An Analysis of Factors Influencing High IMDB Ratings

Group 8

1 Data Description

Source: IMDB film database

Description of variables:

- `film_id`: Unique identifier
- `year`: Year of release
- `length`: Duration (minutes)
- `budget`: Production budget (in $10 million)
- `votes`: Number of viewer votes
- `genre`: Genre of the film
- `rating`: IMDB score from 0–10

Total observations: 2,847 films

Objective of the analysis: To determine which factors of films are associated with an IMDB rating above 7 by using a Generalised Linear Model (GLM).

2 Data Preparing & Cleaning

```r
# Load dataset
raw_data <- read.csv("dataset08.csv")

# Preview the structure of the dataset
glimpse(raw_data)
```

```
Rows: 2,847
Columns: 7
$ film_id <int> 5993, 37190, 43646, 28476, 23975, 50170, 56142, 2287, 17822, 5~
$ year    <int> 1943, 1961, 1987, 1976, 1982, 1936, 1932, 1967, 1983, 2003, 19~
$ length  <int> 65, 87, 79, NA, 88, NA, 75, 100, 82, 15, 86, 96, 150, 86, 102,~
$ budget  <dbl> 15.5, 12.3, 16.4, 12.2, 12.5, 7.0, 12.0, 12.2, 13.4, 13.9, 11.~
$ votes   <int> 42, 6, 161, 5, 97, 146, 14, 8, 141, 20, 121, 119, 5, 14, 48, 1~
$ genre   <chr> "Action", "Drama", "Action", "Documentary", "Action", "Drama",~
$ rating  <dbl> 7.6, 6.0, 7.5, 8.0, 3.5, 4.4, 4.5, 8.4, 3.5, 7.8, 8.2, 2.9, 4.~
```

```
# Remove rows that have missing values in 'length'variable
clean_data <- raw_data %>%
  filter(!is.na(length))

# Convert 'genre' to factor for categorical analysis
clean_data$genre <- as.factor(clean_data$genre)

# Define a function to create new binary response variable 'rating_above7'
rating_rank <- function(rating_column, threshold = 7){
  ifelse(rating_column > threshold, 1, 0)
}
#check the range of 'year' variable
range(clean_data$year) #we can see that range is between 1898 and 2005
```

```
[1] 1898 2005
```

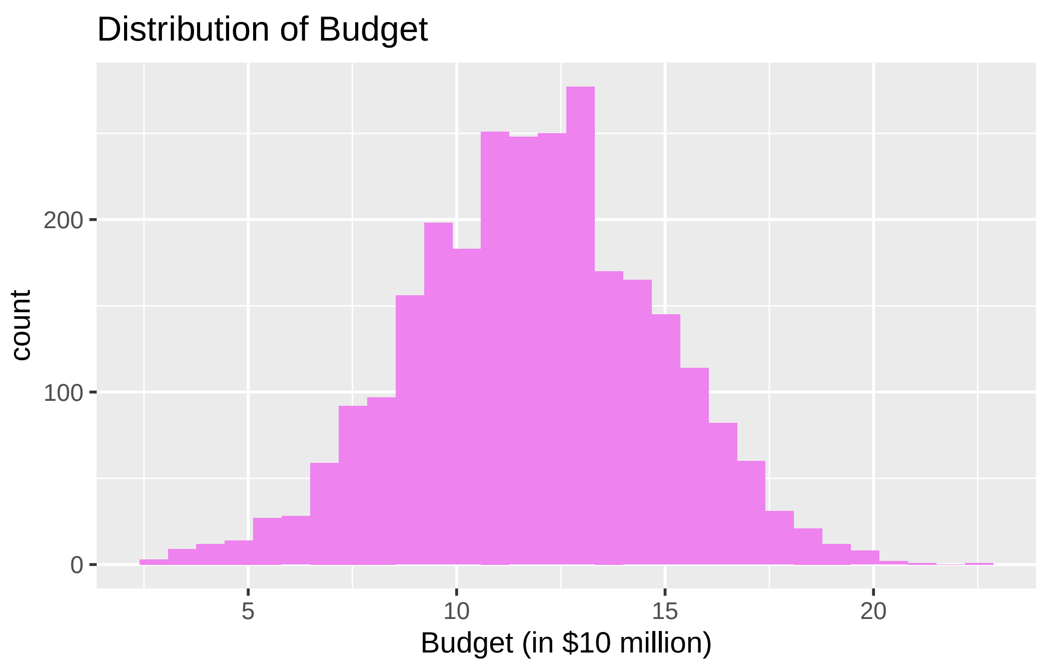
```
# Mutate new variables : binary outcome 'rating_above_7' & 'decade_group'
clean_data <- clean_data %>%
  mutate(
    rating_above_7 = rating_rank(rating),
    decade_group = cut(year,
                       breaks = c(1890, 1930, 1940, 1950, 1960, 1970, 1980, 1990, 2000, 2010),
                       labels = c("1890s-1920s", "1930s", "1940s", "1950s", "1960s", "1970s", "
                       right=FALSE)
  )
#check the missing values in 'budget' & 'votes' variables
sum(is.na(clean_data$budget)) # 0 missing values
```

```
[1] 0
```

```
sum(is.na(clean_data$votes)) # 0 missing values
```

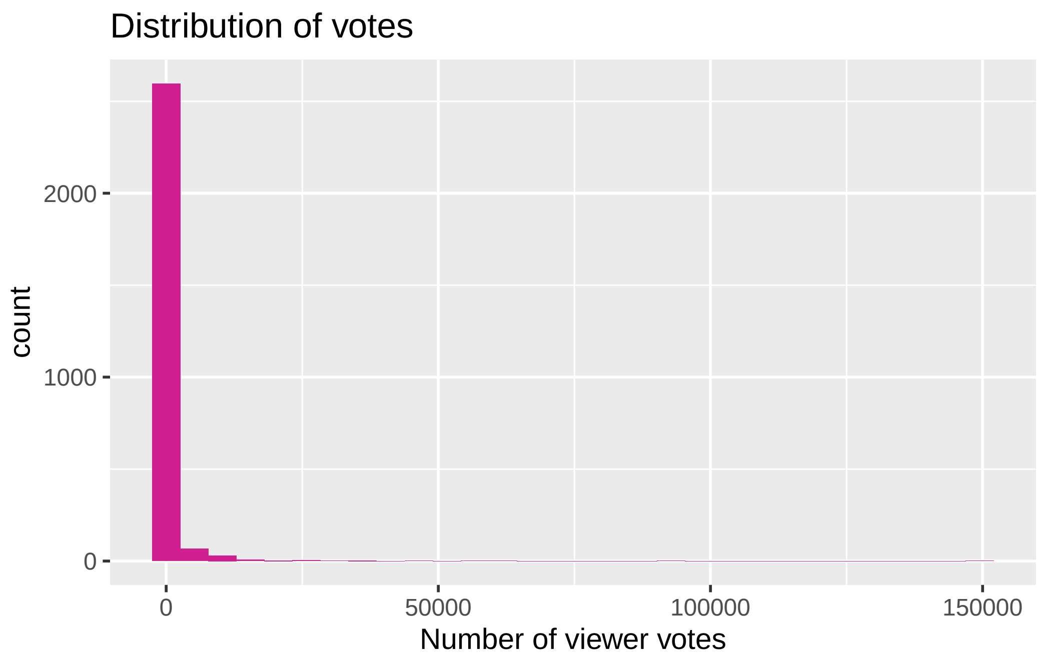
```
[1] 0
```

```
#Visualize the distribution of 'budget'
#If distribution is heavily skewed, log-transformation might be needed
ggplot(clean_data, aes(x = budget)) +
  geom_histogram(bins = 30, fill = "violet") +
  labs(title = "Distribution of Budget", x = "Budget (in $10 million)")
```



```
#Interpretation:
#The 'budget' variable appears approximately normally distributed.

#Visualize the distribution of 'votes'
ggplot(clean_data, aes(x = votes)) +
  geom_histogram(bins = 30, fill = "violetred") +
  labs(title = "Distribution of votes", x = "Number of viewer votes")
```



```
#Interpretation:
#The 'votes' variable is highly right-skewed.
#A log-transformation should be applied before using this variable in modelling.
```

3 Exploratory Data Analysis (EDA)

3.1 Data Overview

From the data overview, we will build summary statistics for both numeric and categorical variables to explore the general characteristics and ranges within the dataset. We highlight the key variables that are particularly interesting to explore for our objective

```
#Built the summary statistics table for Numeric Variables by create function
make_table <- function(data, name, label = NULL) {
  if (is.null(label)) label <- name
  summary_table <- data %>%
    group_by(rating_above_7) %>%
    summarize(
      Mean = mean(.data[[name]], na.rm = TRUE),
      Median = median(.data[[name]], na.rm = TRUE),
      `Std. Dev` = sd(.data[[name]], na.rm = TRUE),
      Minimum = min(.data[[name]], na.rm = TRUE),
      Maximum = max(.data[[name]], na.rm = TRUE),
      IQR = IQR(.data[[name]], na.rm = TRUE),
      `Sample Size` = n(),
      .groups = "drop") %>%
    pivot_longer(-rating_above_7, names_to = "Statistic", values_to = "Value") %>%
    pivot_wider(names_from = rating_above_7, values_from = "Value",
                names_prefix = "Rating > 7 = ") %>% gt() %>%
    fmt_number(columns = starts_with("Rating"), decimals = 2) %>%
    cols_label(
      Statistic = "Statistic",
      `Rating > 7 = 0` = "Rating <= 7",
      `Rating > 7 = 1` = "Rating > 7")
    summary_table %>% as_latex() %>% as.character() %>% cat()}
#Built the summary statistics table for Categorical Variables by create function
make_cat_table <- function(data, cat, group_var = "rating_above_7", var_label = NULL) {
  if (is.null(var_label)) var_label <- cat_var
  group_sym <- sym(group_var)
  cat_sym <- sym(cat)
  tab <- data %>%
    group_by(!!group_sym, !!cat_sym) %>%
    summarize(Count = n(), .groups = "drop") %>%
    group_by(!!group_sym) %>%
    mutate(Percentage = Count / sum(Count) * 100) %>%
    pivot_wider(
      names_from = !!group_sym,
      values_from = c(Count, Percentage),
      names_sep = "_") %>%
    rename(Category = !!cat_sym)
  tab %>% gt() %>% fmt_number(columns = where(is.numeric), decimals = 2) %>%
    cols_label(
      Category = "Category",
      Count_0 = "Count (<= 7)",
      Count_1 = "Count (> 7)",
      Percentage_0 = "% (<= 7)",
      Percentage_1 = "% (> 7)"
    ) %>% as_latex() %>% as.character() %>% cat()}
```

Table 1: Summary Statistics table of Length (minutes)

```r
# Length Table
make_table(clean_data, "length", "Length (minutes)")
```

Statistic	Rating <= 7	Rating > 7
Mean	95.14	56.80
Median	93.00	68.00
Std. Dev	28.28	40.24
Minimum	3.00	1.00
Maximum	480.00	174.00
IQR	20.00	78.00
Sample Size	1,801.00	915.00

Table 2: Summary Statistics table of Budget(in $10 million)

```r
# Budget Table
make_table(clean_data, "budget", "Budget (in $10 million)")
```

Statistic	Rating <= 7	Rating > 7
Mean	11.35	12.90
Median	11.40	12.80
Std. Dev	2.85	2.89
Minimum	2.50	4.00
Maximum	19.60	22.30
IQR	3.90	4.10
Sample Size	1,801.00	915.00

Table 3: Summary Statistics table of Votes

```r
# Votes Table
make_table(clean_data, "votes", "Number of Votes")
```

Statistic	Rating <= 7	Rating > 7
Mean	824.81	387.46
Median	36.00	21.00
Std. Dev	5,485.31	2,093.44
Minimum	5.00	5.00
Maximum	149,494.00	34,666.00
IQR	132.00	59.00
Sample Size	1,801.00	915.00

Table 4: Summary Statistics table of Rating (IMDB score)

```r
# Rating Table
make_table(clean_data, "rating", "IMDB score")
```

Statistic	Rating <= 7	Rating > 7
Mean	4.02	7.99
Median	4.00	8.00
Std. Dev	0.91	0.39
Minimum	0.80	7.10
Maximum	7.00	9.20
IQR	1.10	0.60
Sample Size	1,801.00	915.00

Table 5: Summary Statistics table of Genre

```r
# Film Genre Table
make_cat_table(clean_data, "genre", var_label = "Film Genre")
```

Category	Count (<= 7)	Count (> 7)	% (<= 7)	% (> 7)
Action	709.00	117.00	39.37	12.79
Animation	56.00	118.00	3.11	12.90
Comedy	276.00	371.00	15.32	40.55
Documentary	11.00	138.00	0.61	15.08
Drama	719.00	42.00	39.92	4.59
Romance	28.00	NA	1.55	NA
Short	2.00	129.00	0.11	14.10

Table 6: Summary Statistics table of Decade Group

```r
# Decade Group Table
make_cat_table(clean_data, "decade_group", var_label = "Decade Group")
```

Category	Count (<= 7)	Count (> 7)	% (<= 7)	% (> 7)
1890s-1920s	29.00	37.00	1.61	4.04
1930s	137.00	99.00	7.61	10.82
1940s	132.00	73.00	7.33	7.98
1950s	166.00	71.00	9.22	7.76
1960s	172.00	72.00	9.55	7.87
1970s	197.00	81.00	10.94	8.85
1980s	275.00	88.00	15.27	9.62
1990s	413.00	182.00	22.93	19.89
2000s	280.00	212.00	15.55	23.17

The summary statistics provide a general overview of the dataset and the key characteristic's variables.

- **Length**: As shown in Table 1, Films with high IMDB scores (rating > 7) have a lower average length (mean = 56.8 mins) compared to the films with low IMDB scores (rating $<= 7$)(mean = 95.1 mins).
- **Budget**: As shown in Table 2, There is only a small difference in average production budgets between films with high IMDB scores (rating > 7)(mean = 12.9) and films with

low IMDB scores (rating <= 7)(mean = 11.35).

- **Votes**: As shown in Table 3, Films with high IMDB scores (rating > 7) have a lower average votes (mean = 387.46) compared to the films with low IMDB scores (rating <= 7)(mean = 824.81).
- **Rating**: As shown in Table 4, The sample size of the films with low IMDB scores (rating <= 7)(Sample Size = 1801) is higher than films with high IMDB scores (rating > 7)(Sample Size = 915).
- **Genre**: As shown in Table 5, Comedy appears more frequently in films with high IMDB scores (rating > 7), while Romance is found only among films with lower scores (rating 7).
- **Decade Group**: As shown in Table 6, A significant proportion of films with high IMDB scores (rating > 7) are from the 1990s and 2000s.

3.2 Target Variable Exploration

In this section, we explore the distribution of our target variable (rating_above_7) using a simple bar plot to observe its overall balance.

```r
ggplot(clean_data, aes(x = as.factor(rating_above_7)))+
  geom_bar(fill = "aquamarine4")+
  labs(x = "Rating > 7",
       y = "Count",
       title = "Distribution of Target Variable")+
  theme_bw()
```

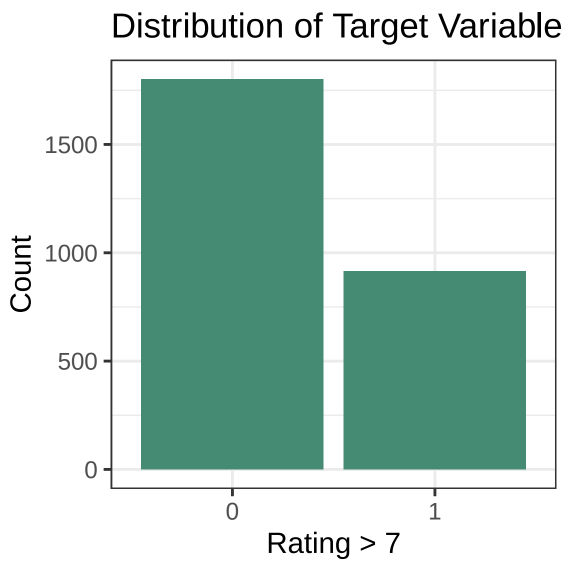


Figure 1: Bar plot of our Target Variable

As shown in Figure 1, there is a clear imbalance in the distribution of our target variable(rating_above). This suggests that model evaluation's part may take this imbalance into the account. Therefore this plot highlights the importance of being aware of this concern when building models.

3.3 Bivariate Analysis with target

In this section, we focus on the relationships among explanatory variables, as well as their associations with the target variable.

3.3.1 Relationships Between Numerical Predictors and Target Variable

```
clean_data %>%
  mutate(votes_bin = ntile(votes, 8)) %>%
  group_by(rating_above_7, votes_bin) %>%
  summarise(n = n(), .groups = "drop") %>%
  group_by(rating_above_7) %>%
  mutate(perc = n / sum(n) * 100) %>%
  ggplot(aes(x = as.factor(rating_above_7), y = perc, fill = as.factor(votes_bin))) +
  geom_col() +
  geom_text(aes(label = round(perc, 1)), position = position_stack(vjust = 0.5), size = 3) +
  labs(
    x = "Rating > 7",
    y = "Percentage",
    fill = "Votes Bin") +
  theme_bw()
```

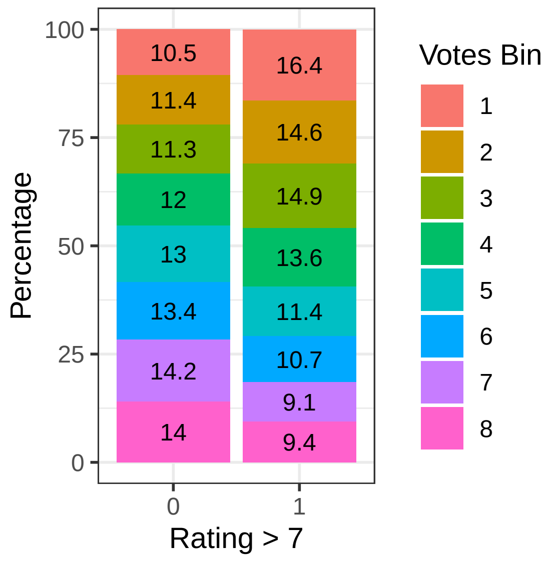


Figure 2: Distribution of Votes Binned into Quantiles by IMDB Rating Group

As shown in Figure 2, the chart shows the distribution of vote counts, divided into 8 quantile-based bins by rating group(Rating <= 7 and Rating >7).

The bin number reflects the relative number of votes, with Bin 1 representing the lowest and Bin 8 the highest.

We can observe that the lower vote bins(Bin1-3) are common among films with high IMDB scores (rating > 7). In contrast, films with low IMDB scores (rating <= 7) tend to appear more in high votes, as shown in bin 7 and 8.

However, we found that votes variable was found to be highly skewed, suggesting the plot might still not fully reflect the actual imbalance of vote in raw scale.

```
ggplot(clean_data, aes(x = budget, y = length, color = as.factor(rating_above_7))) +
  geom_point(alpha = 0.6) +
  scale_color_manual(
    values = c("0" = "darkorange",
               "1" = "darkorchid"),
    labels = c("<= 7", "> 7"))+
  labs(
    x = "Budget (in $10 million)",
    y = "Film Length (minutes)",
    color = "IMDB Scores") +
  theme_bw()
```

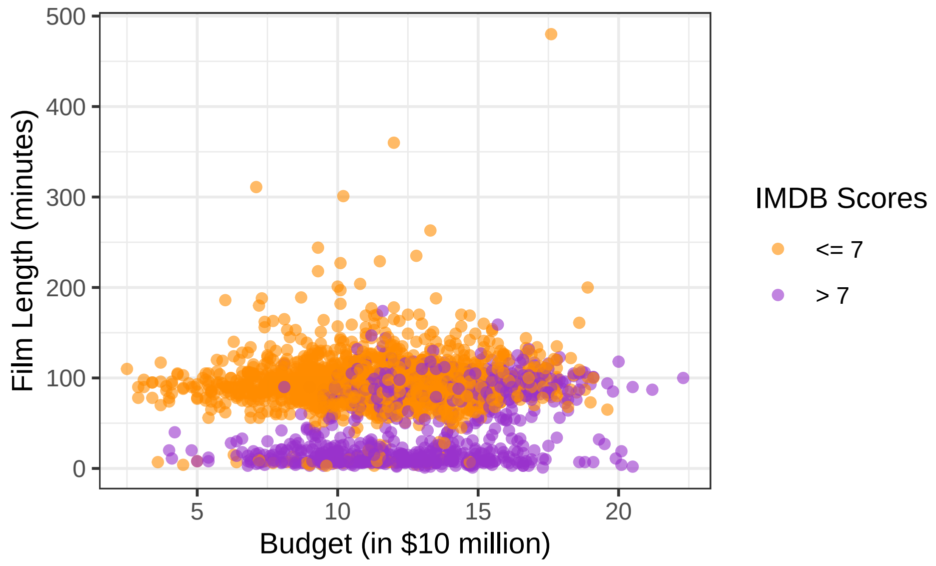


Figure 3: Relationship between Film Length and Budget by IMDB Rating Group

As shown in Figure 3,there is no strong linear relationship between the film length and budget across rating group. However, we can observe a clear separation between the two groups. Films with high IMDB scores (rating > 7) tend to cluster within 50 - 150 minutes in length and have budgets around $100 -150 million. In contrast, films with low IMDB scores (rating <= 7) tend to cluster under 100 minutes, while their budgets remain in a similar range to films with high IMDB scores (rating > 7).

```
ggplot(clean_data, aes(x = rating, y = length, color = as.factor(rating_above_7))) +
  geom_point(alpha = 0.6) +
  scale_color_manual(
    values = c("0" = "darkorange",
               "1" = "darkorchid"),
    labels = c("<= 7", "> 7"))+
  labs(
    x = "IMDB score from 0-10",
    y = "Film Length (minutes)",
    color = "IMDB Scores") +
  theme_bw()
```

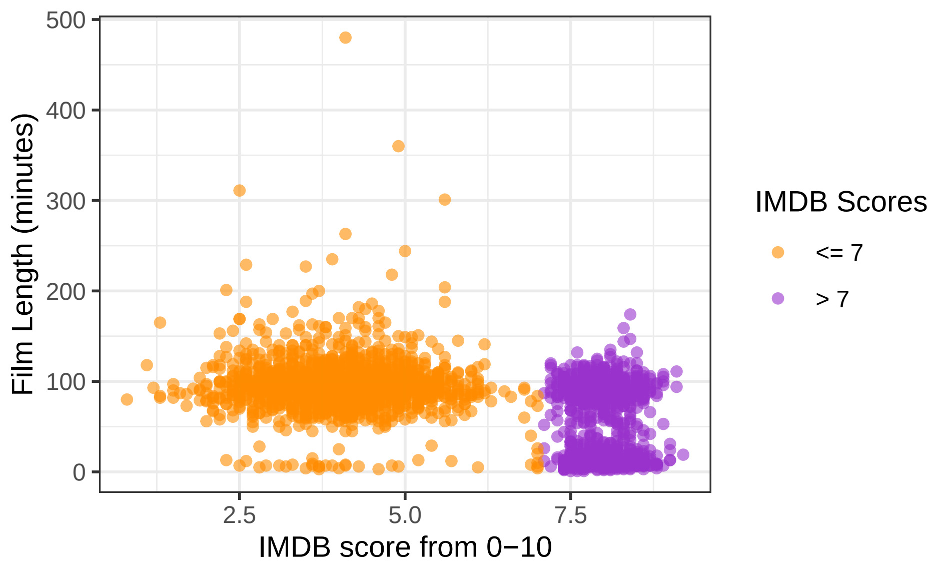


Figure 4: Relationship between Film Length and IMDB score by IMDB Rating Group

As shown in Figure 4, there are no clear relationship between the film length and IMDB score across rating group. However, the plot clearly reflects the separation at a rating of 7. Notably, films with low IMDB scores (rating <= 7) tend to have more variation in film length, including several outliers with unusually long durations. In contrast,the films with high IMDB scores(rating > 7) are more tightly clustered.

3.3.2 Categorical Predictors vs Target Variable

```
ggplot(clean_data, aes(x= genre, fill = as.factor(rating_above_7))) +
  geom_bar(position = "fill", alpha = 0.6) +
  scale_y_continuous(labels = scales::percent_format()) +
  scale_fill_manual(
    values = c("0" = "darkorange",
               "1" = "darkorchid"),
    labels = c("<= 7", "> 7"))+
  labs(x = "Genre",
       y = "Proportion",
      fill = "IMDB Scores")+
  theme_bw()
```

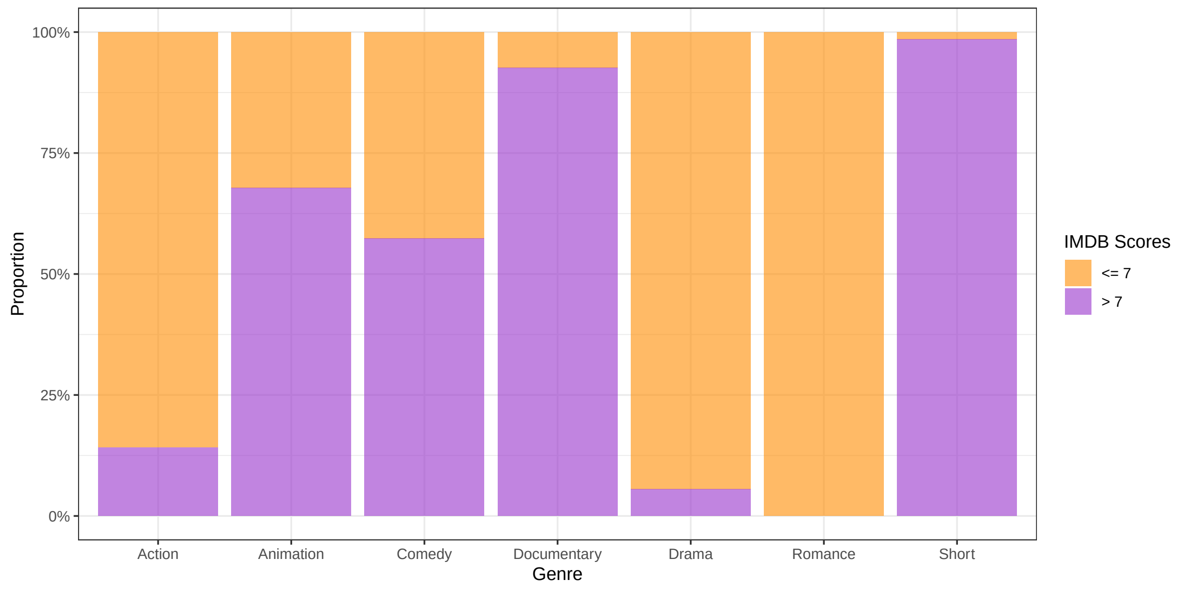


Figure 5: Stacked Bar plot of between Genre of the film and IMDB score

As shown in Figure 5,the plot illustrates the proportion of films with high IMDB scores(rating > 7) and films with low IMDB scores (rating $<= 7$) across different genres. Genres such as Documentary, Short, and Animation have a higher proportion of high-rated films. In contrast, Drama, Action, and Romance tend to have fewer high-rated films. Moreover, Comedy shows a more balanced distribution, making it less clear to classify.

```
ggplot(clean_data, aes(x = decade_group, fill = as.factor(rating_above_7)))+
  geom_bar(position = "fill", alpha = 0.6) +
scale_y_continuous(labels = scales::percent_format()) +
  scale_fill_manual(
    values = c("0" = "darkorange", "1" = "darkorchid"),
    labels = c("<= 7", "> 7")) +
  labs(
    x = "Decade Group",
    y = "Proportion",
    fill = "IMDB Scores") +
  theme_bw()
```

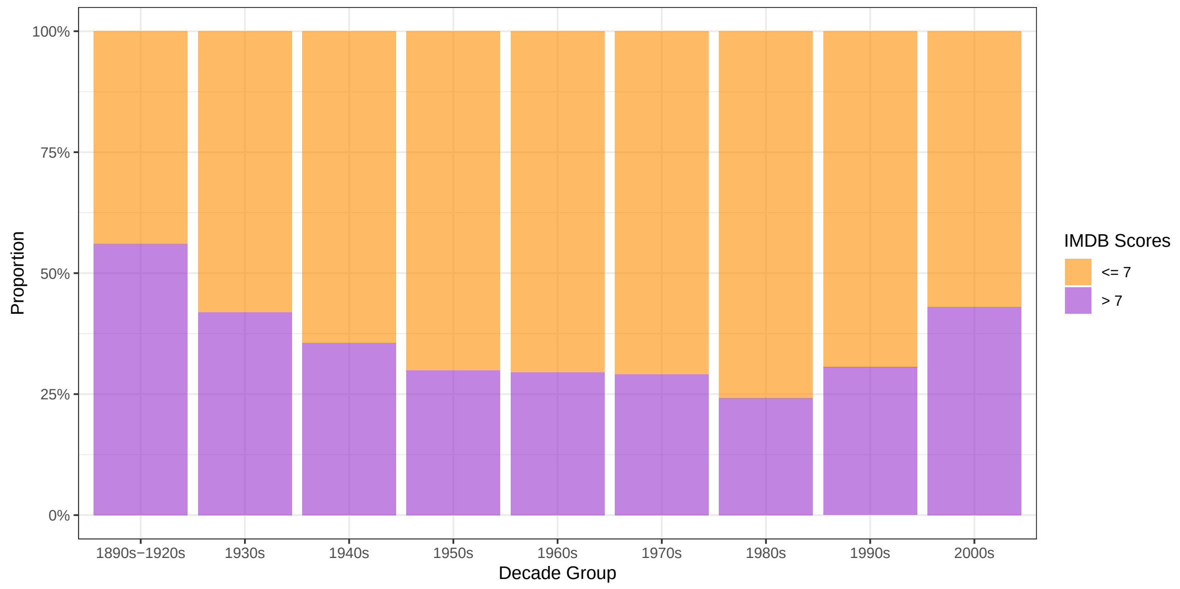


Figure 6: Stacked Bar plot of between Year in Decade and IMDB score

As shown in Figure 6, representing the proportion of films with high IMDB scores(rating > 7) and films with low IMDB scores (rating $<= 7$) across different decades. In this plot, it's difficult to draw strong conclusions from decade-based trends, as the proportions remain fairly similar. However, one notable exception is the 2000s, which show a higher proportion of high-rated films compared to earlier decades.

```r
library(patchwork)
#Boxplot : Length vs Rating
p1 <- ggplot(clean_data, aes(x = as.factor(rating_above_7),
                       y = length,
                       fill = as.factor(rating_above_7))) +
  geom_boxplot(alpha = 0.6) +
  scale_fill_manual(values = c("0" = "darkorange",
                               "1" = "darkorchid"),
                    labels = c("<=7", "> 7"))+
  labs(x = "Rating Groups",
       y = "Film Length (minutes)",
       fill = "IMDB Scores") +
  theme_bw()

#Boxplot : Budget vs Rating
p2 <- ggplot(clean_data, aes(x = as.factor(rating_above_7),
                       y = budget,
                       fill = as.factor(rating_above_7))) +
  geom_boxplot(alpha = 0.6) +
  scale_fill_manual(values = c("0" = "darkorange",
                               "1" = "darkorchid"),
                    labels = c("<= 7", "> 7"))+
  labs(x = "Rating Groups",
       y = "Budget (in $10 million)",
       fill = "IMDB Scores") +
  theme_bw()
#combine the two plots
p1+p2
```

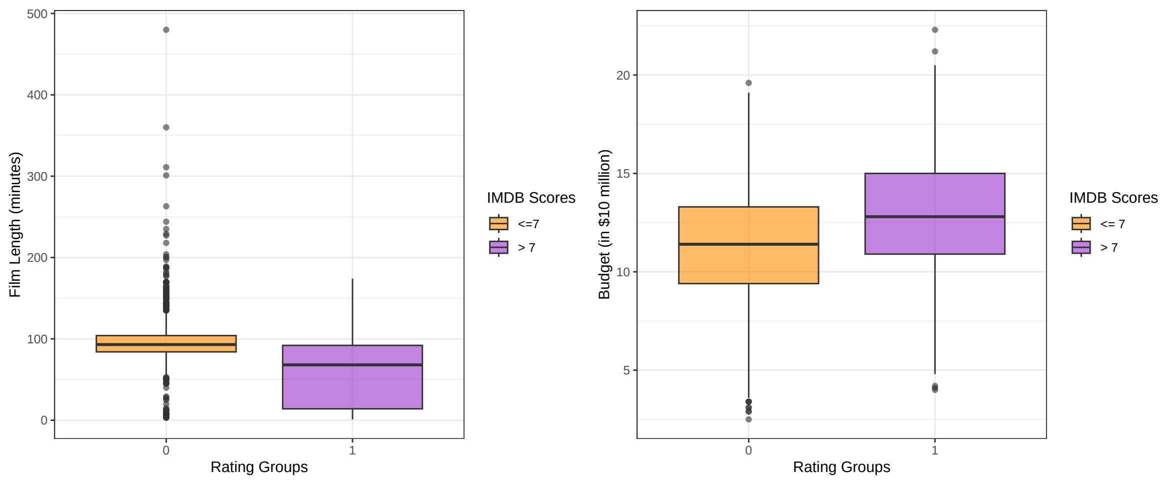


Figure 7: Boxplots comparing Film Length and Budget across IMDB Rating Groups

As shown in Figure 7, these two boxplots compare film length and budget between films with high IMDB scores (rating > 7) and those with lower scores (rating 7). For film length, although high-rated films tend to have a lower median, they also show a wider range in length. In Budget part, we can observe that films with higher ratings generally have a slightly higher median budget, suggesting that larger budgets might be related with better ratings.

```
clean_data %>%
  group_by(year, genre) %>%
  summarize(mean_rating = mean(rating), .groups = "drop") %>%
  ggplot(aes(x = year, y = mean_rating, color = as.factor(genre)))+
  geom_line(linewidth = 0.6) +
  geom_hline(yintercept = 7, linetype = "dashed", color = "gray40")+
  labs(x = "Year",
       y = "Average Rating",
       color = "Genre")+
  theme_bw()
```

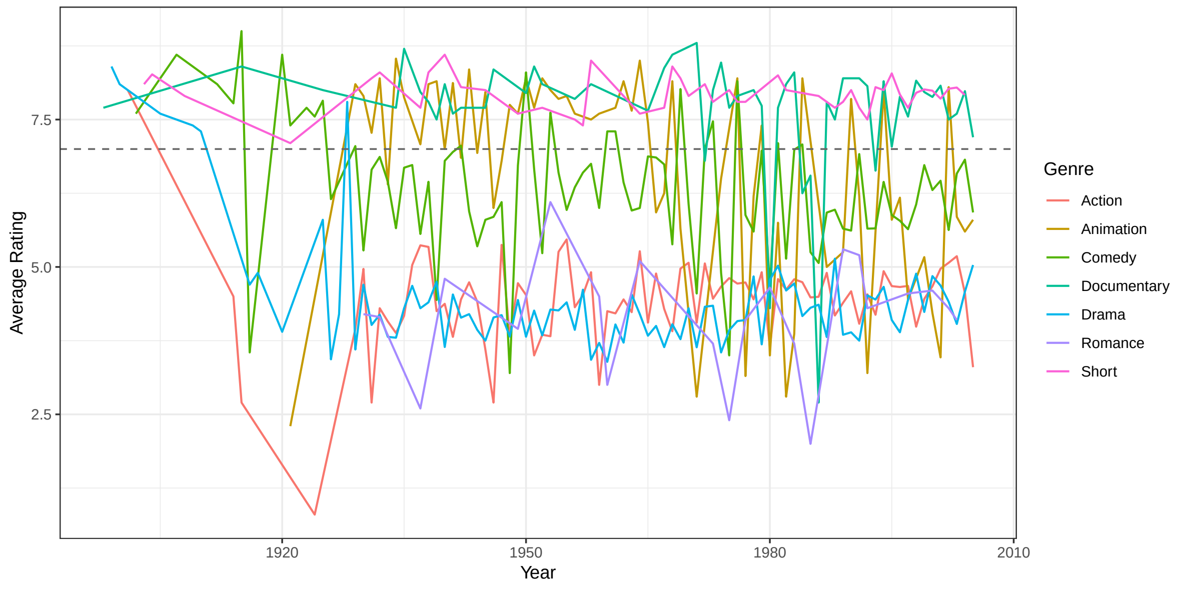


Figure 8: Line chart of average IMDB scores over time for each genre

As shown in Figure 8, the line chart illustrates the trend of average IMDB scores over time for each genre. Each line represents how a genre's average rating changed across the years. Documentary and Short often maintain higher average ratings, while Drama and Romance tend to remain below threshold of 7(indicated by the dashed line).

```
ggplot(clean_data, aes(x = as.factor(rating_above_7), y = rating, fill = genre)) +
  geom_boxplot(alpha = 0.7, position = position_dodge2(preserve = "single")) +
  labs(
    x = "Rating > 7",
    y = "IMDB score from 0-10",
    fill = "Genre") +
  theme_bw()
```

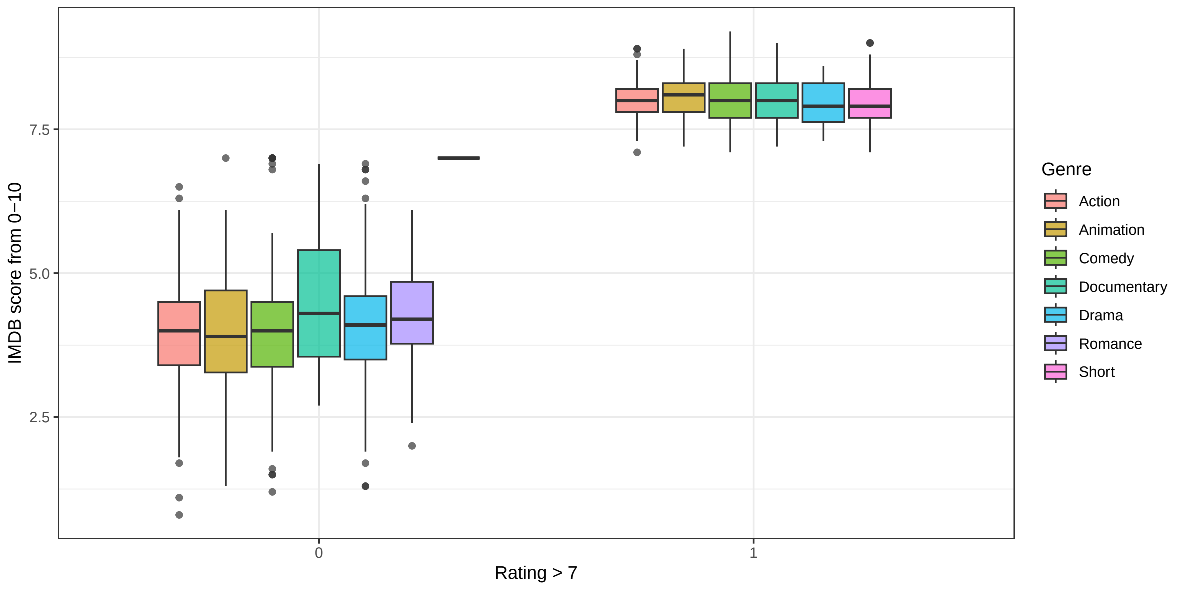


Figure 9: Boxplot of IMDB Ratings by Genre and IMDB Rating Group

As shown in Figure 9, the boxplot displays the distribution of IMDB scores across the different genre by spliting into films with high IMDB scores (rating > 7) and films with low IMDB scores (rating $<= 7$). We observe that this plot reflects separation clearly between the two groups due to the threshold of 7, as expected from the target variable definition.

```
ggplot(clean_data, aes(x = log(votes +1) , fill = as.factor(rating_above_7))) +
  geom_density(alpha = 0.6) +
  scale_fill_manual(values = c("0" = "darkorange", "1" = "darkorchid"),
                    labels = c("<= 7", "> 7")) +
  labs( x = "Number of Votes",
        y = "Density",
        fill = "IMDB Scores") +
        theme_bw()
```

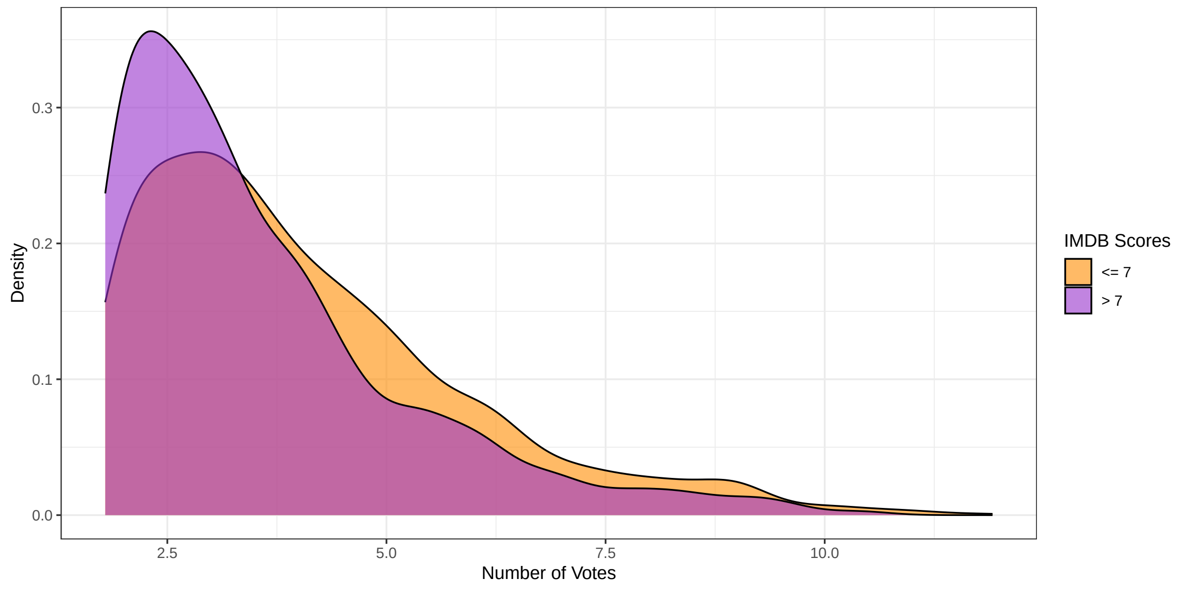


Figure 10: Density plot of Viewer Votes by IMDB Rating Group

As shown in Figure 10, the density plot displays the distribution of the number of votes for films with films with high IMDB scores (rating > 7) compared to films with low IMDB scores (rating $<= 7$). This plot can complements Figure 2 by providing a smoother and more continuous view of the vote distribution across rating groups.

3.4 Correlation Matrix

```
library(corrplot)
# Computed correlation matrix from numeric variables
cor_matrix <- cor(clean_data[, c("length", "budget", "votes", "rating")], use = "complete.obs
# Draw a correlation Heatmap by 'ggplot'
corrplot(cor_matrix, method = "color",
         addCoef.col = "black",
         tl.col = "black",
         tl.cex = 0.8,
         number.cex = 0.7)
```

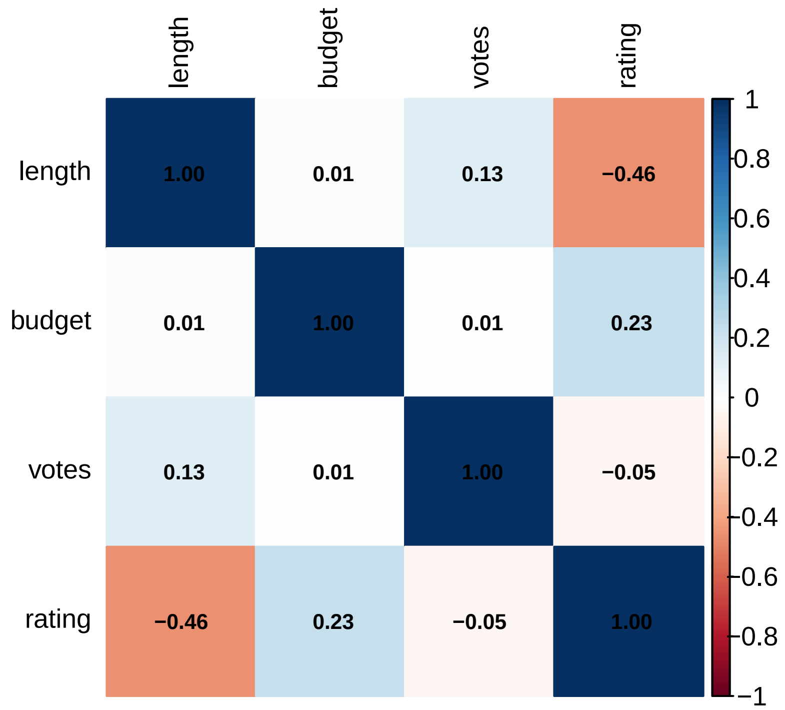


Figure 11: Correlation Heatmap of numeric variables

As shown in Figure 11, this plot shows the correlation between numeric variables. We can see that there is no strong correlation between any pairs of variables, since all values are below 0.5. One thing to notice is that film length has the highest absolute correlation (around -0.46), which suggests a moderate negative relationship, meaning that longer films may tend to get lower ratings. For budget, the correlation is quite weak, so it may not be meaningful to interpret it on its own. Lastly, votes show a very low correlation (around -0.05), which might suggest that we should consider applying a transformation before using it in modeling.

4 Statistical Modelling

In this section, we will perform the modelling of the generalised linear model.

From the visualisation results, the votes variables show a right-skewed (skewed distribution), so a log transformation is needed before modelling:

```
#Performs a log transformation on the votes variable
clean_data=clean_data%>%
  mutate(log_votes=log(votes+1)) #Avoiding the log(0) problem
```

Firstly, to test whether year should be put into the model as a continuous or grouped variable, we fitted a model for each and observed their AIC values:

```
#Fitting the GLM logistic regression model
glm_model=glm(rating_above_7~length+log_votes+budget+genre+year,
                 data=clean_data,
                 family=binomial(link="logit"))
summary(glm_model)
```

```
Call:
glm(formula = rating_above_7 ~ length + log_votes + budget +
    genre + year, family = binomial(link = "logit"), data = clean_data)

Coefficients:
                   Estimate Std. Error z value Pr(>|z|)
(Intercept)      -22.410737   5.848178  -3.832 0.000127 ***
length            -0.058280   0.003691 -15.788  < 2e-16 ***
log_votes          0.060259   0.040508   1.488 0.136857
budget             0.510924   0.030160  16.941  < 2e-16 ***
genreAnimation    -0.168236   0.327364  -0.514 0.607314
genreComedy        3.069028   0.180969  16.959  < 2e-16 ***
genreDocumentary   5.648565   0.446796  12.642  < 2e-16 ***
genreDrama        -1.568914   0.239578  -6.549 5.81e-11 ***
genreRomance     -14.620723 390.700297  -0.037 0.970149
genreShort         3.978589   0.795084   5.004 5.62e-07 ***
year               0.009445   0.002987   3.162 0.001565 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 3470.8  on 2715  degrees of freedom
Residual deviance: 1454.4  on 2705  degrees of freedom
AIC: 1476.4

Number of Fisher Scoring iterations: 15
```

```r
glm_model1=glm(rating_above_7~length+log_votes+budget+genre+decade_group,
                 data=clean_data,
                 family=binomial(link="logit"))
summary(glm_model1)
```

```
Call:
glm(formula = rating_above_7 ~ length + log_votes + budget +
    genre + decade_group, family = binomial(link = "logit"),
    data = clean_data)

Coefficients:
                    Estimate Std. Error z value Pr(>|z|)
(Intercept)        -4.377283   0.633780  -6.907 4.96e-12 ***
length             -0.059689   0.003833 -15.573  < 2e-16 ***
log_votes           0.074184   0.041124   1.804   0.0712 .
budget              0.514208   0.030408  16.910  < 2e-16 ***
genreAnimation     -0.231877   0.332119  -0.698   0.4851
genreComedy         3.104698   0.183114  16.955  < 2e-16 ***
genreDocumentary    5.701071   0.451668  12.622  < 2e-16 ***
genreDrama         -1.554340   0.241963  -6.424 1.33e-10 ***
genreRomance      -14.603838 389.730571  -0.037   0.9701
genreShort          4.019767   0.805525   4.990 6.03e-07 ***
decade_group1930s  -0.046639   0.543830  -0.086   0.9317
decade_group1940s   0.455167   0.561532   0.811   0.4176
decade_group1950s   0.475142   0.561143   0.847   0.3971
decade_group1960s   0.935925   0.562839   1.663   0.0963 .
decade_group1970s   0.877435   0.565057   1.553   0.1205
decade_group1980s   0.614581   0.553248   1.111   0.2666
decade_group1990s   0.564173   0.542689   1.040   0.2985
decade_group2000s   0.978918   0.539545   1.814   0.0696 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 3470.8  on 2715  degrees of freedom
Residual deviance: 1444.0  on 2698  degrees of freedom
AIC: 1480

Number of Fisher Scoring iterations: 15
```

```r
AIC(glm_model,glm_model1)
```

```
           df      AIC
glm_model  11 1476.389
glm_model1 18 1479.967
```

From the results, the model with year as a continuous variable has lower AIC values and significant variables, so we will use this model for subsequent stepwise regressions.

```r
#Stepwise regression
best_model=stepAIC(glm_model,direction="both")
```

```
Start:  AIC=1476.39
rating_above_7 ~ length + log_votes + budget + genre + year

            Df Deviance    AIC
<none>           1454.4 1476.4
- log_votes  1   1456.6 1476.6
- year       1   1464.6 1484.6
- length     1   1834.4 1854.4
- budget     1   1878.2 1898.2
- genre      6   2431.2 2441.2
```

```r
summary(best_model)
```

```

Call:
glm(formula = rating_above_7 ~ length + log_votes + budget +
    genre + year, family = binomial(link = "logit"), data = clean_data)

Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)     -22.410737   5.848178  -3.832 0.000127 ***
length           -0.058280   0.003691 -15.788  < 2e-16 ***
log_votes         0.060259   0.040508   1.488 0.136857
budget            0.510924   0.030160  16.941  < 2e-16 ***
genreAnimation   -0.168236   0.327364  -0.514 0.607314
genreComedy       3.069028   0.180969  16.959  < 2e-16 ***
genreDocumentary  5.648565   0.446796  12.642  < 2e-16 ***
genreDrama       -1.568914   0.239578  -6.549 5.81e-11 ***
```

```
genreRomance     -14.620723 390.700297  -0.037 0.970149    
genreShort         3.978589   0.795084   5.004 5.62e-07 ***
year               0.009445   0.002987   3.162 0.001565 ** 
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 3470.8  on 2715  degrees of freedom
Residual deviance: 1454.4  on 2705  degrees of freedom
AIC: 1476.4

Number of Fisher Scoring iterations: 15
```

```
AIC(glm_model,best_model)
```

```
           df      AIC
glm_model  11 1476.389
best_model 11 1476.389
```

After the stepwise regression method, it is found that the AIC of the model is the same as the original model, but some of the variables of the original model are not significant, after that we will continue to search for the best model by eliminating the non-significant variables.

```
#Model selection by removing insignificant variables
clean_data_selected=clean_data%>%
  filter(genre%in%c("Comedy","Documentary","Drama","Short"))
glm_model_reduced=glm(rating_above_7~length+log_votes+budget+genre+year,
                      family=binomial(link="logit"),data=clean_data_selected)
summary(glm_model_reduced)
```

```
Call:
glm(formula = rating_above_7 ~ length + log_votes + budget + 
    genre + year, family = binomial(link = "logit"), data = clean_data_selected)

Coefficients:
                   Estimate Std. Error z value Pr(>|z|)    
(Intercept)      -23.792795   7.465561  -3.187  0.00144 ** 
length            -0.061252   0.004711 -13.001  < 2e-16 ***
log_votes         -0.065077   0.050232  -1.296  0.19513    
```

```
budget              0.442562   0.037770  11.717  < 2e-16 ***
genreDocumentary    2.391256   0.426869   5.602 2.12e-08 ***
genreDrama         -4.704494   0.291035 -16.165  < 2e-16 ***
genreShort          0.213984   0.812499   0.263  0.79227
year                0.012525   0.003815   3.283  0.00103 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2275.92  on 1687  degrees of freedom
Residual deviance:  843.52  on 1680  degrees of freedom
AIC: 859.52

Number of Fisher Scoring iterations: 7
```

From the results, log_votes and genreShort are still not significant and we will continue with the culling.

```
#Model selection by removing insignificant variables
clean_data_selected=clean_data%>%
  filter(genre%in%c("Comedy","Documentary","Drama"))
glm_model_reduced1=glm(rating_above_7~length+budget+genre+year,
                       family=binomial(link="logit"),data=clean_data_selected)
summary(glm_model_reduced1)
```

```
Call:
glm(formula = rating_above_7 ~ length + budget + genre + year,
    family = binomial(link = "logit"), data = clean_data_selected)

Coefficients:
                   Estimate Std. Error z value Pr(>|z|)
(Intercept)      -22.958874   7.345806  -3.125  0.00178 **
length            -0.064203   0.004709 -13.635  < 2e-16 ***
budget             0.455413   0.038834  11.727  < 2e-16 ***
genreDocumentary   2.501551   0.428456   5.839 5.27e-09 ***
genreDrama        -4.733854   0.294997 -16.047  < 2e-16 ***
year               0.012016   0.003740   3.213  0.00131 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2023.54  on 1556  degrees of freedom
Residual deviance:  816.71  on 1551  degrees of freedom
AIC: 828.71

Number of Fisher Scoring iterations: 7
```

```
AIC(glm_model_reduced,glm_model_reduced1)
```

```
                   df      AIC
glm_model_reduced   8 859.5184
glm_model_reduced1  6 828.7069
```

After this exclusion, the resulting model variables were all significant and had the smallest AIC values, and we will use the model for subsequent evaluations.

5 Model Diagnostics

In this section, we will perform model diagnostics on the resulting model.

First we will look at the goodness-of-fit of the model by calculating the pseudo R²:

```
#Evaluating the goodness-of-fit of the model
#Pseudo R²
pR2=1-(glm_model_reduced1$deviance/glm_model_reduced1$null.deviance)
print(pR2)
```

```
[1] 0.5963962
```

In GLM (logistic regression), the pseudo R^2 can be used to measure the explanatory power: as can be seen from the results, the pseudo R^2 is 0.60, which proves that the model has some explanatory power.

Next, we will perform a residual analysis:

```
#Residual Analysis
#Getting the residuals
residuals_data=data.frame(Index=1:length(residuals(glm_model_reduced1)),
                          Residuals=residuals(glm_model_reduced1,type="deviance"))
```

```
#Plotting Residuals and Trendlines
ggplot(residuals_data,aes(x=Index,y=Residuals))+
  geom_point(alpha=0.5,color="black")+
  geom_smooth(method="loess",color="red",se=FALSE)+
  labs(x="Index",y="Deviance Residuals")+
  theme_minimal()
```

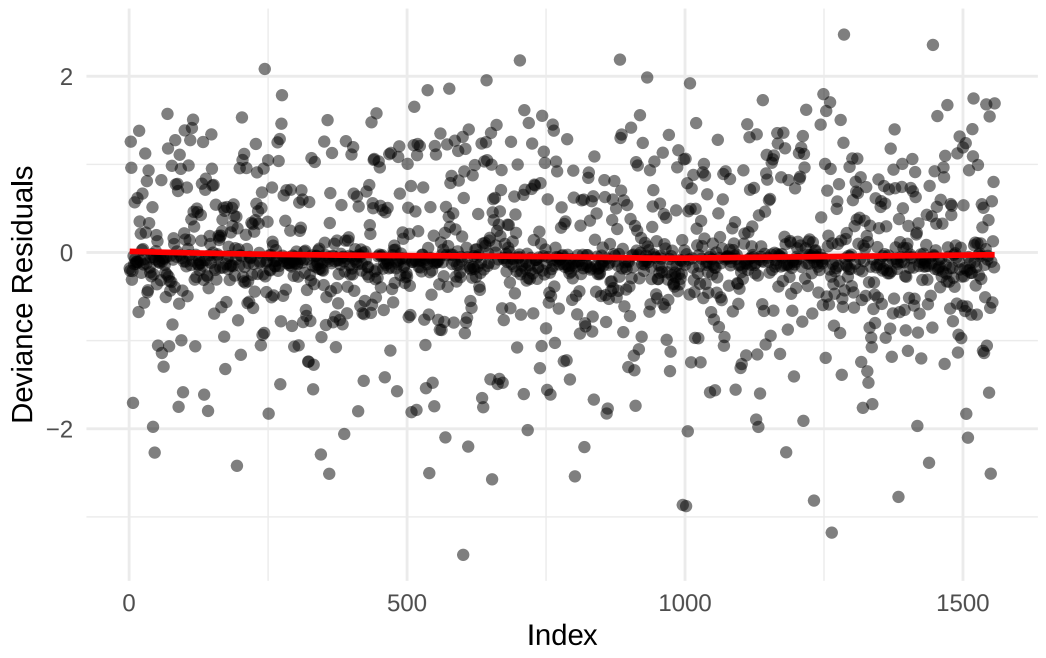


Figure 12: Residual Plot with LOESS Smoothing

```
#Plotting Histogram of Residuals
ggplot(residuals_data,aes(x=Residuals))+
  geom_histogram(bins=30,col="white",fill="lightblue")+
  labs(x="Residuals",y="Count")
```

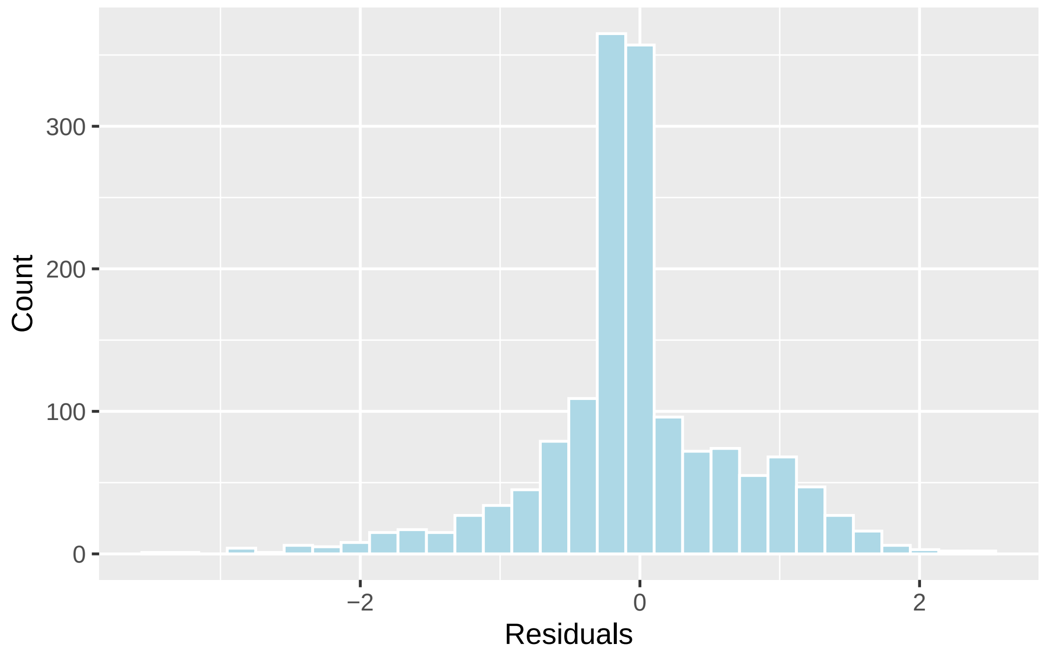


Figure 13: Histogram of Residuals

The two residual plots show that the model is overall good and acceptable.

Next we will calculate the ROC curve and AUC values to observe the predictive power of the model.

```
#Assessment of predictive capacity
#predictive probability
pred_probs=predict(glm_model_reduced1,type="response")
```

```
#Calculate ROC curve & AUC
roc_obj=roc(clean_data_selected$rating_above_7,pred_probs)
plot(roc_obj,col="blue")
```

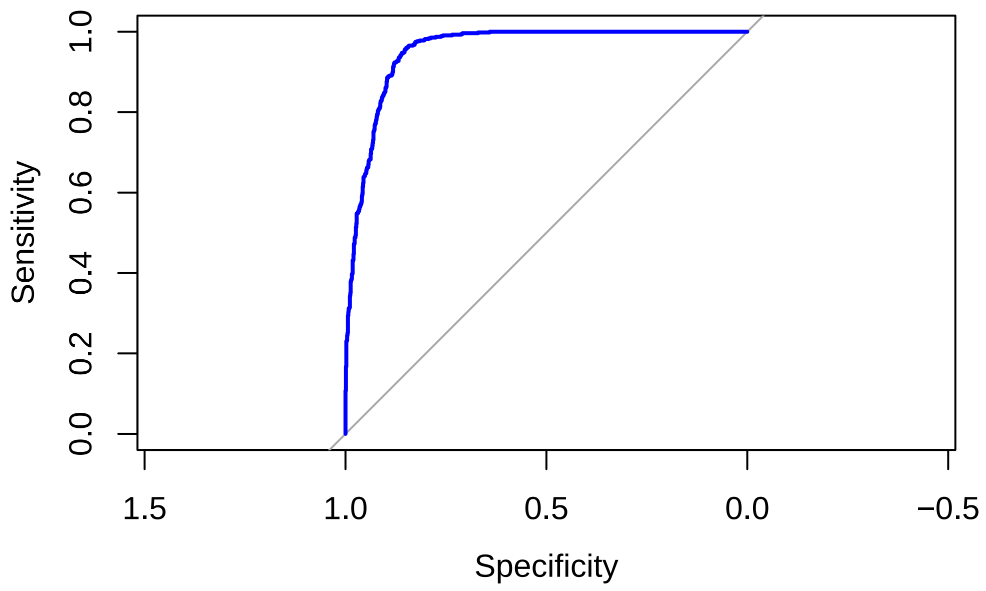


Figure 14: Plot of ROC

```
auc(roc_obj)  #View AUC values
```

```
Area under the curve: 0.9544
```

Area under the curve is 0.9544, which means ths model is good.

```
#Calculate the confusion matrix
pred_class=ifelse(pred_probs>0.5,1,0)
conf_matrix=confusionMatrix(as.factor(pred_class),as.factor(clean_data_selected$rating_above_
print(conf_matrix)
```

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 915  93
         1  91 458

               Accuracy : 0.8818
```

```
                 95% CI : (0.8647, 0.8974)
    No Information Rate : 0.6461
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.7414

 Mcnemar's Test P-Value : 0.9412

            Sensitivity : 0.9095
            Specificity : 0.8312
         Pos Pred Value : 0.9077
         Neg Pred Value : 0.8342
             Prevalence : 0.6461
         Detection Rate : 0.5877
   Detection Prevalence : 0.6474
      Balanced Accuracy : 0.8704

       'Positive' Class : 0
```

By calculating the confusion matrix, Accuracy = 88%, the model predicts more accurately overall and the model performs well and can be used for further analysis or optimisation.

```
#Multicollinearity check
vif(glm_model_reduced1)
```

```
           GVIF Df GVIF^(1/(2*Df))
length 1.602052  1        1.265722
budget 1.303684  1        1.141790
genre  1.692097  2        1.140529
year   1.078064  1        1.038299
```

In the model, the VIF values of all the variables are close to 1, indicating that there is little or no covariance between these variables. Therefore, the model is stable with respect to multicollinearity and no further treatment of covariance is required.

6 Conclusions