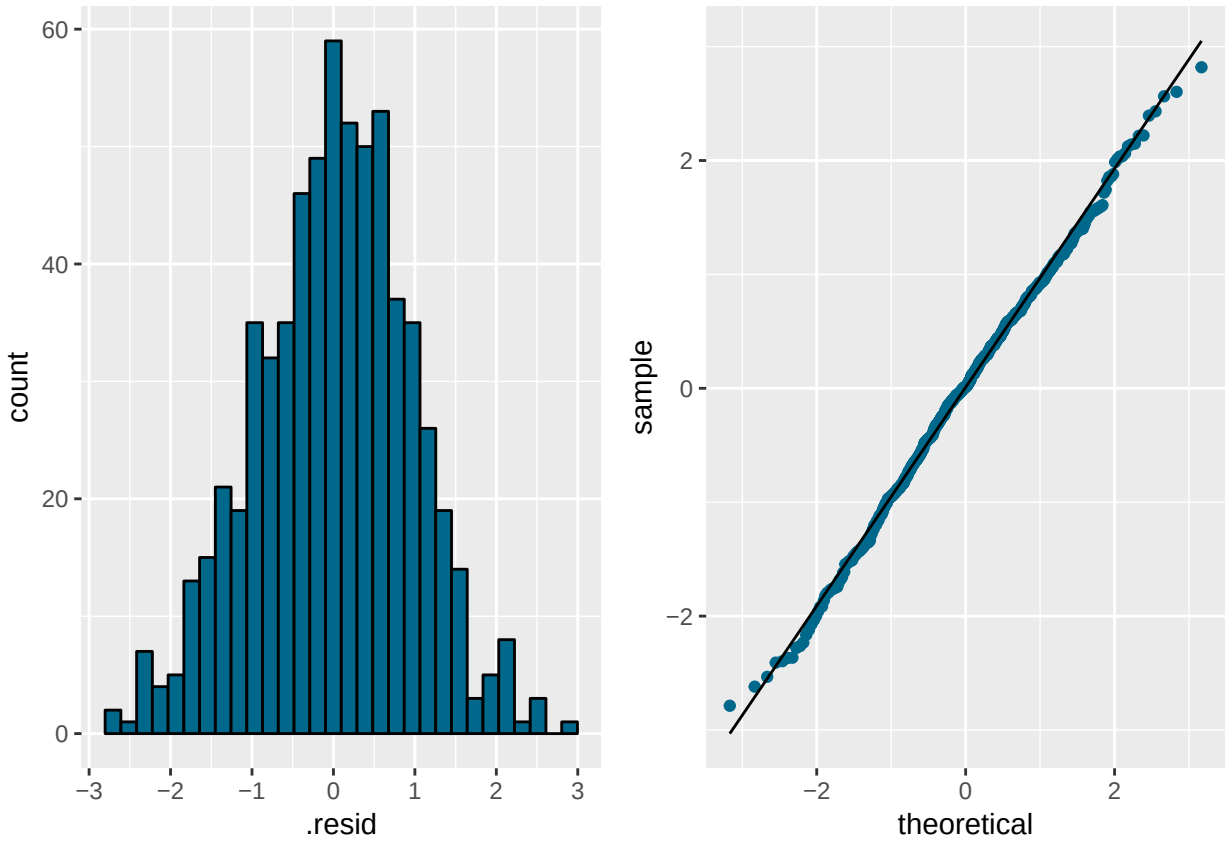
Lets plot the distribution and

```
residual_distribution <- ggplot(data=full_mlr_model, aes(x=.resid))+
  geom_histogram(color="black",fill = "deepskyblue4")

residual_qq_plot<- ggplot(data=full_mlr_model, aes(sample=.resid))+
  geom_qq(color = "deepskyblue4") +
  geom_qq_line()

grid.arrange(residual_distribution, residual_qq_plot, ncol=2)
```

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

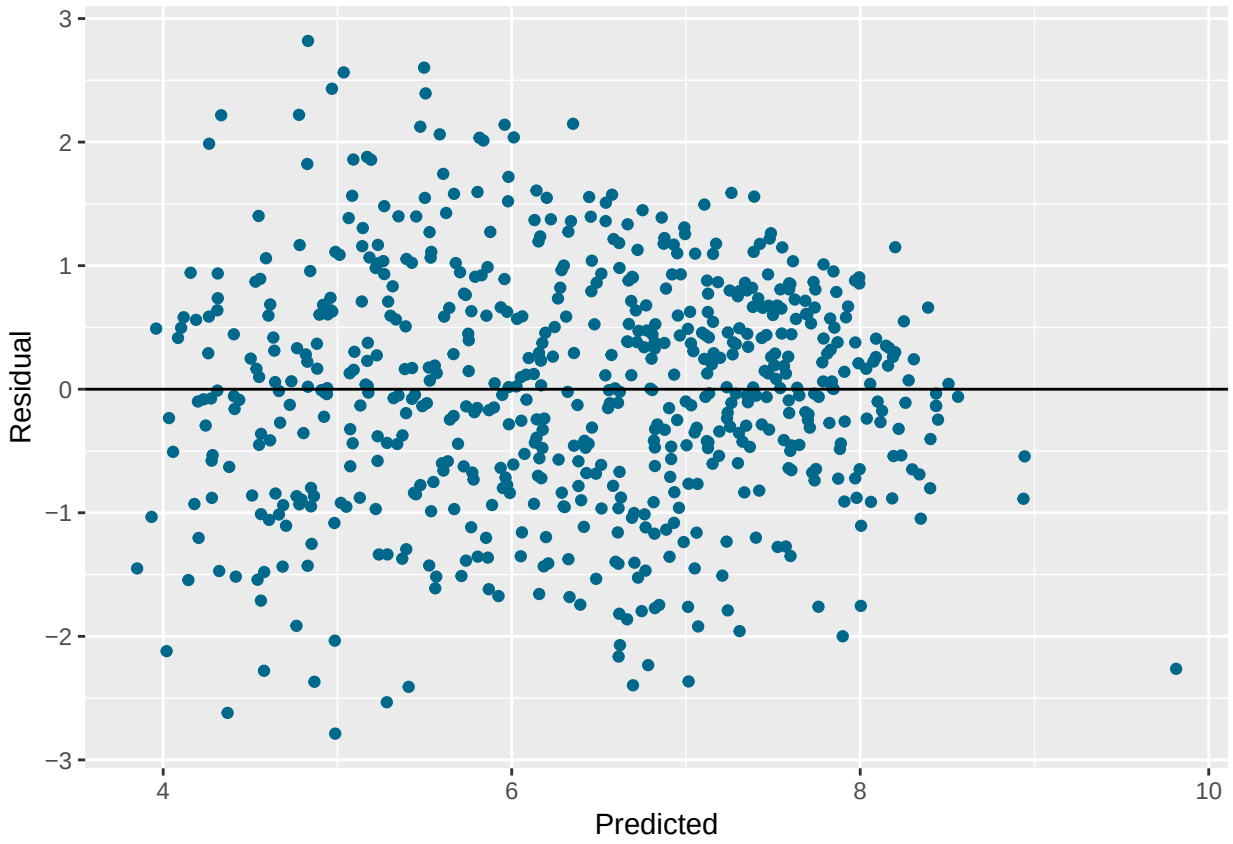


We can confidently state that the residuals are fairly normal distributed around the mean 0.

4.1.3 Constant variability of residuals:

Let's plot the residuals vs the predicted values to account for the model as a whole:

```
ggplot(full_mlr_model, aes(x = .fitted, y =  .resid) ) +
  geom_point(color = "deepskyblue4") +
  geom_hline(yintercept = 0) +
  labs(x="Predicted", y="Residual")
```



We can see that the variability of residuals stays around the same value and the scatter stays random while the predicted value increase, so the condition here is met.

4.1.4 Independant Residuals:

The movies were **randomly**, and movie rate data isn't a time series type of data, so we can safely say that the Independent Residuals condition is met as well.

Conclusion: we can see that all the conditions for an MLR model are met here, so we can be very confident that we want be getting any biases in our predictions.

4.2 Model Selection:

Let's look at the model's summary so we can see what we are working with here:

```
summary(full_mlr_model)
```

```
##
## Call:
## lm(formula = movie_rating ~ ., data = refined_data_set)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.78721 -0.63840  0.01323  0.65674  2.81894
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     3.389193   0.257734  13.150  < 2e-16 ***
```

```
## genreAnimation                   0.230125   0.346222   0.665 0.506501
## genreArt House & International  0.486232   0.285518   1.703 0.089062 .
## genreComedy                     -0.097688   0.159673  -0.612 0.540889
## genreDocumentary                 0.767433   0.194959   3.936 9.19e-05 ***
## genreDrama                       0.153780   0.136906   1.123 0.261756
## genreHorror                     -0.505145   0.235736  -2.143 0.032506 *
## genreMusical & Performing Arts  0.657912   0.308305   2.134 0.033229 *
## genreMystery & Suspense         -0.145096   0.176565  -0.822 0.411514
## genreOther                       0.102309   0.272272   0.376 0.707222
## genreScience Fiction & Fantasy -0.542311   0.343943  -1.577 0.115353
## runtime                          0.007964   0.002223   3.582 0.000368 ***
## critics_score                    0.035299   0.001516  23.284  < 2e-16 ***
## best_actor_winyes               -0.065279   0.113724  -0.574 0.566160
## best_actress_winyes             -0.065237   0.125629  -0.519 0.603742
## best_dir_winyes                  0.054491   0.159164   0.342 0.732194
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9654 on 634 degrees of freedom
## Multiple R-squared:  0.5984, Adjusted R-squared:  0.5889
## F-statistic: 62.99 on 15 and 634 DF,  p-value: < 2.2e-16
```

the p-value associated with the F-statistic for the whole model suggests that there is at least one slope that is not zero which means that there is an explanatory variable that can be used to construct a linear regression model.

We will be using backward selection with p-value, for two reasons, first we will be sure that all the variables included affect the rating of the movie so we can suggest the most important things to work on for a new movie to receive a good rating, and second this method required fewer iterations so it will be favorable when time is a limited resource which is usually the case.

4.2.1 First iteration:

Let's see the p-values so we can decide what variables to remove:

```
format_regression_summary <- function(.data) {
    kable(.data) %>%
    kable_styling(bootstrap_options = "striped", full_width = F, position = "left" )  %>%
    column_spec(2, bold = T, color="white", background = "gray") %>%
    column_spec(5, bold = T, color="white", background = "#D6576E")
  }
```

```
tidy(full_mlr_model) %>% format_regression_summary
```

term	estimate	std.error	statistic	p.value
(Intercept)	3.3891935	0.2577341	13.1499613	0.0000000
genreAnimation	0.2301245	0.3462221	0.6646732	0.5065012
genreArt House & International	0.4862320	0.2855181	1.7029813	0.0890617
genreComedy	-0.0976879	0.1596729	-0.6118005	0.5408891
genreDocumentary	0.7674333	0.1949591	3.9363817	0.0000919
genreDrama	0.1537802	0.1369065	1.1232500	0.2617565
genreHorror	-0.5051450	0.2357361	-2.1428409	0.0325059
genreMusical & Performing Arts	0.6579120	0.3083055	2.1339614	0.0332290
genreMystery & Suspense	-0.1450965	0.1765647	-0.8217751	0.4115138
genreOther	0.1023086	0.2722718	0.3757591	0.7072217
genreScience Fiction & Fantasy	-0.5423105	0.3439434	-1.5767434	0.1153534
runtime	0.0079638	0.0022235	3.5816685	0.0003676
critics_score	0.0352989	0.0015160	23.2841610	0.0000000
best_actor_winyes	-0.0652793	0.1137238	-0.5740162	0.5661604
best_actress_winyes	-0.0652372	0.1256286	-0.5192865	0.6037422
best_dir_winyes	0.0544914	0.1591644	0.3423594	0.7321939

Let's start by removing the best_dir_win variable since it has the highest p-value

```
mlr_model_without_dir <-
  lm(movie_rating~.-best_dir_win,
     data=refined_data_set)

  tidy(mlr_model_without_dir) %>% format_regression_summary
```

term	estimate	std.error	statistic	p.value
(Intercept)	3.3742645	0.2538418	13.2927835	0.0000000
genreAnimation	0.2285229	0.3459497	0.6605669	0.5091297
genreArt House & International	0.4825120	0.2851129	1.6923541	0.0910690
genreComedy	-0.0975765	0.1595615	-0.6115292	0.5410682
genreDocumentary	0.7622501	0.1942352	3.9243668	0.0000965
genreDrama	0.1522116	0.1367346	1.1131900	0.2660481
genreHorror	-0.5044240	0.2355628	-2.1413571	0.0326252
genreMusical & Performing Arts	0.6553222	0.3079984	2.1276809	0.0337482
genreMystery & Suspense	-0.1451458	0.1764419	-0.8226268	0.4110291
genreOther	0.0994722	0.2719565	0.3657650	0.7146623
genreScience Fiction & Fantasy	-0.5396201	0.3436145	-1.5704227	0.1168149
runtime	0.0081121	0.0021793	3.7222753	0.0002150
critics_score	0.0353651	0.0015026	23.5364523	0.0000000
best_actor_winyes	-0.0639671	0.1135802	-0.5631889	0.5735051
best_actress_winyes	-0.0641514	0.1255012	-0.5111615	0.6094157

we can see that the R-squared stayed the same while the Adjusted R-squared improved (which expected since we dropped one predictor)

4.2.2 Second iteration:

now let's drop the best_actress_win variable since it has the highest p-value:

```
mlr_model_without_dir_actress <-
  lm(movie_rating~.-best_dir_win-best_actress_win,
     data=refined_data_set)

  tidy(mlr_model_without_dir_actress) %>% format_regression_summary
```

term	estimate	std.error	statistic	p.value
(Intercept)	3.3933667	0.2509301	13.5231524	0.0000000
genreAnimation	0.2187697	0.3452225	0.6337065	0.5265002
genreArt House & International	0.4774458	0.2847750	1.6765717	0.0941177
genreComedy	-0.1054595	0.1587222	-0.6644280	0.5066572
genreDocumentary	0.7604741	0.1940913	3.9181256	0.0000989
genreDrama	0.1440656	0.1357239	1.0614607	0.2888835
genreHorror	-0.5067915	0.2353804	-2.1530741	0.0316882
genreMusical & Performing Arts	0.6577124	0.3077840	2.1369287	0.0329847
genreMystery & Suspense	-0.1539786	0.1754917	-0.8774123	0.3805940
genreOther	0.0935565	0.2715523	0.3445250	0.7305654
genreScience Fiction & Fantasy	-0.5403361	0.3434121	-1.5734336	0.1161158
runtime	0.0079411	0.0021523	3.6896578	0.0002438
critics_score	0.0353428	0.0015011	23.5452081	0.0000000
best_actor_winyes	-0.0679437	0.1132476	-0.5999570	0.5487486

4.2.3 Third iteration:

now let's drop the best_actor_win variable since it has the highest p-value:

```
mlr_model_without_dir_actress_actor <-
  lm(movie_rating~.-best_dir_win-best_actress_win-best_actor_win,
     data=refined_data_set)

  tidy(mlr_model_without_dir_actress_actor)%>% format_regression_summary
```

term	estimate	std.error	statistic	p.value
(Intercept)	3.4162933	0.2478787	13.7821173	0.0000000
genreAnimation	0.2138432	0.3449514	0.6199228	0.5355303
genreArt House & International	0.4843759	0.2843977	1.7031640	0.0890252
genreComedy	-0.1063686	0.1586353	-0.6705229	0.5027675
genreDocumentary	0.7634200	0.1939317	3.9365410	0.0000918
genreDrama	0.1413297	0.1355791	1.0424153	0.2976146
genreHorror	-0.5028112	0.2351687	-2.1380875	0.0328895
genreMusical & Performing Arts	0.6625424	0.3075240	2.1544409	0.0315799
genreMystery & Suspense	-0.1631344	0.1747390	-0.9335889	0.3508697
genreOther	0.0904674	0.2713670	0.3333766	0.7389597
genreScience Fiction & Fantasy	-0.5337494	0.3430641	-1.5558302	0.1202451
runtime	0.0076532	0.0020970	3.6495364	0.0002842
critics_score	0.0353340	0.0015002	23.5522953	0.0000000

4.3 Interpretation of model parameters:

```
best_model <- mlr_model_without_dir_actress_actor

tidy(best_model)%>%
  kable() %>%
  kable_styling(bootstrap_options = "striped", full_width = F, position = "left" ) %>%
  column_spec(2, bold = T, color="white", background = "#D6576E")
```

term	estimate	std.error	statistic	p.value
(Intercept)	**3.4162933**	0.2478787	13.7821173	0.0000000
genreAnimation	**0.2138432**	0.3449514	0.6199228	0.5355303
genreArt House & International	**0.4843759**	0.2843977	1.7031640	0.0890252
genreComedy	**-0.1063686**	0.1586353	-0.6705229	0.5027675
genreDocumentary	**0.7634200**	0.1939317	3.9365410	0.0000918
genreDrama	**0.1413297**	0.1355791	1.0424153	0.2976146
genreHorror	**-0.5028112**	0.2351687	-2.1380875	0.0328895
genreMusical & Performing Arts	**0.6625424**	0.3075240	2.1544409	0.0315799
genreMystery & Suspense	**-0.1631344**	0.1747390	-0.9335889	0.3508697
genreOther	**0.0904674**	0.2713670	0.3333766	0.7389597
genreScience Fiction & Fantasy	**-0.5337494**	0.3430641	-1.5558302	0.1202451
runtime	**0.0076532**	0.0020970	3.6495364	0.0002842
critics_score	**0.0353340**	0.0015002	23.5522953	0.0000000

the equation of the regression model is:

$$
\begin{aligned}
movie\_rating =& 3.4162933 + \\
& 0.2138432 * genre_{animation} + 0.484375892 * genre_{art\_house\_international} + -0.106368569 * genre_{comedy} + \\
& 0.7634200 * genre_{documentary} + 0.1413297 * genre_{drama} + -0.5028112 * genre_{horror} + \\
& 0.6625424 * genre_{musical\_performing\_arts} + -0.1631344 * genre_{mystry\_suspense} + 0.0904674 * genre_{other} + \\
& -0.5337494 * genre_{scifi\_fantasy} + 0.0076532 * runtime + 0.0353340 * critics\_score
\end{aligned}
$$

if a variable has a subscript, that mean if the variable equals to the specified category in the subscript we will put a 1 in the value of the variable there and 0 else where for the entire variable levels.

we will just pick there values to interpret, one level of the genre variable and the two continuous variables run-time and critics_score:

- all else held constant, if the genre of a movie is a documentary, we expect to observe on average an additional 0.76 in its score.
- all else held constant, for every minute added to the run-time of movie we expect to see, on average, a 0.007 increase in the movie rating
- all else held constant, for every point that critics add to the rating of a movie we are expected to see, on average, a 0.03 increase in its rating.

Conclusion:

for the *association* between:

- critics_score,
- genre,
- best_actor_win,
- best_actress_win,
- best_dir_win,
- run-time

and:

- movie_rating

the data we have, suggests that for regression the following variable are the most significant ones to decide movie_rating:

- critics_score,
- genre,
- run-time

So we can state that there is *NO association*, in a regression setting, between getting a good movie rating and hiring a lot of Oscar winners: actors, actress nor directors.

Part 5: Prediction

5.1 Predicting movie rating for Finding Dory:

We will run our predictions on the popular animation movie: **Finding Dory** directed by *Andrew Stanton & Angus MacLane*, let's check if the movie is in already in the data-set

```
any(movies$title == " Finding Dory")
```

```
## [1] FALSE
```

So this movie isn't present in the data-set.

I gathered the information from Rotten Tomatoes and IMDb, this is the data we need, the explanatory variables:

- critics_score: 94%
- genre: Animation
- run-time: 100min

to compute the response variable movie_rating we need:

- IMDB audience rating: 7.3/10
- RT audience score: 84/100

```
real_movie_rating <- (84/20)+(7.3/2)
real_movie_rating
```

```
## [1] 7.85
```

this movie got a 7.85 average rating from audience.

Let's construct the data.frame that we will use to compute the predicted rating:

```
third_test_movie <-
  data.frame("critics_score"=94,
             "genre"="Animation",
             "runtime"=100,
             "best_actor_win"="no",
             "best_actress_win"="no",
             "best_dir_win"="no")
```

Note: best_actor_win, best_actress_win, best_dir_win although present here in the input data they don't affect the prediction, they're included for the predict function to work properly.

Lets predict the movie_rating using our model:

```
predicted_movie_rating <- predict(mlr_model_without_dir_actress_actor, third_test_movie)
predicted_movie_rating <- as.numeric(predicted_movie_rating)
predicted_movie_rating
```

```
## [1] 7.716858
```

Let's change the value of best_actor_win, best_actress_win, best_dir_win to see if they'll affect the predicted value:

```
third_test_movie_altered <-
  data.frame("critics_score"=94,
             "genre"="Animation",
             "runtime"=100,
             "best_actor_win"="yes",
             "best_actress_win"="yes",
             "best_dir_win"="yes")
predict(mlr_model_without_dir_actress_actor, third_test_movie_altered)
```

```
##        1
## 7.716858
```

so the value stays the same as mentioned in the Note above.

Lets compute the difference between the prediction and real value:

```
real_movie_rating - predicted_movie_rating
```

```
## [1] 0.1331416
```

so we missed the real value only with about 0.13 this is a good prediction value, given that we used a simple linear regression model.

Let's compute the confidence interval for this prediction:

```
predict(mlr_model_without_dir_actress_actor, third_test_movie, interval = "predict", level=0.95)
```

```
##        fit      lwr     upr
## 1 7.716858 5.717807 9.71591
```

the model predicts, with 95% confidence, that "Finding Dory", will get a score between 5.71 and 9.71.

Part 6: Conclusion

To answer our original research question, we can say that from all the variables:

- critics_score,
- genre,
- best_actor_win,
- best_actress_win,
- best_dir_win,
- run-time,

We can use critics_score, genre and run-time to build a multiple linear regression model with all those variables being significant predictor of the how the audience will rate a 2016 movie.

as for the problems with this analysis:

- we have data limited to 2016, we can extrapolate to all movies
- we have data from an observational study, we can't get causation conclusions
- a large confidence interval which suggests that the model has a lot of variance while making a prediction.