FinalProject_IMDB

Adithya Daine Manjunath, Charan Ashwath, Tejas Padavalamane

2023-11-29

Packages

```r
library(tidyverse)
library(ggplot2)
library(tigerstats)
library(reticulate)
library(MASS)
library(MLmetrics)
library(dplyr)
```

R Markdown

```r
IMDB_data <- read.csv("~/Downloads/Movie Ratings.csv", sep=",")
view(IMDB_data)
summary(IMDB_data)
```

```
##      Film               Genre            Rotten.Tomatoes.Ratings..
##  Length:562         Length:562         Min.   : 0.0
##  Class :character   Class :character   1st Qu.:25.0
##  Mode  :character   Mode  :character   Median :46.0
##                                        Mean   :47.4
##                                        3rd Qu.:70.0
##                                        Max.   :97.0
##  Audience.Ratings.. Budget..million... Year.of.release
##  Min.   : 0.00      Min.   :  0.0      Min.   :2007
##  1st Qu.:47.00      1st Qu.: 20.0      1st Qu.:2008
##  Median :58.00      Median : 35.0      Median :2009
##  Mean   :58.83      Mean   : 50.1      Mean   :2009
##  3rd Qu.:72.00      3rd Qu.: 65.0      3rd Qu.:2010
##  Max.   :96.00      Max.   :300.0      Max.   :2011
```

Data Preparation: Show the information of the dataset. E.g.# of observations, # of attributes, data types, missing values, etc.

##Obsrvations

```r
dim(IMDB_data)
```

```
## [1] 562   6
```

Here, we can see that the IMDB movie dataset has 562 rows and 6 columns.

##Attributes

```r
print("IMDB Data")
```

```
## [1] "IMDB Data"
```

```r
str(IMDB_data)
```

```
## 'data.frame':    562 obs. of  6 variables:
##  $ Film                     : chr  "(500) Days of Summer " "10,000 B.C." "12 Rounds " "127 Hours" ..
##  $ Genre                    : chr  "Comedy" "Adventure" "Action" "Adventure" ...
##  $ Rotten.Tomatoes.Ratings..: int  87 9 30 93 55 39 40 50 43 93 ...
##  $ Audience.Ratings..       : int  81 44 52 84 70 63 71 57 48 93 ...
##  $ Budget..million...       : int  8 105 20 18 20 200 30 32 28 8 ...
##  $ Year.of.release          : int  2009 2008 2009 2010 2009 2009 2008 2007 2011 2011 ...
```

The str() function is used to obtain the structure of the selected dataset, in this case the IMDB dataset.

##Null values

```r
#is.na(IMDB_data)
#colSums(is.na(IMDB_data))
sum(is.na(IMDB_data))
```

```
## [1] 0
```

There are no null values

##Mean, Minimum and Maximum Critics Rating

```r
mean(IMDB_data$Rotten.Tomatoes.Ratings..)
```

```
## [1] 47.40391
```

```r
min(IMDB_data$Rotten.Tomatoes.Ratings..)
```

```
## [1] 0
```

```r
max(IMDB_data$Rotten.Tomatoes.Ratings..)
```

```
## [1] 97
```

The average ratings provided by the critics is 47.40 with a minimum of 0 and a maximum of 97.

##Mean, Minimum and Maximum Audience Rating

```r
mean(IMDB_data$Audience.Ratings..)
```

```
## [1] 58.83096
```

```r
min(IMDB_data$Audience.Ratings..)
```

```
## [1] 0
```

```r
max(IMDB_data$Audience.Ratings..)
```

```
## [1] 96
```

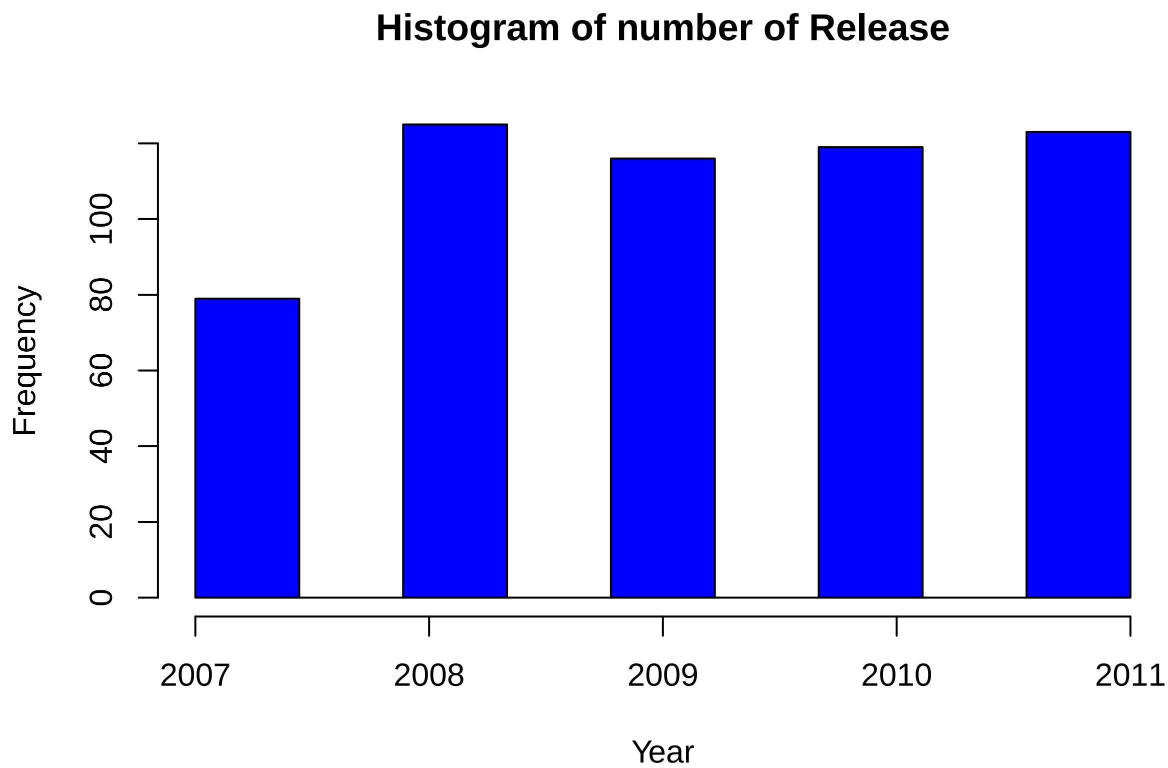
The average ratings provided by the audience is 58.83 with a minimum of 0 and a maximum of 96.

##Histogram, Histogram Overlay, Barplot, Barplot Overlay, Boxplot

#HISTOGRAM

```r
#HISTOGRAM Year of Release
maxpotints <- max(IMDB_data$Year.of.release)
minpoints <- min(IMDB_data$Year.of.release)

#
hist(IMDB_data$Year.of.release,breaks = seq(minpoints,maxpotints,l=10), col = "blue", main = "Histogram
```




```
movies_per_year <- table(IMDB_data$Year.of.release)
movies_per_year
```

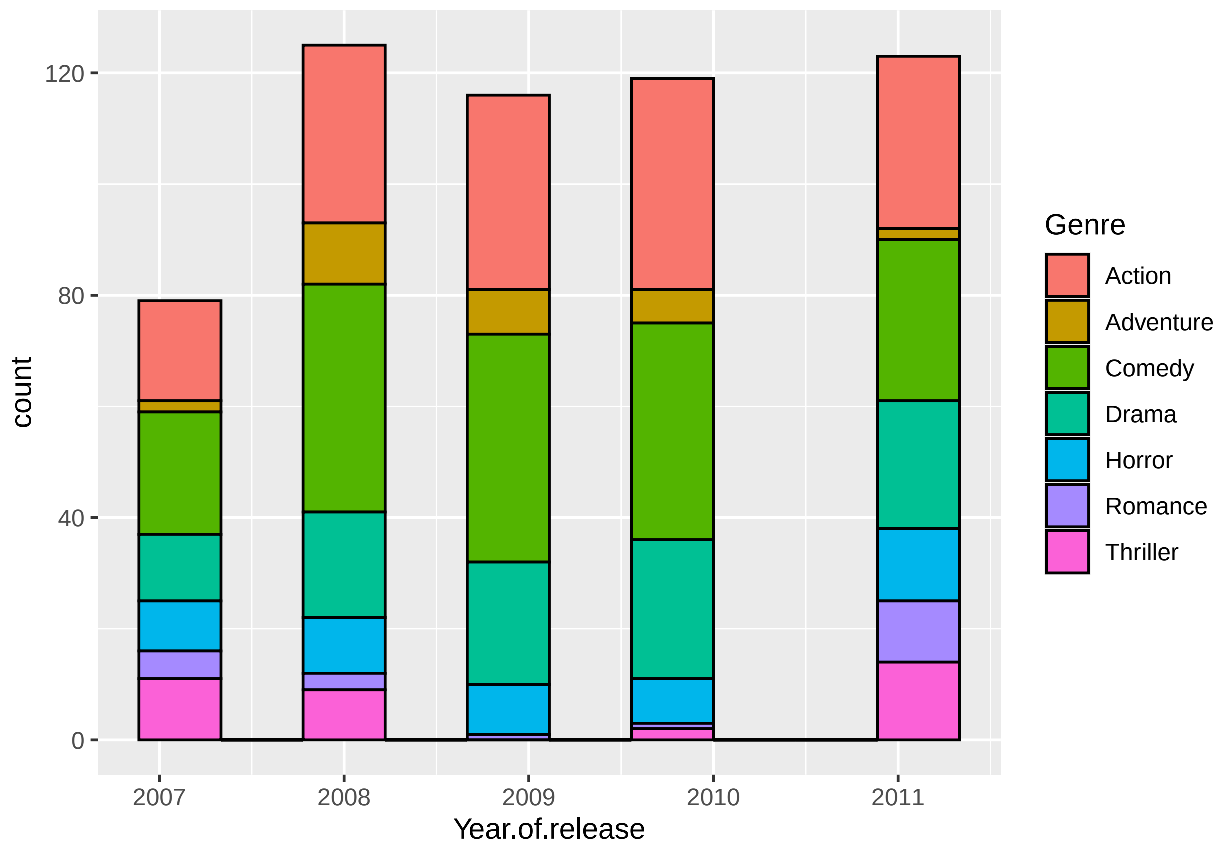
```
##
## 2007 2008 2009 2010 2011
##   79  125  116  119  123
```

From the above histogram representation, we can see that we have obtained the dataset of the IMDB spanning from the year 2007 to 2011. As you can see, there were more movies released in the year 2008 and less movies were released in the year 2007.

#HISTOGRAM WITH OVERLAY

```
#
#ggplot(IMDB_data) +
#geom_histogram(mapping = aes(x = Year.of.release),bins = 10)

#HISTOGRAM WITH OVERLAY
ggplot(IMDB_data, aes(x = Year.of.release)) +
geom_histogram(aes(fill = Genre),bins = 10, color = "black")
```

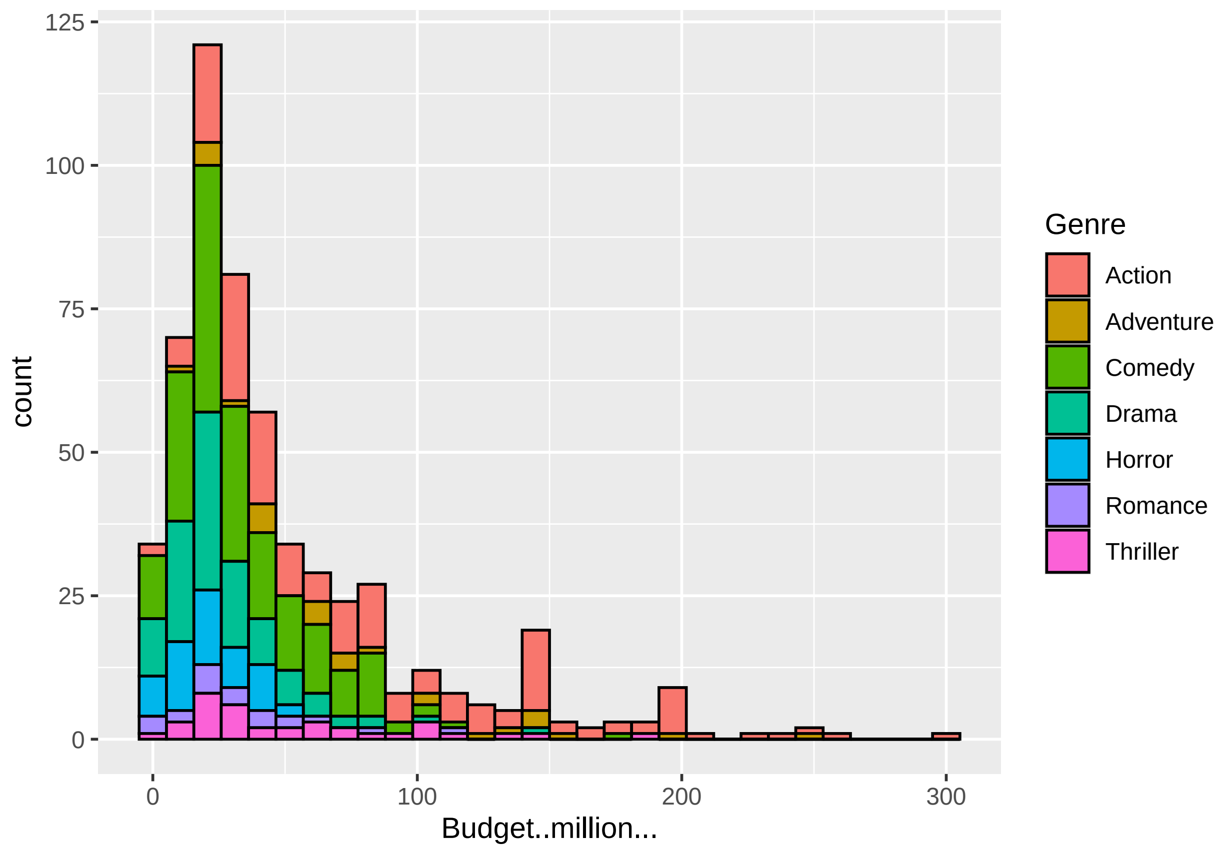



```
movies_per_genre_year <- table(IMDB_data$Year.of.release, IMDB_data$Genre)
movies_per_genre_year
```

```
##
##        Action Adventure Comedy Drama Horror Romance Thriller
##   2007     18         2     22    12      9       5       11
##   2008     32        11     41    19     10       3        9
##   2009     35         8     41    22      9       1        0
##   2010     38         6     39    25      8       1        2
##   2011     31         2     29    23     13      11       14
```

This graph provides the representation of the number of movies released each year with respect to the genres. From this, we can find that the "Comedy" genre has the most releases and the "Romance" genre has the least releases.

```
ggplot(IMDB_data, aes(x = Budget..million...)) +
geom_histogram(aes(fill = Genre),bins = 30, color = "black")
```




```
#PARAMETERS
total_budget_per_genre <- aggregate(Budget..million... ~ Genre, data = IMDB_data, sum)
least_budget_genre <- total_budget_per_genre[which.min(total_budget_per_genre$Budget..million...), ]
most_budget_genre <- total_budget_per_genre[which.max(total_budget_per_genre$Budget..million...), ]


total_budget_per_genre
```

```
##       Genre Budget..million...
## 1    Action              13033
## 2 Adventure               2363
## 3    Comedy               6211
## 4     Drama               2813
## 5    Horror               1062
## 6   Romance                709
## 7  Thriller               1968
```

```
least_budget_genre
```

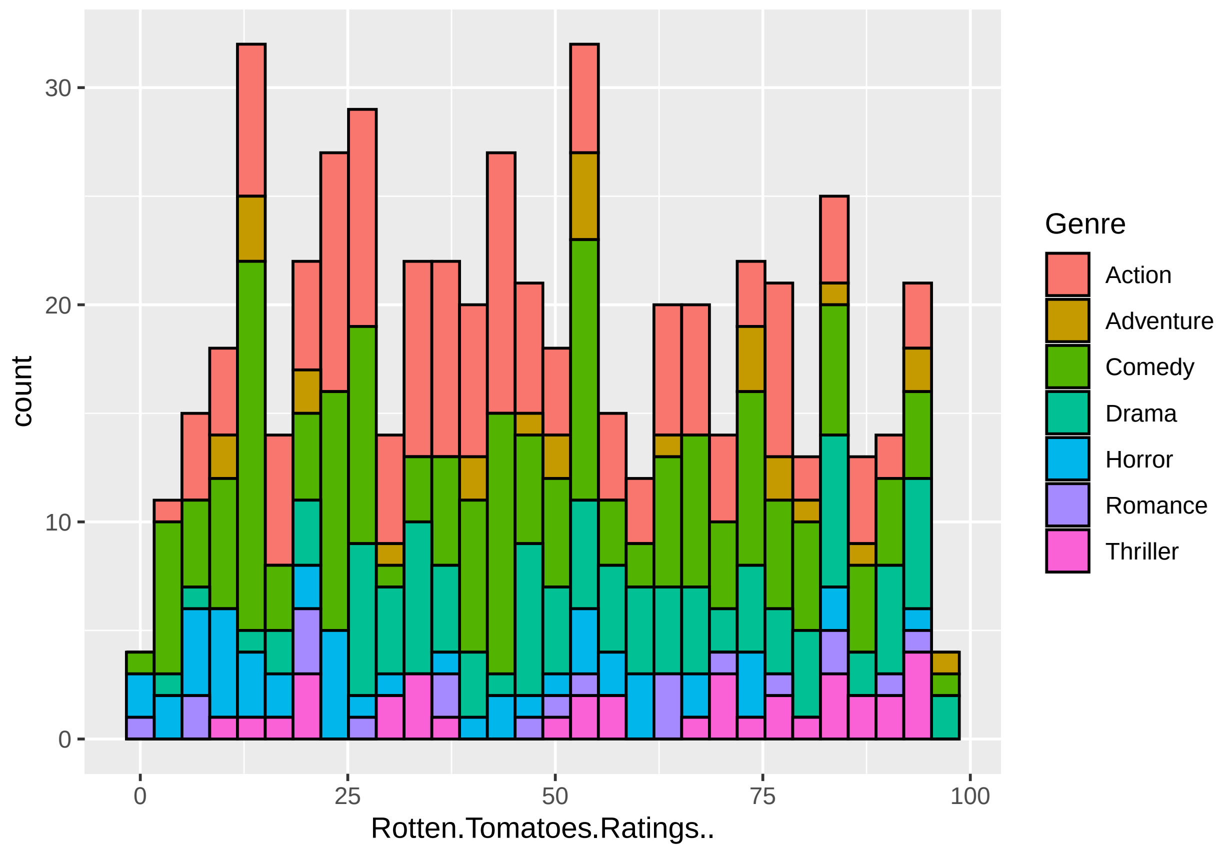
```
##     Genre Budget..million...
## 6 Romance                709
```

```
most_budget_genre
```

```
##     Genre Budget..million...
## 1 Action               13033
```

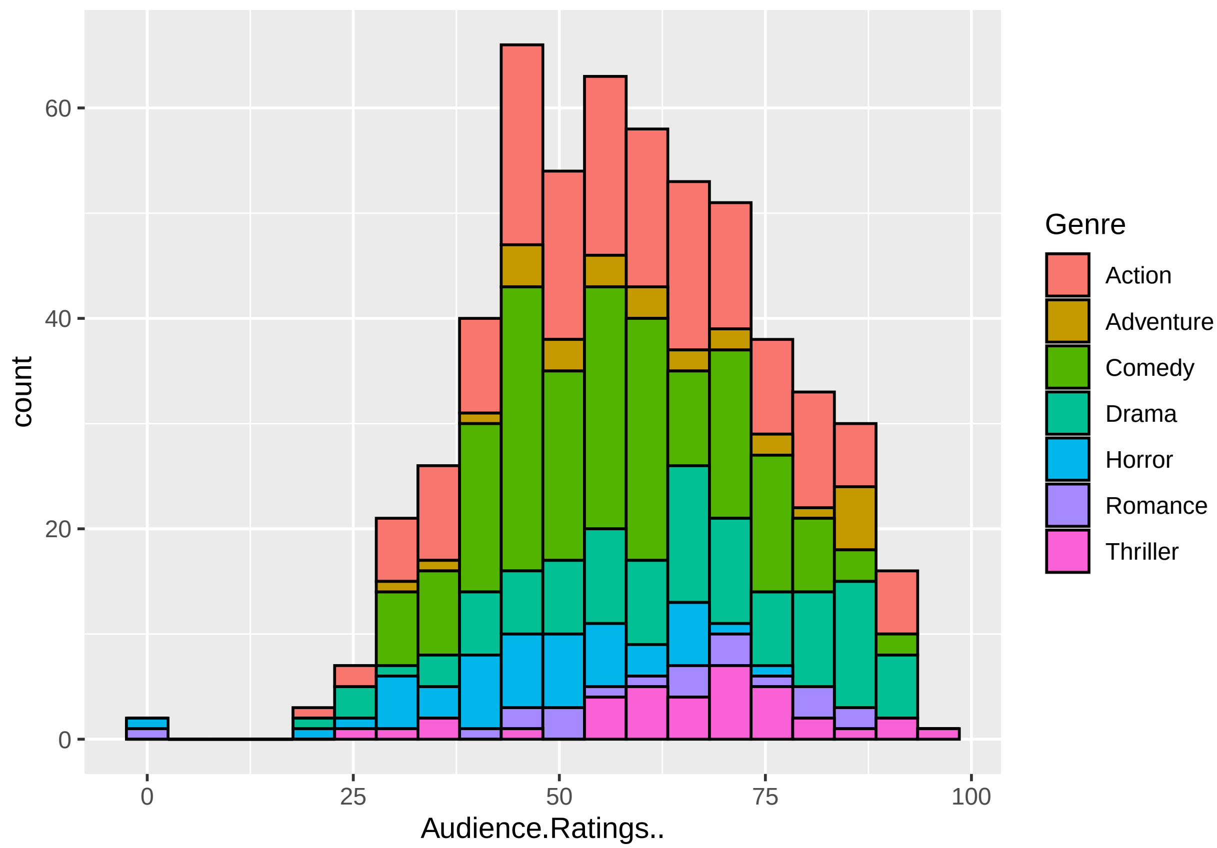
From the above graph, we can conclude that most of the movies have a budget of around a million to 40 million. We can also find out that it takes more budget to make an Action movies (From the representation, only the Action genre has spent around 300M budget) when compared to any other genre.

```r
ggplot(IMDB_data, aes(x = Rotten.Tomatoes.Ratings..)) +
geom_histogram(aes(fill = Genre),bins = 30, color = "black")
```



> The above histogram overlay represents the Critics Rating with respect to each and every genre.

```r
ggplot(IMDB_data, aes(x = Audience.Ratings..)) +
geom_histogram(aes(fill = Genre),bins = 20, color = "black")
```




```
#ggplot(data=IMDB_data,aes(x=Budget..million...)) + geom_histogram(binwidth=10,color='black',aes(fill =
```

The above histogram overlay shows the audience rating. From the representation, we can say that it is normally distributed.

#BOXPLOT

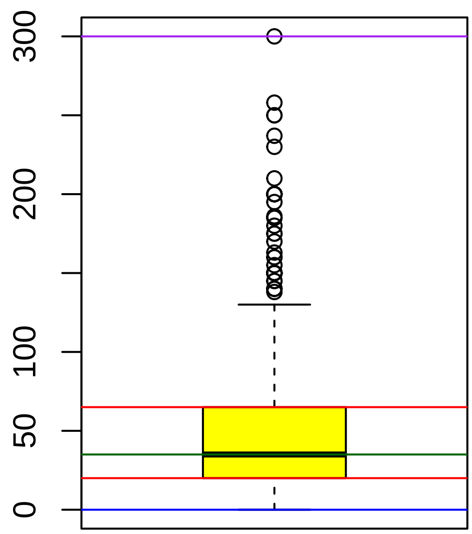
```
#BOXPLOT
#boxplot(IMDB_data$Year.of.release, horizontal = FALSE,col = "green")

par(mfrow=c(1,2))
boxplot(IMDB_data$Budget..million..., horizontal = FALSE,col = "yellow")
abline(h=min(IMDB_data$Budget..million...), col="Blue")
abline(h=max(IMDB_data$Budget..million...), col="purple")
abline(h=median(IMDB_data$Budget..million...), col="darkgreen")
abline(h=quantile(IMDB_data$Budget..million...,c(0.25,0.75)), col="red")

#BOXPLOT OF GENRE AND YEAR OF RELEASE
#ggplot(IMDB_data, aes(y = Year.of.release, x = Genre)) +
  #geom_boxplot()

#BOXPLOT OF GENRE AND CRITICS RATING
#ggplot(IMDB_data, aes(y = Rotten.Tomatoes.Ratings.., x = Genre)) +
#geom_boxplot()
```

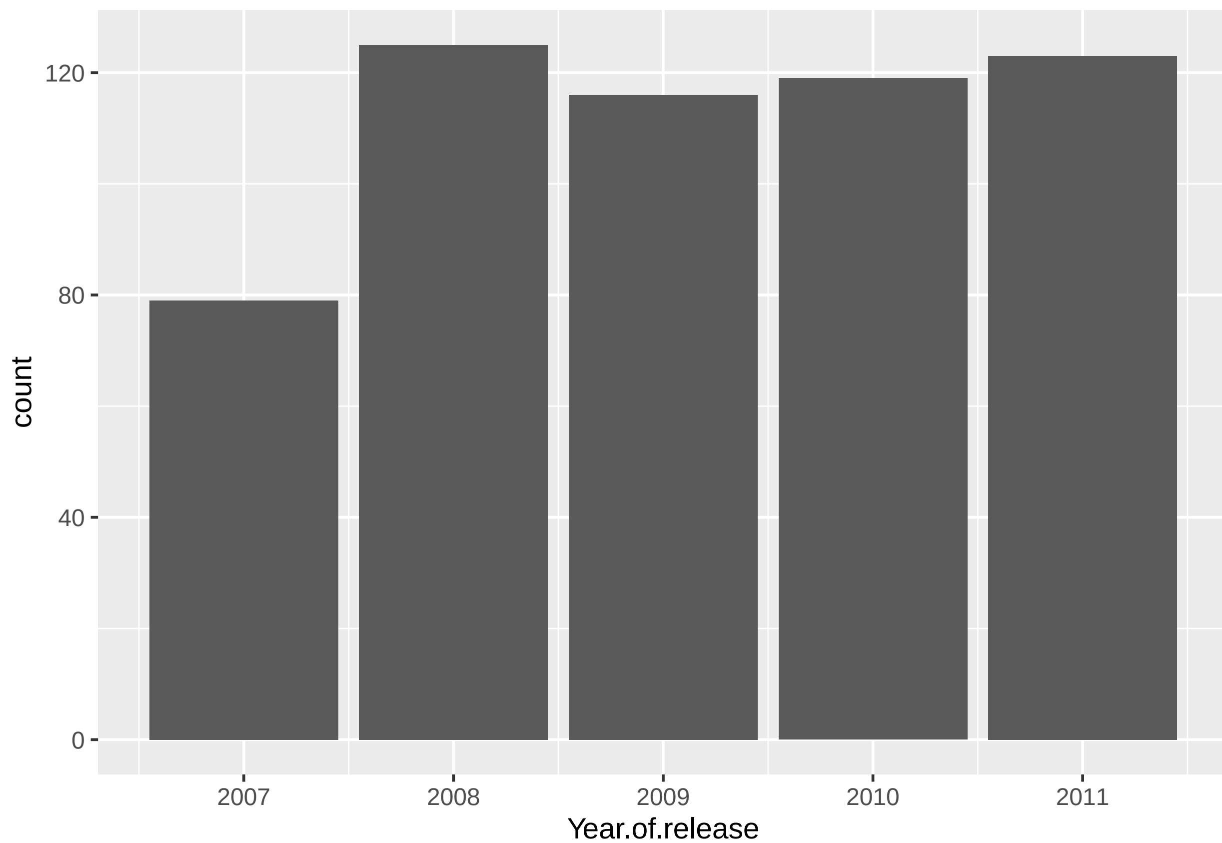
```r
#BOXPLOT OF GENRE AND AUDIENCE RATING
#ggplot(IMDB_data, aes(y = Audience.Ratings.., x = Genre)) +
#geom_boxplot()
```



> the graph above is boxplot overlay which shows the minimum, maximum and average money spent on movies.we can clearly see that minimum money spent on the movie is less than a million and maximum money spent on the movie is 300 million and overall average budget spent to make movies is around 40 million.

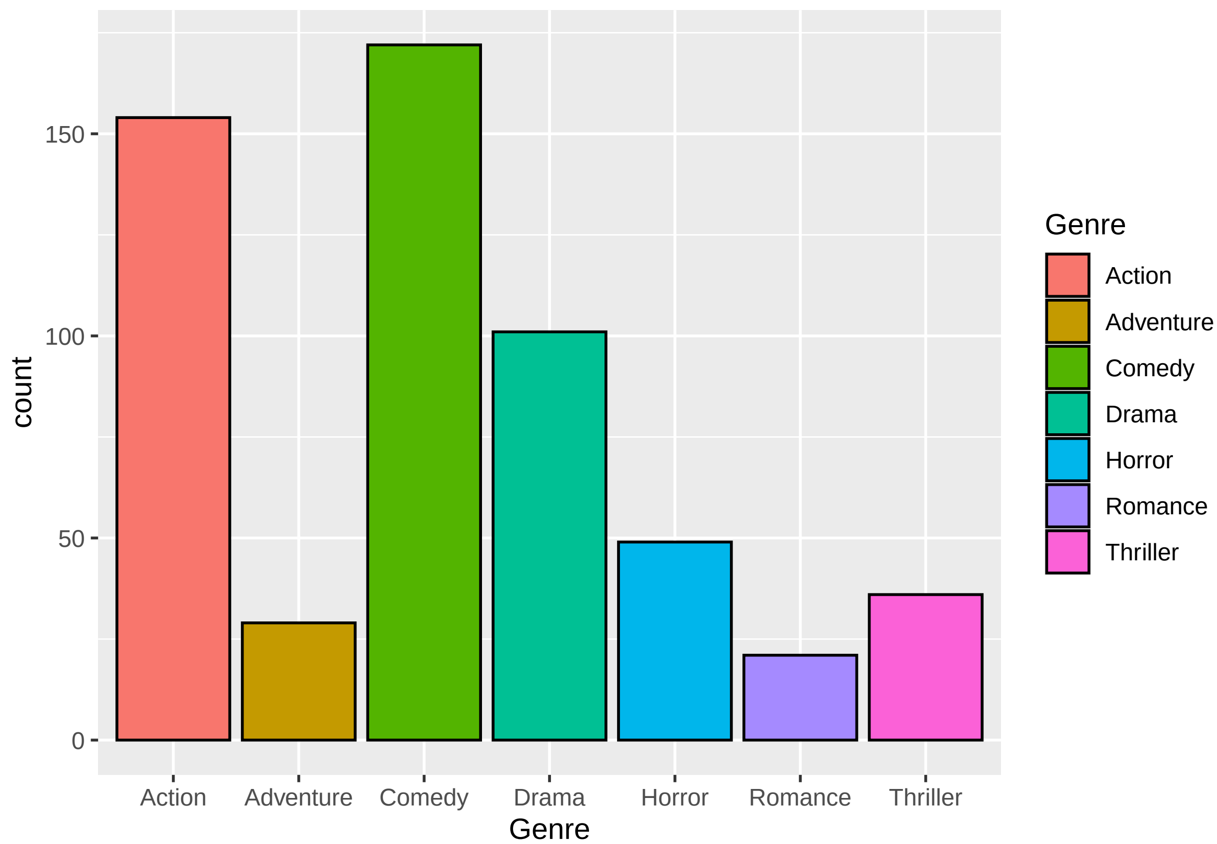
#BARPLOT

```r
#BARPLOT
ggplot(IMDB_data) +
geom_bar(mapping = aes(x = Year.of.release))
```



#BARPLOT WITH OVERLAY

```
#BARPLOT WITH OVERLAY
ggplot(IMDB_data, aes(x = Genre)) +
geom_bar(aes(fill = Genre), color = "black")
```




```
genre_counts <- table(IMDB_data$Genre)
genre_counts
```

```
##
##    Action Adventure    Comedy     Drama    Horror   Romance  Thriller
##       154        29       172       101        49        21        36
```

The above barplot shows the representation of the number of movies released per genre. Here, we can see that the "Comedy" genre has the most releases with around 170 movies between the years 2007 to 2011. THe genre "romance" has the least releases with less that 25 releases between the same timeframe.

Data Analysis

a Hypothesis Testing: Construct a hypothesis testing with null and alternative hypotheses. Use the appropriate test to get the conclusion.

#HYPOTHESIS TESTING

```
#Performing a Two-Sample T-test:
comedy <- filter(IMDB_data, Genre == "Comedy")
drama <- filter(IMDB_data, Genre == "Drama")
#t.test(comedy$Rotten.Tomatoes.Ratings.., drama$Rotten.Tomatoes.Ratings..)
```

```r
mean(comedy$Rotten.Tomatoes.Ratings..)
```

```
## [1] 44.9186
```

```r
mean(drama$Rotten.Tomatoes.Ratings..)
```

```
## [1] 56.47525
```

```r
# State hypotheses

#NULL HYPOTHESIS
H0 <- mean(comedy$Rotten.Tomatoes.Ratings..) == mean(drama$Rotten.Tomatoes.Ratings..)

#ALTERNATIVE HYPOTHESIS
H1 <- mean(comedy$Rotten.Tomatoes.Ratings..) != mean(drama$Rotten.Tomatoes.Ratings..)

# Perform t-test
t.test(comedy$Rotten.Tomatoes.Ratings.., drama$Rotten.Tomatoes.Ratings..)
```

```
##
##  Welch Two Sample t-test
##
## data:  comedy$Rotten.Tomatoes.Ratings.. and drama$Rotten.Tomatoes.Ratings..
## t = -3.6333, df = 224.73, p-value = 0.0003467
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -17.824511  -5.288774
## sample estimates:
## mean of x mean of y
##  44.91860  56.47525
```

> The p-value in this case is 0.0003467, which is less than 0.05. As a result, we reject the null hypothesis. The alternate theory is correct. The average rating difference between comedy and drama films is statistically significant, with dramas receiving higher ratings.

#SIMPLE LINEAR REGRESSION

```r
#SIMPLE LINEAR REGRESSION
lm_model <- lm(Audience.Ratings.. ~ Genre, data = IMDB_data)
summary(lm_model)
```

```
##
## Call:
## lm(formula = Audience.Ratings.. ~ Genre, data = IMDB_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -62.333 -10.721   0.348  12.279  36.593
##
## Coefficients:
```

```
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       58.721      1.307  44.939  < 2e-16 ***
## GenreAdventure     4.003      3.282   1.220   0.2231
## GenreComedy       -2.314      1.799  -1.286   0.1989
## GenreDrama         5.705      2.076   2.748   0.0062 **
## GenreHorror      -11.333      2.660  -4.261 2.39e-05 ***
## GenreRomance       3.613      3.772   0.958   0.3386
## GenreThriller      6.863      3.002   2.286   0.0226 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 16.22 on 555 degrees of freedom
## Multiple R-squared:  0.08139,    Adjusted R-squared:  0.07145
## F-statistic: 8.195 on 6 and 555 DF,  p-value: 1.644e-08
```

##To predict estimated audience ratings for genre_drama

```r
new <- data.frame(Genre="Drama")
predict(lm_model,new)
```

```
##        1
## 64.42574
```

To Predict Confidence interval

```r
predict(lm_model, new, interval = "confidence")
```

```
##        fit      lwr      upr
## 1 64.42574 61.25644 67.59505
```

#To Predict Prediction Interval

```r
predict(lm_model, new, interval = "prediction")
```

```
##        fit      lwr      upr
## 1 64.42574 32.41731 96.43418
```

#SPLIT THE Original DATASET INTO TWO FOR PROBABILITY

```r
DataSet <- sample(2, nrow(IMDB_data), replace = TRUE, prob=c(0.8, 0.2))
IMDB_Training <- IMDB_data[DataSet == 1,]
IMDB_Test <- IMDB_data[DataSet == 2,]
dim(IMDB_Training)
```

```
## [1] 448   6
```

```r
dim(IMDB_Test)
```

```
## [1] 114   6
```

#MULTIPLE LINEAR REGRESSION

```r
#MULTIPLE LINEAR REGRESSION
 IMDB_data$Genre <- as.factor(IMDB_data$Genre)
MLR <- lm(Audience.Ratings.. ~ Rotten.Tomatoes.Ratings.. + Budget..million... + Genre, data = IMDB_Trai
summary(MLR)
```

```
##
## Call:
## lm(formula = Audience.Ratings.. ~ Rotten.Tomatoes.Ratings.. +
##     Budget..million... + Genre, data = IMDB_Training)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -45.562  -7.531   0.321   7.785  29.237
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)               35.14470    1.83298  19.174  < 2e-16 ***
## Rotten.Tomatoes.Ratings..  0.39247    0.02250  17.440  < 2e-16 ***
## Budget..million...         0.07624    0.01356   5.622 3.37e-08 ***
## GenreAdventure             0.34650    2.57431   0.135   0.8930
## GenreComedy                1.31526    1.61945   0.812   0.4171
## GenreDrama                 5.04200    1.85745   2.714   0.0069 **
## GenreHorror               -5.33784    2.40877  -2.216   0.0272 *
## GenreRomance               8.50450    3.45719   2.460   0.0143 *
## GenreThriller              3.22572    2.68845   1.200   0.2308
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.11 on 439 degrees of freedom
## Multiple R-squared:  0.4907, Adjusted R-squared:  0.4814
## F-statistic: 52.87 on 8 and 439 DF,  p-value: < 2.2e-16
```

#Prediction

```r
pred <- predict(object=MLR,newdata = IMDB_Test)
summary(pred)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   31.33   49.20   56.38   57.94   66.71   83.26
```

##MULTIPLE LINEAR REGRESSION MAE and MSE

```r
library(MLmetrics)
MAE(y_pred = pred, y_true = IMDB_Test$Audience.Ratings..)
```

```
## [1] 10.01444
```

```r
MSE(y_pred = pred, y_true = IMDB_Test$Audience.Ratings..)
```

```
## [1] 166.4779
```

#Forward selection

```r
library(MASS)
 #To create a null model
i <- lm( Audience.Ratings..~ 1, data=IMDB_Training)
IMDB_data$Genre <- as.factor(IMDB_data$Genre)
#To create a full model
all <- lm(Audience.Ratings..~Rotten.Tomatoes.Ratings.. + Budget..million... + Genre + Year.of.release, d
#To perform forward stepwise regression
forward <- stepAIC(i, direction = 'forward' ,scope=formula(all))
```

```
## Start:  AIC=2529.59
## Audience.Ratings.. ~ 1
##
##                             Df Sum of Sq    RSS    AIC
## + Rotten.Tomatoes.Ratings..  1     53211  73128 2286.6
## + Genre                      6     12750 113589 2493.9
## + Budget..million...         1      4641 121698 2514.8
## + Year.of.release            1       711 125629 2529.1
## <none>                                   126339 2529.6
##
## Step:  AIC=2286.64
## Audience.Ratings.. ~ Rotten.Tomatoes.Ratings..
##
##                      Df Sum of Sq   RSS    AIC
## + Budget..million...  1    5085.3 68043 2256.3
## + Genre               6    4152.3 68976 2272.4
## + Year.of.release     1    1472.2 71656 2279.5
## <none>                            73128 2286.6
##
## Step:  AIC=2256.35
## Audience.Ratings.. ~ Rotten.Tomatoes.Ratings.. + Budget..million...
##
##                   Df Sum of Sq   RSS    AIC
## + Genre            6    3699.2 64344 2243.3
## + Year.of.release  1    1428.5 66614 2248.8
## <none>                         68043 2256.3
##
## Step:  AIC=2243.31
## Audience.Ratings.. ~ Rotten.Tomatoes.Ratings.. + Budget..million... +
##     Genre
##
##                   Df Sum of Sq   RSS    AIC
## + Year.of.release  1    1472.2 62871 2234.9
## <none>                         64344 2243.3
##
```

```
## Step:  AIC=2234.94
## Audience.Ratings.. ~ Rotten.Tomatoes.Ratings.. + Budget..million... +
##     Genre + Year.of.release
```

```r
#view results of forward stepwise regression
forward$anova
```

```
## Stepwise Model Path
## Analysis of Deviance Table
##
## Initial Model:
## Audience.Ratings.. ~ 1
##
## Final Model:
## Audience.Ratings.. ~ Rotten.Tomatoes.Ratings.. + Budget..million... +
##     Genre + Year.of.release
##
##
##                         Step Df  Deviance Resid. Df Resid. Dev      AIC
## 1                                               447  126339.43 2529.587
## 2 + Rotten.Tomatoes.Ratings..  1 53211.257       446   73128.17 2286.639
## 3       + Budget..million...  1  5085.329       445   68042.84 2256.349
## 4                    + Genre  6  3699.236       439   64343.61 2243.305
## 5          + Year.of.release  1  1472.227       438   62871.38 2234.936
```

```r
#To view full model
summary(forward)
```

```
##
## Call:
## lm(formula = Audience.Ratings.. ~ Rotten.Tomatoes.Ratings.. +
##     Budget..million... + Genre + Year.of.release, data = IMDB_Training)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -47.344  -7.355   0.249   8.195  27.682
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)               2761.50731  851.30777   3.244  0.00127 **
## Rotten.Tomatoes.Ratings..    0.39674    0.02231  17.783  < 2e-16 ***
## Budget..million...           0.07557    0.01342   5.630 3.22e-08 ***
## GenreAdventure              -0.34173    2.55664  -0.134  0.89373
## GenreComedy                  0.88491    1.60827   0.550  0.58245
## GenreDrama                   4.83202    1.83934   2.627  0.00892 **
## GenreHorror                 -5.34565    2.38377  -2.243  0.02543 *
## GenreRomance                 9.08888    3.42618   2.653  0.00827 **
## GenreThriller                2.86669    2.66291   1.077  0.28229
## Year.of.release             -1.35692    0.42370  -3.203  0.00146 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.98 on 438 degrees of freedom
```

```
## Multiple R-squared:  0.5024, Adjusted R-squared:  0.4921
## F-statistic: 49.13 on 9 and 438 DF,  p-value: < 2.2e-16
```

#Forward selection MAE and MSE

```r
pred_forward <-predict(object = forward, newdata = IMDB_Test)
MAE(y_pred = pred_forward, y_true = IMDB_Test$Audience.Ratings..)
```

```
## [1] 10.11228
```

```r
MSE(y_pred = pred_forward, y_true = IMDB_Test$Audience.Ratings..)
```

```
## [1] 166.5489
```

#Backward selection

```r
backward <- stepAIC (all, direction='backward')
```

```
## Start:  AIC=2234.94
## Audience.Ratings.. ~ Rotten.Tomatoes.Ratings.. + Budget..million... +
##     Genre + Year.of.release
##
##                             Df Sum of Sq    RSS    AIC
## <none>                                    62871 2234.9
## - Year.of.release            1      1472  64344 2243.3
## - Genre                      6      3743  66614 2248.8
## - Budget..million...         1      4550  67421 2264.2
## - Rotten.Tomatoes.Ratings..  1     45393 108265 2476.4
```

```r
backward$anova
```

```
## Stepwise Model Path
## Analysis of Deviance Table
##
## Initial Model:
## Audience.Ratings.. ~ Rotten.Tomatoes.Ratings.. + Budget..million... +
##     Genre + Year.of.release
##
## Final Model:
## Audience.Ratings.. ~ Rotten.Tomatoes.Ratings.. + Budget..million... +
##     Genre + Year.of.release
##
##
##   Step Df Deviance Resid. Df Resid. Dev      AIC
## 1                        438   62871.38 2234.936
```

```r
summary(backward)
```

```
##
## Call:
## lm(formula = Audience.Ratings.. ~ Rotten.Tomatoes.Ratings.. +
##     Budget..million... + Genre + Year.of.release, data = IMDB_Training)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -47.344  -7.355   0.249   8.195  27.682
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)               2761.50731  851.30777   3.244  0.00127 **
## Rotten.Tomatoes.Ratings..    0.39674    0.02231  17.783  < 2e-16 ***
## Budget..million...           0.07557    0.01342   5.630 3.22e-08 ***
## GenreAdventure              -0.34173    2.55664  -0.134  0.89373
## GenreComedy                  0.88491    1.60827   0.550  0.58245
## GenreDrama                   4.83202    1.83934   2.627  0.00892 **
## GenreHorror                 -5.34565    2.38377  -2.243  0.02543 *
## GenreRomance                 9.08888    3.42618   2.653  0.00827 **
## GenreThriller                2.86669    2.66291   1.077  0.28229
## Year.of.release             -1.35692    0.42370  -3.203  0.00146 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.98 on 438 degrees of freedom
## Multiple R-squared:  0.5024, Adjusted R-squared:  0.4921
## F-statistic: 49.13 on 9 and 438 DF,  p-value: < 2.2e-16
```

#Backward selection MAE and MSE

```r
pred_backward <-predict(object = backward, newdata = IMDB_Test)
MAE(y_pred = pred_backward, y_true = IMDB_Test$Audience.Ratings..)
```

```
## [1] 10.11228
```

```r
MSE(y_pred = pred_backward, y_true = IMDB_Test$Audience.Ratings..)
```

```
## [1] 166.5489
```

#Both direction

```r
both <- stepAIC (i, direction='both',scope = formula(all),trace = 0)
both$anova
```

```
## Stepwise Model Path
## Analysis of Deviance Table
##
## Initial Model:
## Audience.Ratings.. ~ 1
##
## Final Model:
```

```
## Audience.Ratings.. ~ Rotten.Tomatoes.Ratings.. + Budget..million... +
##     Genre + Year.of.release
##
##
##                         Step Df  Deviance Resid. Df Resid. Dev      AIC
## 1                                               447  126339.43 2529.587
## 2 + Rotten.Tomatoes.Ratings..  1 53211.257       446   73128.17 2286.639
## 3        + Budget..million...  1  5085.329       445   68042.84 2256.349
## 4                     + Genre  6  3699.236       439   64343.61 2243.305
## 5           + Year.of.release  1  1472.227       438   62871.38 2234.936
```

```r
summary(both)
```

```
##
## Call:
## lm(formula = Audience.Ratings.. ~ Rotten.Tomatoes.Ratings.. +
##     Budget..million... + Genre + Year.of.release, data = IMDB_Training)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -47.344  -7.355   0.249   8.195  27.682
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)               2761.50731  851.30777   3.244  0.00127 **
## Rotten.Tomatoes.Ratings..    0.39674    0.02231  17.783  < 2e-16 ***
## Budget..million...           0.07557    0.01342   5.630 3.22e-08 ***
## GenreAdventure              -0.34173    2.55664  -0.134  0.89373
## GenreComedy                  0.88491    1.60827   0.550  0.58245
## GenreDrama                   4.83202    1.83934   2.627  0.00892 **
## GenreHorror                 -5.34565    2.38377  -2.243  0.02543 *
## GenreRomance                 9.08888    3.42618   2.653  0.00827 **
## GenreThriller                2.86669    2.66291   1.077  0.28229
## Year.of.release             -1.35692    0.42370  -3.203  0.00146 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.98 on 438 degrees of freedom
## Multiple R-squared:  0.5024, Adjusted R-squared:  0.4921
## F-statistic: 49.13 on 9 and 438 DF,  p-value: < 2.2e-16
```

#Both direction MAE and MSE

```r
pred_both <-predict(object = both, newdata = IMDB_Test)
MAE(y_pred = pred_both, y_true = IMDB_Test$Audience.Ratings..)
```

```
## [1] 10.11228
```

```r
MSE(y_pred = pred_both, y_true = IMDB_Test$Audience.Ratings..)
```

```
## [1] 166.5489
```