EDA & DataPreprocessing

2024-10-12

Loading and Data Exploration

Loading libraries and reading the data Loading libraries

```
library(knitr)
library(ggplot2)
library(plyr)
library(dplyr)
library(tidyr)
library(corrplot)
library(caret)
library(gridExtra)
library(scales)
library(Rmisc)
library(ggrepel)
library(randomForest)
library(psych)
library(xgboost)
library(stringr)
library(GGally)
library(psych)
library(lubridate)
library(igraph)
library(ggraph)
library(reshape2)
```

Reading data

```
df <- read.csv("IMDB_data_Fall_2024.csv")
```

```
dim(df)
```

Data size, type, and structure

```
## [1] 1930   42
```

```
str(df)
```

```
## 'data.frame':    1930 obs. of  42 variables:
##  $ movie_title        : chr  "August: Osage County" "Radio" "Coach Carter" "The Possession" ...
##  $ movie_id           : int  2 12 15 20 22 23 26 31 38 39 ...
##  $ imdb_link          : chr  "http://www.imdb.com/title/tt1322269/?ref_=fn_tt_tt_1" "http://www.imdb
##  $ imdb_score         : num  7.3 6.9 7.2 5.9 7.6 6.4 7.1 8.1 7.1 6.5 ...
##  $ movie_budget       : int  25000000 35000000 30000000 14000000 8000000 20000000 22700000 25000000 4
##  $ release_day        : int  10 24 14 20 22 17 24 21 21 14 ...
##  $ release_month      : chr  "Jan" "Oct" "Jan" "Aug" ...
##  $ release_year       : int  2014 2003 2005 2012 1979 2006 1987 2007 1998 2007 ...
```

```
##  $ duration           : int  121 109 136 92 112 105 96 122 110 95 ...
##  $ language           : chr  "English" "English" "English" "English" ...
##  $ country            : chr  "USA" "USA" "USA" "USA" ...
##  $ maturity_rating    : chr  "R" "PG" "PG-13" "PG-13" ...
##  $ aspect_ratio       : num  2.35 1.85 2.35 2.35 1.85 1.85 1.85 2.35 2.35 1.85 ...
##  $ distributor        : chr  "The Weinstein Company" "Columbia Pictures Corporation" "Paramount Pictu
##  $ nb_news_articles   : int  2141 331 223 620 97 173 408 4135 1723 378 ...
##  $ director           : chr  "John Wells" "Michael Tollin" "Thomas Carter" "Ole Bornedal" ...
##  $ actor1             : chr  "Benedict Cumberbatch" "Alfre Woodard" "Channing Tatum" "Kyra Sedgwick"
##  $ actor1_star_meter  : int  259 2735 573 2047 102 573 12294 628 547 358742 ...
##  $ actor2             : chr  "Meryl Streep" "Riley Smith" "Rick Gonzalez" "Madison Davenport" ...
##  $ actor2_star_meter  : int  559 3915 4793 1769 5062 370 13732 2450 1054 3086 ...
##  $ actor3             : chr  "Julia Roberts" "Debra Winger" "Robert Ri'chard" "Natasha Calis" ...
##  $ actor3_star_meter  : int  513 1845 6729 11963 5451 3711 8419 3592 3001 642 ...
##  $ colour_film        : chr  "Color" "Color" "Color" "Color" ...
##  $ genres             : chr  "Drama" "Biography|Drama|Sport" "Drama|Sport" "Horror|Thriller" ...
##  $ nb_faces           : int  3 1 0 0 0 0 2 0 1 4 ...
##  $ plot_keywords      : chr  "based on play|incestuous relationship|pedophilia|secret|teenage daught
##  $ action             : int  0 0 0 0 0 0 0 0 1 0 ...
##  $ adventure          : int  0 0 0 0 0 0 1 0 0 0 ...
##  $ scifi              : int  0 0 0 0 0 0 1 0 0 0 ...
##  $ thriller           : int  0 0 0 1 0 0 0 1 0 0 ...
##  $ musical            : int  0 0 0 0 0 0 0 0 0 1 ...
##  $ romance            : int  0 0 0 0 0 1 0 0 0 1 ...
##  $ western            : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ sport              : int  0 1 1 0 0 0 0 0 0 0 ...
##  $ horror             : int  0 0 0 1 0 0 0 0 1 0 ...
##  $ drama              : int  1 1 1 0 1 0 0 1 0 0 ...
##  $ war                : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ animation          : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ crime              : int  0 0 0 0 1 0 0 1 0 0 ...
##  $ movie_meter_IMDBpro: int  4000 8556 3940 5452 4722 2446 2294 513 697 6854 ...
##  $ cinematographer    : chr  "Adriano Goldman" "Don Burgess" "Sharone Meir" "Dan Laustsen" ...
##  $ production_company : chr  "The Weinstein Company" "Revolution Studios" "Coach Carter" "Ghost Hous
```

```
summary(df)
```

```
##  movie_title           movie_id      imdb_link           imdb_score
##  Length:1930        Min.   :    2   Length:1930        Min.   :1.900
##  Class :character   1st Qu.: 2528   Class :character   1st Qu.:5.900
##  Mode  :character   Median : 5802   Mode  :character   Median :6.600
##                     Mean   : 7067                      Mean   :6.512
##                     3rd Qu.:10604                      3rd Qu.:7.300
##                     Max.   :21838                      Max.   :9.300
##   movie_budget       release_day    release_month       release_year
##  Min.   :  560000   Min.   : 1.00   Length:1930        Min.   :1936
##  1st Qu.: 8725000   1st Qu.: 9.00   Class :character   1st Qu.:1997
##  Median :18000000   Median :17.00   Mode  :character   Median :2004
##  Mean   :20973774   Mean   :15.95                      Mean   :2001
##  3rd Qu.:30000000   3rd Qu.:23.00                      3rd Qu.:2010
##  Max.   :55000000   Max.   :30.00                      Max.   :2018
##     duration       language           country          maturity_rating
##  Min.   : 37.0   Length:1930        Length:1930        Length:1930
##  1st Qu.: 96.0   Class :character   Class :character   Class :character
##  Median :106.0   Mode  :character   Mode  :character   Mode  :character
```

```
##  Mean   :109.7
##  3rd Qu.:118.0
##  Max.   :330.0
##   aspect_ratio   distributor        nb_news_articles    director
##  Min.   :1.180   Length:1930        Min.   :    0.0   Length:1930
##  1st Qu.:1.850   Class :character   1st Qu.:   78.0   Class :character
##  Median :2.350   Mode  :character   Median :  286.0   Mode  :character
##  Mean   :2.096                      Mean   :  770.6
##  3rd Qu.:2.350                      3rd Qu.:  845.5
##  Max.   :2.760                      Max.   :60620.0
##     actor1          actor1_star_meter    actor2          actor2_star_meter
##  Length:1930        Min.   :      9   Length:1930        Min.   :      3
##  Class :character   1st Qu.:    505   Class :character   1st Qu.:   1895
##  Mode  :character   Median :   1888   Mode  :character   Median :   3986
##                     Mean   :  21190                      Mean   :  17114
##                     3rd Qu.:   4665                      3rd Qu.:   7667
##                     Max.   :8342201                      Max.   :5529461
##     actor3          actor3_star_meter colour_film          genres
##  Length:1930        Min.   :      8   Length:1930        Length:1930
##  Class :character   1st Qu.:   3075   Class :character   Class :character
##  Mode  :character   Median :   5856   Mode  :character   Mode  :character
##                     Mean   :  35469
##                     3rd Qu.:  12250
##                     Max.   :6292982
##     nb_faces     plot_keywords          action         adventure
##  Min.   : 0.00   Length:1930        Min.   :0.0000   Min.   :0.0000
##  1st Qu.: 0.00   Class :character   1st Qu.:0.0000   1st Qu.:0.0000
##  Median : 1.00   Mode  :character   Median :0.0000   Median :0.0000
##  Mean   : 1.44                      Mean   :0.2005   Mean   :0.1264
##  3rd Qu.: 2.00                      3rd Qu.:0.0000   3rd Qu.:0.0000
##  Max.   :31.00                      Max.   :1.0000   Max.   :1.0000
##      scifi           thriller         musical           romance
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.00000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.:0.0000
##  Median :0.0000   Median :0.0000   Median :0.00000   Median :0.0000
##  Mean   :0.1083   Mean   :0.2979   Mean   :0.07047   Mean   :0.2451
##  3rd Qu.:0.0000   3rd Qu.:1.0000   3rd Qu.:0.00000   3rd Qu.:0.0000
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.00000   Max.   :1.0000
##     western            sport             horror          drama
##  Min.   :0.00000   Min.   :0.00000   Min.   :0.000   Min.   :0.0000
##  1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.000   1st Qu.:0.0000
##  Median :0.00000   Median :0.00000   Median :0.000   Median :1.0000
##  Mean   :0.01762   Mean   :0.04819   Mean   :0.113   Mean   :0.5492
##  3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.000   3rd Qu.:1.0000
##  Max.   :1.00000   Max.   :1.00000   Max.   :1.000   Max.   :1.0000
##       war            animation           crime        movie_meter_IMDBpro
##  Min.   :0.00000   Min.   :0.00000   Min.   :0.0000   Min.   :    71
##  1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.0000   1st Qu.:  2836
##  Median :0.00000   Median :0.00000   Median :0.0000   Median :  5406
##  Mean   :0.03627   Mean   :0.01036   Mean   :0.2161   Mean   : 11612
##  3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.0000   3rd Qu.: 10198
##  Max.   :1.00000   Max.   :1.00000   Max.   :1.0000   Max.   :849550
##  cinematographer    production_company
##  Length:1930        Length:1930
```

```
##  Class :character   Class :character
##  Mode  :character   Mode  :character
##
##
##
```

There are 41 predictors and 1 target variable (imdb_score). All of them match their expected data types, so there's no need to modify the datatype for any feature.

drop some columns

First we drop some columns that we won't use in our model at all. After getting rid of the titles, urls, and plot_keywords, there're 38 predictors and the target variable imdb_score.

```
df$movie_title <- NULL
df$imdb_link <- NULL
df$plot_keywords <- NULL

dim(df)
```

```
## [1] 1930   39
```

Business questions to explore through EDA Before jumping into EDA analysis, we've outlined some business questions to guide us through the process. By answering these questions, we hope to uncover insights that will inform our feature selection and interactions, potentially improving the performance of our predictive model.

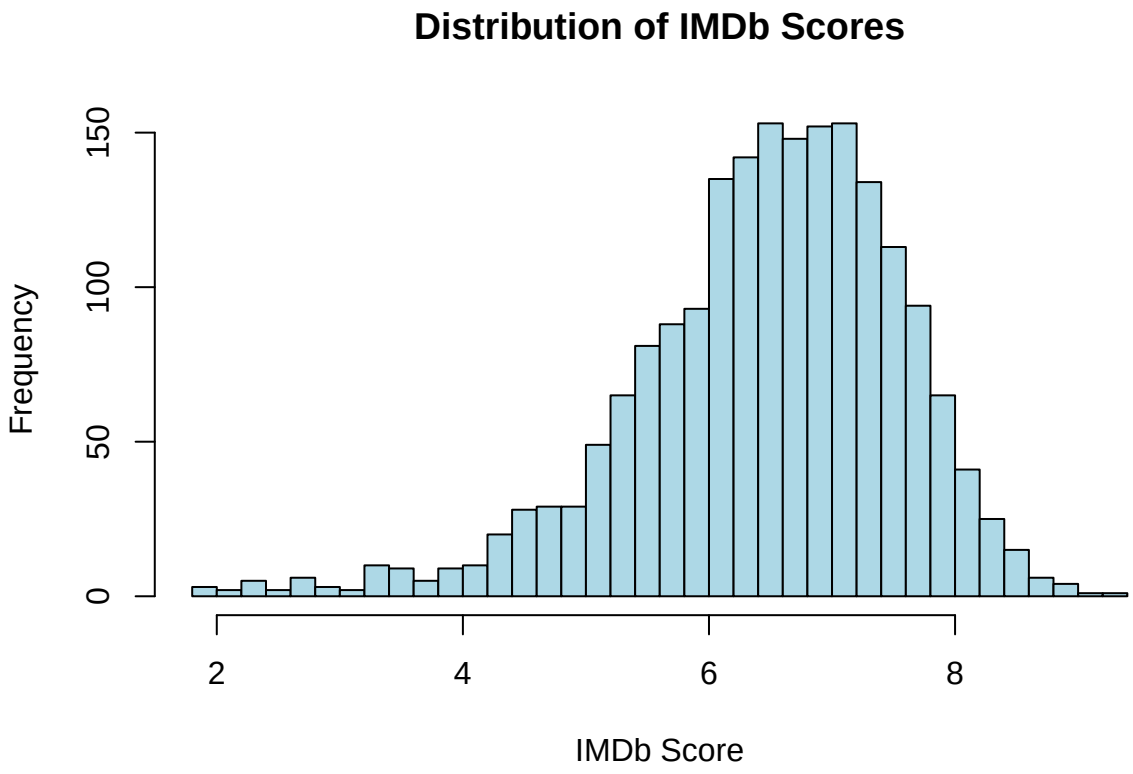
- What predictors have high correlation with IMDb scores? Is there any possible collinearity between predictors?
- Does the release timing affect IMDb scores?
- How does the combination of multiple genres in a movie affect its IMDb score?
- How does genre popularity differ across regions?
- How does the presence of top 10 personnel impact IMDb scores?
- Look into movies associated with the top 10 personnel for each predictor. Do different personnel associate with certain countries? What types of movie genres tend to have the highest average IMDb scores among these top personnel?

We will explore these questions in our upcoming analysis. By doing so, we aim to gain deeper insights into the relationships between features. This will help us determine which interactions between variables are worth including in our model to improve model performance.

High-level veiw of dataset To get a feel for the dataset, first we look at shape of the target.

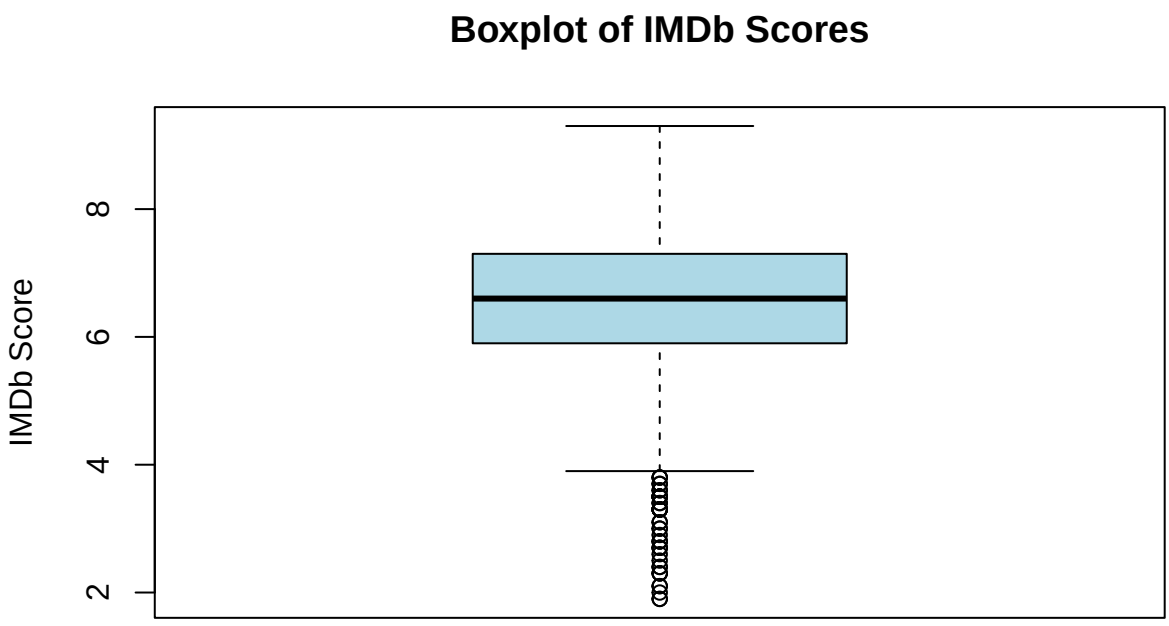
Target variable: imdb_score histogram

```
hist(df$imdb_score, main = "Distribution of IMDb Scores", xlab = "IMDb Score", col = "lightblue", breaks
```



boxplot

```
boxplot(df$imdb_score, main = "Boxplot of IMDb Scores", ylab = "IMDb Score", col = "lightblue")
```

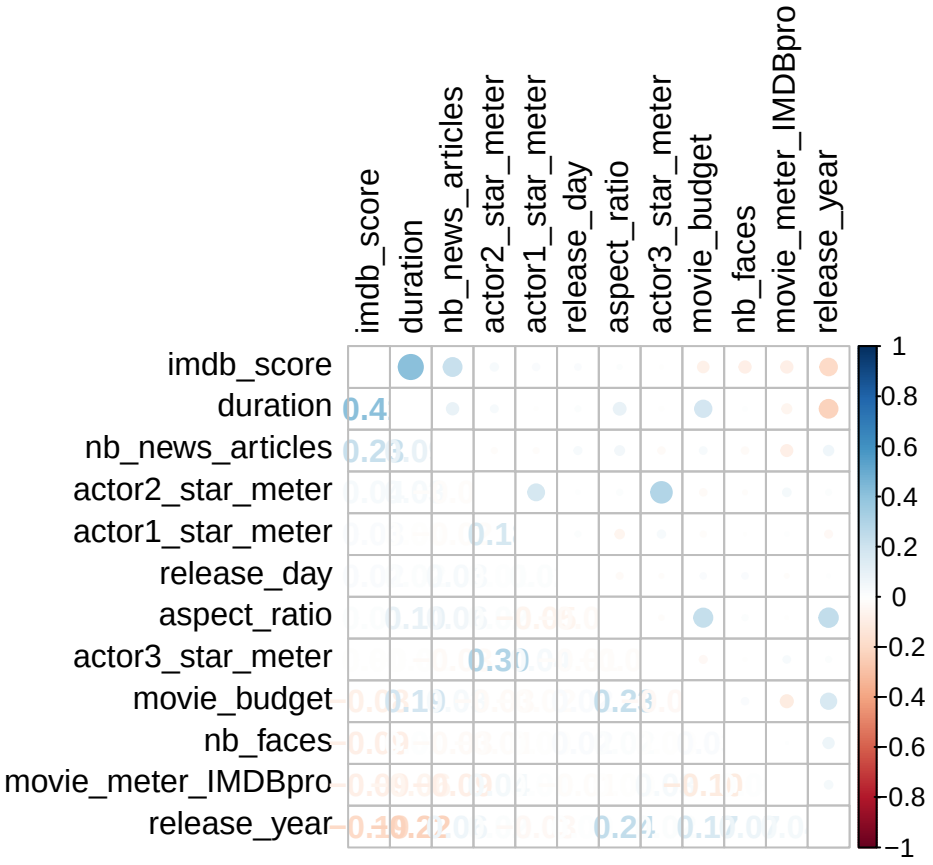


The IMDb Scores are slightly left-skewed, but it's not severe enough to be a major concern. This distribution makes sense because people usually rate good movies higher while reserving lower ratings for particularly bad experiences, which are less common. Since many viewers are more inclined to watch movies they expect to enjoy, this could lead to higher overall ratings.

Numerical predictors For numerical variables, we first looking at their correlations and distribution.

Correlations with imdb_score

```
numerical_cols <- df[, c("imdb_score", "movie_budget", "release_day", "release_year", "duration",
                         "aspect_ratio", "nb_news_articles", "actor1_star_meter", "actor2_star_meter",
                         "actor3_star_meter", "nb_faces", "movie_meter_IMDBpro")]

cor_matrix <- cor(numerical_cols, use = "pairwise.complete.obs")
# sort on decreasing correlations with imdb_score
cor_sorted <- cor_matrix[order(-cor_matrix[, "imdb_score"]), order(-cor_matrix["imdb_score", ])]
# plot it
corrplot.mixed(cor_sorted, tl.col = "black", tl.pos = "lt")
```



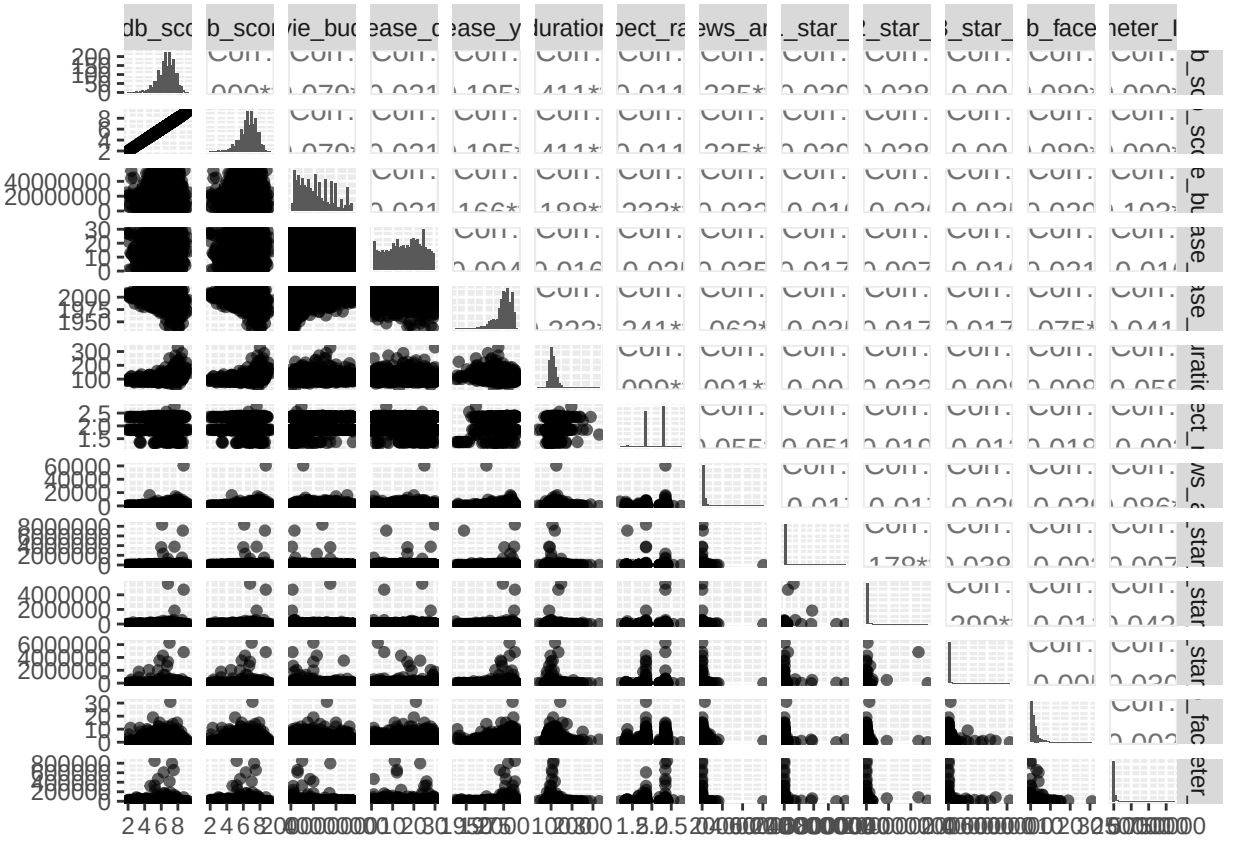
Scatter plot matrix

```
options(scipen=999)

ggpairs(df[c("imdb_score", colnames(numerical_cols))],
        upper = list(continuous = wrap("cor", size = 4)),
        lower = list(continuous = wrap("points", alpha = 0.6)),
        diag = list(continuous = "barDiag"),
        progress = FALSE, message = FALSE, warnings = FALSE)
```

```
## Warning in warn_if_args_exist(list(...)): Extra arguments: 'message',
## 'warnings' are being ignored.  If these are meant to be aesthetics, submit them
## using the 'mapping' variable within ggpairs with ggplot2::aes or
## ggplot2::aes_string.
```

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

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



There is a moderate positive correlation between movie duration and IMDb score. This suggests that longer movies tend to have slightly higher ratings on average. Overall, the correlations shown above are quite low, no matter it's between target or predictors themselves.

Note: Since Pearson's correlation only captures linear relationships, non-linear interactions between certain variables and IMDb scores might exist.

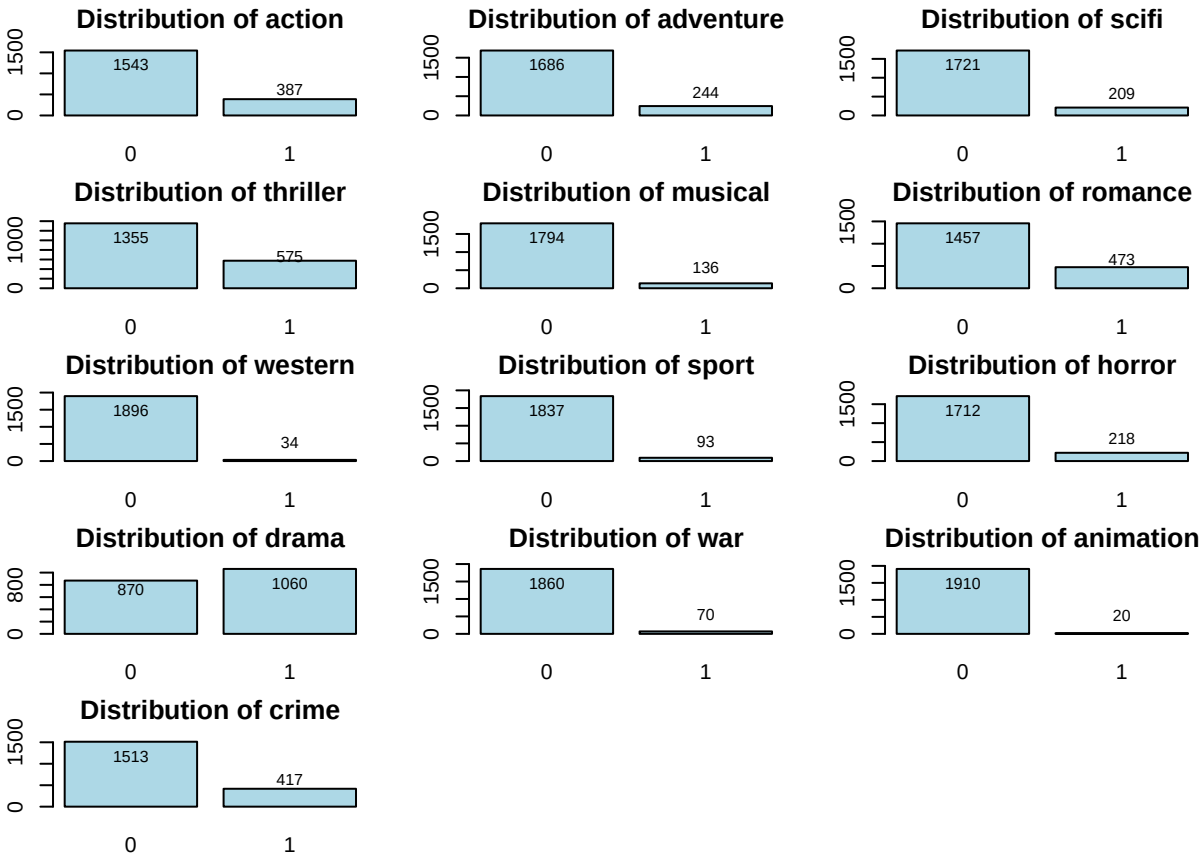
Categorical predictors For categorical variables, we first look at the dummy variables to see their distributions and correlations. Since some categorical variables in string format may require further conversion and can be more complex, we'll address those in the later sections and explore them in more detail then.

Dummy variables - genres

1. bar chart

There're imbalances presenting in some predictors.

```r
par(mfrow = c(5, 3), mar = c(2, 2, 2, 2))

genre_columns <- colnames(df)[grepl("action|adventure|scifi|thriller|musical|romance|western|sport|horro

for (genre in genre_columns) {

  genre_counts <- table(df[[genre]])

  bar_heights <- barplot(genre_counts, main = paste("Distribution of", genre), col = "lightblue", ylim =

  text(x = bar_heights, y = genre_counts / 2, labels = genre_counts, cex = 0.8, col = "black", pos = 3)

}

par(mfrow = c(1, 1), mar = c(5, 4, 4, 2) + 0.1)
```

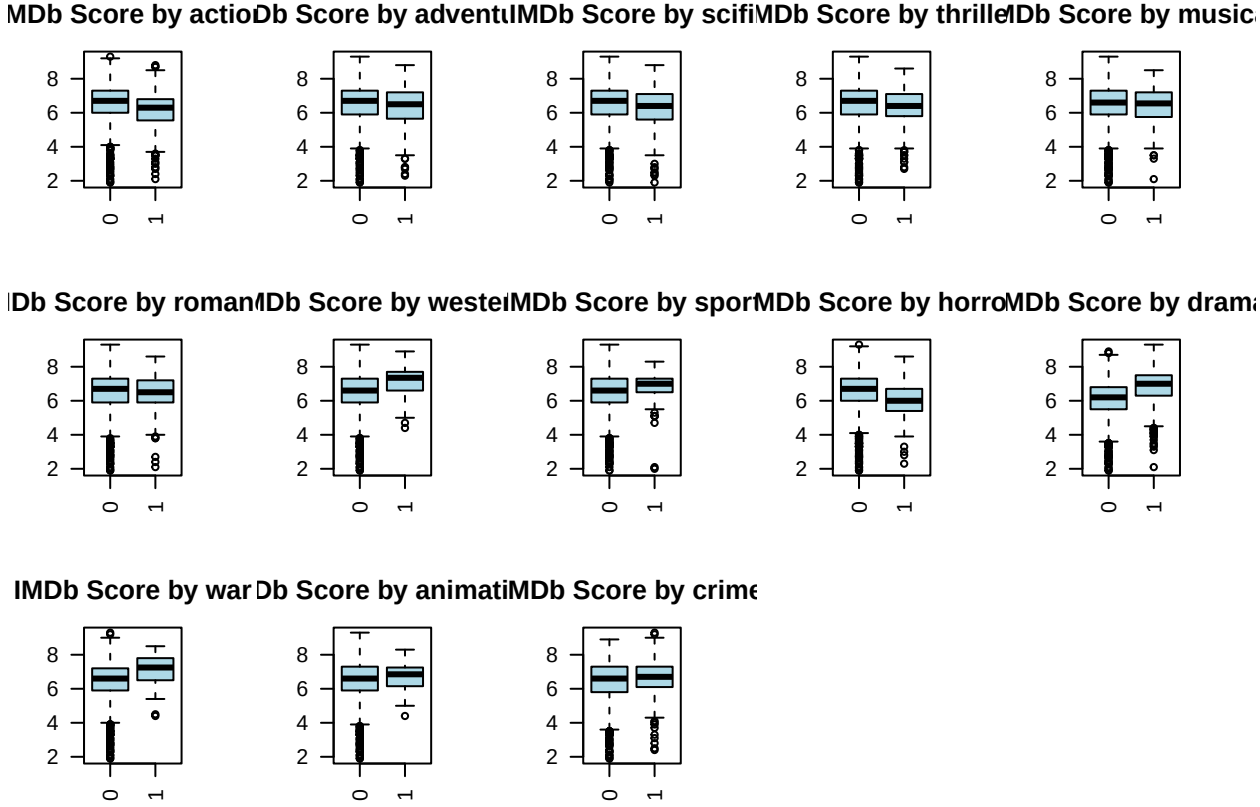


2. Boxplots

Some genres show insignificant differences in terms of ratings, indicating that we might consider dropping them from our analysis.

```r
par(mfrow = c(3, 5), mar = c(3, 3, 3, 3))

for (genre in genre_columns) {
  boxplot(df$imdb_score ~ df[[genre]],
          main = paste("IMDb Score by", genre),
          ylab = "IMDb Score",
```

```
          xlab = genre,
          col = "lightblue",
          las = 2)  # las = 2 makes the axis labels perpendicular to the axis
}

par(mfrow = c(1, 1), mar = c(5, 4, 4, 2) + 0.1)
```



3. Point Biserial Correlation

*** specifically suited for binary-continuous relationships, it measures the strength and direction of the association between a binary variable and a continuous variable.

Drama has the strongest positive correlation with IMDb score, suggesting drama movies tend to be rated higher than average.

```
for (genre in genre_columns) {
  correlation <- biserial(df$imdb_score, df[[genre]])
  print(paste(genre, ":", correlation))
}
```

```
##
## [1] "action : -0.227064322901513"
##
## [1] "adventure : -0.107108853641586"
##
## [1] "scifi : -0.156732577464518"
##
## [1] "thriller : -0.105581869620907"
##
## [1] "musical : -0.0429522776395504"
```

```
##
## [1] "romance : -0.020355293686273"
##
## [1] "western : 0.198305451004262"
##
## [1] "sport : 0.11759999228579"
##
## [1] "horror : -0.274239811445477"
##
## [1] "drama : 0.42494608034447"
##
## [1] "war : 0.255005038531814"
##
## [1] "animation : 0.0610537711481328"
##
## [1] "crime : 0.0862764275417247"
```

Missing data, feature splitting, label encoding, and factorizing variables

Missing values *** R uses NA for missing values, while Python uses NaN for missing values. If the dataset contains other non-standard missing values (" ","NaN", "None"), R might not recognize them as NA.

```
colnames(df)[colSums(is.na(df)) > 0]
```

```
## character(0)
```

Language is the only feature having missing data.

```
for (col in colnames(df)){
  na_checking = df[df[col] == "" | df[col] == "NaN" | df[col] == "None", ]
  if (nrow(na_checking) > 0) {
    print(col)
  }
}
```

```
## [1] "language"
```

There are only two missing values in the language variable, and we'll address that in the next part.

```
df[df$language == "" | df$language == "NaN" | df$language == "None", ]
```

```
##      movie_id imdb_score movie_budget release_day release_month release_year
## 868      5032        8.5      4000000           1           Feb         2012
## 1604    12945        7.4     12500000          12           Feb         1982
##      duration language country maturity_rating aspect_ratio
## 868       102     None     USA           PG-13         2.35
## 1604      100     None  Canada               R         2.35
##                   distributor nb_news_articles            director
## 868              ICM Partners              112          Ron Fricke
## 1604 Films sans FrontiÌÄå¬res               31 Jean-Jacques Annaud
##                       actor1 actor1_star_meter                       actor2
## 868  Collin Alfredo St. Dic           1504888 Balinese Tari Legong Dancers
## 1604         Rae Dawn Chong              1153               Everett McGill
##      actor2_star_meter              actor3 actor3_star_meter colour_film
## 868             546238 Puti Sri Candra Dewi            582819       Color
## 1604              6876        Gary Schwartz             30057       Color
##                       genres nb_faces action adventure scifi thriller musical
## 868        Documentary|Music        0      0         0     0        0       1
```

```
## 1604 Adventure|Drama|History        0      0         1     0        0       0
##      romance western sport horror drama war animation crime movie_meter_IMDBpro
## 868        0       0     0      0     0   0         0     0                9155
## 1604       0       0     0      0     1   0         0     0                6820
##      cinematographer                    production_company
## 868       Ron Fricke                      Bali Film Center
## 1604 Claude Agostini International Cinema Corporation (ICC)
```

EDA - categorical predictors This section focuses on the essential task of cleaning and addressing any imperfections for later analysis, specifically for categorical predictors. We review all the features, addressing them one by one. This process includes handling missing values, converting variable types, encoding, and performing feature extractions.

Language Because the test dataset has no variation in this predictor (all movies are in English), it doesn't provide useful information for predicting IMDb scores. So, we decide to drop it.

(skip the following analysis)

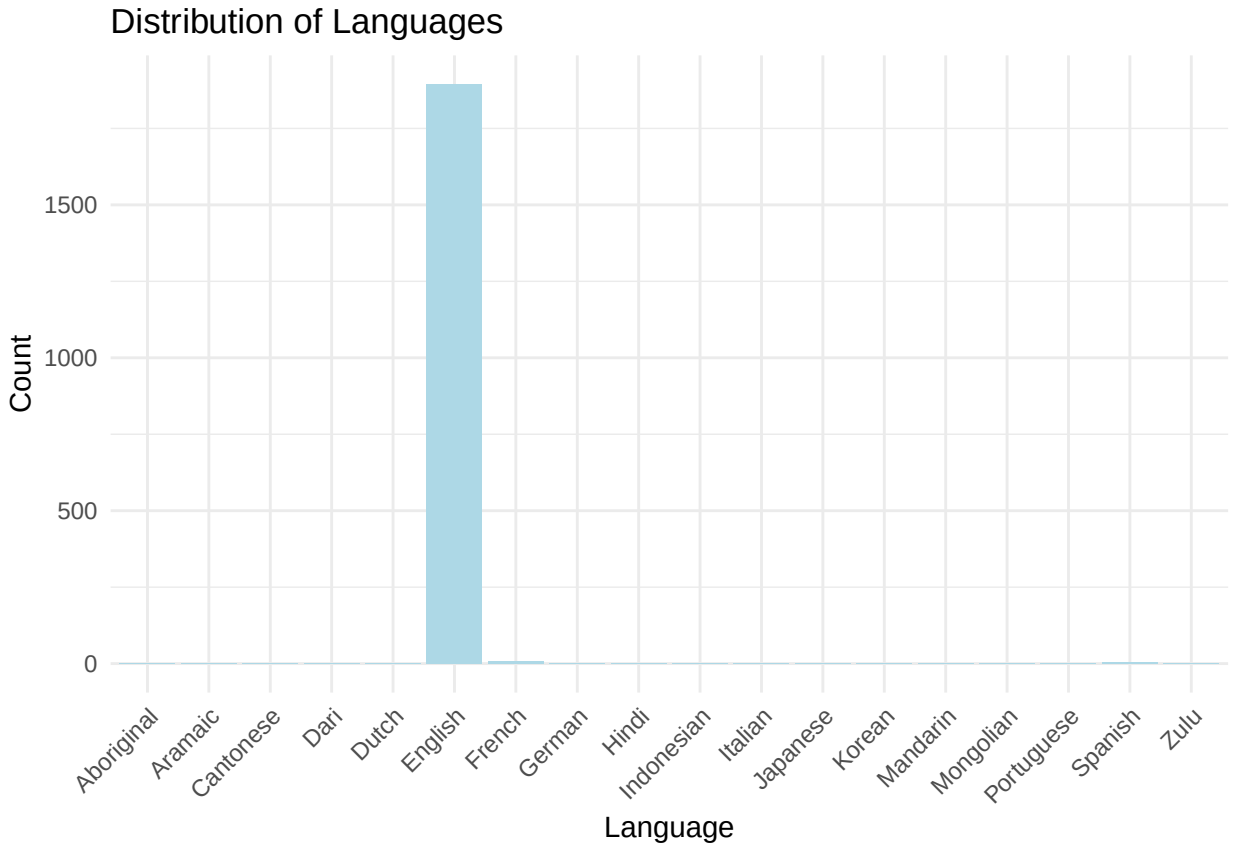
Since the two missing movies are from the North America region, it's reasonable to impute their language as English.

```
df$language[df$language == "None"] <- "English"
nrow(df[df$language == "" | df$language == "NaN" | df$language == "None", ]) # 0
```

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

bar plot

```
ggplot(df, aes(x = language)) +
  geom_bar(fill = "lightblue") +
  labs(title = "Distribution of Languages", x = "Language", y = "Count") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

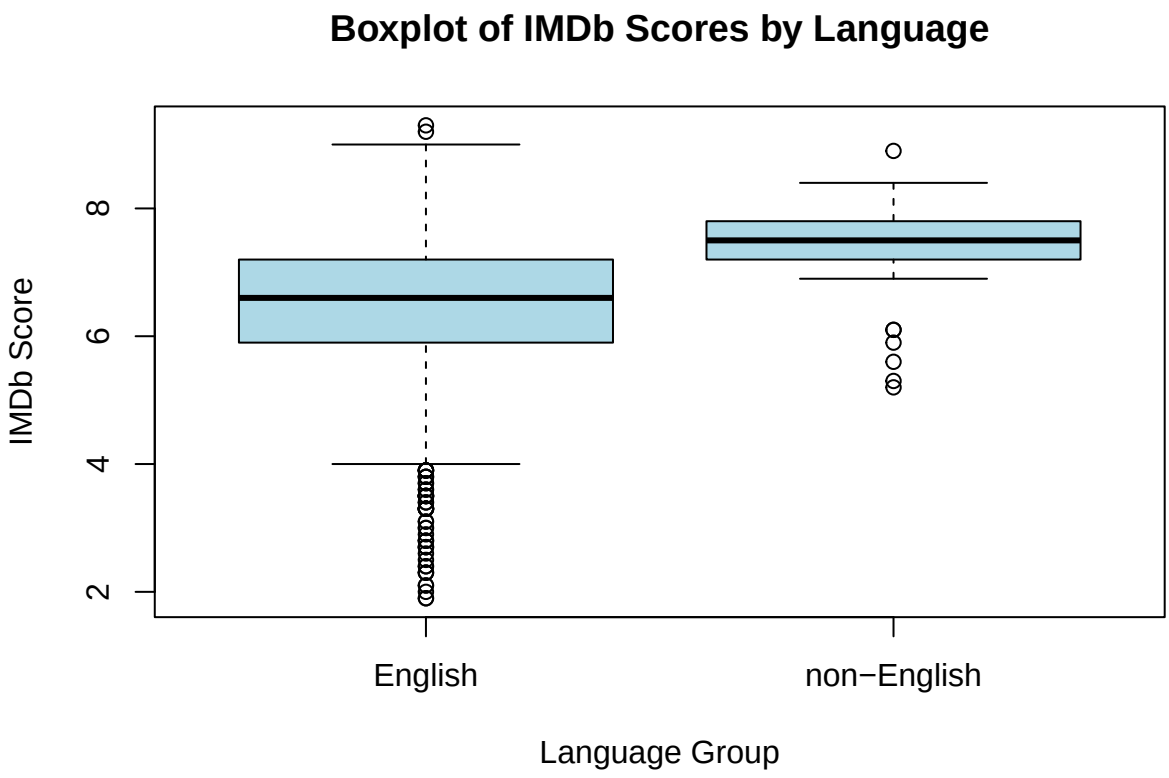


Since this column has an imbalanced distribution, with many granular languages that have little impact on the model, it would be better to group all non-English languages into a 'Non-English' category. This approach can simplify the model while still preserving important information about the language factor by keeping only 'English' and 'Non-English'.

```
df$language[df$language != "English"] <- "non-English"
```

box plot

```
boxplot(imdb_score ~ language, data = df, main = "Boxplot of IMDb Scores by Language",
        xlab = "Language Group", ylab = "IMDb Score", col = "lightblue")
```




```
table(df$language)
```

```
##
##     English non-English
##        1894          36
```

The boxplot suggests a significant difference in IMDb scores between English and non-English movies. A pairwise t-test can formally determine whether this new classification is effective by assessing the impact of language on IMDb scores. If we reject the null hypothesis, it indicates that language could be a significant predictor of IMDb scores.

pairwise t-test

(Conservative ranking: Scheffé > Bonferroni > Tukey > Fisher's LSD)

We choose a middle conservative t-test approach (Bonferroni), to assess the robustness of the newly reclassified levels.

```
pairwise_results <- pairwise.t.test(df$imdb_score, df$language, p.adjust.method = "bonferroni")
print(pairwise_results)
```

```
##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  df$imdb_score and df$language
##
##             English
## non-English 0.00001
##
## P value adjustment method: bonferroni
```

This result indicates that there is a statistically significant difference in the mean IMDb scores between English and non-English movies, concluding that the language classification impacts the IMDb score.

Genres Split the string format genres and remove duplicate ones with dummy.

Steps:

1. Splitting the genres and storing in a new column as a list
2. Identify unique genres from "Genres" after splitting
3. Filter out genres that aren't represented as columns
4. Create new dummified columns for those extracted, unique genres
5. Assign 1 / 0 to these new dummy variables corresponding to extracted genres list in each movie

Splitting the genres and storing in a new column as a list

```
df$genre_list <- strsplit(as.character(df$genres), "\\|")
head(df$genre_list)
```

```
## [[1]]
## [1] "Drama"
##
## [[2]]
## [1] "Biography" "Drama"     "Sport"
##
## [[3]]
## [1] "Drama" "Sport"
##
## [[4]]
## [1] "Horror"   "Thriller"
##
## [[5]]
## [1] "Biography" "Crime"     "Drama"
##
## [[6]]
## [1] "Comedy"  "Romance"
```

Identify unique genres from "Genres" after splitting

```
unique_genres <- tolower(unique(unlist(df$genre_list)))
unique_genres
```

```
##  [1] "drama"       "biography"   "sport"       "horror"      "thriller"
##  [6] "crime"       "comedy"      "romance"     "adventure"   "sci-fi"
## [11] "action"      "music"       "fantasy"     "history"     "mystery"
## [16] "family"      "war"         "musical"     "western"     "animation"
## [21] "documentary"
```

Filter out genres that aren't represented as columns

```
existing_genre_columns <- colnames(df)[colnames(df) %in% unique_genres]
existing_genre_columns
```

```
##  [1] "action"    "adventure" "thriller"  "musical"   "romance"   "western"
##  [7] "sport"     "horror"    "drama"     "war"       "animation" "crime"
```

Create new dummified columns for those extracted unique genres

&

Assign 1 / 0 to these new dummy variables corresponding to extracted genres list in each movie

```r
for (genre in unique_genres) {

  if (!(genre %in% existing_genre_columns)) {

    df[[genre]] <- sapply(df$genre_list, function(x) ifelse(str_to_title(genre) %in% x, 1, 0))

  }
}

df$genre_list <- NULL

head(df)
```

```
##   movie_id imdb_score movie_budget release_day release_month release_year
## 1        2        7.3     25000000          10           Jan         2014
## 2       12        6.9     35000000          24           Oct         2003
## 3       15        7.2     30000000          14           Jan         2005
## 4       20        5.9     14000000          20           Aug         2012
## 5       22        7.6      8000000          22           Jun         1979
## 6       23        6.4     20000000          17           Mar         2006
##   duration language country maturity_rating aspect_ratio
## 1      121  English     USA               R         2.35
## 2      109  English     USA              PG         1.85
## 3      136  English     USA           PG-13         2.35
## 4       92  English     USA           PG-13         2.35
## 5      112  English     USA              PG         1.85
## 6      105  English     USA           PG-13         1.85
##                     distributor nb_news_articles       director
## 1          The Weinstein Company             2141     John Wells
## 2 Columbia Pictures Corporation              331 Michael Tollin
## 3            Paramount Pictures              223  Thomas Carter
## 4                     Lionsgate              620   Ole Bornedal
## 5            Paramount Pictures               97     Don Siegel
## 6       Lakeshore International              173   Andy Fickman
##                 actor1 actor1_star_meter               actor2
## 1 Benedict Cumberbatch               259          Meryl Streep
## 2        Alfre Woodard              2735           Riley Smith
## 3       Channing Tatum               573         Rick Gonzalez
## 4        Kyra Sedgwick              2047      Madison Davenport
## 5       Clint Eastwood               102       Patrick McGoohan
## 6       Channing Tatum               573 Alexandra Breckenridge
##   actor2_star_meter          actor3 actor3_star_meter colour_film
## 1               559   Julia Roberts               513       Color
## 2              3915    Debra Winger              1845       Color
## 3              4793 Robert Ri'chard              6729       Color
## 4              1769   Natasha Calis             11963       Color
## 5              5062       Fred Ward              5451       Color
## 6               370    Laura Ramsey              3711       Color
##                  genres nb_faces action adventure scifi thriller musical
## 1                 Drama        3      0         0     0        0       0
## 2 Biography|Drama|Sport        1      0         0     0        0       0
## 3           Drama|Sport        0      0         0     0        0       0
## 4       Horror|Thriller        0      0         0     0        1       0
## 5 Biography|Crime|Drama        0      0         0     0        0       0
```

```
## 6        Comedy|Romance        0      0         0     0        0       0
##   romance western sport horror drama war animation crime movie_meter_IMDBpro
## 1       0       0     0      0     1   0         0     0                4000
## 2       0       0     1      0     1   0         0     0                8556
## 3       0       0     1      0     1   0         0     0                3940
## 4       0       0     0      1     0   0         0     0                5452
## 5       0       0     0      0     1   0         0     1                4722
## 6       1       0     0      0     0   0         0     0                2446
##   cinematographer    production_company biography comedy sci-fi music fantasy
## 1 Adriano Goldman The Weinstein Company         0      0      0     0       0
## 2     Don Burgess    Revolution Studios         1      0      0     0       0
## 3    Sharone Meir          Coach Carter         0      0      0     0       0
## 4    Dan Laustsen  Ghost House Pictures         0      0      0     0       0
## 5   Bruce Surtees    Paramount Pictures         1      0      0     0       0
## 6   Greg Gardiner           DreamWorks         0      1      0     0       0
##   history mystery family documentary
## 1       0       0      0           0
## 2       0       0      0           0
## 3       0       0      0           0
## 4       0       0      0           0
## 5       0       0      0           0
## 6       0       0      0           0
```

```
dim(df)
```

```
## [1] 1930   48
```

```
colnames(df)
```

```
##  [1] "movie_id"            "imdb_score"          "movie_budget"
##  [4] "release_day"         "release_month"       "release_year"
##  [7] "duration"            "language"            "country"
## [10] "maturity_rating"     "aspect_ratio"        "distributor"
## [13] "nb_news_articles"    "director"            "actor1"
## [16] "actor1_star_meter"   "actor2"              "actor2_star_meter"
## [19] "actor3"              "actor3_star_meter"   "colour_film"
## [22] "genres"              "nb_faces"            "action"
## [25] "adventure"           "scifi"               "thriller"
## [28] "musical"             "romance"             "western"
## [31] "sport"               "horror"              "drama"
## [34] "war"                 "animation"           "crime"
## [37] "movie_meter_IMDBpro" "cinematographer"     "production_company"
## [40] "biography"           "comedy"              "sci-fi"
## [43] "music"               "fantasy"             "history"
## [46] "mystery"             "family"              "documentary"
```

```
# extracted genres: "biography", "comedy", "music", "fantasy", "history", "mystery", "family", "documen
```

Transforming them into a tidy data frame makes visual analysis easier.

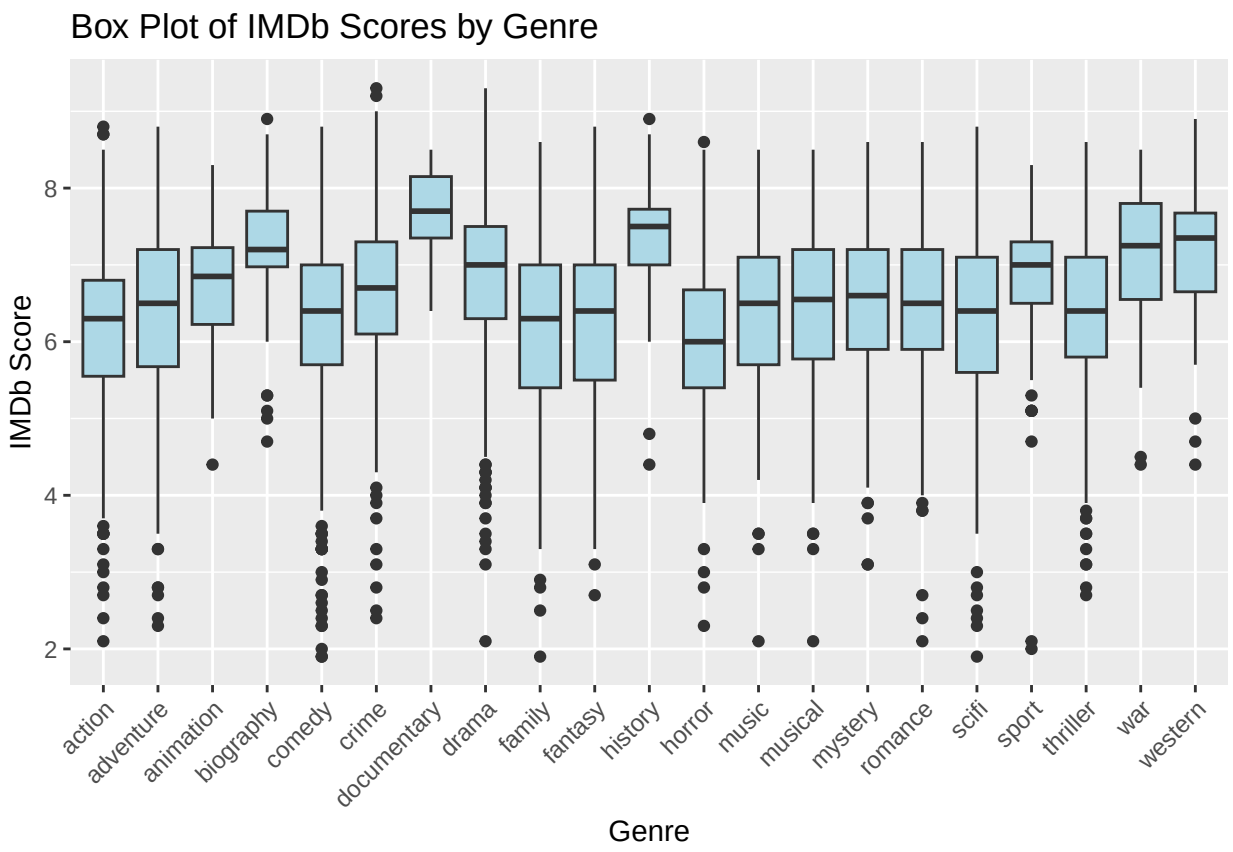
```
genres_all <- c("action", "adventure", "scifi", "thriller", "musical", "romance",
                "western", "sport", "horror", "drama", "war", "animation",
                "crime", "biography", "comedy", "music", "fantasy",
                "history", "mystery", "family", "documentary")

df_long <- pivot_longer(df, cols = all_of(genres_all), names_to = "genre", values_to = "genre_indicator
df_long <- df_long[df_long$genre_indicator == 1, ]
```

```r
head(df_long)
```

```
## # A tibble: 6 x 29
##   movie_id imdb_score movie_budget release_day release_month release_year
##      <int>      <dbl>        <int>       <int> <chr>                <int>
## 1        2        7.3     25000000          10 Jan                   2014
## 2       12        6.9     35000000          24 Oct                   2003
## 3       12        6.9     35000000          24 Oct                   2003
## 4       12        6.9     35000000          24 Oct                   2003
## 5       15        7.2     30000000          14 Jan                   2005
## 6       15        7.2     30000000          14 Jan                   2005
## # i 23 more variables: duration <int>, language <chr>, country <chr>,
## #   maturity_rating <chr>, aspect_ratio <dbl>, distributor <chr>,
## #   nb_news_articles <int>, director <chr>, actor1 <chr>,
## #   actor1_star_meter <int>, actor2 <chr>, actor2_star_meter <int>,
## #   actor3 <chr>, actor3_star_meter <int>, colour_film <chr>, genres <chr>,
## #   nb_faces <int>, movie_meter_IMDBpro <int>, cinematographer <chr>,
## #   production_company <chr>, `sci-fi` <dbl>, genre <chr>, ...
```

Box plot

```r
ggplot(df_long, aes(x = genre, y = imdb_score)) +
  geom_boxplot(fill = "lightblue") +
  labs(title = "Box Plot of IMDb Scores by Genre",
       x = "Genre",
       y = "IMDb Score") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



As the “animation”, “documentary” are less than 30, they could be group into “other”.

```
table(df_long$genre)
```

```
##
##      action   adventure   animation   biography      comedy       crime
##         387         244          20         152         766         417
## documentary       drama      family     fantasy     history      horror
##          11        1060         146         172          76         218
##       music     musical     mystery     romance       scifi       sport
##         100         136         203         473         209          93
##    thriller         war     western
##         575          70          34
```

Look at the t-test to further filter out genres types that could be grouped together!

```
for (genre in genres_all) {
  # Welch's t-test
  t_test_results <- t.test(df$imdb_score[df[[genre]] == 1], df$imdb_score[df[[genre]] == 0])

  # Fetch and print insignificant genres (threshold: p-value = 0.05)
  if (t_test_results$p.value >= 0.05) {
    print(paste(genre, "- p-value:", t_test_results$p.value))
  }
}
```

```
## [1] "musical - p-value: 0.337662947281095"
## [1] "romance - p-value: 0.475827407419031"
## [1] "animation - p-value: 0.420847318364638"
## [1] "music - p-value: 0.0939739158062037"
## [1] "mystery - p-value: 0.863901049829263"
```

Group those undesired genres into “other_genres”

```
grouped_genres <- c("animation", "documentary", "musical", "romance", "music", "mystery")

df$other_genre <- 0

for (col in grouped_genres) {
  df$other_genre[df[[col]] == 1] <- 1
}

drop_genres_cols <- c("genres", "animation", "documentary", "musical", "romance", "music", "mystery")

df[drop_genres_cols] <- NULL

colnames(df)
```

```
##  [1] "movie_id"            "imdb_score"          "movie_budget"
##  [4] "release_day"         "release_month"       "release_year"
##  [7] "duration"            "language"            "country"
## [10] "maturity_rating"     "aspect_ratio"        "distributor"
## [13] "nb_news_articles"    "director"            "actor1"
## [16] "actor1_star_meter"   "actor2"              "actor2_star_meter"
## [19] "actor3"              "actor3_star_meter"   "colour_film"
## [22] "nb_faces"            "action"              "adventure"
## [25] "scifi"               "thriller"            "western"
```

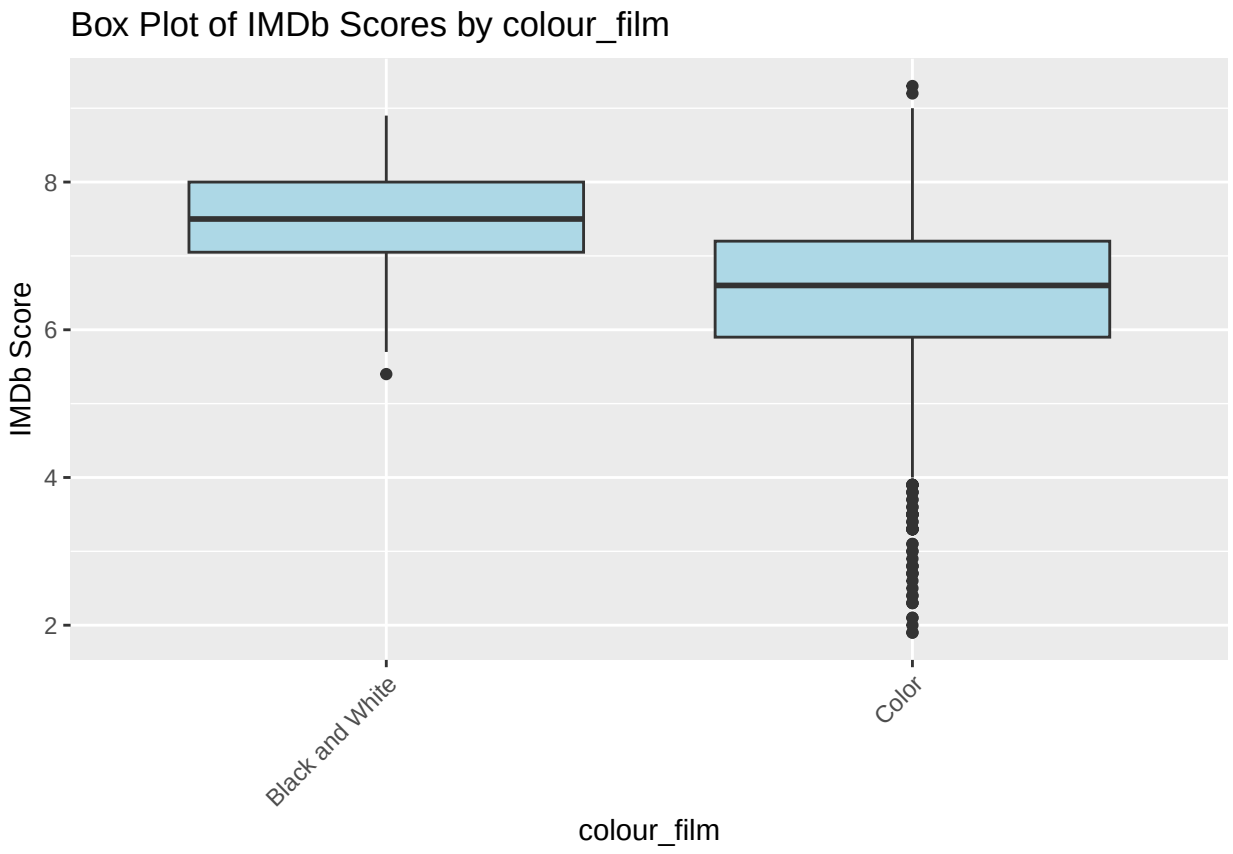
```
## [28] "sport"               "horror"              "drama"
## [31] "war"                 "crime"               "movie_meter_IMDBpro"
## [34] "cinematographer"     "production_company"  "biography"
## [37] "comedy"              "sci-fi"              "fantasy"
## [40] "history"             "family"              "other_genre"
```

Color Film Because the test dataset has no variation in this predictor (all movies are in colour), it doesn't provide useful information for predicting IMDb scores. So, we decide to drop it.

(skip the following analysis)

There is a significant difference between the levels. Although the movies we are going to predict are all in color, making it reasonable to drop this variable, for now, we'll keep it to check its interaction effect with other predictors.

```
ggplot(df, aes(x = colour_film, y = imdb_score)) +
  geom_boxplot(fill = "lightblue") +
  labs(title = "Box Plot of IMDb Scores by colour_film",
       x = "colour_film",
       y = "IMDb Score") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

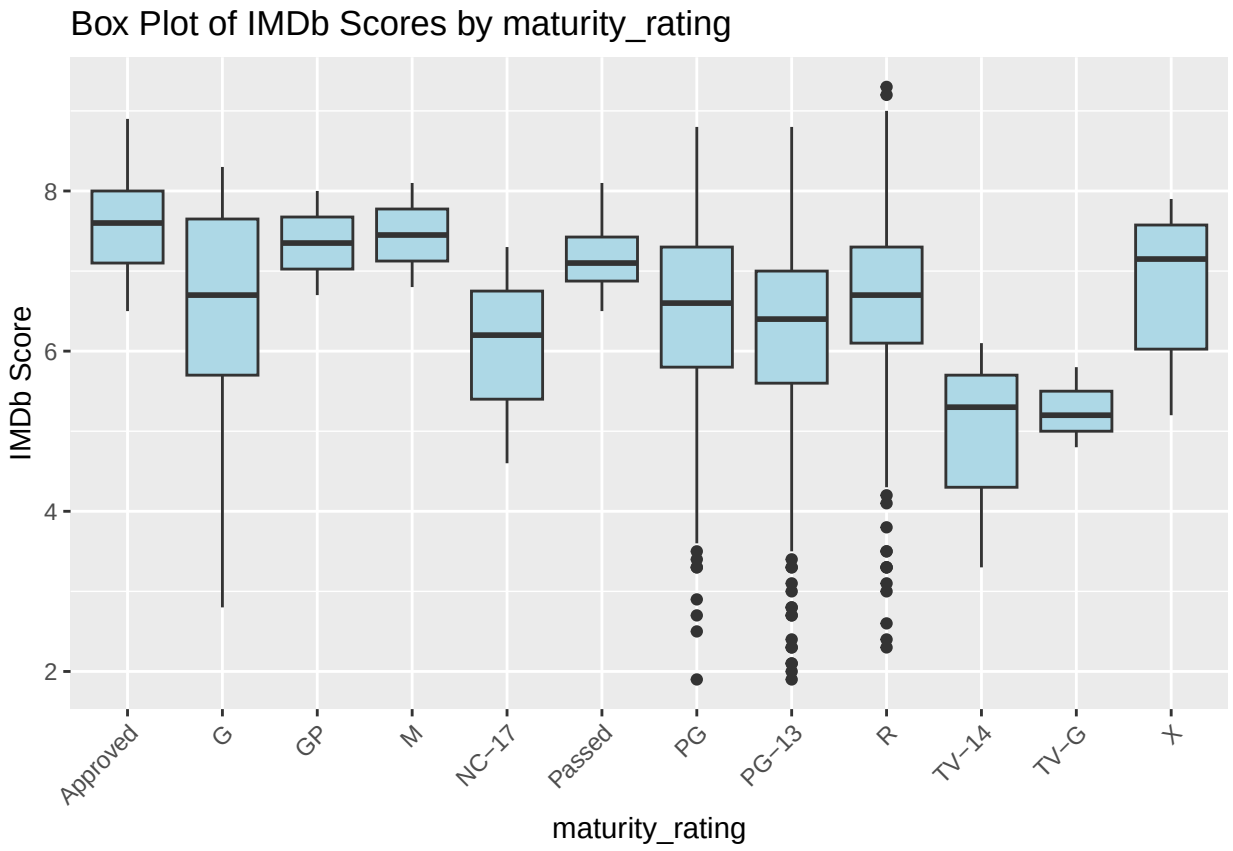


maturity_rating

- Approved: Generally used for older content (suitable for all ages, but often lacks specificity).
- G (General Audience) / TV-G (General Audience): Suitable for all ages (0+).
- GP (General Parental Guidance): Suggested parental guidance (ages 5+).
- PG (Parental Guidance): Some material may not be suitable for children (ages 7+).

- PG-13 (Parents Strongly Cautioned): Some material may be inappropriate for children under 13 (ages 13+).
- TV-14 (Parents Strongly Cautioned): Some material may be inappropriate for children under 14 (ages 14+).
- R (Restricted): Restricted to viewers over 17 or 18, parents are strongly cautioned (ages 17+).
- NC-17 (No One 17 and Under Admitted): Explicit content, unsuitable for minors (ages 17+).
- M (Mature): Content for mature audiences only (typically ages 17+).
- X (Adult Only): Explicit content, suitable for adults only (ages 18+).
- Passed: A classification indicating acceptance, typically seen in older content.

```
ggplot(df, aes(x = maturity_rating, y = imdb_score)) +
  geom_boxplot(fill = "lightblue") +
  labs(title = "Box Plot of IMDb Scores by maturity_rating",
       x = "maturity_rating",
       y = "IMDb Score") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



Given the presence of rare levels in this feature, we reclassify them into three categories based on the age definitions mentioned earlier to help with dimension reduction and enhance the robustness of the levels.

```
table(df$maturity_rating)
```

```
##
## Approved        G       GP        M    NC-17   Passed       PG    PG-13
##       21       34        2        2        3        4      255      582
##        R    TV-14     TV-G        X
##     1013        3        3        8
```

Ordinal Levels -

Based on the usual rating hierarchy, the sorted ordinal levels for maturity ratings could be:

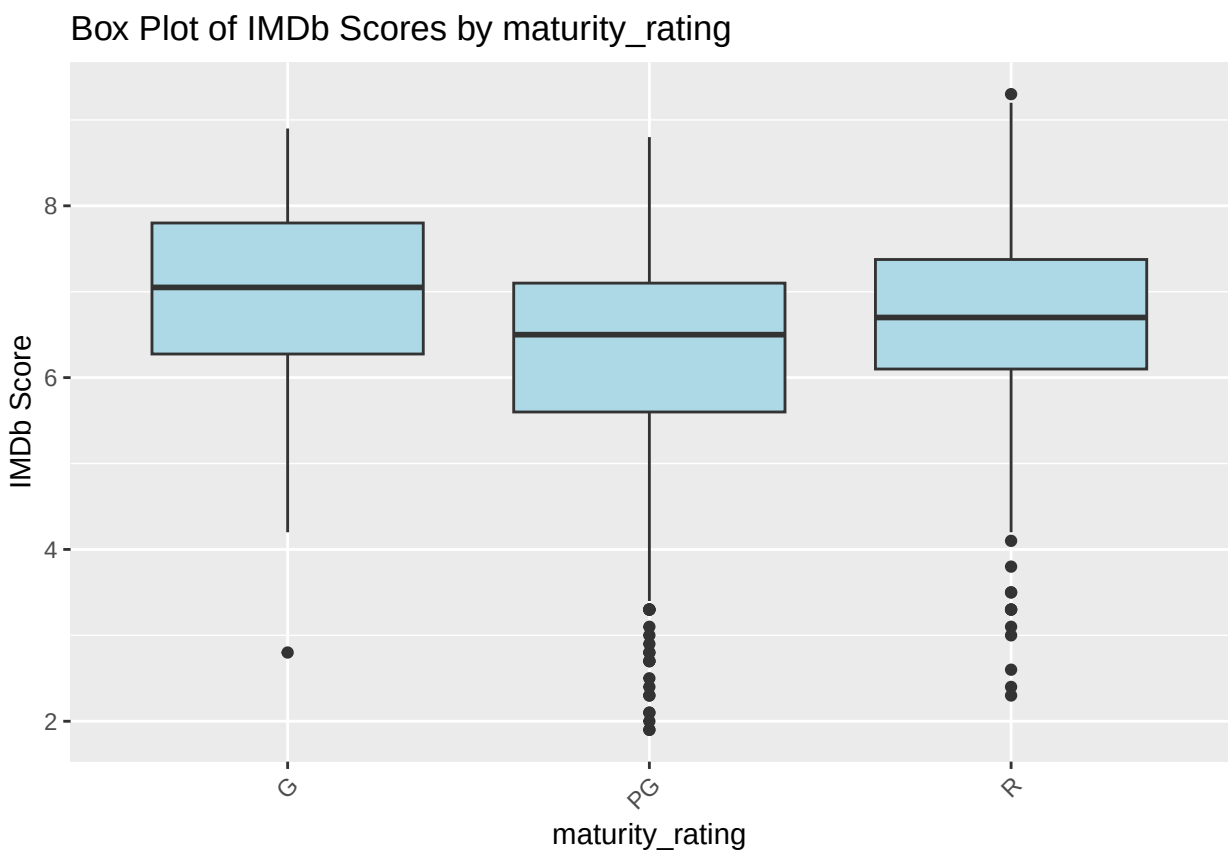
- $<$ 7 yo: Approved, G, TV-G, GP
- $<$ 17/18 yo: PG, PG-13, TV-14
- $=$ 17/18: R, NC-17, M, Passed, X

```
group_G <- c("Approved", "TV-G", "GP")
group_PG <- c("PG-13", "TV-14")
group_R <- c("NC-17", "M", "Passed", "X")

df$maturity_rating[df$maturity_rating %in% group_G] <- "G"
df$maturity_rating[df$maturity_rating %in% group_PG] <- "PG"
df$maturity_rating[df$maturity_rating %in% group_R] <- "R"

table(df$maturity_rating)
```

```
##
##    G   PG    R
##   60  840 1030
```

G and R are quite similar in terms of imdb_score.

```
ggplot(df, aes(x = maturity_rating, y = imdb_score)) +
  geom_boxplot(fill = "lightblue") +
  labs(title = "Box Plot of IMDb Scores by maturity_rating",
       x = "maturity_rating",
       y = "IMDb Score") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



t-test

```
pairwise_results <- pairwise.t.test(df$imdb_score, df$maturity_rating, p.adjust.method = "bonferroni")
print(pairwise_results)
```

```
##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  df$imdb_score and df$maturity_rating
##
##    G       PG
## PG 0.00019 -
## R  0.41722 0.0000000000012
##
## P value adjustment method: bonferroni
```

There is no significant difference in IMDb scores between the R and G ratings. However, combining them into a single group seems inappropriate, and there may be potential interaction effects with other predictors. So, we decide to keep all three levels.

```
table(df$country)
```

country

```
##
##          Aruba      Australia        Belgium         Brazil         Canada
##              1             23              1              1             38
##          China       Colombia Czech Republic        Denmark         France
##              2              1              1              1             40
##        Georgia        Germany         Greece      Hong Kong        Hungary
##              1             34              1              4              1
##          India      Indonesia        Ireland          Italy          Japan
##              1              1              5              8              5
##     Kyrgyzstan         Mexico    Netherlands    New Zealand  Official site
##              1              1              2              5              1
##           Peru         Russia   South Africa    South Korea          Spain
##              1              1              5              2              7
##         Taiwan             UK            USA   West Germany
##              1            177           1555              1
```

We have 34 countries in the dataset, but some only have a single sample, making them quite rare. We approach it in two ways:

1. Reclassifying the countries by continent.
2. Grouping them into representative countries and creating an 'Other Countries' category for the rest.

When modeling, we'll evaluate which approach performs better.

Approach 1: Re-classify countries by continent

```
Africa <- c("South Africa", "Kenya", "Nigeria", "Egypt", "Ghana", "Morocco", "Uganda")
Asia <- c("China", "India", "Japan", "South Korea", "Indonesia", "Kyrgyzstan", "Russia", "Taiwan", "Hong
Europe <- c("Belgium", "Czech Republic", "Denmark", "France", "Georgia", "Germany", "Greece", "Hungary"
North_America <- c("USA", "Canada", "Mexico")
Central_South_America <- c("Aruba", "Brazil", "Colombia", "Peru")
Australia <- c("Australia", "New Zealand")
Other <- c("Official site")
```

```
df$continent <- 0

df$continent[df$country %in% Africa] <- "Africa"
df$continent[df$country %in% Asia] <- "Asia"
df$continent[df$country %in% Europe] <- "Europe"
df$continent[df$country %in% North_America] <- "North America"
df$continent[df$country %in% Central_South_America] <- "Central South America"
df$continent[df$country %in% Australia] <- "Australia"
df$continent[df$country %in% Other] <- "Other"

table(df$continent)
```

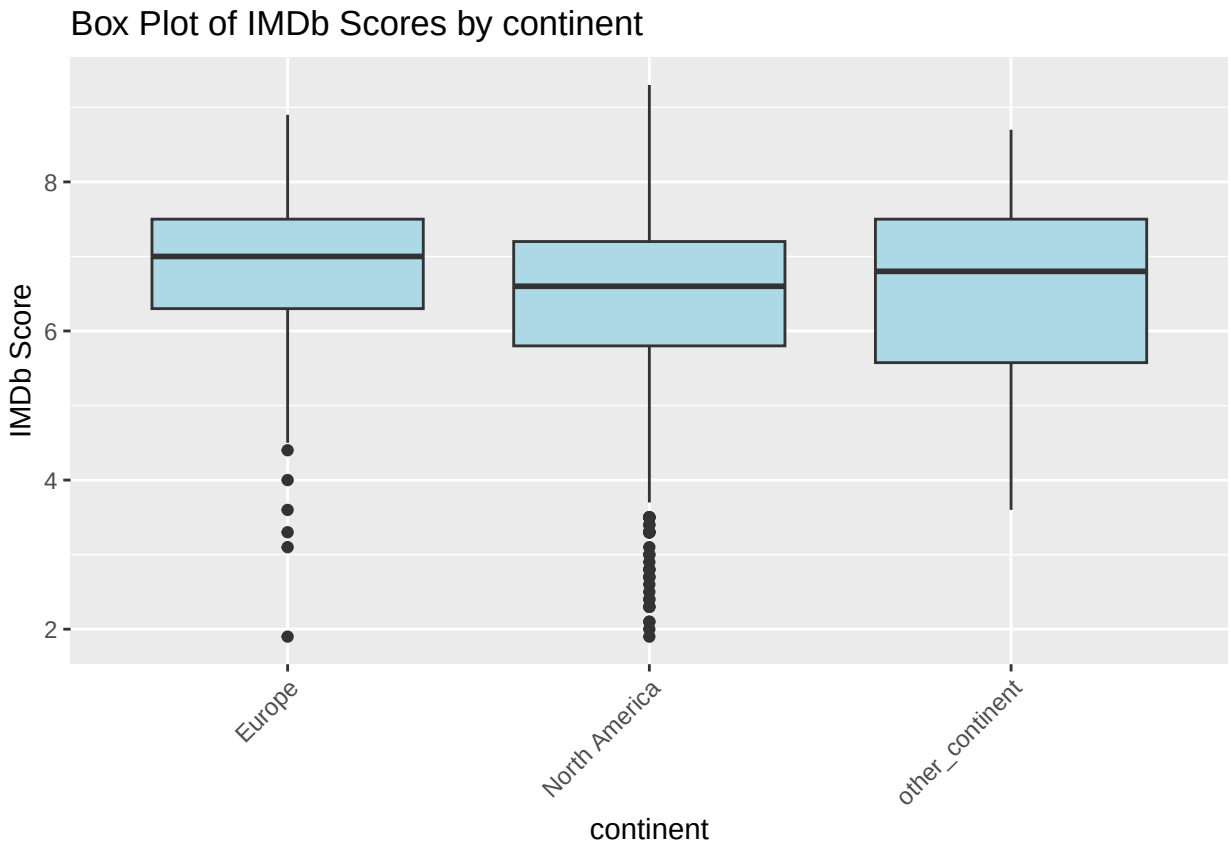
```
##
##                 Africa                   Asia              Australia
##                      5                     18                     28
## Central South America                 Europe          North America
##                      4                    280                   1594
##                  Other
##                      1
```

Since Africa, Asia, Australia, Central South America, Other each have fewer than 30 samples, which limits their statistical representation, we further consolidate these continents into "other_continents" category.

```
other_continent <- c("Africa", "Asia", "Australia", "Central South America", "Other")
df$continent[df$continent %in% other_continent] <- "other_continent"

table(df$continent)
```

```
##
##          Europe   North America other_continent
##             280            1594              56
```

```
ggplot(df, aes(x = continent, y = imdb_score)) +
  geom_boxplot(fill = "lightblue") +
  labs(title = "Box Plot of IMDb Scores by continent",
       x = "continent",
       y = "IMDb Score") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



t-test

```
pairwise_results <- pairwise.t.test(df$imdb_score, df$continent, p.adjust.method = "bonferroni")
print(pairwise_results)
```

```
##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  df$imdb_score and df$continent
##
##                 Europe     North America
## North America   0.00000015 -
## other_continent 0.38       1.00
##
## P value adjustment method: bonferroni
```

Although the t-test shows no significant difference in IMDb scores between 'other_continent' and Europe or between 'other_continent' and North America, keeping 'other_continent' as a separate category still provides additional granularity and may help capture nuanced interactions with other predictors. We'll see.

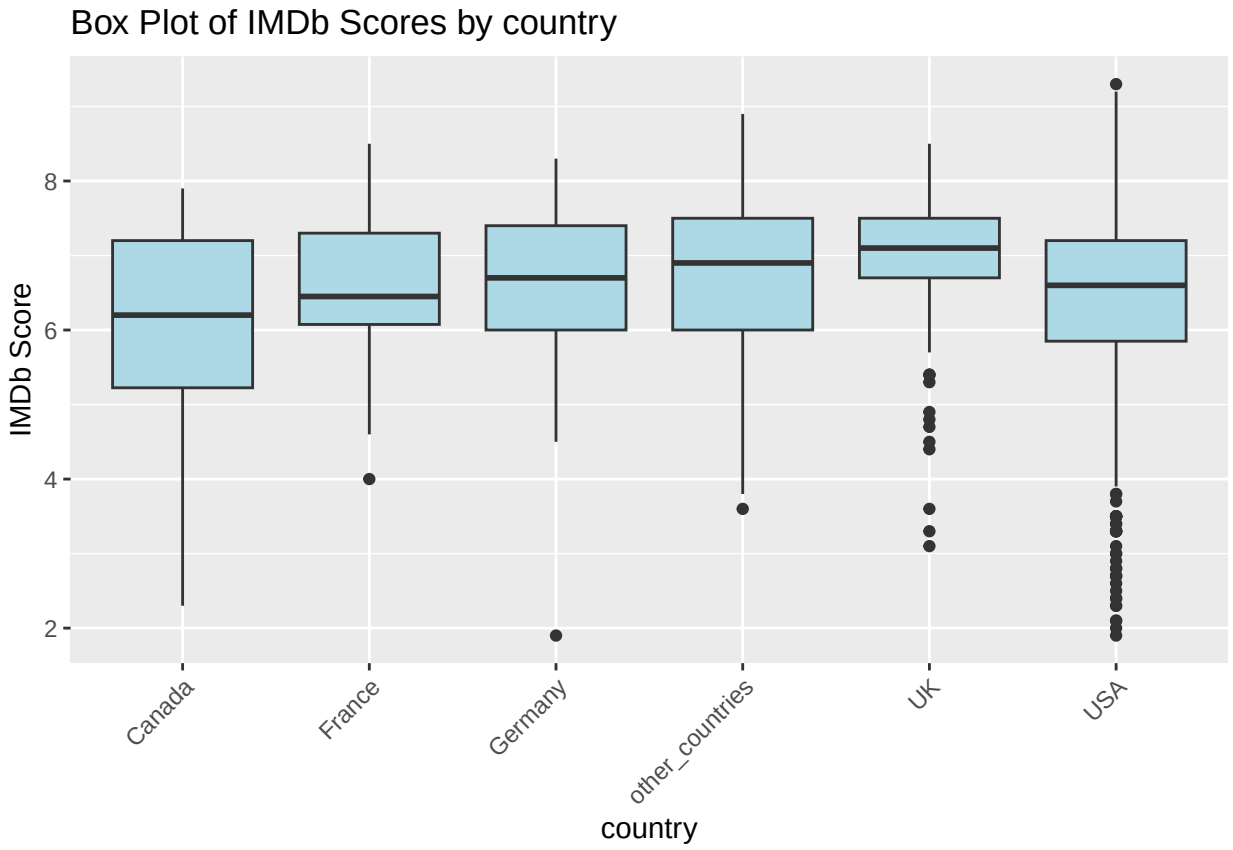
Approach 2: Re-classify countries by frequncy

```
over_30_countries <- c("Germany", "West Germany", "Canada", "France", "UK", "USA")
over_30_germany <- c("West Germany")

df$country[!(df$country %in% over_30_countries)] <- "other_countries"
df$country[df$country %in% over_30_germany] <- "Germany"

table(df$country)
```

```
##
##          Canada          France         Germany other_countries              UK
##              38              40              35              85             177
##             USA
##            1555
```

Box-plot

```
ggplot(df, aes(x = country, y = imdb_score)) +
  geom_boxplot(fill = "lightblue") +
  labs(title = "Box Plot of IMDb Scores by country",
       x = "country",
       y = "IMDb Score") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



t-test

```
pairwise_results <- pairwise.t.test(df$imdb_score, df$country, p.adjust.method = "bonferroni")
print(pairwise_results)
```

```
##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  df$imdb_score and df$country
##
##                Canada  France Germany other_countries UK
## France         0.293   -      -       -               -
## Germany        0.782   1.000  -       -               -
## other_countries 0.023  1.000  1.000   -               -
```

```
## UK              0.00002257 0.855  0.416   0.959           -
## USA             0.201      1.000  1.000   0.860           0.00000015
##
## P value adjustment method: bonferroni
```

Despite the lack of strong significance in most comparisons, keeping the 'other_countries' category allows for finer granularity and could help capture nuanced interactions with other predictors. Who knows what might happen when we run some non-linear or non-parametric models!!!

Personnel The way we approach personnel-related predictors is as follows:

Steps:

1. Identify the top 10 entries for each predictor
2. Create new dummy columns indicating whether a movie falls within the top 10 for each personnel predictor

Identify the top 10 entries for each predictor

```
personnel_cols <- c("director", "actor1", "actor2", "actor3", "cinematographer", "production_company",

top_10_list <- list()

for (col in personnel_cols) {
  top_10 <- head(sort(table(df[[col]]), decreasing = TRUE), 10)

  # Store the top 10 in the list with the column name as the key
  top_10_list[[col]] <- top_10

  cat("Top 10 for", col, ":\n")
  print(top_10)
  cat("-------------------------\n")

  sum_top_10 <- sum(top_10)
  cat("Sum of Top 10:", sum_top_10, "\n")
  cat("-------------------------\n")
}
```

```
## Top 10 for director :
##
##          Woody Allen     Steven Spielberg       Clint Eastwood
##                   18                   12                   11
##            Spike Lee    Steven Soderbergh      Martin Scorsese
##                   11                   10                    9
##       Barry Levinson       Bobby Farrelly Francis Ford Coppola
##                    8                    8                    8
##      Joel Schumacher
##                    8
## -------------------------
## Sum of Top 10: 103
## -------------------------
## Top 10 for actor1 :
##
##     Robert De Niro        Bill Murray       J.K. Simmons       Kevin Spacey
##                 30                 17                 17                 17
##      Jason Statham      Harrison Ford        Johnny Depp  Denzel Washington
##                 15                 14                 14                 13
```

```
## Scarlett Johansson       Keanu Reeves
##                 13                 12
## -------------------------
## Sum of Top 10: 162
## -------------------------
## Top 10 for actor2 :
##
##   Morgan Freeman  Charlize Theron        Brad Pitt Chazz Palminteri
##                9                7                6                6
##       Demi Moore     Meryl Streep     James Franco    Jason Flemyng
##                6                6                5                5
##    Jay Hernandez     Kate Winslet
##                5                5
## -------------------------
## Sum of Top 10: 60
## -------------------------
## Top 10 for actor3 :
##
##          Hope Davis      Ben Mendelsohn          John Heard       Robert Duvall
##                   6                   5                   5                   5
##        Steve Carell       Thomas Lennon          Anne Heche          Bob Gunton
##                   5                   5                   4                   4
##        Bruce McGill Clifton Collins Jr.
##                   4                   4
## -------------------------
## Sum of Top 10: 47
## -------------------------
## Top 10 for cinematographer :
##
##            multiple       Roger Deakins          Mark Irwin         John Bailey
##                  79                  18                  17                  16
##         Andrew Dunn       Jack N. Green Matthew F. Leonetti       Robert Elswit
##                  13                  13                  13                  13
##         Dean Cundey         Don Burgess
##                  12                  12
## -------------------------
## Sum of Top 10: 206
## -------------------------
## Top 10 for production_company :
##
##            Universal Pictures            Paramount Pictures
##                           110                            99
## Columbia Pictures Corporation                 Warner Bros.
##                           96                            76
##               New Line Cinema         Twentieth Century Fox
##                           75                            70
##     Metro-Goldwyn-Mayer (MGM)           Touchstone Pictures
##                           38                            31
##                    DreamWorks                      Miramax
##                           29                            29
## -------------------------
## Sum of Top 10: 653
## -------------------------
## Top 10 for distributor :
```

```
##
##                  Warner Bros.            Universal Pictures
##                           169                           146
##            Paramount Pictures         Twentieth Century Fox
##                           138                           126
## Columbia Pictures Corporation             New Line Cinema
##                           113                            73
##          Buena Vista Pictures                       Miramax
##                            60                            44
##                United Artists     Metro-Goldwyn-Mayer (MGM)
##                            40                            39
## -------------------------
## Sum of Top 10: 948
## -------------------------
```

Create new dummy columns indicating whether a movie falls within the top 10 for each personnel predictor

```
for (col in personnel_cols) {

  top_10_names <- names(top_10_list[[col]])

  new_col_name <- paste0("top_", col)

  df[[new_col_name]] <- ifelse(df[[col]] %in% top_10_names, 1, 0)
}

df[personnel_cols] <- NULL

colnames(df)
```

```
##  [1] "movie_id"               "imdb_score"             "movie_budget"
##  [4] "release_day"            "release_month"          "release_year"
##  [7] "duration"               "language"               "country"
## [10] "maturity_rating"        "aspect_ratio"           "nb_news_articles"
## [13] "actor1_star_meter"      "actor2_star_meter"      "actor3_star_meter"
## [16] "colour_film"            "nb_faces"               "action"
## [19] "adventure"              "scifi"                  "thriller"
## [22] "western"                "sport"                  "horror"
## [25] "drama"                  "war"                    "crime"
## [28] "movie_meter_IMDBpro"    "biography"              "comedy"
## [31] "sci-fi"                 "fantasy"                "history"
## [34] "family"                 "other_genre"            "continent"
## [37] "top_director"           "top_actor1"             "top_actor2"
## [40] "top_actor3"             "top_cinematographer"    "top_production_company"
## [43] "top_distributor"
```

```
head(df)
```

```
##   movie_id imdb_score movie_budget release_day release_month release_year
## 1        2        7.3     25000000          10           Jan         2014
## 2       12        6.9     35000000          24           Oct         2003
## 3       15        7.2     30000000          14           Jan         2005
## 4       20        5.9     14000000          20           Aug         2012
## 5       22        7.6      8000000          22           Jun         1979
## 6       23        6.4     20000000          17           Mar         2006
##   duration language country maturity_rating aspect_ratio nb_news_articles
```

```
## 1      121  English     USA              R         2.35             2141
## 2      109  English     USA             PG         1.85              331
## 3      136  English     USA             PG         2.35              223
## 4       92  English     USA             PG         2.35              620
## 5      112  English     USA             PG         1.85               97
## 6      105  English     USA             PG         1.85              173
##   actor1_star_meter actor2_star_meter actor3_star_meter colour_film nb_faces
## 1               259               559               513       Color        3
## 2              2735              3915              1845       Color        1
## 3               573              4793              6729       Color        0
## 4              2047              1769             11963       Color        0
## 5               102              5062              5451       Color        0
## 6               573               370              3711       Color        0
##   action adventure scifi thriller western sport horror drama war crime
## 1      0         0     0        0       0     0      0     1   0     0
## 2      0         0     0        0       0     1      0     1   0     0
## 3      0         0     0        0       0     1      0     1   0     0
## 4      0         0     0        1       0     0      1     0   0     0
## 5      0         0     0        0       0     0      0     1   0     1
## 6      0         0     0        0       0     0      0     0   0     0
##   movie_meter_IMDBpro biography comedy sci-fi fantasy history family
## 1                4000         0      0      0       0       0      0
## 2                8556         1      0      0       0       0      0
## 3                3940         0      0      0       0       0      0
## 4                5452         0      0      0       0       0      0
## 5                4722         1      0      0       0       0      0
## 6                2446         0      1      0       0       0      0
##   other_genre     continent top_director top_actor1 top_actor2 top_actor3
## 1           0 North America            0          0          1          0
## 2           0 North America            0          0          0          0
## 3           0 North America            0          0          0          0
## 4           0 North America            0          0          0          0
## 5           0 North America            0          0          0          0
## 6           1 North America            0          0          0          0
##   top_cinematographer top_production_company top_distributor
## 1                   0                      0               0
## 2                   1                      0               1
## 3                   0                      0               1
## 4                   0                      0               0
## 5                   0                      1               1
## 6                   0                      1               0
```

Release timing Even though the upcoming movies we're predicting are all released in November, and the timing predictors might seem irrelevant, they could still be valuable. If historical data reveals significant trends or patterns associated with **specific days of the week**, **release periods within a month (early, mid, or late month)**, or whether **it's a weekday or weekend**, these predictors could help improve the model's performance.

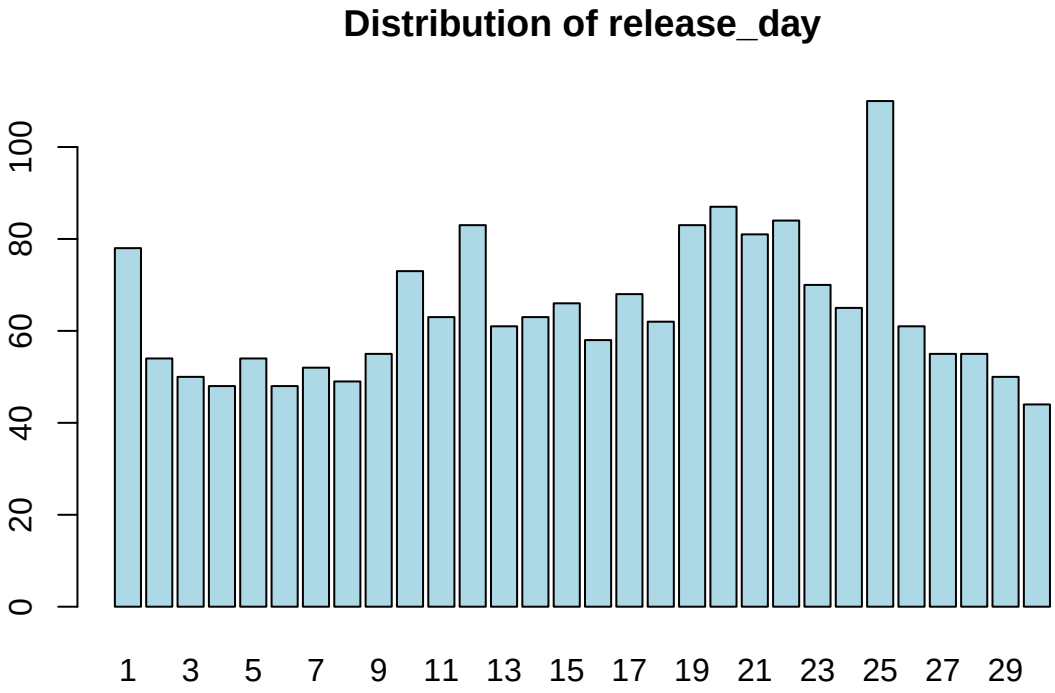
bar chart (Distribution)

We can classify the day into three groups representing early, mid, and late parts of a month to see whether there's a trend.

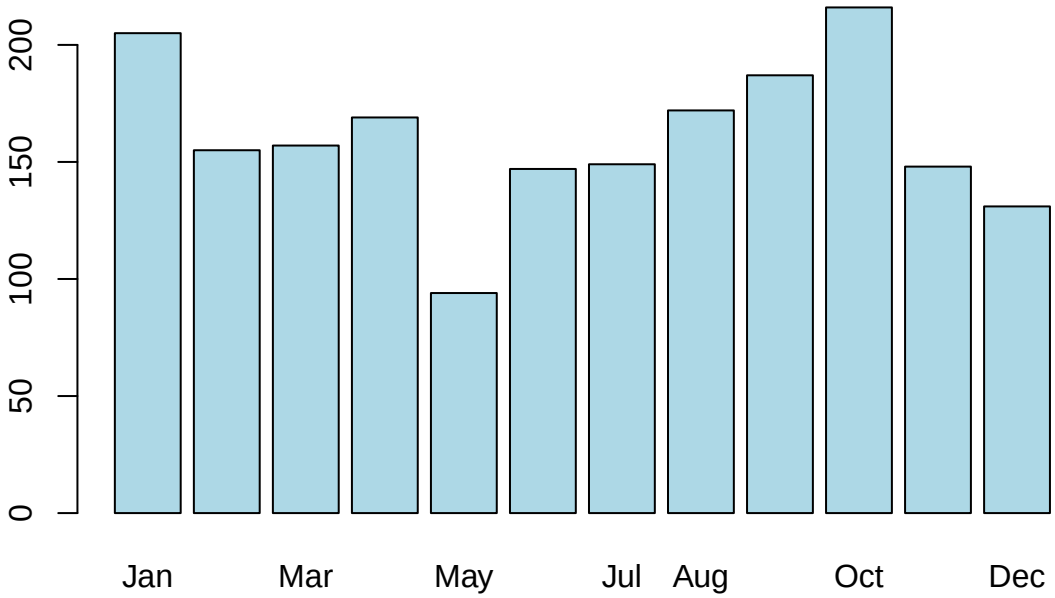
```
# convert the chr to factor, so that the plot will present in order
df$release_month <- factor(df$release_month, levels = c("Jan", "Feb", "Mar", "Apr", "May", "Jun",
```

```
                                                    "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"))

df$release_day_group <- cut(as.numeric(df$release_day),
                            breaks = c(0, 10, 20, 31),
                            labels = c("Early Month", "Mid Month", "Late Month"),
                            right = TRUE)

release_cols <- colnames(df)[grepl("release_day|release_day_group|release_month|release_year", colnames
for (col in release_cols) {
  print(col)
  print(table(df[[col]]))
  barplot(table(df[[col]]), main=paste("Distribution of", col), col="lightblue")
}
```

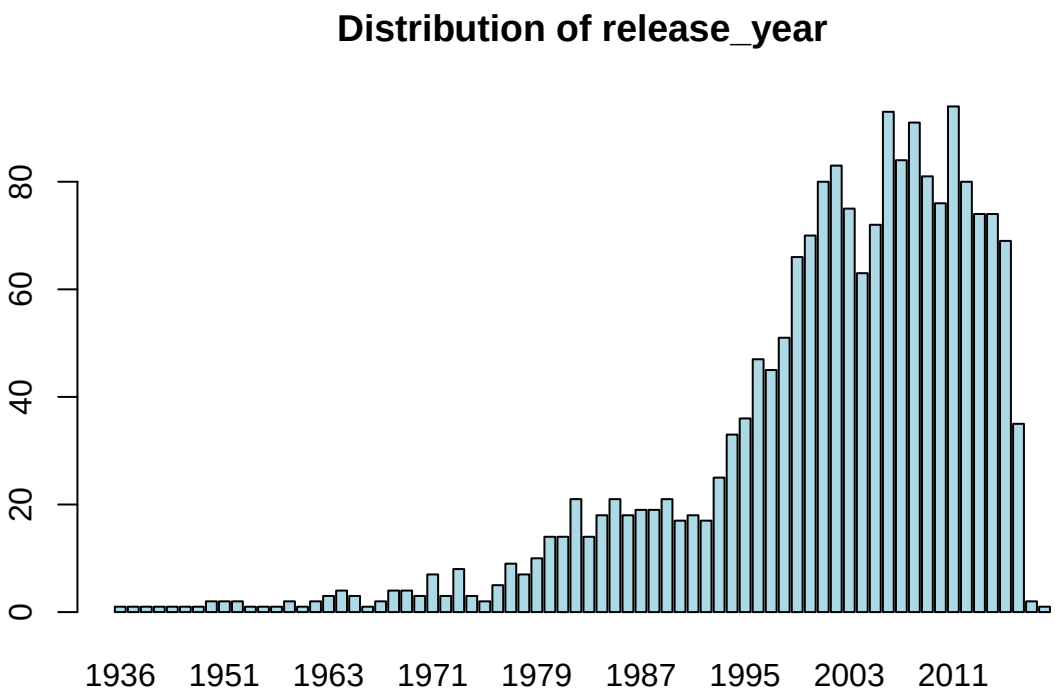
```
## [1] "release_day"
##
##   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20
##  78  54  50  48  54  48  52  49  55  73  63  83  61  63  66  58  68  62  83  87
##  21  22  23  24  25  26  27  28  29  30
##  81  84  70  65 110  61  55  55  50  44
```



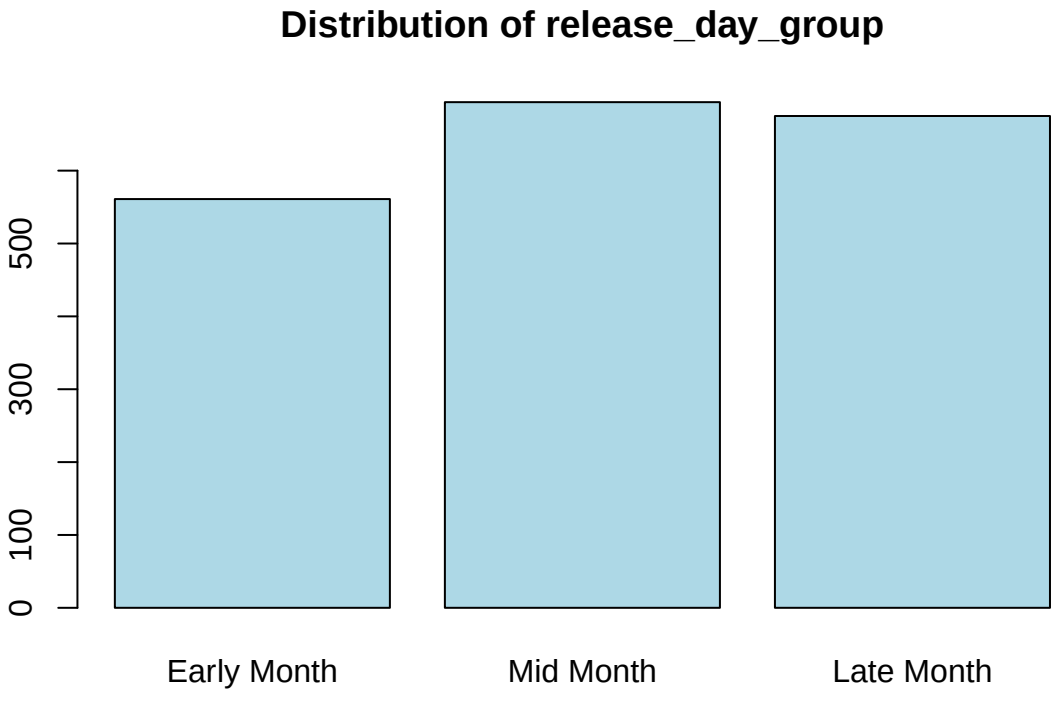

```
## [1] "release_month"
##
## Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
## 205 155 157 169  94 147 149 172 187 216 148 131
```




```
## [1] "release_year"
##
## 1936 1938 1939 1940 1941 1943 1944 1947 1951 1954 1955 1957 1958 1960 1961 1962
##    1    1    1    1    1    1    1    2    2    2    1    1    1    2    1    2
## 1963 1964 1965 1966 1967 1968 1969 1970 1971 1972 1973 1974 1975 1976 1977 1978
##    3    4    3    1    2    4    4    3    7    3    8    3    2    5    9    7
## 1979 1980 1981 1982 1983 1984 1985 1986 1987 1988 1989 1990 1991 1992 1993 1994
##   10   14   14   21   14   18   21   18   19   19   21   17   18   17   25   33
## 1995 1996 1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010
##   36   47   45   51   66   70   80   83   75   63   72   93   84   91   81   76
## 2011 2012 2013 2014 2015 2016 2017 2018
##   94   80   74   74   69   35    2    1
```

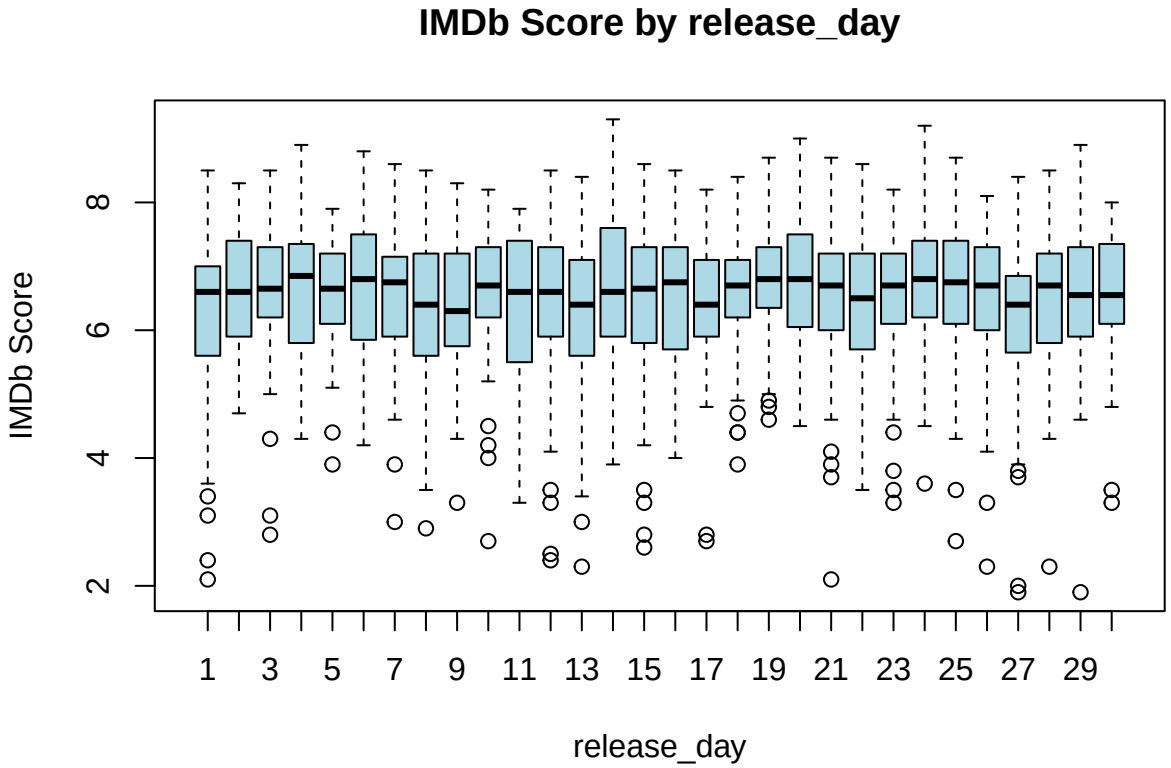


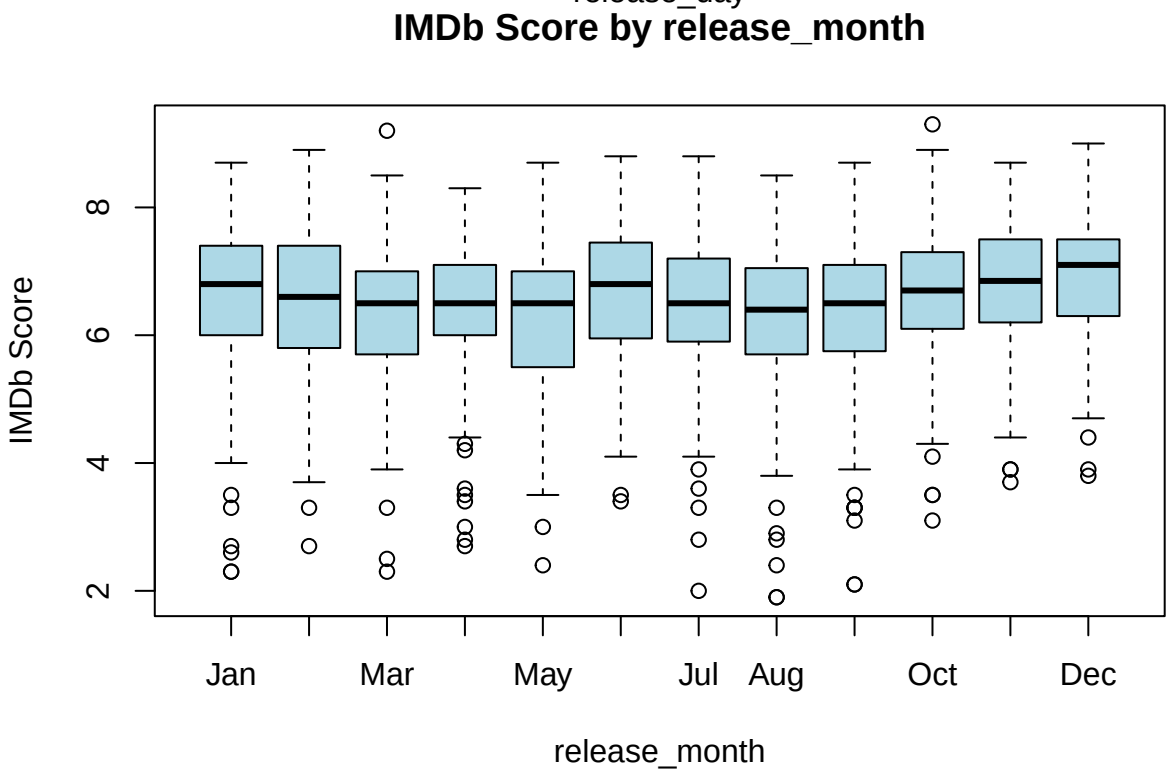

```
## [1] "release_day_group"
##
## Early Month   Mid Month  Late Month
##         561         694         675
```

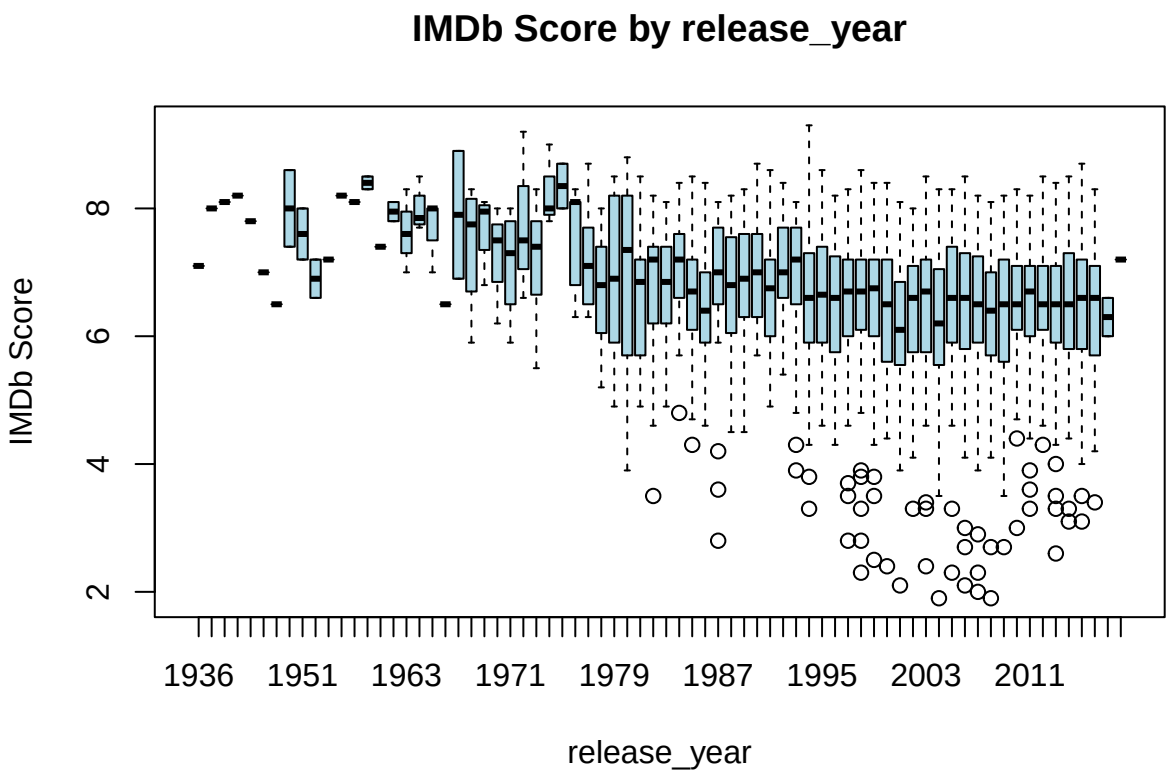


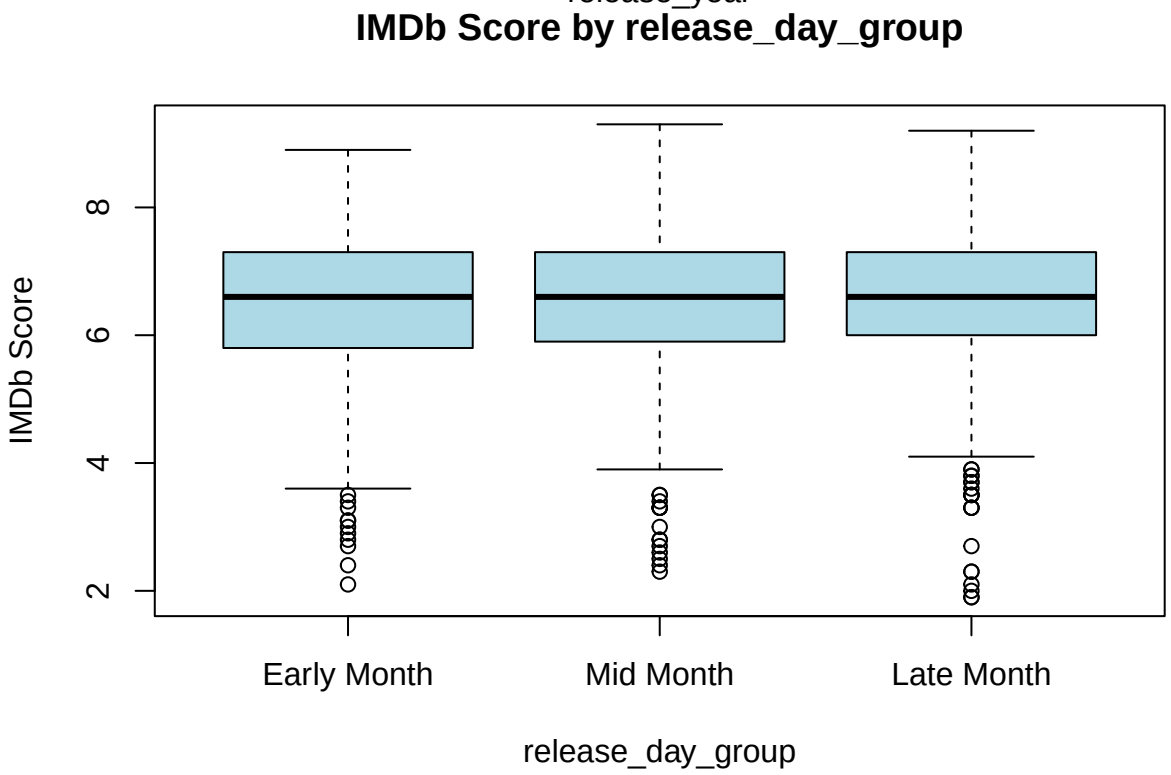
box plot

```
for (col in release_cols) {
  boxplot(df$imdb_score ~ df[[col]], main=paste("IMDb Score by", col), ylab="IMDb Score", xlab=col, col
}
```


IMDb Score by release_day
IMDb Score
2
4
6
8
1
3
5
7
9
11
13
15
17
19
21
23
25
27
29
release_day


IMDb Score by release_month
IMDb Score
2
4
6
8
Jan
Mar
May
Jul
Aug
Oct
Dec
release_month





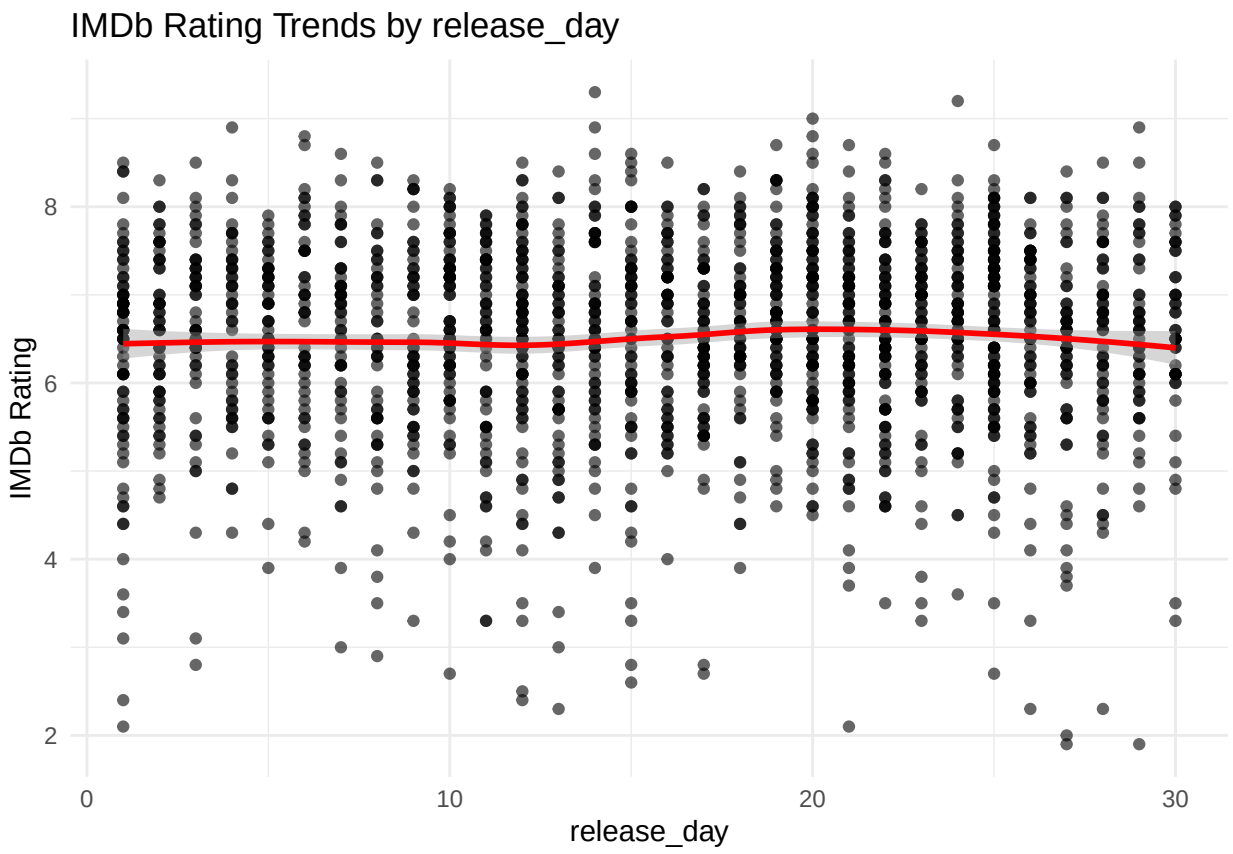
scatter plot

```
for (col in release_cols) {

  p <- ggplot(df, aes_string(x = col, y = "imdb_score")) +
    geom_point(color = "black", alpha = 0.6) +
```

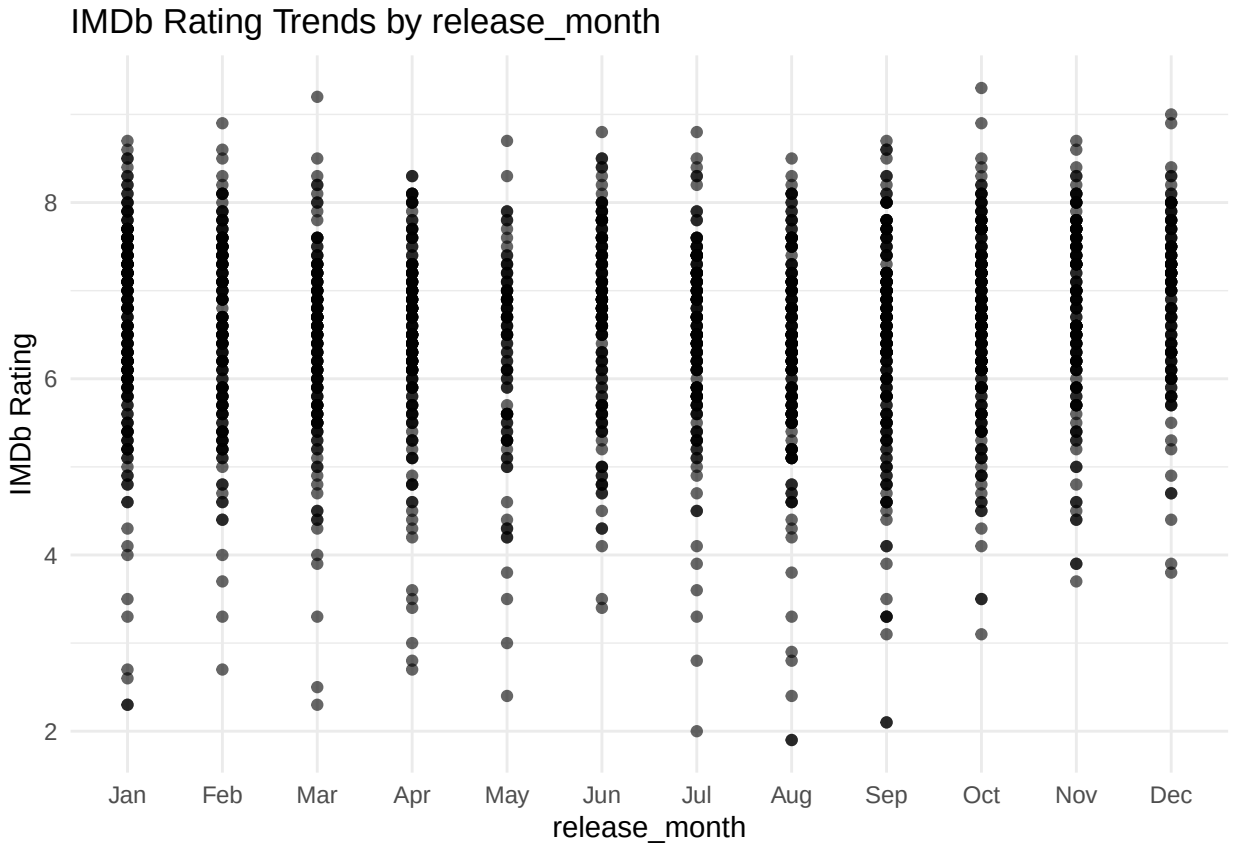
```
    geom_smooth(method = "loess", color = "red", se = TRUE) +
    labs(x = col, y = "IMDb Rating", title = paste("IMDb Rating Trends by", col)) +
    theme_minimal()  # Cleaner theme

  print(p)
}
```

```
## Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
## i Please use tidy evaluation idioms with `aes()`.
## i See also `vignette("ggplot2-in-packages")` for more information.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

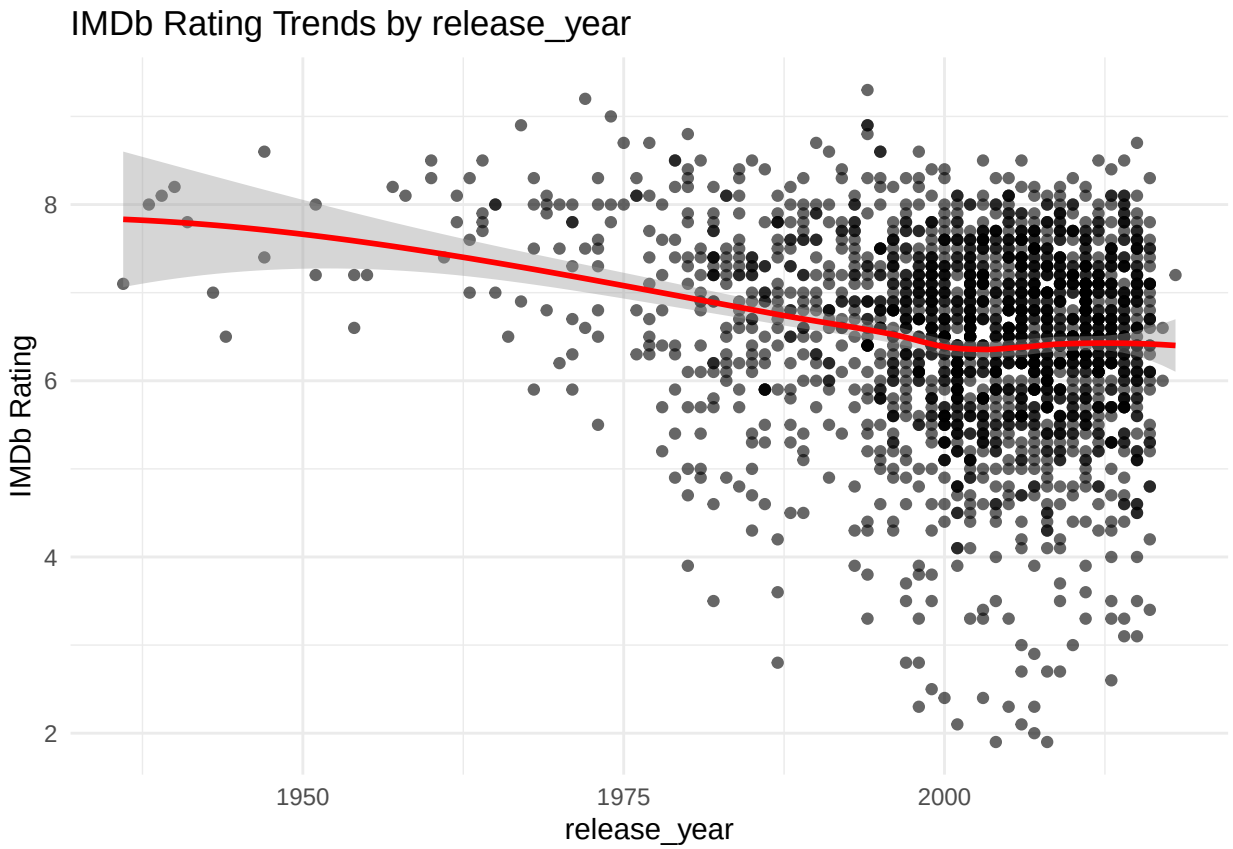
```
## `geom_smooth()` using formula = 'y ~ x'
```



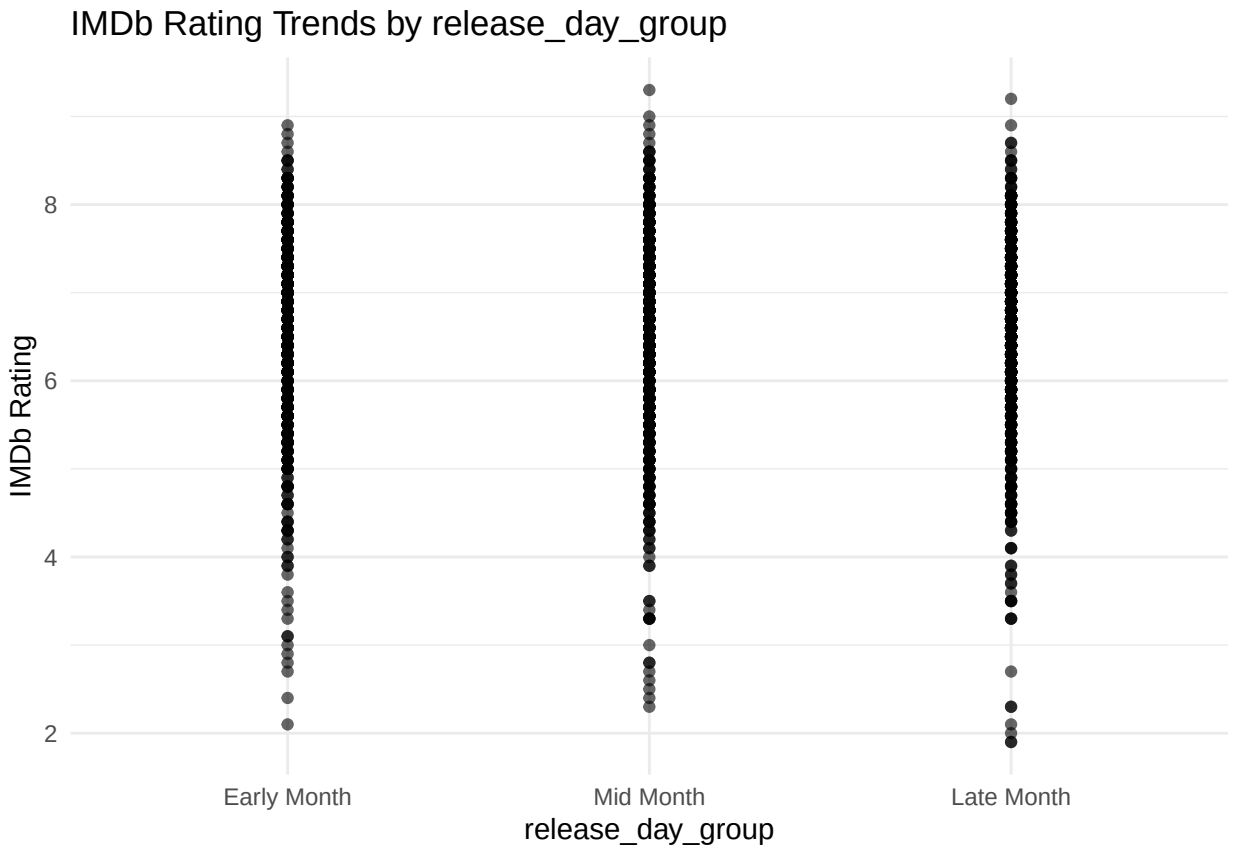

```
## `geom_smooth()` using formula = 'y ~ x'
```




```
## `geom_smooth()` using formula = 'y ~ x'
```




```
## `geom_smooth()` using formula = 'y ~ x'
```



From the analysis above, it seems like time-related predictors are useless. The decreasing trend observed in the year dimension is likely due to sample size rather than an actual decline in IMDb scores. Also, no discernible pattern spotted, regardless of the time granularity analyzed.

Mon ~ Sun / Weekday or Weekend

One more thing we can take with them is to analyze whether the release day falls on a weekday / weekend, or on a specific day of the week, has any effect on the IMDb scores.

```
df$date <- as.Date(with(df, paste(release_year, release_month, release_day, sep = "-")), "%Y-%b-%d")

# Get the day of the week (1 = Sunday, 7 = Saturday)
df$day_of_week <- wday(df$date, label = TRUE)
df$weekday_or_weekend <- ifelse(df$day_of_week %in% c("Sat", "Sun"), "Weekend", "Weekday")

head(df[c("release_year", "release_month", "release_day", "day_of_week", "weekday_or_weekend")])
```

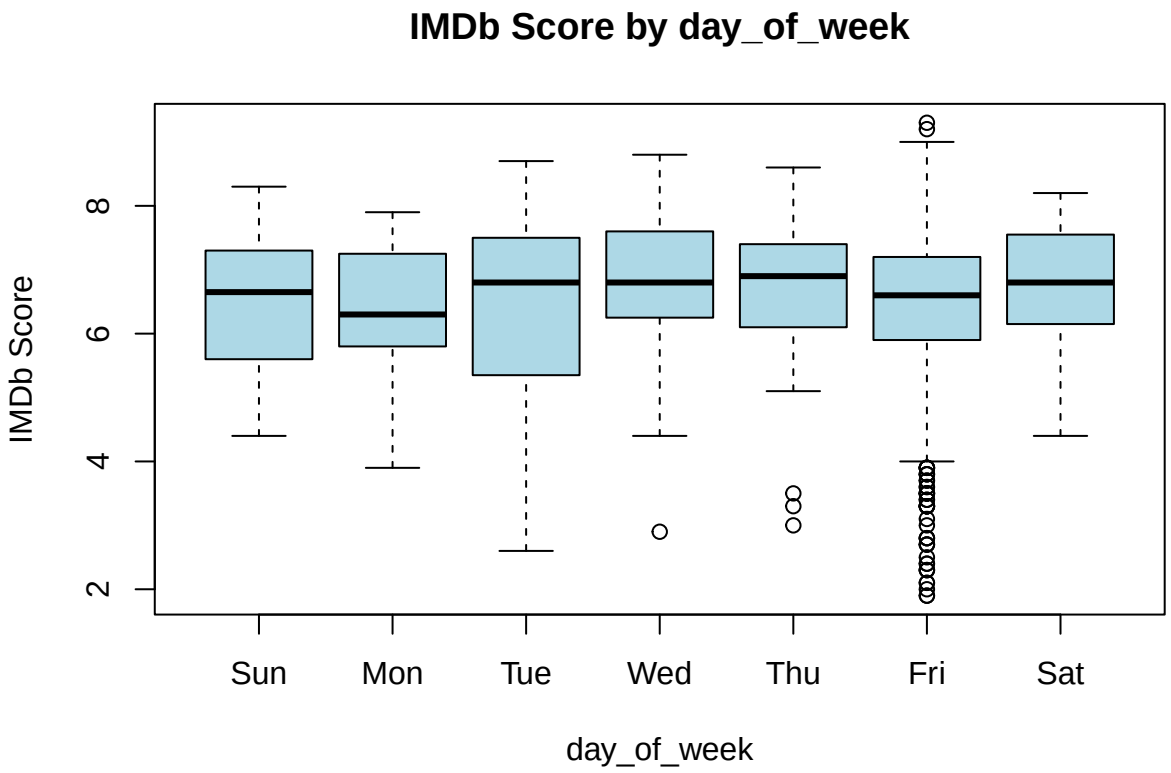
```
##   release_year release_month release_day day_of_week weekday_or_weekend
## 1         2014           Jan          10         Fri            Weekday
## 2         2003           Oct          24         Fri            Weekday
## 3         2005           Jan          14         Fri            Weekday
## 4         2012           Aug          20         Mon            Weekday
## 5         1979           Jun          22         Fri            Weekday
## 6         2006           Mar          17         Fri            Weekday
```

box plots

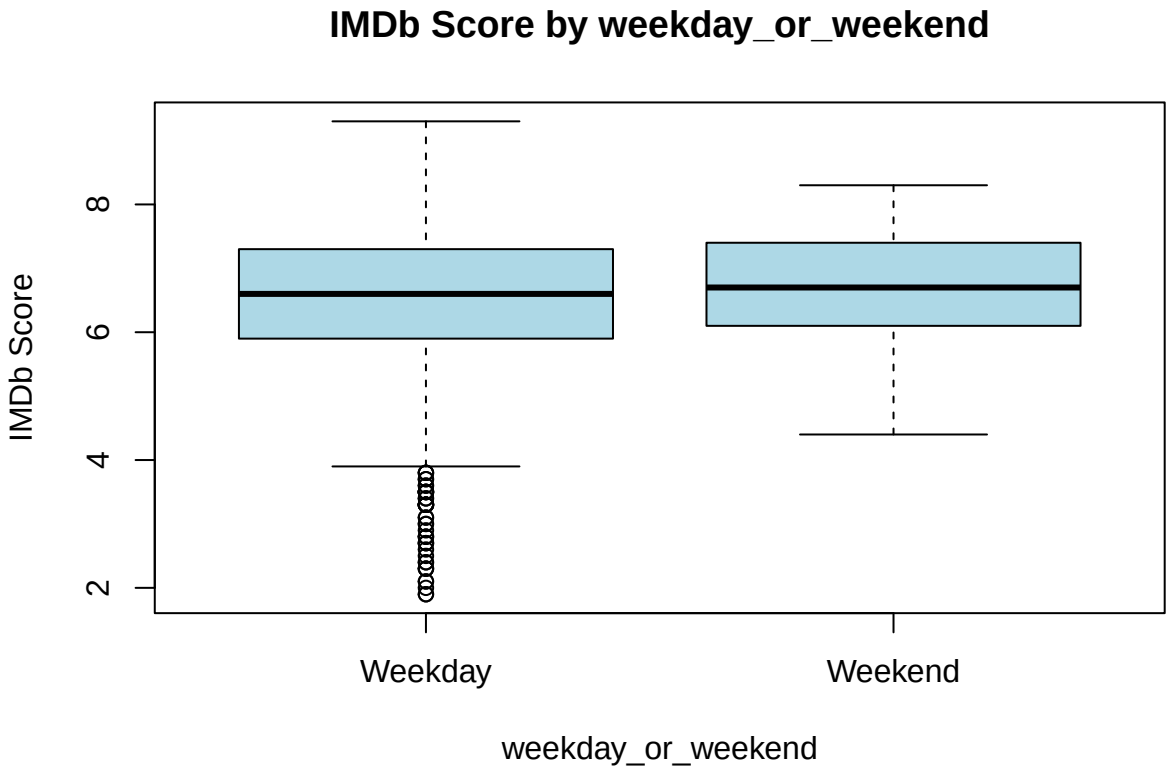
```
for (col in c("day_of_week", "weekday_or_weekend", "release_month", "release_day_group")) {
  print(table(df[col]))
  boxplot(df$imdb_score ~ df[[col]], main=paste("IMDb Score by", col), ylab="IMDb Score", xlab=col, col
```

```
}
```

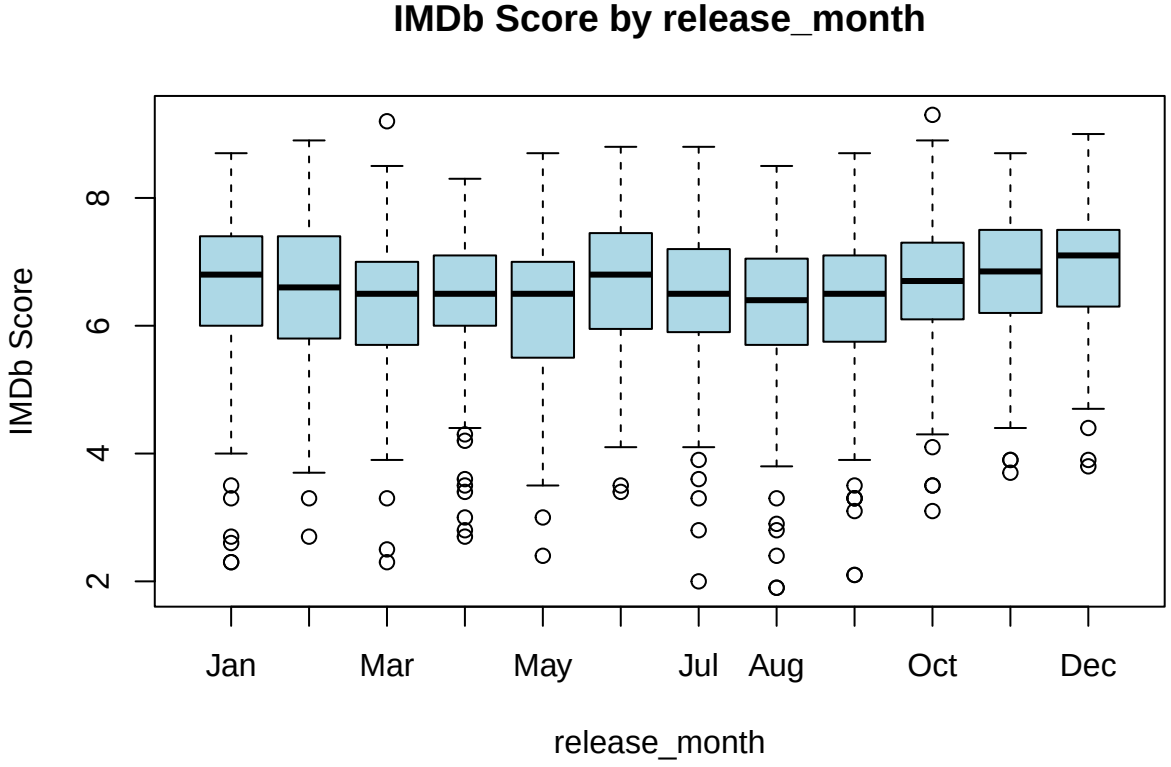
```
## day_of_week
##  Sun  Mon  Tue  Wed  Thu  Fri  Sat
##   38   24   55  179   67 1523   44
```




```
## weekday_or_weekend
## Weekday Weekend
##    1848      82
```

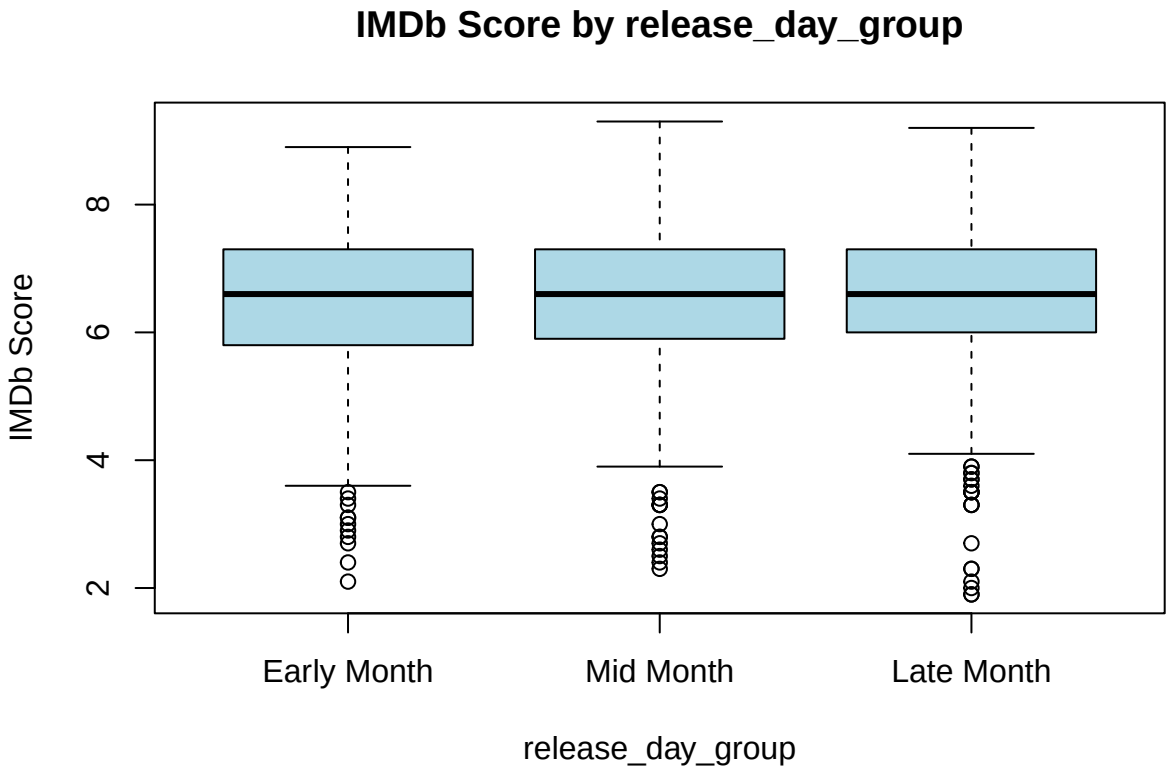



```
## release_month
## Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
## 205 155 157 169  94 147 149 172 187 216 148 131
```




```
## release_day_group
## Early Month   Mid Month  Late Month
```

```
##         561         694         675
```



pairwise t-test

Since the movies we're going to predict are all being released in Oct or Nov, it makes more sense to focus on time-related predictors at a granularity of month or smaller. Given that the day unit is too small, we've decided to just focus on two groups: 'weekend / weekday,' and 'early / mid / late month.'

```
time_cols <- c("day_of_week", "weekday_or_weekend", "release_day_group")

for (col in time_cols) {

    pairwise_results <- pairwise.t.test(df$imdb_score, df[[col]], p.adjust.method = "bonferroni")
    print(pairwise_results)

}
```

```
##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  df$imdb_score and df[[col]]
##
##     Sun    Mon    Tue    Wed    Thu    Fri
## Mon 1.0000 -      -      -      -      -
## Tue 1.0000 1.0000 -      -      -      -
## Wed 1.0000 1.0000 0.2871 -      -      -
## Thu 1.0000 1.0000 1.0000 1.0000 -      -
## Fri 1.0000 1.0000 1.0000 0.0013 1.0000 -
## Sat 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000
##
## P value adjustment method: bonferroni
```

```
##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  df$imdb_score and df[[col]]
##
##         Weekday
## Weekend 0.25
##
## P value adjustment method: bonferroni
##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  df$imdb_score and df[[col]]
##
##            Early Month Mid Month
## Mid Month  1           -
## Late Month 1           1
##
## P value adjustment method: bonferroni
```

Although there is no significant difference in IMDb scores between levels within each group, this interpretation relies solely on linear perspectives. In a similar fashion to previous predictors, retaining these variables could still be valuable for capturing potential interactions with other predictors. Also, they can enhance visual analysis when exploring trend patterns.

Drop useless columns

```
drop_release_cols <- c("date", "release_year", "release_month", "release_day")
df[drop_release_cols] <- NULL

colnames(df)
```

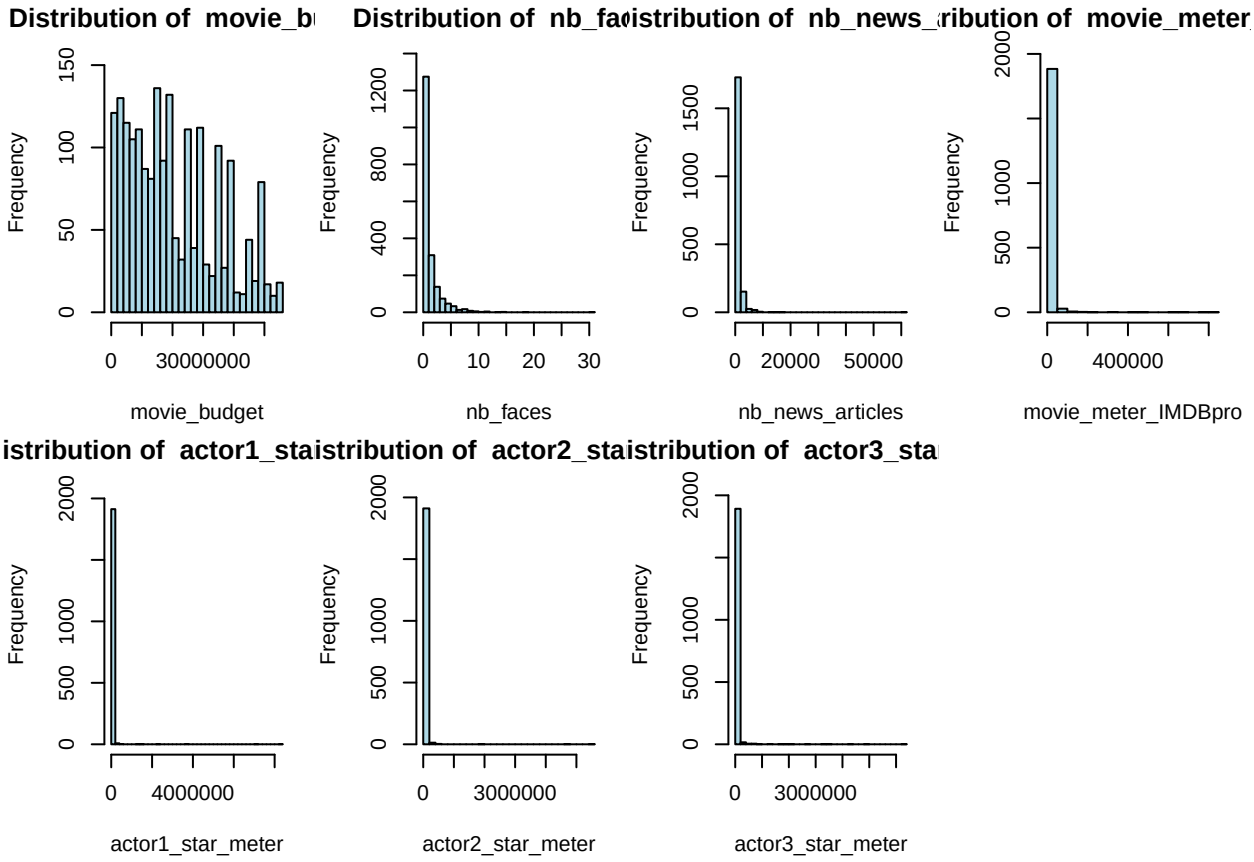
```
##  [1] "movie_id"               "imdb_score"             "movie_budget"
##  [4] "duration"               "language"               "country"
##  [7] "maturity_rating"        "aspect_ratio"           "nb_news_articles"
## [10] "actor1_star_meter"      "actor2_star_meter"      "actor3_star_meter"
## [13] "colour_film"            "nb_faces"               "action"
## [16] "adventure"              "scifi"                  "thriller"
## [19] "western"                "sport"                  "horror"
## [22] "drama"                  "war"                    "crime"
## [25] "movie_meter_IMDBpro"    "biography"              "comedy"
## [28] "sci-fi"                 "fantasy"                "history"
## [31] "family"                 "other_genre"            "continent"
## [34] "top_director"           "top_actor1"             "top_actor2"
## [37] "top_actor3"             "top_cinematographer"    "top_production_company"
## [40] "top_distributor"        "release_day_group"      "day_of_week"
## [43] "weekday_or_weekend"
```

EDA - numerical predictors Moving on to the numerical variables. Similar to the approach taken with categorical predictors, we conduct outlier detection, skewness, scaling and normalization, feature engineering, or data type conversion.

Histograms

```
num_cols <- c("movie_budget", "nb_faces", "nb_news_articles", "movie_meter_IMDBpro", "actor1_star_meter

options(scipen = 999)
```

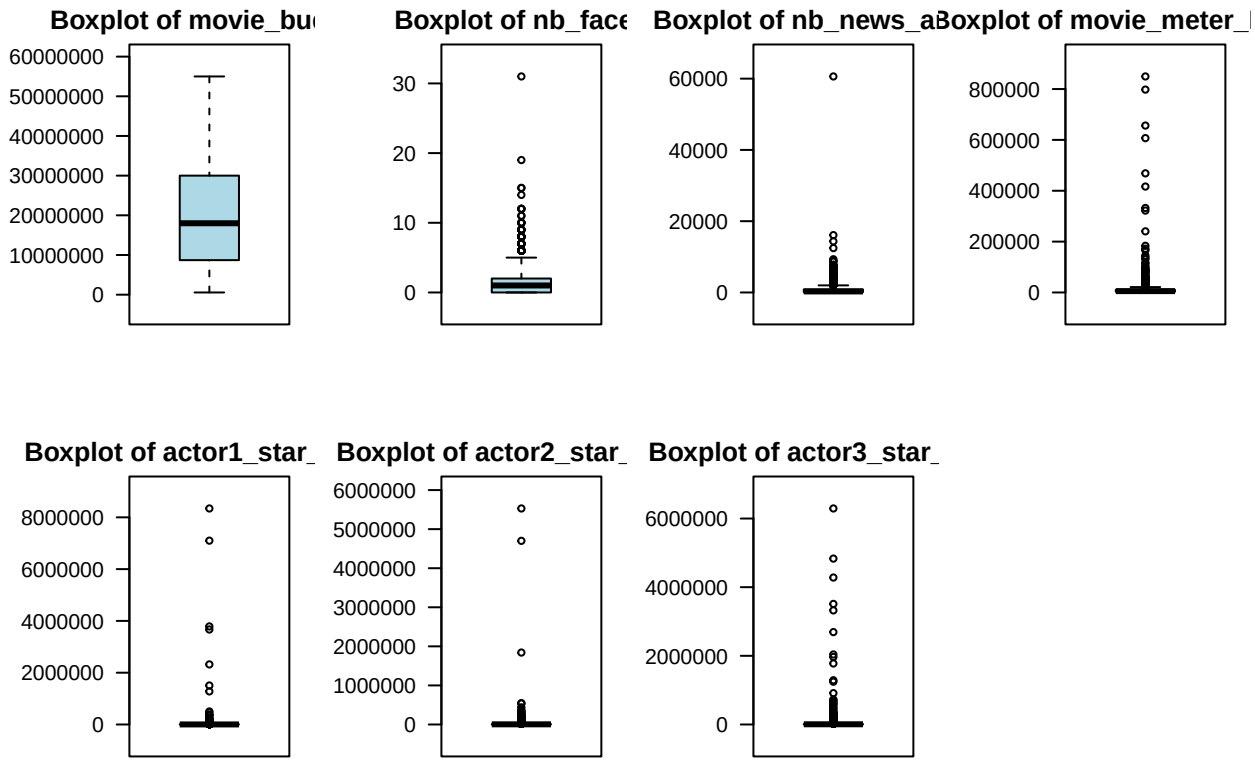
```r
par(mfrow = c(2, 4), mar = c(4, 4, 2, 1))

for (col in num_cols) {
  hist(df[[col]],
       main = paste("Distribution of ", col),
       xlab = col,
       col = "lightblue",
       breaks = 30,
       ylim = c(0, max(table(cut(df[[col]], breaks = 30)), na.rm = TRUE) * 1.1) # Adjust ylim to add so
  )
}

par(mfrow = c(1, 1), mar = c(5, 4, 4, 2) + 0.1)
```



Box Plots

```r
par(mfrow = c(2, 4), mar = c(4, 5, 2, 1), oma = c(0, 0, 0, 0))

for (col in num_cols) {

  boxplot(df[[col]],
          main = paste("Boxplot of", col),
          col = "lightblue",
          las = 2,
          ylim = range(df[[col]], na.rm = TRUE) + c(-1, 1) * diff(range(df[[col]], na.rm = TRUE)) * 0.1
          )
```

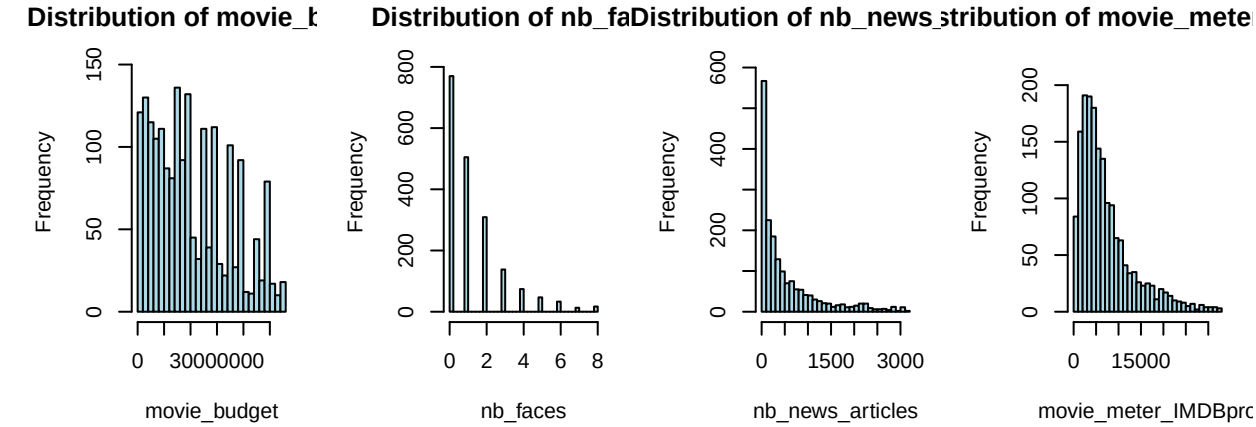
```
}

par(mfrow = c(1, 1), mar = c(5, 4, 4, 2) + 0.1)
```

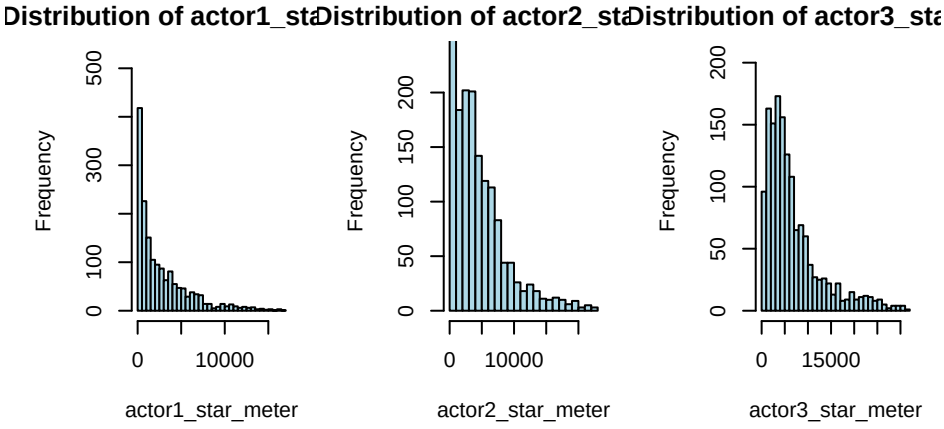


Except for movie_budget, all predictors show a significant right skew due to outliers. We thus remove those fall outside of 3 IQR from the dataset.

```
cleaned_df <- df

par(mfrow = c(2, 4), mar = c(4, 5, 2, 1), oma = c(0, 0, 0, 0))

for (col in num_cols) {

  Q1 <- quantile(cleaned_df[[col]], 0.25, na.rm = TRUE)
  Q3 <- quantile(cleaned_df[[col]], 0.75, na.rm = TRUE)
  IQR <- Q3 - Q1

  lower_bound <- Q1 - 3 * IQR
  upper_bound <- Q3 + 3 * IQR

  cleaned_df <- cleaned_df[cleaned_df[[col]] >= lower_bound & cleaned_df[[col]] <= upper_bound, ]

  hist(cleaned_df[[col]],
       main = paste("Distribution of", col),
       xlab = col,
       col = "lightblue",
       breaks = 30,
       ylim = c(0, max(table(cut(cleaned_df[[col]], breaks = 30)), na.rm = TRUE) * 1.1)
      )
}
```

```
par(mfrow = c(1, 1), mar = c(5, 4, 4, 2) + 0.1)
```






```
print(dim(cleaned_df)) #  dropped rows = 479, drop rate = 25%
```

```
## [1] 1451   43
```

log transformation

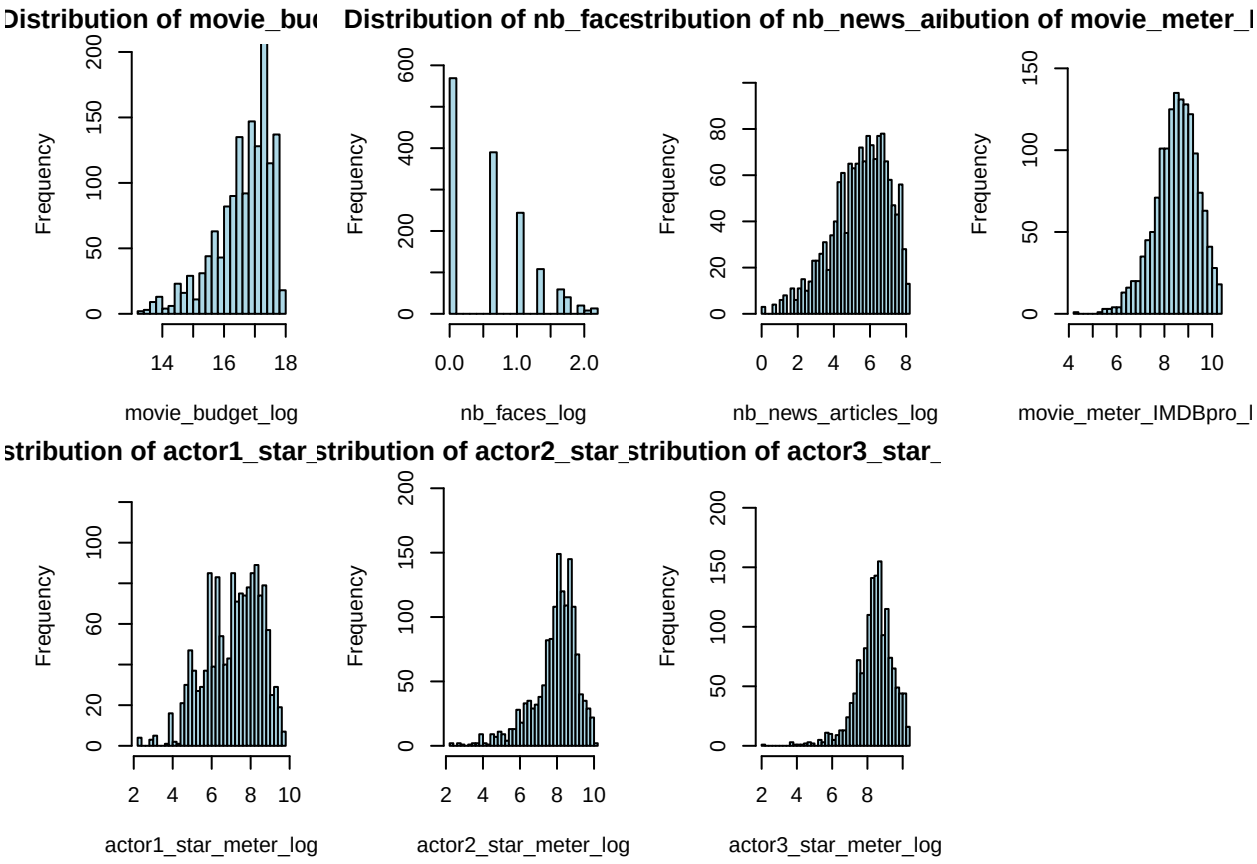
Since after removing 3 IQR, predictor are still right skewed, we decide to use log transformation to reduce skewness.

```
for (col in num_cols) {
  # use log (x + 1) since there're 0 in some predictors
  cleaned_df[[paste0(col, "_log")]] <- log(cleaned_df[[col]] + 1)
  }
```

histogram

```
log_num_cols <- paste0(num_cols, "_log")

par(mfrow = c(2, 4), mar = c(4, 5, 2, 1), oma = c(0, 0, 0, 0))

for (col in log_num_cols) {
 hist(cleaned_df[[col]],
         main = paste("Distribution of", col),
         xlab = col,
         col = "lightblue",
         breaks = 30,
```

```
        ylim = c(0, max(table(cut(cleaned_df[[col]], breaks = 30)), na.rm = TRUE) * 1.1) # Adjust ylim
      )
  }

par(mfrow = c(1, 1), mar = c(5, 4, 4, 2) + 0.1)
```



While distributions of some predictors are left skewed, but they appear to be relatively symmetric than extremely skewed originally after transformation.

Note:

- For the actor star meter variables, since they all show similar patterns, we might consider creating a single feature that combines this information to reduce dimensionality (could be like taking the average log star meter of the top 3 actors)
- For those left-skewed distributions after log transformation, we might consider:
  a) Using a different transformation such as square root, cube root, Box-Cox transformation. . . .
  b) Binning the data into categories.
  c) Using the raw data if our modeling technique doesn't follow normality (e.g., tree-based methods. . .)

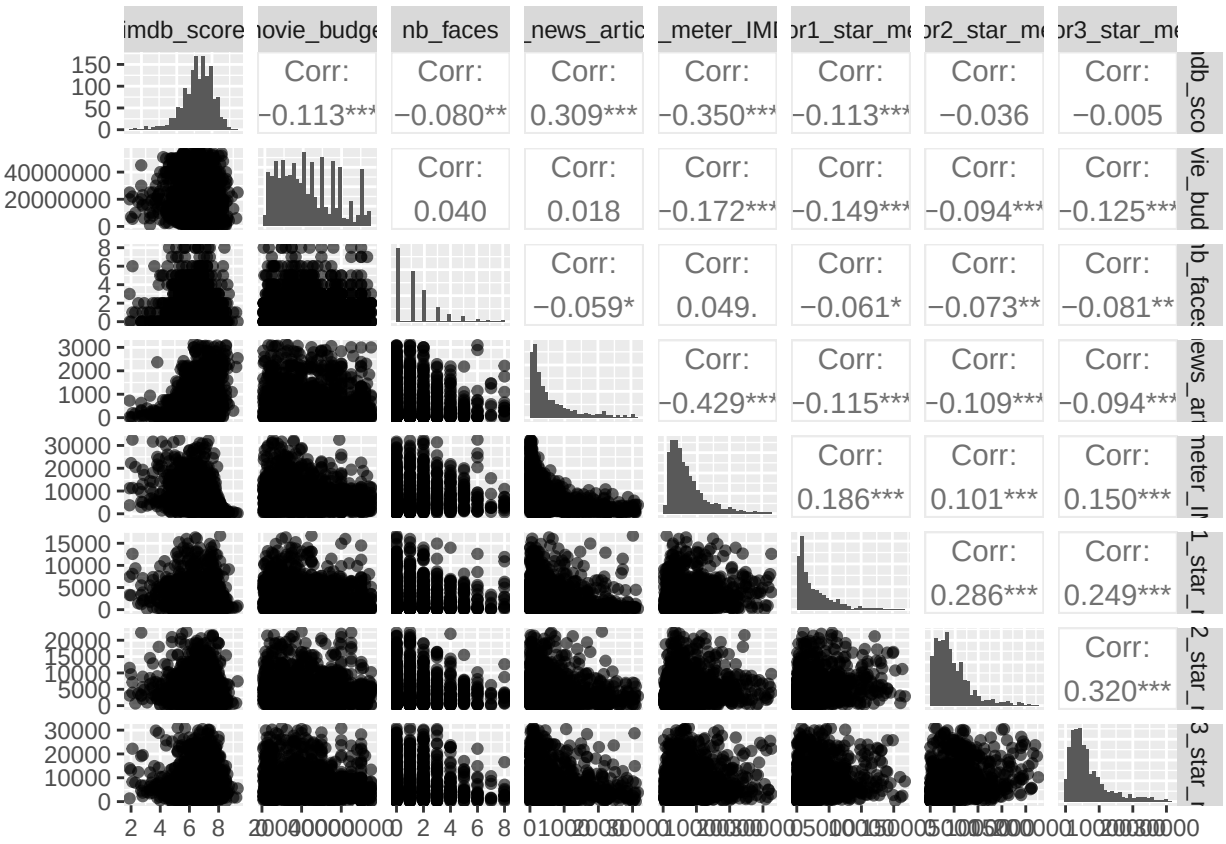
Scatter Plots & Correlation matrix

```
options(scipen=999)

ggpairs(cleaned_df[c("imdb_score", num_cols)],
        upper = list(continuous = wrap("cor", size = 4)),
        lower = list(continuous = wrap("points", alpha = 0.6)),
        diag = list(continuous = "barDiag"),
```

```
        progress = FALSE, message = FALSE, warnings = FALSE)
```

```
## Warning in warn_if_args_exist(list(...)): Extra arguments: 'message',
## 'warnings' are being ignored.  If these are meant to be aesthetics, submit them
## using the 'mapping' variable within ggpairs with ggplot2::aes or
## ggplot2::aes_string.
```

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



Except for the moderate negative correlation between the number of news articles and IMDbPro movie meter rankings, suggesting that movies with more media coverage tend to have better (lower) IMDbPro rankings, many of the other correlations are relatively weak.

The relationships between the target are not straightforward, indicate potential non-linear : movie_meter_IMDBpro, actor star meters shows some curvature;

We could further explore potential complex interactions between variables through cross-analysis.

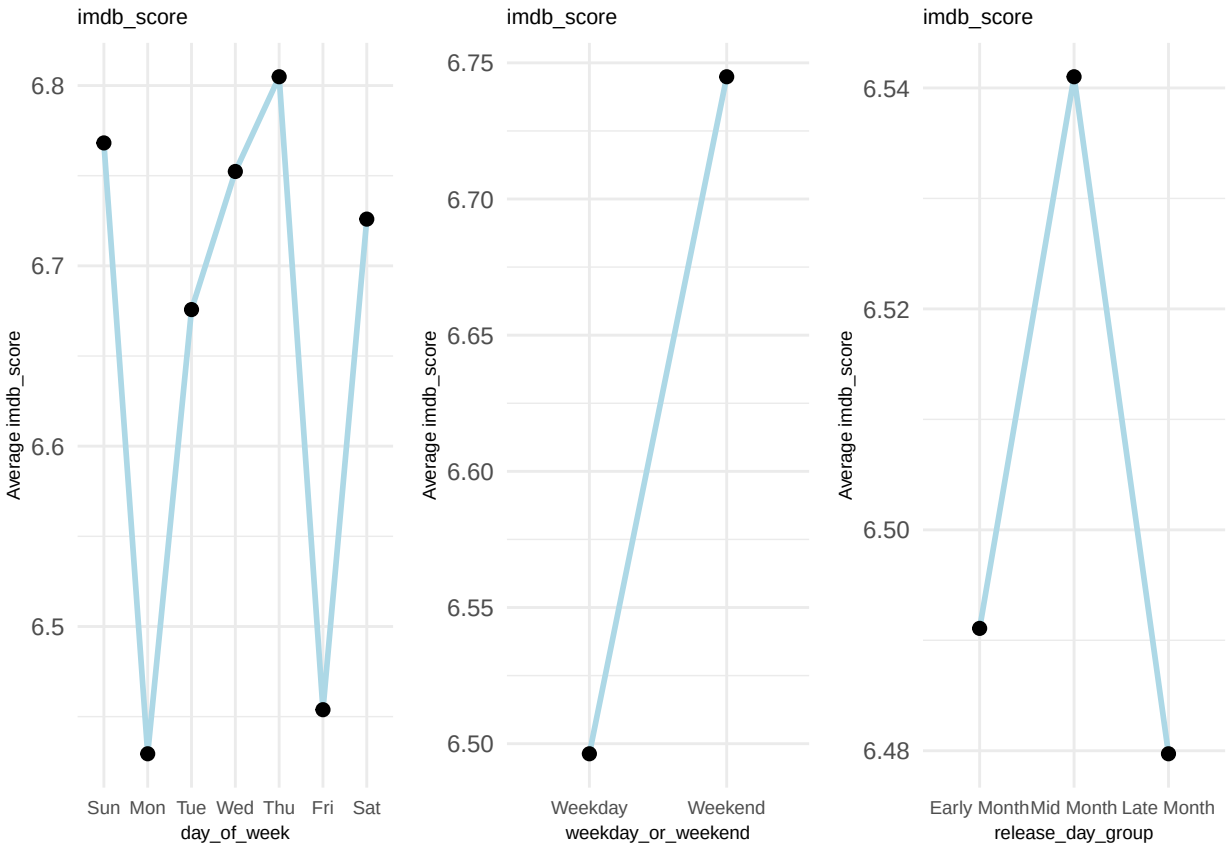
Trending Analysis Target variable

```
plot_list <- list()
```

```r
for (time in time_cols) {

  df_summary <- cleaned_df %>%
    group_by(.data[[time]]) %>%
    summarise(mean_value = mean(.data[["imdb_score"]], na.rm = TRUE))

  p <- ggplot(df_summary, aes_string(x = time, y = "mean_value", group = 1)) +
    geom_line(color = "lightblue", size = 1) +
    geom_point(color = "black", size = 2) +
    labs(title = "imdb_score",
         x = time,
         y = "Average imdb_score") +
    theme_minimal() +
    theme(
      plot.title = element_text(size = 8),
      axis.title.y = element_text(size = 7),
      axis.title.x = element_text(size = 7),
      axis.text.x = element_text(size = 7)
    )

  plot_list <- c(plot_list, list(p))
}
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```r
grid.arrange(grobs = plot_list, ncol = 3, nrow = 1)
```

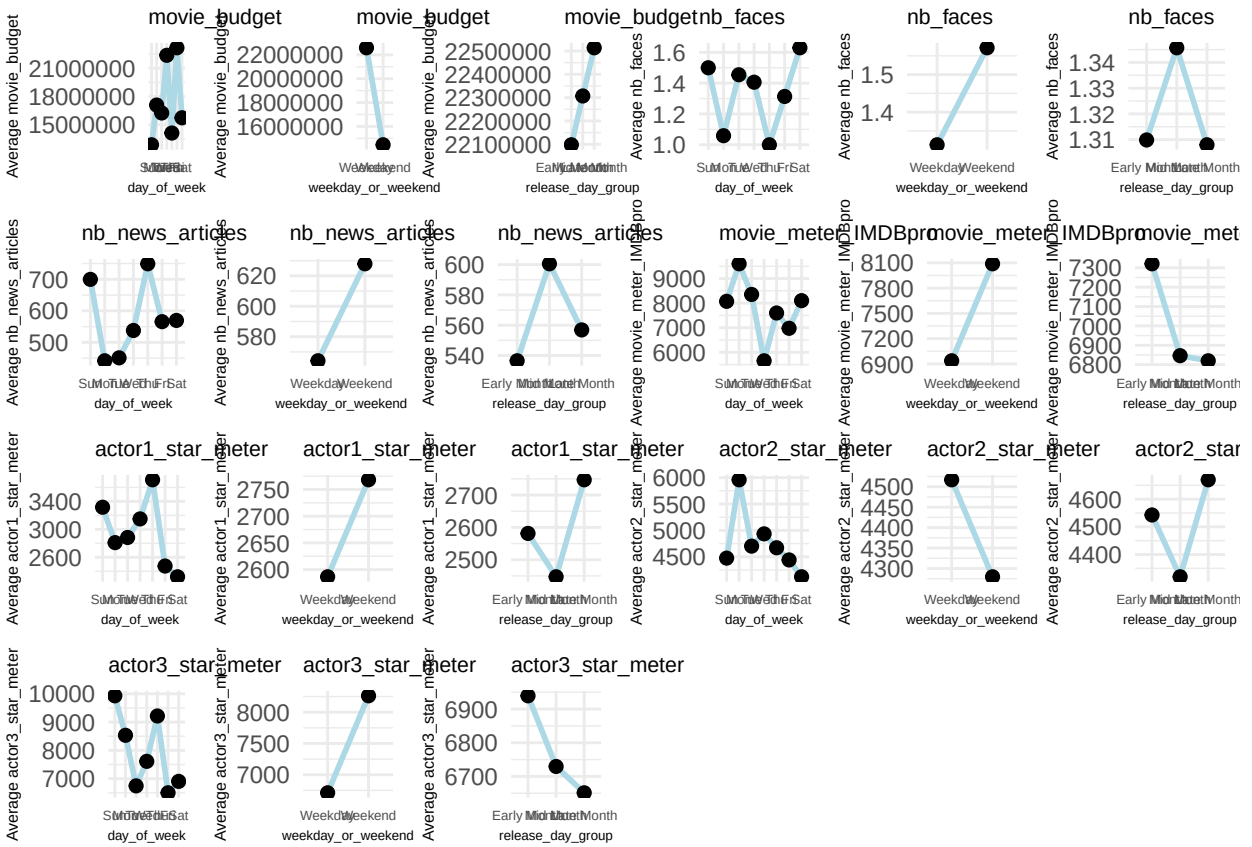


Movies released on weekends or mid-month tend to have higher rating scores, but their effect seems moderate.

Numerical predictors

```
plot_list <- list()

for (col in num_cols) {

  for (time in time_cols) {

    df_summary <- cleaned_df %>%
      group_by(.data[[time]]) %>%
      summarise(mean_value = mean(.data[[col]], na.rm = TRUE))

    p <- ggplot(df_summary, aes_string(x = time, y = "mean_value", group = 1)) +
      geom_line(color = "lightblue", size = 1) +
      geom_point(color = "black", size = 2) +
      labs(title = col,
           x = time,
           y = paste("Average", col)) +
      theme_minimal() +
      theme(
        plot.title = element_text(size = 8),
        axis.title.y = element_text(size = 6),
        axis.title.x = element_text(size = 5),
        axis.text.x = element_text(size = 5)
      )
```

```
    plot_list <- c(plot_list, list(p))
  }
}

grid.arrange(grobs = plot_list, ncol = 6, nrow = 4)
```



From the trend plots, it indicates that movies released during different timeframes may have varying effects on the predictors.

Answer those business questions

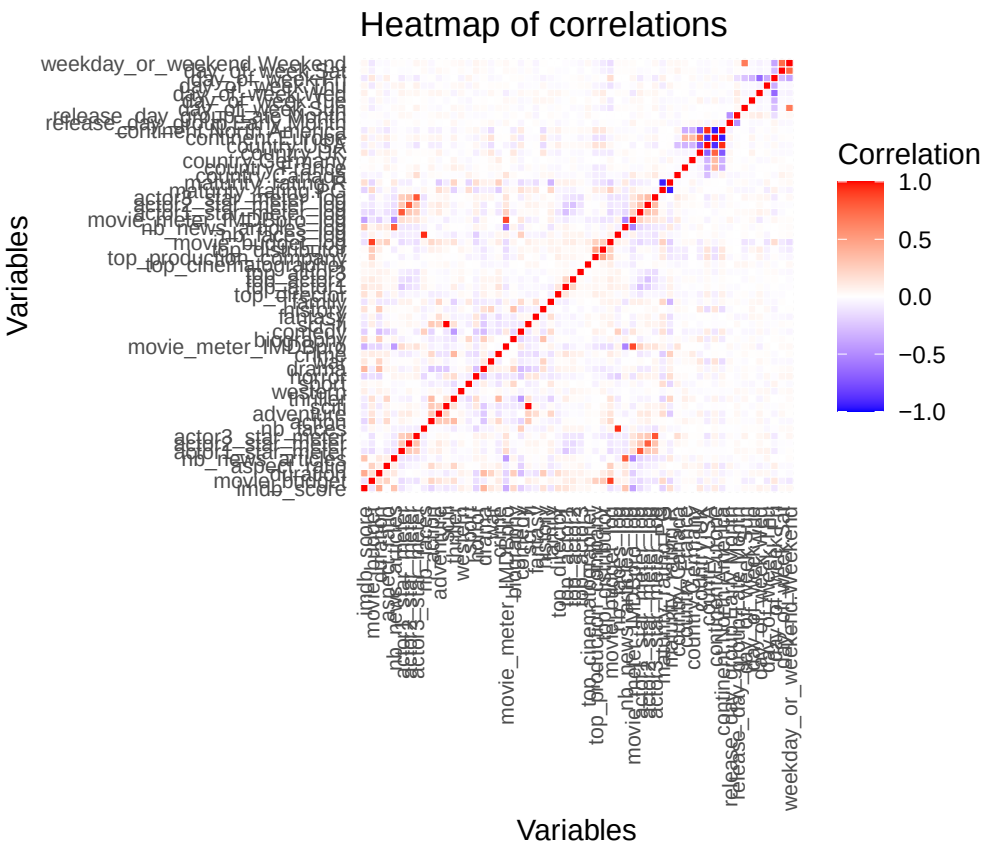
Q1. What predictors have high correlation with IMDb scores? Is there any possible collinearity between predictors?

correlation with target variables

```
options(contrasts = c("contr.treatment", "contr.treatment"))  # Use simple dummy coding for unordered f

factor_columns <- c("maturity_rating", "country", "continent", "release_day_group", "day_of_week", "weel

# dummy base setting
cleaned_df$maturity_rating <- relevel(as.factor(cleaned_df$maturity_rating), ref = "G")
cleaned_df$country <- relevel(as.factor(cleaned_df$country), ref = "other_countries")
cleaned_df$continent <- relevel(as.factor(cleaned_df$continent), ref = "other_continent")
cleaned_df$release_day_group <- relevel(as.factor(cleaned_df$release_day_group), ref = "Mid Month")
cleaned_df$day_of_week <- factor(cleaned_df$day_of_week, ordered = FALSE)
cleaned_df$day_of_week <- relevel(cleaned_df$day_of_week, ref = "Mon")
```

```
cleaned_df$weekday_or_weekend <- relevel(as.factor(cleaned_df$weekday_or_weekend), ref = "Weekday")

dummy <- dummyVars(~ ., data = cleaned_df[, factor_columns], fullRank = TRUE)  # fullRank = TRUE exclud

dummy_data <- predict(dummy, newdata = cleaned_df)
dummy_data <- as.data.frame(dummy_data)

numeric_cols <- cleaned_df[, !names(cleaned_df) %in% c("movie_id", "colour_film", "language", "maturity_
df_dummies <- cbind(numeric_cols, dummy_data)
```

```
correlation_matrix <- cor(df_dummies)

melted_correlation_matrix <- melt(correlation_matrix)

ggplot(melted_correlation_matrix, aes(x = Var1, y = Var2, fill = value)) +
  geom_tile(color = "white") +  # Color of the grid
  scale_fill_gradient2(low = "blue", high = "red", mid = "white",
                       midpoint = 0, limit = c(-1, 1), space = "Lab",
                       name="Correlation") +
  theme_minimal() +  # Use a minimal theme
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, size = 8, hjust = 1)) +
  theme(axis.text.y = element_text(size = 8)) +
  coord_fixed() +  # Fix the aspect ratio
  labs(title = "Heatmap of correlations", x = "Variables", y = "Variables")
```



Some predictors are kept in the dataset for performance testing later, so their high correlation is not a primary focus (e.g., continent and country, day of the week vs. weekday/weekend, log vs. non-log versions).

However, several important correlations are worth noting within specific predictor groups

1. Significant Correlations Within Predictor Levels

There are notable correlations between related variables such as actor star meters, country-specific variables, day-of-week variables, and time-of-month variables (late vs. early month). Also, there is a perfect negative correlation between maturity ratings (R vs. PG), indicating these categories are mutually exclusive, making it necessary to include only one to avoid multicollinearity.

2. IMDbPro vs. Number of News Articles

- Positive Correlation

The number of news articles and IMDbPro movie meter (log version) show a moderate-to-high positive correlation, indicating that movies with more media coverage tend to perform better in IMDbPro rankings. Since both metrics capture elements of movie visibility and popularity, they may exhibit collinearity. Testing their individual impact on the model will help identify which is more predictive.

- Negative Correlation

There's also a moderate-to-strong negative correlation between the number of news articles and IMDbPro (log). This suggests that while increased media attention may draw more visibility, it doesn't always translate into higher IMDbPro rankings, reflecting that more attention doesn't necessarily equate to better audience perceptions or ratings.

correlation with target variables

```
cor_with_target <- correlation_matrix[, "imdb_score"]

cor_with_target_sorted <- sort(cor_with_target, decreasing = TRUE)

cor_with_target_sorted
```

```
##                    imdb_score                      duration
##                   1.000000000                   0.433015742
##                         drama                nb_news_articles
##                   0.375767111                   0.308650039
##          nb_news_articles_log                     biography
##                   0.303789740                   0.202023926
##             maturity_rating.R                  top_director
##                   0.167992124                   0.153277779
##                       history                    top_actor1
##                   0.150056416                   0.141185613
##                           war                    country.UK
##                   0.122788083                   0.120618722
##               continent.Europe                       western
##                   0.103418698                   0.084314500
##                day_of_week.Wed                         crime
##                   0.072902199                   0.071060055
##                    top_actor2                day_of_week.Thu
##                   0.067846603                   0.048444595
##           top_cinematographer                         sport
##                   0.044560334                   0.042884264
##    weekday_or_weekend.Weekend                    top_actor3
##                   0.042499349                   0.036078113
##                day_of_week.Sun                day_of_week.Sat
##                   0.030940879                   0.028830618
##                day_of_week.Tue                top_distributor
##                   0.024697900                   0.019918482
```

```
##                country.France        top_production_company
##                     0.009104568                   0.005085957
##         actor3_star_meter_log               country.Germany
##                    -0.001398208                  -0.003852941
##             actor3_star_meter release_day_group.Early Month
##                    -0.004813908                  -0.008301512
##                    aspect_ratio  release_day_group.Late Month
##                    -0.011194516                  -0.017468456
##             actor2_star_meter                country.Canada
##                    -0.035917149                  -0.053305006
##                       adventure         actor2_star_meter_log
##                    -0.069835851                  -0.072162916
##                         fantasy                   country.USA
##                    -0.072358915                  -0.073769715
##                    nb_faces_log                      thriller
##                    -0.074315374                  -0.079852150
##                        nb_faces       continent.North America
##                    -0.080232465                  -0.094867631
##                day_of_week.Fri                         scifi
##                    -0.101087441                  -0.103990978
##                         sci-fi                  movie_budget
##                    -0.103990978                  -0.113055194
##             actor1_star_meter                        family
##                    -0.113398236                  -0.117493151
##              movie_budget_log         actor1_star_meter_log
##                    -0.130304344                  -0.149713291
##                          action            maturity_rating.PG
##                    -0.173889364                  -0.185668664
##                         horror                        comedy
##                    -0.187035755                  -0.206094621
##           movie_meter_IMDBpro       movie_meter_IMDBpro_log
##                    -0.350340455                  -0.406532456
```

Positive correlation 1. Strong (> 0.3): duration, drama, number of news articles 2. Moderate (> 0.15): biography, maturity_rating.R, top_director, history, history, war, country.UK

Negative correlation 1. Strong (< -0.3): movie_meter_IMDBpro, No matter it's non-log or log version 2. Moderate (< -0.15): comedy, horror, maturity_rating.PG

Many other predictors show moderate to weak correlations, suggesting they may not have a strong, direct impact on IMDb scores individually. However, these weaker predictors could still contribute valuable insights through interactions or combinations with other features in the model, or in non-linear models.

Duration, drama, and number of news articles are the strongest positive influences, while the movie meter and certain genres or maturity ratings show the strongest negative effects. These insights suggest that they are likely to play a crucial role in the linear predictive model.

Q2. Does the release timing (day of the week, weekday or weekend, part of the month) affect IMDb scores?

Movies released on weekends or mid-month tend to have higher rating scores, but their effect seems moderate.

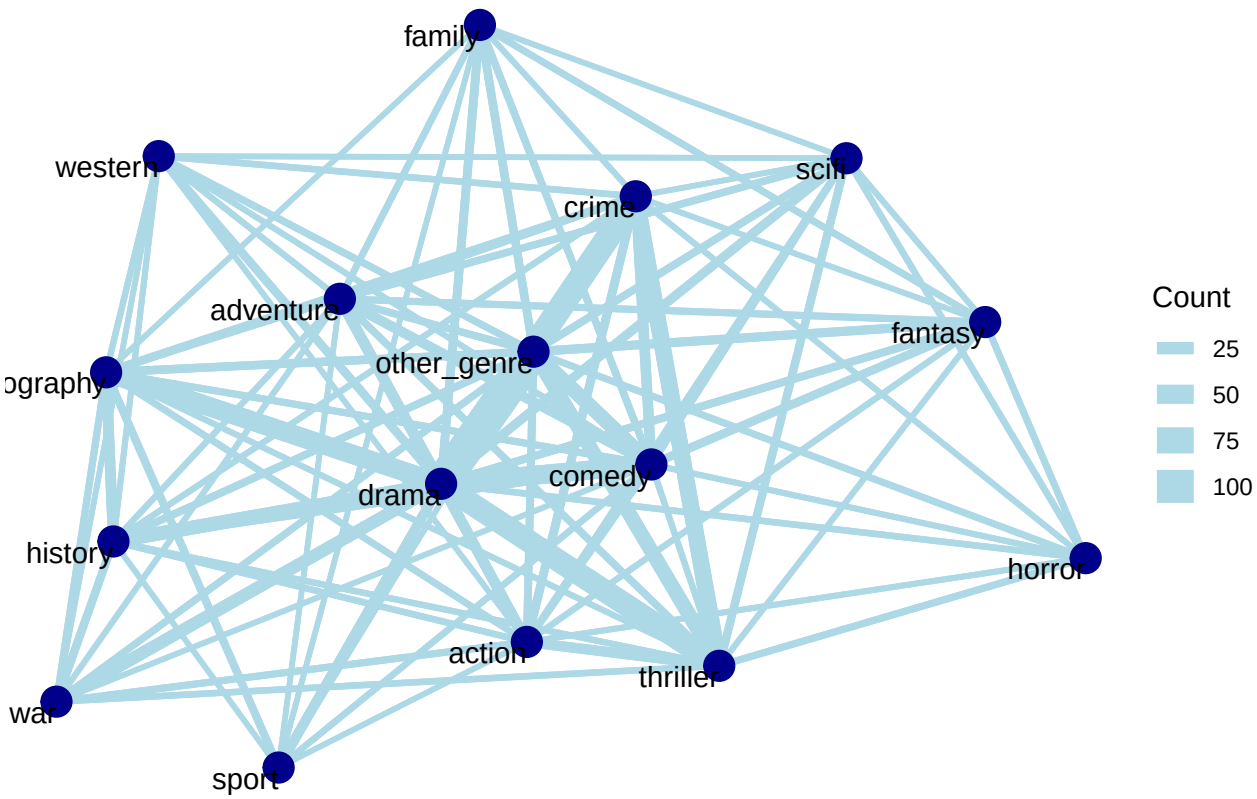
Q3. How does the combination of multiple genres in a movie affect its IMDb score?

network graph

```
genres_retained <- c("action", "adventure", "scifi", "thriller", "western",
                     "sport", "horror", "drama", "war", "crime", "biography",
```

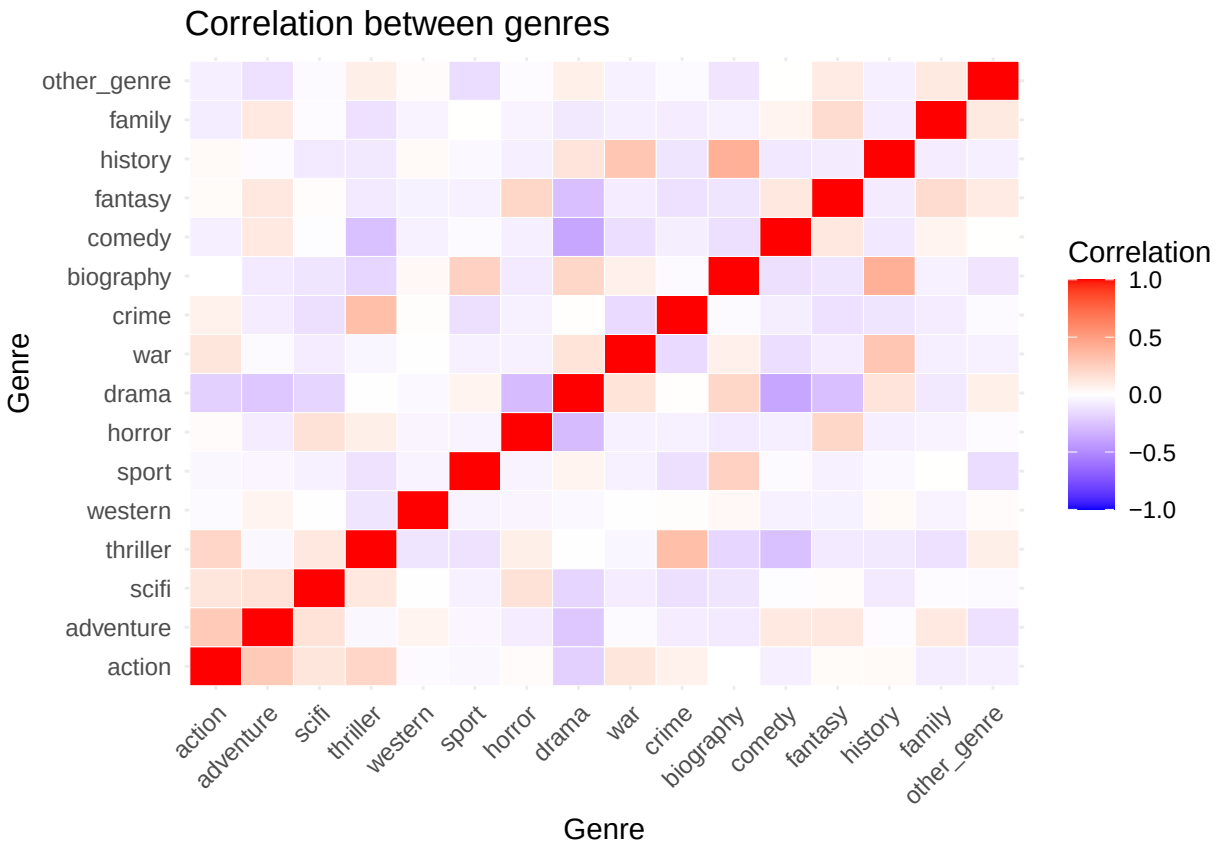
```
                    "comedy", "fantasy", "history", "family", "other_genre")

high_rated_movies <- cleaned_df %>% filter(imdb_score >= 7.3) # 3rd Qu = 7.3

genre_matrix <- high_rated_movies %>% select(all_of(genres_retained))

# co-occurrence matrix calculation
co_occurrence_matrix <- as.matrix(t(genre_matrix)) %*% as.matrix(genre_matrix)

co_occurrence_df <- melt(co_occurrence_matrix) # need to convert to a dataframe
colnames(co_occurrence_df) <- c("Genre1", "Genre2", "Count")

# remove 0 and self-pairings
co_occurrence_df <- co_occurrence_df %>% filter(Count > 0 & Genre1 != Genre2)

graph <- graph_from_data_frame(co_occurrence_df, directed = FALSE)

ggraph(graph, layout = "fr") +
  geom_edge_link(aes(edge_width = Count), edge_colour = "lightblue") +
  geom_node_point(size = 5, color = "darkblue") +
  geom_node_text(aes(label = name), vjust = 1, hjust = 1, size = 4) +
  theme_void() +
  labs(title = "Co-occurrence network of genres in high rated movies")
```

```
## Warning: The `trans` argument of `continuous_scale()` is deprecated as of ggplot2 3.5.0.
## i Please use the `transform` argument instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



correlation heatmap

```
genre_corr_matrix <- cor(genre_matrix)

ggplot(melt(genre_corr_matrix), aes(x = Var1, y = Var2, fill = value)) +
  geom_tile(color = "white") +
  scale_fill_gradient2(low = "blue", high = "red", mid = "white", midpoint = 0, limit = c(-1, 1), space
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, vjust = 1, hjust=1)) +
  labs(title = "Correlation between genres",
       x = "Genre",
       y = "Genre")
```



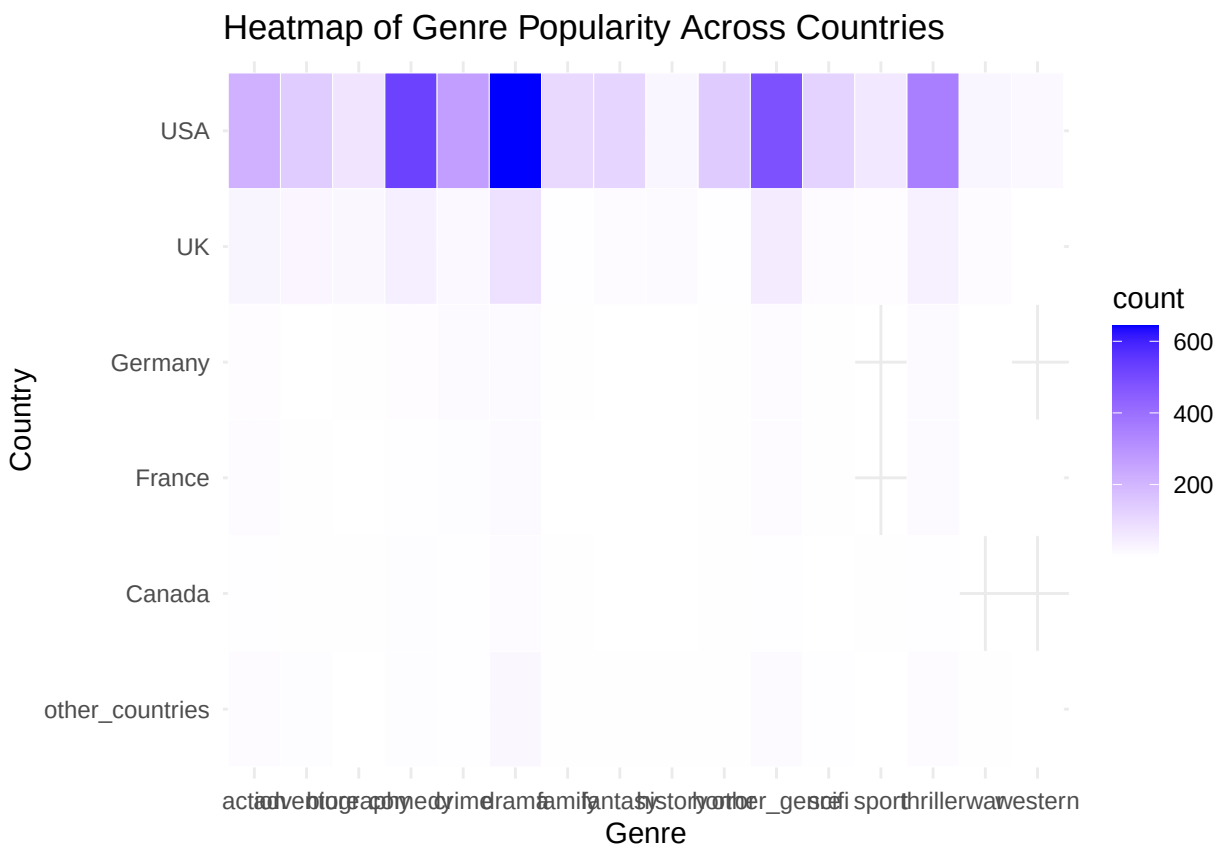
Some types of genres often co-occurring in high-rated movies. This suggests that introducing interaction terms for strongly correlated genre pairs could potentially be valuable for the model. (Action-Adventure, Action-Sci-Fi, Adventure-Sci-Fi, History-Biography, War-Biography, Thriller-Crime, Thriller-Action)

(*** other potential approach: add athother feature "genre diversity" by counting the number of genres per movie)

Q4. How does genre popularity differ across regions?

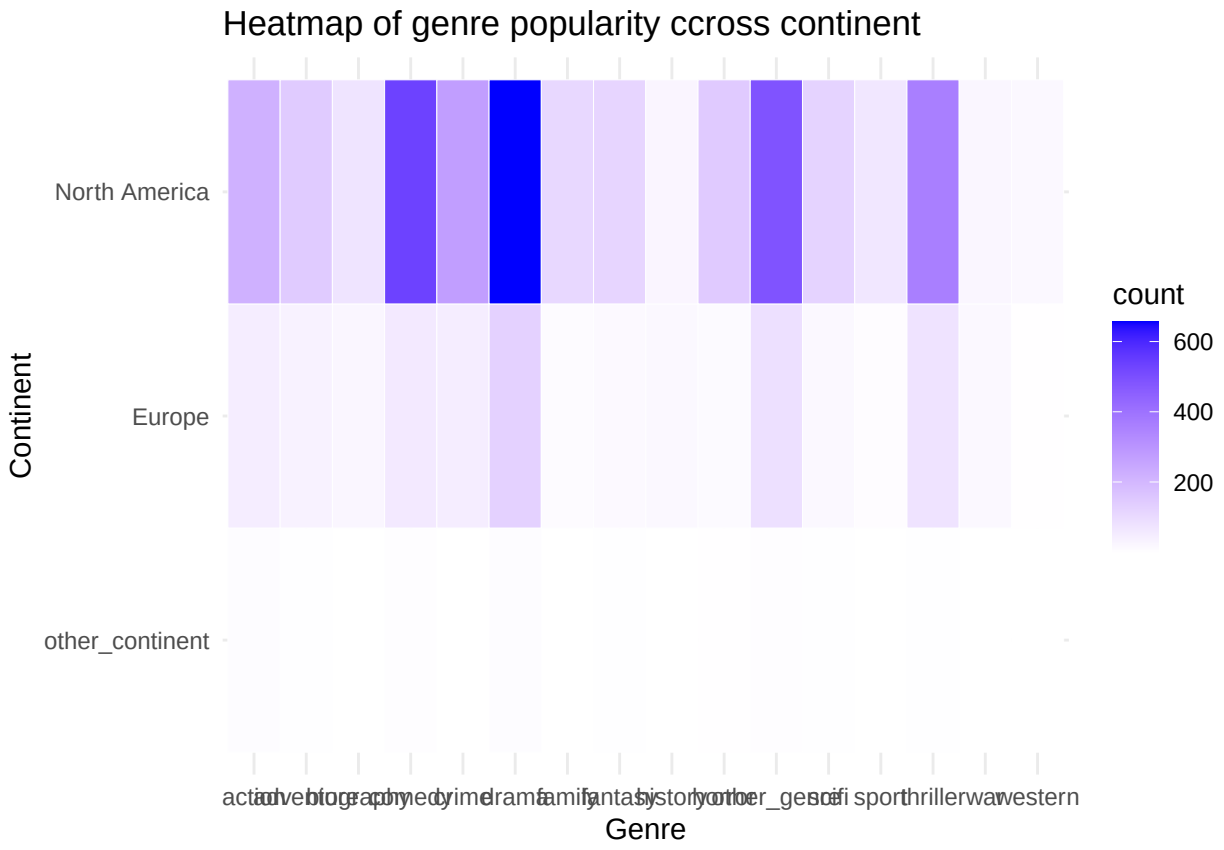
country

```
df_genres <- pivot_longer(cleaned_df, cols = all_of(genres_retained), names_to = "genre", values_to = "g
df_genres <- df_genres[df_genres$genre_indicator == 1, ]

genre_country_summary <- df_genres %>%
  group_by(country, genre) %>%
  summarise(count = n(), .groups = 'drop')

ggplot(genre_country_summary, aes(x = genre, y = country, fill = count)) +
  geom_tile(color = "white") +
  scale_fill_gradient(low = "white", high = "blue") +
  theme_minimal() +
  labs(title = "Heatmap of Genre Popularity Across Countries", x = "Genre", y = "Country")
```



continent

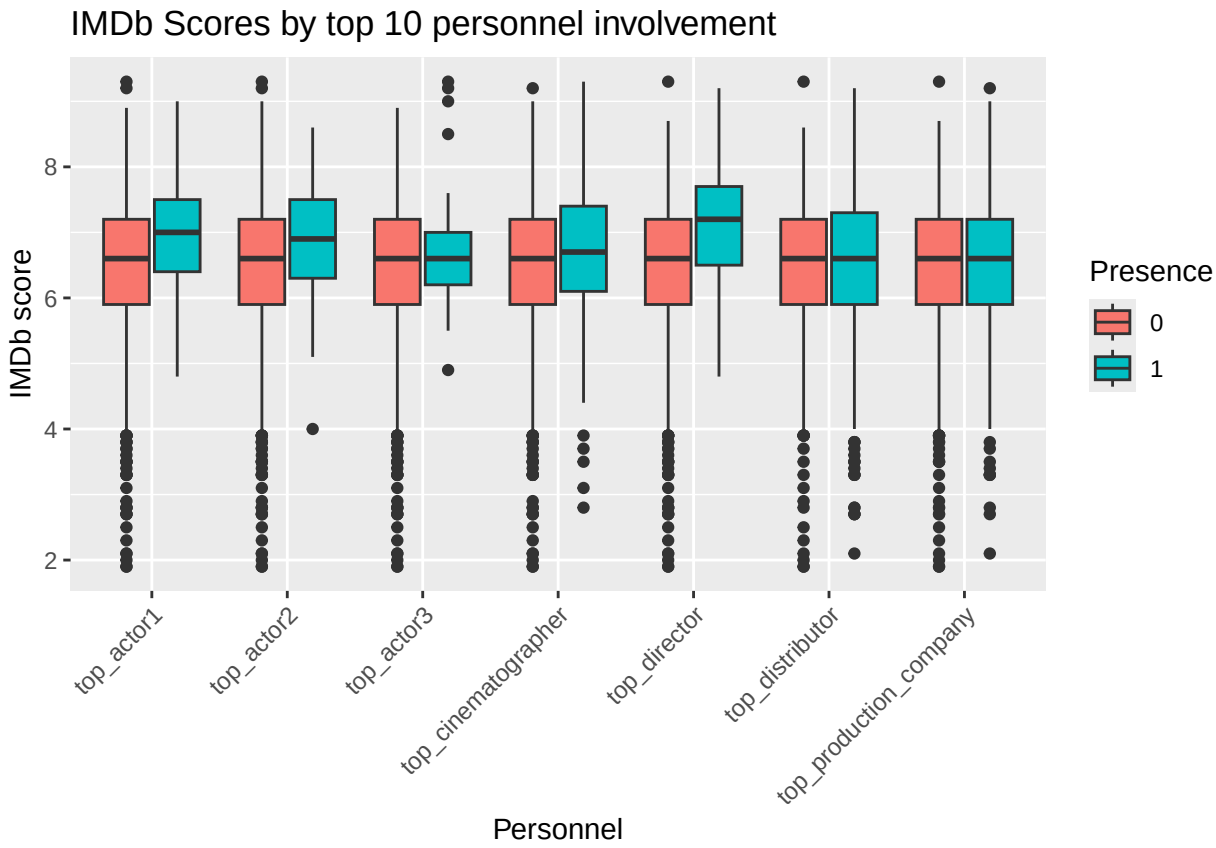
```
genre_continent_summary <- df_genres %>%
  group_by(continent, genre) %>%
  summarise(count = n(), .groups = 'drop')

ggplot(genre_continent_summary, aes(x = genre, y = continent, fill = count)) +
  geom_tile(color = "white") +
  scale_fill_gradient(low = "white", high = "blue") +
  theme_minimal() +
  labs(title = "Heatmap of genre popularity ccross continent", x = "Genre", y = "Continent")
```



Due to the impact of sample size, the findings from the heatmaps are likely to be unreliable. To handle class imbalance, we might consider techniques like weighted sampling or stratification to ensure that the influence of different regions is properly balanced in the model.

Q5. How does the presence of top 10 personnel impact IMDb scores?

```
imdb_scores_by_personnel <- df_dummies %>%
  pivot_longer(cols = starts_with("top_"), names_to = "Personnel", values_to = "Presence")

ggplot(imdb_scores_by_personnel, aes(x = Personnel, y = imdb_score, fill = factor(Presence))) +
  geom_boxplot() +
  labs(title = "IMDb Scores by top 10 personnel involvement",
       x = "Personnel",
       y = "IMDb score",
       fill = "Presence") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



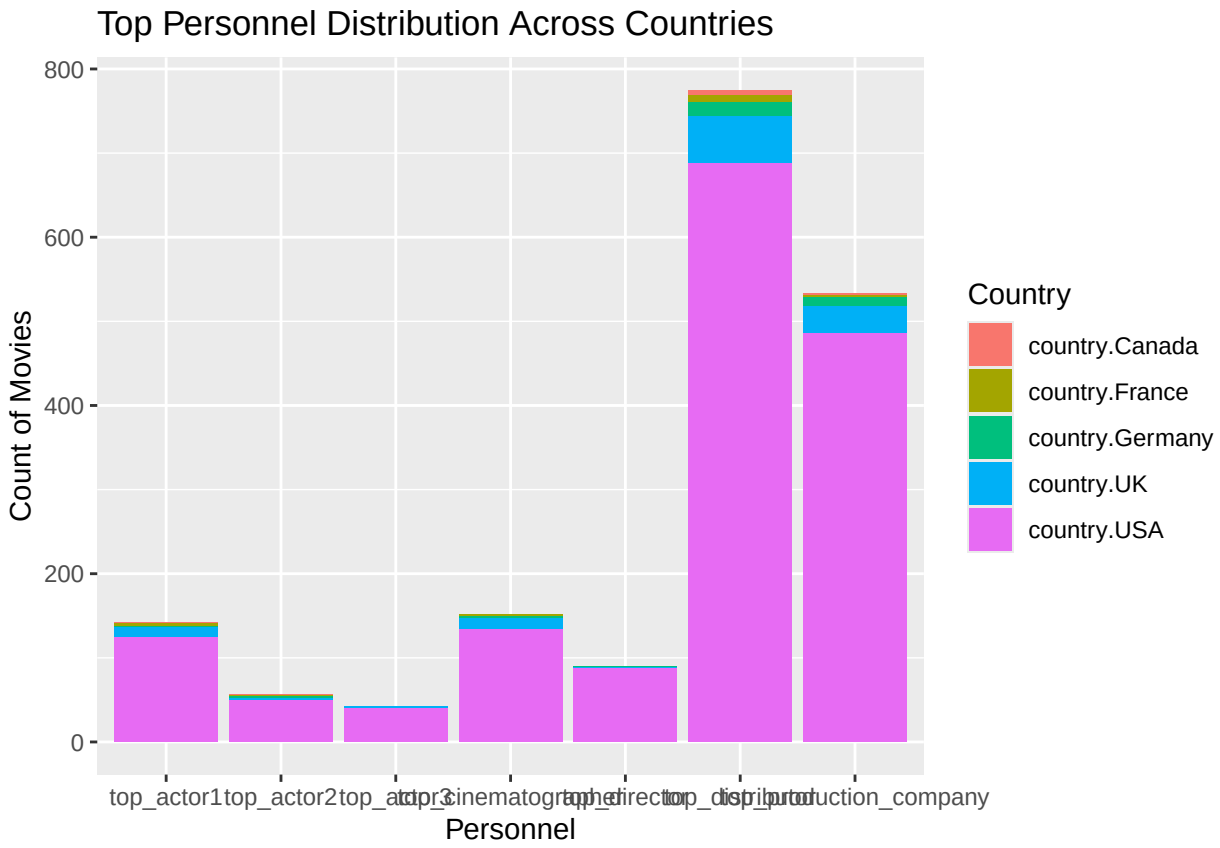
The involvement of top 10 personnel, particularly directors and lead actors, generally correlates with higher IMDb scores. This indicates that including top personnel as features might contribute to predictive model, supporting our approach to feature engineering for personnel-related variables.

Q6. Look into movies associated with the top 10 personnel for each predictor. Do different personnel associate with certain countries? What types of movie genres tend to have the highest average IMDb scores among these top personnel?

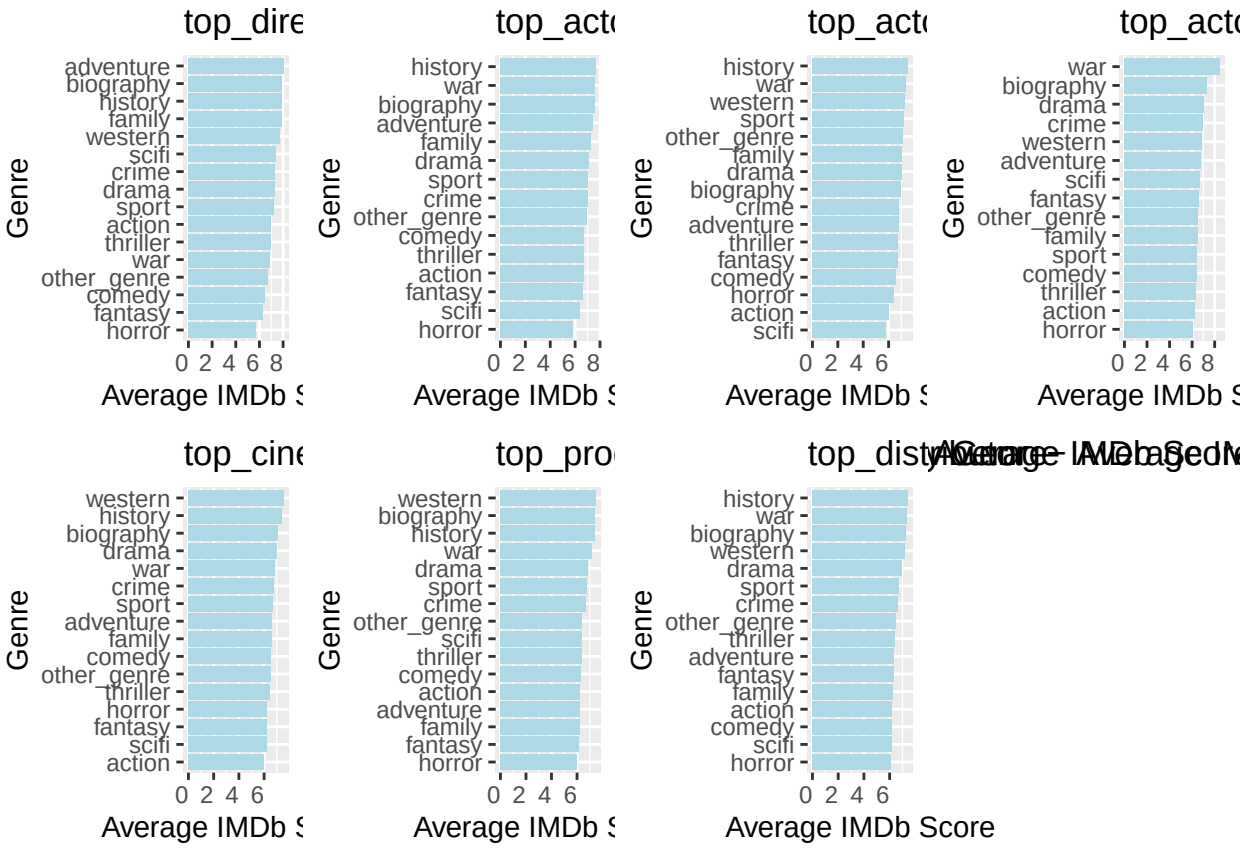
```
country_by_personnel <- df_dummies %>%
  pivot_longer(cols = starts_with("top_"), names_to = "Personnel", values_to = "Presence") %>%
  filter(Presence == 1) %>%
  group_by(Personnel) %>%
  summarise(across(starts_with("country"), sum))

country_by_personnel_melted <- melt(country_by_personnel)
```

```
## Using Personnel as id variables
```

```
ggplot(country_by_personnel_melted, aes(x = Personnel, y = value, fill = variable)) +
  geom_bar(stat = "identity", position = "stack") +
  labs(title = "Top Personnel Distribution Across Countries", x = "Personnel", y = "Count of Movies", f
```




```
top_personnels <- c("top_director", "top_actor1", "top_actor2", "top_actor3", "top_cinematographer", "to

plot_list <- list()

for (personnel in top_personnels) {
  df_personnel <- cleaned_df %>% filter(!!sym(personnel) == 1)

  df_long <- df_personnel %>%
    pivot_longer(cols = all_of(genres_retained), names_to = "genre", values_to = "genre_indicator") %>%
    filter(genre_indicator == 1)

  genre_avg_scores <- df_long %>%
    group_by(genre) %>%
    summarise(avg_score = mean(imdb_score, na.rm = TRUE), .groups = 'drop') %>%
    arrange(desc(avg_score))  # Sort by average score in descending order

  p <- ggplot(genre_avg_scores, aes(x = reorder(genre, avg_score), y = avg_score)) +
    geom_bar(stat = "identity", fill = "lightblue") +
    labs(title = paste(personnel, " - Average IMDb Scores by Genre"), x = "Genre", y = "Average IMDb Sc
    theme(axis.text.x = element_text(hjust = 1)) +
    coord_flip()

  plot_list[[personnel]] <- p
}

grid.arrange(grobs = plot_list, ncol = 4, nrow = ceiling(length(plot_list) / 4))
```



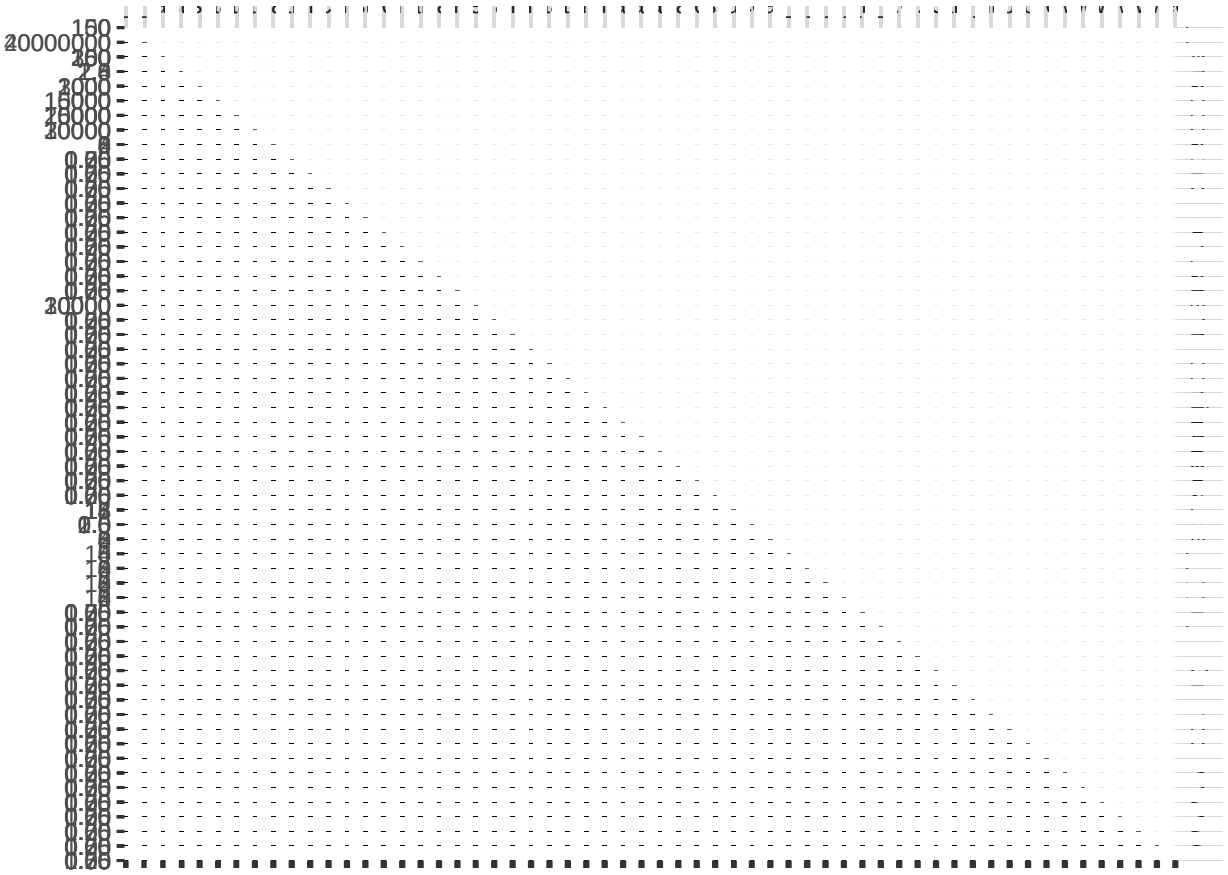
As mentioned above, regarding country distributions may be unreliable due to imbalanced country sample size. However, for genre preferences, some of which tend to rank high in average IMDb scores across different personnel categories (eg., history, western). This suggests that introducing interaction terms for specific pairs could potentially be valuable for the model. (top_director-adventure, top_actor1-history, top_actor2-history, top_actor3-war, top_cinematographer-western, top_production_company-western, top_distributor-history)

Regression Analysis Overview (heteroskedasticity, collinearity) Heteroskedasticity test

```
X <- colnames(df_dummies)[colnames(df_dummies) != "imdb_score"]

# Set options to suppress scientific notation
options(scipen=999)

# Create the ggpairs plot without the ignored arguments
ggpairs(df_dummies[c("imdb_score", X)],
        upper = list(continuous = wrap("cor", size = 4)),
        lower = list(continuous = wrap("points", alpha = 0.6)),
        diag = list(continuous = "barDiag"),
        progress = FALSE)
```

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

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



Heteroskedasticity test

```
library(car)
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:psych':
##
##     logit
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
no_log_cols <- c("movie_budget", "nb_faces", "movie_meter_IMDBpro", "actor1_star_meter", "actor2_star_m

# No log transformation, excluding continent and day_of_week
X_noLog_country_weekday_or_weekend <- df_dummies[
  !grepl("_log$", colnames(df_dummies)) &
  !grepl("^continent", colnames(df_dummies)) &
  !grepl("^day_of_week", colnames(df_dummies))
]

# No log transformation, including country and excluding day_of_week
X_noLog_country_day_of_week <- df_dummies[
  !grepl("_log$", colnames(df_dummies)) &
```

```
  !grepl("^continent", colnames(df_dummies)) &
  !grepl("^day_of_week", colnames(df_dummies))
]

# No log transformation, including continent and excluding weekday_or_weekend
X_noLog_continent_weekday_or_weekend <- df_dummies[
  !grepl("_log$", colnames(df_dummies)) &
  !grepl("^country", colnames(df_dummies)) &
  !grepl("^weekday_or_weekend$", colnames(df_dummies))
]

# No log transformation, including continent and excluding day_of_week
X_noLog_continent_day_of_week <- df_dummies[
  !grepl("_log$", colnames(df_dummies)) &
  !grepl("^country", colnames(df_dummies)) &
  !grepl("^day_of_week", colnames(df_dummies))
]

# Log transformation, excluding continent and including weekday_or_weekend
X_Log_country_weekday_or_weekend <- df_dummies[
  !grepl("_log$", colnames(df_dummies)) &
  !grepl("^continent", colnames(df_dummies)) &
  !grepl("^day_of_week", colnames(df_dummies))
]

# Log transformation, including country and excluding day_of_week
X_Log_country_day_of_week <- df_dummies[
  !grepl("_log$", colnames(df_dummies)) &
  !grepl("^continent", colnames(df_dummies)) &
  !grepl("^day_of_week", colnames(df_dummies))
]

# Log transformation, including continent and excluding weekday_or_weekend
X_Log_continent_weekday_or_weekend <- df_dummies[
  !grepl("_log$", colnames(df_dummies)) &
  !grepl("^country", colnames(df_dummies)) &
  !grepl("^weekday_or_weekend$", colnames(df_dummies))
]

# Log transformation, including continent and excluding day_of_week
X_Log_continent_day_of_week <- df_dummies[
  !grepl("_log$", colnames(df_dummies)) &
  !grepl("^country", colnames(df_dummies)) &
  !grepl("^day_of_week", colnames(df_dummies))
]

# Create a list of data frames for the models
df_lists <- list(
  X_noLog_country_weekday_or_weekend,
  X_noLog_country_day_of_week,
  X_noLog_continent_weekday_or_weekend,
  X_noLog_continent_day_of_week,
  X_Log_country_weekday_or_weekend,
```

```
  X_Log_country_day_of_week,
  X_Log_continent_weekday_or_weekend,
  X_Log_continent_day_of_week
)

heteroskedastic_predictors_list <- list()

for (df_index in seq_along(df_lists)) {
  df <- df_lists[[df_index]]
  heteroskedastic_predictors <- c()

  for (pred in colnames(df)[-1]) {

    lm_model <- lm(imdb_score ~ get(pred), data = df)

    test_result <- ncvTest(lm_model)

    # threshold = 0.05
    if (test_result$p < 0.05) {
      heteroskedastic_predictors <- c(heteroskedastic_predictors, pred)
    }
  }

  heteroskedastic_predictors_list[[df_index]] <- heteroskedastic_predictors
}

for (i in seq_along(heteroskedastic_predictors_list)) {
  cat(sprintf("Heteroskedastic predictors for data frame %d:\n", i))
  print(heteroskedastic_predictors_list[[i]])
  cat("============================================================\n")
}
```

```
## Heteroskedastic predictors for data frame 1:
##  [1] "movie_budget"           "duration"               "aspect_ratio"
##  [4] "nb_news_articles"       "actor1_star_meter"      "actor2_star_meter"
##  [7] "actor3_star_meter"      "scifi"                  "thriller"
## [10] "drama"                  "war"                    "crime"
## [13] "movie_meter_IMDBpro"    "biography"              "sci-fi"
## [16] "history"                "family"                 "top_actor1"
## [19] "top_production_company" "maturity_rating.PG"     "maturity_rating.R"
## [22] "country.UK"
## ============================================================
## Heteroskedastic predictors for data frame 2:
##  [1] "movie_budget"           "duration"               "aspect_ratio"
##  [4] "nb_news_articles"       "actor1_star_meter"      "actor2_star_meter"
##  [7] "actor3_star_meter"      "scifi"                  "thriller"
## [10] "drama"                  "war"                    "crime"
## [13] "movie_meter_IMDBpro"    "biography"              "sci-fi"
## [16] "history"                "family"                 "top_actor1"
## [19] "top_production_company" "maturity_rating.PG"     "maturity_rating.R"
## [22] "country.UK"
## ============================================================
## Heteroskedastic predictors for data frame 3:
##  [1] "movie_budget"           "duration"               "aspect_ratio"
```

```
##  [4] "nb_news_articles"       "actor1_star_meter"      "actor2_star_meter"
##  [7] "actor3_star_meter"      "scifi"                  "thriller"
## [10] "drama"                  "war"                    "crime"
## [13] "movie_meter_IMDBpro"    "biography"              "sci-fi"
## [16] "history"                "family"                 "top_actor1"
## [19] "top_production_company" "maturity_rating.PG"     "maturity_rating.R"
## ========================================================
## Heteroskedastic predictors for data frame 4:
##  [1] "movie_budget"           "duration"               "aspect_ratio"
##  [4] "nb_news_articles"       "actor1_star_meter"      "actor2_star_meter"
##  [7] "actor3_star_meter"      "scifi"                  "thriller"
## [10] "drama"                  "war"                    "crime"
## [13] "movie_meter_IMDBpro"    "biography"              "sci-fi"
## [16] "history"                "family"                 "top_actor1"
## [19] "top_production_company" "maturity_rating.PG"     "maturity_rating.R"
## ========================================================
## Heteroskedastic predictors for data frame 5:
##  [1] "movie_budget"           "duration"               "aspect_ratio"
##  [4] "nb_news_articles"       "actor1_star_meter"      "actor2_star_meter"
##  [7] "actor3_star_meter"      "scifi"                  "thriller"
## [10] "drama"                  "war"                    "crime"
## [13] "movie_meter_IMDBpro"    "biography"              "sci-fi"
## [16] "history"                "family"                 "top_actor1"
## [19] "top_production_company" "maturity_rating.PG"     "maturity_rating.R"
## [22] "country.UK"
## ========================================================
## Heteroskedastic predictors for data frame 6:
##  [1] "movie_budget"           "duration"               "aspect_ratio"
##  [4] "nb_news_articles"       "actor1_star_meter"      "actor2_star_meter"
##  [7] "actor3_star_meter"      "scifi"                  "thriller"
## [10] "drama"                  "war"                    "crime"
## [13] "movie_meter_IMDBpro"    "biography"              "sci-fi"
## [16] "history"                "family"                 "top_actor1"
## [19] "top_production_company" "maturity_rating.PG"     "maturity_rating.R"
## [22] "country.UK"
## ========================================================
## Heteroskedastic predictors for data frame 7:
##  [1] "movie_budget"           "duration"               "aspect_ratio"
##  [4] "nb_news_articles"       "actor1_star_meter"      "actor2_star_meter"
##  [7] "actor3_star_meter"      "scifi"                  "thriller"
## [10] "drama"                  "war"                    "crime"
## [13] "movie_meter_IMDBpro"    "biography"              "sci-fi"
## [16] "history"                "family"                 "top_actor1"
## [19] "top_production_company" "maturity_rating.PG"     "maturity_rating.R"
## ========================================================
## Heteroskedastic predictors for data frame 8:
##  [1] "movie_budget"           "duration"               "aspect_ratio"
##  [4] "nb_news_articles"       "actor1_star_meter"      "actor2_star_meter"
##  [7] "actor3_star_meter"      "scifi"                  "thriller"
## [10] "drama"                  "war"                    "crime"
## [13] "movie_meter_IMDBpro"    "biography"              "sci-fi"
## [16] "history"                "family"                 "top_actor1"
## [19] "top_production_company" "maturity_rating.PG"     "maturity_rating.R"
## ========================================================
```

Heteroskedasticity is present in the dataset. The non-constant variance test has identified several predictors that show signs of heteroskedasticity when plotted against the target variable (likely IMDb score). Key heteroskedastic predictors include:

- Continuous variables: movie_budget, duration, aspect_ratio, nb_news_articles, actor star meters, and movie_meter_IMDBpro. Genre variables: scifi, thriller, drama, war, crime, biography, history, and family.
- Categorical variables: top_actor1, top_production_company, maturity ratings (PG and R), and country (UK).

These flagged predictors suggest that the variance of the residuals changes across their range when predicting the target variable. This heteroskedasticity should be considered when building predictive models, as it may affect the efficiency and reliability of standard regression techniques.

simple linear regressions between Y and each predictor xi

```
library(broom)  # For tidy output of regression results

regression_results_list <- list()

for (df_index in seq_along(df_lists)) {
  df <- df_lists[[df_index]]  # get the current data frame
  regression_results <- data.frame(Predictor = character(),
                                   P_Value = numeric(),
                                   R_Squared = numeric(),
                                   stringsAsFactors = FALSE)

  # loop through each predictor in the current data frame
  for (pred in colnames(df)[-1]) {  # xclude imdb_score

    lm_model <- lm(imdb_score ~ get(pred), data = df)

    model_summary <- summary(lm_model)

    p_value <- round(coef(model_summary)[2, 4], 4)
    r_squared <- model_summary$r.squared

    regression_results <- rbind(regression_results,
                                data.frame(Predictor = pred,
                                           P_Value = p_value,
                                           R_Squared = r_squared,
                                           stringsAsFactors = FALSE))
  }

  regression_results_list[[df_index]] <- regression_results
}

for (i in seq_along(regression_results_list)) {
  cat(sprintf("Regression Results for Data Frame %d:\n", i))
  print(regression_results_list[[i]])
  cat("============================================================\n")
}
```

```
## Regression Results for Data Frame 1:
##                       Predictor P_Value     R_Squared
## 1                   movie_budget  0.0000 0.01278147681
```

```
## 2                      duration  0.0000 0.18750263272
## 3                  aspect_ratio  0.6701 0.00012531718
## 4               nb_news_articles  0.0000 0.09526484658
## 5              actor1_star_meter  0.0000 0.01285915988
## 6              actor2_star_meter  0.1715 0.00129004162
## 7              actor3_star_meter  0.8546 0.00002317371
## 8                       nb_faces  0.0022 0.00643724846
## 9                         action  0.0000 0.03023751080
## 10                     adventure  0.0078 0.00487704615
## 11                         scifi  0.0001 0.01081412342
## 12                      thriller  0.0023 0.00637636583
## 13                       western  0.0013 0.00710893487
## 14                         sport  0.1025 0.00183906010
## 15                        horror  0.0000 0.03498237378
## 16                         drama  0.0000 0.14120092159
## 17                           war  0.0000 0.01507691335
## 18                         crime  0.0068 0.00504953143
## 19           movie_meter_IMDBpro  0.0000 0.12273843439
## 20                     biography  0.0000 0.04081366654
## 21                        comedy  0.0000 0.04247499288
## 22                        sci-fi  0.0001 0.01081412342
## 23                       fantasy  0.0058 0.00523581258
## 24                       history  0.0000 0.02251692792
## 25                        family  0.0000 0.01380464056
## 26                  top_director  0.0000 0.02349407756
## 27                    top_actor1  0.0000 0.01993337720
## 28                    top_actor2  0.0097 0.00460316160
## 29                    top_actor3  0.1696 0.00130163023
## 30           top_cinematographer  0.0897 0.00198562334
## 31        top_production_company  0.8465 0.00002586695
## 32                top_distributor  0.4484 0.00039674594
## 33            maturity_rating.PG  0.0000 0.03447285276
## 34             maturity_rating.R  0.0000 0.02822135362
## 35                country.Canada  0.0423 0.00284142364
## 36                country.France  0.7290 0.00008289316
## 37               country.Germany  0.8834 0.00001484515
## 38                     country.UK  0.0000 0.01454887621
## 39                    country.USA  0.0049 0.00544197078
## 40 release_day_group.Early Month  0.7520 0.00006891511
## 41  release_day_group.Late Month  0.5061 0.00030514697
## 42    weekday_or_weekend.Weekend  0.1056 0.00180619468
## ========================================================
## Regression Results for Data Frame 2:
##                        Predictor P_Value     R_Squared
## 1                    movie_budget  0.0000 0.01278147681
## 2                      duration  0.0000 0.18750263272
## 3                  aspect_ratio  0.6701 0.00012531718
## 4               nb_news_articles  0.0000 0.09526484658
## 5              actor1_star_meter  0.0000 0.01285915988
## 6              actor2_star_meter  0.1715 0.00129004162
## 7              actor3_star_meter  0.8546 0.00002317371
## 8                       nb_faces  0.0022 0.00643724846
## 9                         action  0.0000 0.03023751080
## 10                     adventure  0.0078 0.00487704615
```

```
## 11                         scifi  0.0001 0.01081412342
## 12                      thriller  0.0023 0.00637636583
## 13                       western  0.0013 0.00710893487
## 14                         sport  0.1025 0.00183906010
## 15                        horror  0.0000 0.03498237378
## 16                         drama  0.0000 0.14120092159
## 17                           war  0.0000 0.01507691335
## 18                         crime  0.0068 0.00504953143
## 19           movie_meter_IMDBpro  0.0000 0.12273843439
## 20                     biography  0.0000 0.04081366654
## 21                        comedy  0.0000 0.04247499288
## 22                        sci-fi  0.0001 0.01081412342
## 23                       fantasy  0.0058 0.00523581258
## 24                       history  0.0000 0.02251692792
## 25                        family  0.0000 0.01380464056
## 26                  top_director  0.0000 0.02349407756
## 27                    top_actor1  0.0000 0.01993337720
## 28                    top_actor2  0.0097 0.00460316160
## 29                    top_actor3  0.1696 0.00130163023
## 30           top_cinematographer  0.0897 0.00198562334
## 31        top_production_company  0.8465 0.00002586695
## 32                top_distributor  0.4484 0.00039674594
## 33             maturity_rating.PG  0.0000 0.03447285276
## 34              maturity_rating.R  0.0000 0.02822135362
## 35                 country.Canada  0.0423 0.00284142364
## 36                 country.France  0.7290 0.00008289316
## 37                country.Germany  0.8834 0.00001484515
## 38                     country.UK  0.0000 0.01454887621
## 39                    country.USA  0.0049 0.00544197078
## 40 release_day_group.Early Month  0.7520 0.00006891511
## 41  release_day_group.Late Month  0.5061 0.00030514697
## 42    weekday_or_weekend.Weekend  0.1056 0.00180619468
## ======================================================
## Regression Results for Data Frame 3:
##                         Predictor P_Value     R_Squared
## 1                    movie_budget  0.0000 0.01278147681
## 2                        duration  0.0000 0.18750263272
## 3                    aspect_ratio  0.6701 0.00012531718
## 4                nb_news_articles  0.0000 0.09526484658
## 5              actor1_star_meter  0.0000 0.01285915988
## 6              actor2_star_meter  0.1715 0.00129004162
## 7              actor3_star_meter  0.8546 0.00002317371
## 8                        nb_faces  0.0022 0.00643724846
## 9                          action  0.0000 0.03023751080
## 10                      adventure  0.0078 0.00487704615
## 11                         scifi  0.0001 0.01081412342
## 12                      thriller  0.0023 0.00637636583
## 13                       western  0.0013 0.00710893487
## 14                         sport  0.1025 0.00183906010
## 15                        horror  0.0000 0.03498237378
## 16                         drama  0.0000 0.14120092159
## 17                           war  0.0000 0.01507691335
## 18                         crime  0.0068 0.00504953143
## 19           movie_meter_IMDBpro  0.0000 0.12273843439
```

```
## 20                     biography  0.0000 0.04081366654
## 21                        comedy  0.0000 0.04247499288
## 22                        sci-fi  0.0001 0.01081412342
## 23                       fantasy  0.0058 0.00523581258
## 24                       history  0.0000 0.02251692792
## 25                        family  0.0000 0.01380464056
## 26                  top_director  0.0000 0.02349407756
## 27                    top_actor1  0.0000 0.01993337720
## 28                    top_actor2  0.0097 0.00460316160
## 29                    top_actor3  0.1696 0.00130163023
## 30           top_cinematographer  0.0897 0.00198562334
## 31        top_production_company  0.8465 0.00002586695
## 32               top_distributor  0.4484 0.00039674594
## 33            maturity_rating.PG  0.0000 0.03447285276
## 34             maturity_rating.R  0.0000 0.02822135362
## 35              continent.Europe  0.0001 0.01069542716
## 36       continent.North America  0.0003 0.00899986744
## 37 release_day_group.Early Month  0.7520 0.00006891511
## 38  release_day_group.Late Month  0.5061 0.00030514697
## 39               day_of_week.Sun  0.2389 0.00095733797
## 40               day_of_week.Tue  0.3472 0.00060998625
## 41               day_of_week.Wed  0.0055 0.00531473065
## 42               day_of_week.Thu  0.0651 0.00234687880
## 43               day_of_week.Fri  0.0001 0.01021867077
## 44               day_of_week.Sat  0.2724 0.00083120452
## 45    weekday_or_weekend.Weekend  0.1056 0.00180619468
## ========================================================
## Regression Results for Data Frame 4:
##                         Predictor P_Value     R_Squared
## 1                    movie_budget  0.0000 0.01278147681
## 2                        duration  0.0000 0.18750263272
## 3                    aspect_ratio  0.6701 0.00012531718
## 4                nb_news_articles  0.0000 0.09526484658
## 5              actor1_star_meter  0.0000 0.01285915988
## 6              actor2_star_meter  0.1715 0.00129004162
## 7              actor3_star_meter  0.8546 0.00002317371
## 8                        nb_faces  0.0022 0.00643724846
## 9                          action  0.0000 0.03023751080
## 10                    adventure  0.0078 0.00487704615
## 11                        scifi  0.0001 0.01081412342
## 12                     thriller  0.0023 0.00637636583
## 13                      western  0.0013 0.00710893487
## 14                        sport  0.1025 0.00183906010
## 15                       horror  0.0000 0.03498237378
## 16                        drama  0.0000 0.14120092159
## 17                          war  0.0000 0.01507691335
## 18                        crime  0.0068 0.00504953143
## 19           movie_meter_IMDBpro  0.0000 0.12273843439
## 20                     biography  0.0000 0.04081366654
## 21                        comedy  0.0000 0.04247499288
## 22                        sci-fi  0.0001 0.01081412342
## 23                       fantasy  0.0058 0.00523581258
## 24                       history  0.0000 0.02251692792
## 25                        family  0.0000 0.01380464056
```

```
## 26                  top_director  0.0000 0.02349407756
## 27                    top_actor1  0.0000 0.01993337720
## 28                    top_actor2  0.0097 0.00460316160
## 29                    top_actor3  0.1696 0.00130163023
## 30           top_cinematographer  0.0897 0.00198562334
## 31         top_production_company  0.8465 0.00002586695
## 32                top_distributor  0.4484 0.00039674594
## 33             maturity_rating.PG  0.0000 0.03447285276
## 34              maturity_rating.R  0.0000 0.02822135362
## 35               continent.Europe  0.0001 0.01069542716
## 36        continent.North America  0.0003 0.00899986744
## 37 release_day_group.Early Month  0.7520 0.00006891511
## 38  release_day_group.Late Month  0.5061 0.00030514697
## 39    weekday_or_weekend.Weekend  0.1056 0.00180619468
## ======================================================
## Regression Results for Data Frame 5:
##                         Predictor P_Value     R_Squared
## 1                    movie_budget  0.0000 0.01278147681
## 2                        duration  0.0000 0.18750263272
## 3                    aspect_ratio  0.6701 0.00012531718
## 4                nb_news_articles  0.0000 0.09526484658
## 5               actor1_star_meter  0.0000 0.01285915988
## 6               actor2_star_meter  0.1715 0.00129004162
## 7               actor3_star_meter  0.8546 0.00002317371
## 8                        nb_faces  0.0022 0.00643724846
## 9                          action  0.0000 0.03023751080
## 10                      adventure  0.0078 0.00487704615
## 11                          scifi  0.0001 0.01081412342
## 12                       thriller  0.0023 0.00637636583
## 13                        western  0.0013 0.00710893487
## 14                          sport  0.1025 0.00183906010
## 15                         horror  0.0000 0.03498237378
## 16                          drama  0.0000 0.14120092159
## 17                            war  0.0000 0.01507691335
## 18                          crime  0.0068 0.00504953143
## 19           movie_meter_IMDBpro  0.0000 0.12273843439
## 20                      biography  0.0000 0.04081366654
## 21                         comedy  0.0000 0.04247499288
## 22                         sci-fi  0.0001 0.01081412342
## 23                        fantasy  0.0058 0.00523581258
## 24                        history  0.0000 0.02251692792
## 25                         family  0.0000 0.01380464056
## 26                  top_director  0.0000 0.02349407756
## 27                    top_actor1  0.0000 0.01993337720
## 28                    top_actor2  0.0097 0.00460316160
## 29                    top_actor3  0.1696 0.00130163023
## 30           top_cinematographer  0.0897 0.00198562334
## 31         top_production_company  0.8465 0.00002586695
## 32                top_distributor  0.4484 0.00039674594
## 33             maturity_rating.PG  0.0000 0.03447285276
## 34              maturity_rating.R  0.0000 0.02822135362
## 35                 country.Canada  0.0423 0.00284142364
## 36                 country.France  0.7290 0.00008289316
## 37                country.Germany  0.8834 0.00001484515
```

```
## 38                    country.UK  0.0000 0.01454887621
## 39                   country.USA  0.0049 0.00544197078
## 40 release_day_group.Early Month  0.7520 0.00006891511
## 41  release_day_group.Late Month  0.5061 0.00030514697
## 42    weekday_or_weekend.Weekend  0.1056 0.00180619468
## ========================================================
## Regression Results for Data Frame 6:
##                        Predictor P_Value     R_Squared
## 1                    movie_budget  0.0000 0.01278147681
## 2                        duration  0.0000 0.18750263272
## 3                    aspect_ratio  0.6701 0.00012531718
## 4                nb_news_articles  0.0000 0.09526484658
## 5               actor1_star_meter  0.0000 0.01285915988
## 6               actor2_star_meter  0.1715 0.00129004162
## 7               actor3_star_meter  0.8546 0.00002317371
## 8                        nb_faces  0.0022 0.00643724846
## 9                          action  0.0000 0.03023751080
## 10                      adventure  0.0078 0.00487704615
## 11                          scifi  0.0001 0.01081412342
## 12                       thriller  0.0023 0.00637636583
## 13                        western  0.0013 0.00710893487
## 14                          sport  0.1025 0.00183906010
## 15                         horror  0.0000 0.03498237378
## 16                          drama  0.0000 0.14120092159
## 17                            war  0.0000 0.01507691335
## 18                          crime  0.0068 0.00504953143
## 19           movie_meter_IMDBpro  0.0000 0.12273843439
## 20                      biography  0.0000 0.04081366654
## 21                         comedy  0.0000 0.04247499288
## 22                         sci-fi  0.0001 0.01081412342
## 23                        fantasy  0.0058 0.00523581258
## 24                        history  0.0000 0.02251692792
## 25                         family  0.0000 0.01380464056
## 26                   top_director  0.0000 0.02349407756
## 27                     top_actor1  0.0000 0.01993337720
## 28                     top_actor2  0.0097 0.00460316160
## 29                     top_actor3  0.1696 0.00130163023
## 30            top_cinematographer  0.0897 0.00198562334
## 31         top_production_company  0.8465 0.00002586695
## 32                 top_distributor  0.4484 0.00039674594
## 33              maturity_rating.PG  0.0000 0.03447285276
## 34               maturity_rating.R  0.0000 0.02822135362
## 35                  country.Canada  0.0423 0.00284142364
## 36                  country.France  0.7290 0.00008289316
## 37                 country.Germany  0.8834 0.00001484515
## 38                    country.UK  0.0000 0.01454887621
## 39                   country.USA  0.0049 0.00544197078
## 40 release_day_group.Early Month  0.7520 0.00006891511
## 41  release_day_group.Late Month  0.5061 0.00030514697
## 42    weekday_or_weekend.Weekend  0.1056 0.00180619468
## ========================================================
## Regression Results for Data Frame 7:
##                        Predictor P_Value     R_Squared
## 1                    movie_budget  0.0000 0.01278147681
```

```
## 2                       duration  0.0000 0.18750263272
## 3                   aspect_ratio  0.6701 0.00012531718
## 4                nb_news_articles  0.0000 0.09526484658
## 5               actor1_star_meter  0.0000 0.01285915988
## 6               actor2_star_meter  0.1715 0.00129004162
## 7               actor3_star_meter  0.8546 0.00002317371
## 8                        nb_faces  0.0022 0.00643724846
## 9                          action  0.0000 0.03023751080
## 10                      adventure  0.0078 0.00487704615
## 11                          scifi  0.0001 0.01081412342
## 12                       thriller  0.0023 0.00637636583
## 13                        western  0.0013 0.00710893487
## 14                          sport  0.1025 0.00183906010
## 15                         horror  0.0000 0.03498237378
## 16                          drama  0.0000 0.14120092159
## 17                            war  0.0000 0.01507691335
## 18                          crime  0.0068 0.00504953143
## 19            movie_meter_IMDBpro  0.0000 0.12273843439
## 20                      biography  0.0000 0.04081366654
## 21                         comedy  0.0000 0.04247499288
## 22                         sci-fi  0.0001 0.01081412342
## 23                        fantasy  0.0058 0.00523581258
## 24                        history  0.0000 0.02251692792
## 25                         family  0.0000 0.01380464056
## 26                   top_director  0.0000 0.02349407756
## 27                     top_actor1  0.0000 0.01993337720
## 28                     top_actor2  0.0097 0.00460316160
## 29                     top_actor3  0.1696 0.00130163023
## 30            top_cinematographer  0.0897 0.00198562334
## 31          top_production_company  0.8465 0.00002586695
## 32                 top_distributor  0.4484 0.00039674594
## 33              maturity_rating.PG  0.0000 0.03447285276
## 34               maturity_rating.R  0.0000 0.02822135362
## 35                continent.Europe  0.0001 0.01069542716
## 36         continent.North America  0.0003 0.00899986744
## 37 release_day_group.Early Month  0.7520 0.00006891511
## 38  release_day_group.Late Month  0.5061 0.00030514697
## 39                 day_of_week.Sun  0.2389 0.00095733797
## 40                 day_of_week.Tue  0.3472 0.00060998625
## 41                 day_of_week.Wed  0.0055 0.00531473065
## 42                 day_of_week.Thu  0.0651 0.00234687880
## 43                 day_of_week.Fri  0.0001 0.01021867077
## 44                 day_of_week.Sat  0.2724 0.00083120452
## 45    weekday_or_weekend.Weekend  0.1056 0.00180619468
## ======================================================
## Regression Results for Data Frame 8:
##                       Predictor P_Value     R_Squared
## 1                   movie_budget  0.0000 0.01278147681
## 2                       duration  0.0000 0.18750263272
## 3                   aspect_ratio  0.6701 0.00012531718
## 4                nb_news_articles  0.0000 0.09526484658
## 5               actor1_star_meter  0.0000 0.01285915988
## 6               actor2_star_meter  0.1715 0.00129004162
## 7               actor3_star_meter  0.8546 0.00002317371
```

```
## 8                       nb_faces  0.0022 0.00643724846
## 9                         action  0.0000 0.03023751080
## 10                     adventure  0.0078 0.00487704615
## 11                         scifi  0.0001 0.01081412342
## 12                      thriller  0.0023 0.00637636583
## 13                       western  0.0013 0.00710893487
## 14                         sport  0.1025 0.00183906010
## 15                        horror  0.0000 0.03498237378
## 16                         drama  0.0000 0.14120092159
## 17                           war  0.0000 0.01507691335
## 18                         crime  0.0068 0.00504953143
## 19           movie_meter_IMDBpro  0.0000 0.12273843439
## 20                     biography  0.0000 0.04081366654
## 21                        comedy  0.0000 0.04247499288
## 22                        sci-fi  0.0001 0.01081412342
## 23                       fantasy  0.0058 0.00523581258
## 24                       history  0.0000 0.02251692792
## 25                        family  0.0000 0.01380464056
## 26                  top_director  0.0000 0.02349407756
## 27                    top_actor1  0.0000 0.01993337720
## 28                    top_actor2  0.0097 0.00460316160
## 29                    top_actor3  0.1696 0.00130163023
## 30           top_cinematographer  0.0897 0.00198562334
## 31        top_production_company  0.8465 0.00002586695
## 32                top_distributor  0.4484 0.00039674594
## 33             maturity_rating.PG  0.0000 0.03447285276
## 34              maturity_rating.R  0.0000 0.02822135362
## 35               continent.Europe  0.0001 0.01069542716
## 36       continent.North America  0.0003 0.00899986744
## 37 release_day_group.Early Month  0.7520 0.00006891511
## 38  release_day_group.Late Month  0.5061 0.00030514697
## 39    weekday_or_weekend.Weekend  0.1056 0.00180619468
## ========================================================
```

Based on the p-values you provided, the variables with the most linear predictive power for predicting imdb_score (p-value < 0.01) are: movie_budget, duration, nb_news_articles, actor1_star_meter, action, horror, biography, comedy, maturity_rating.PG, maturity_rating.R, country.UK, country.USA, scifi.

These variables are statistically significant predictors, with duration showing the highest R-squared value, indicating it has the strongest linear predictive power among the listed variables.

Exporting data

exporting to xlsx

```
library(writexl)
write_xlsx(df_dummies, "IMDb_processed.xlsx")
```