# IMDb Prediction Challenge

MGSC 661-075: Multivariate Statistics Midterm Project

Cine-mettrics

Lakshya A., Om S., Nandani Y., Valentin N., Tomy P.

November 2023

## 1 Introduction

In a galaxy not so far, far away, cinephiles and movie enthusiasts embark on an epic journey to unlock the secrets that November 2023 holds. With the spectral glow of stars as our guiding light, we aim to anticipate the ratings of 12 colossal blockbusters set to grace the silver screen in November 2023.

Harnessing the Force of data, we delve deep into the archives of IMDB, a treasure trove of cinematic history containing over a 1,000 films. The wisdom of ages, distilled into bits and bytes of information, becomes our guiding star map. Our task? To extract the cosmic patterns and understand the celestial alchemy that have birthed timeless classics and box office supernovas.

As a group given high-stakes responsibility, we embrace the spirit of Jedi wisdom, using cutting-edge predictive analytics, machine learning, and a dash of cinematic intuition to foresee the critical acclaim and audience reception that these forthcoming blockbusters shall command.

So, climb aboard our spin-off of Obi-Wan Delta-7 Aethersprite-Class light interceptor, and buckle up as we navigate the analytic starscape with precision and intuition.

May the force be with you!

## 2 Data description

Before diving into the intricacies of our model, it's crucial to describe the data and discuss the preprocessing steps that were taken to prepare the dataset for analysis. These steps included data cleaning, transformation, and feature selection, which are essential for ensuring the reliability and robustness of the predictive model.

### 2.1 IMDb Score

The IMDb score is the target variable that we aim to predict. Figure 1 shows the distribution of IMDb scores in the training set. The distribution of IMDb scores appears to be left-skewed. This skewness indicates that while most films achieve a moderate level of success according to IMDb scores, the number of films rated poorly by viewers is relatively small.

## 2.2 Movie budget

The budget of a movie is the total amount of money spent on producing the film. The budget is a continuous variable, and the distribution of the budget is shown in Figure 2a. It is skewed to the right , indicating that a larger number of films have smaller budgets.

The bivariate relationship between the IMDb score and the budget is shown in Figure 2b. There are no specific trends/patterns that can be observed from the scatter plot, but it's important to note that when the budget is high we do not see any movies with a low IMDb score.

## 2.3 Release year

The release year is the year in which the movie was released. The year of release was excluded, likely because it did not provide significant predictive power for the IMDb score

## 2.4 Release month

The release month is the month in which the movie was released. The number of movies released in each month is shown in Figure 3a.

From the given dataset, we see that January and October have the highest number of movies released, coinciding with the holiday season.

The bivariate relationship between the IMDb score and the release month is shown in Figure 3b. No noticeable trends emerge from the box plot.

## 2.5 Number of faces

The number of faces is the number of actors/actresses in the movie poster. The distribution of the number of faces is shown in Figure 4.

We see that the number of faces is skewed to the right, indicating that a larger number of movies have a smaller number of faces.

## 2.6 Duration

The duration is the length of the movie in minutes. The boxplot of the duration is shown in Figure 5a.

The bivariate relationship between the IMDb score and the duration is shown in Figure 5b. We see that longer running movies tend to have higher IMDb scores.

## 2.7 Genre

The distribution of ratings across genres is shown in Figure 6.

We see that there is some variablity amongst the median ratings across genres, with the highest median rating being for the "Drama" and "War" genre.

## 2.8 Language, Color and Country

The dataset demonstrated a homogeneity in language (primarily English), country of production (primarily USA), and the color type (primarily Color). These features showed limited variability, and thus were considered for exclusion when building predictive models, as they would not significantly contribute to the variance in IMDb scores.

## 2.9 Aspect Ratio and Maturity Rating

When examining these two features, it was noted that the aspect ratios were mostly "2.35" or "1.85", and the maturity ratings were commonly "R", "PG-13", "PG", or "G". To streamline the analysis and model building, values that fell outside these common categories were treated as "Others", thereby reducing the complexity of the dataset.

## 2.10 Actor Star Meter

In analyzing the impact of various predictors on IMDb scores, all predictors were found to be statistically significant except for the `"actor_star_meters"`. This suggests that the star power, as quantified by this meter, does not have a significant impact on the rating of the movie.

## 2.11 Distributor

The dataset contains movies with $\approx$ 1000 unique distributors. To reduce the complexity of the dataset, we looked at the top distributors by number of movies released (as shown in Figure 7).

The IMDb scores of movies released by top 10 distributors show some variability, with the highest median rating being for "Miramax".

## 2.12 Plot Keywords

To extract features from the plot keywords, we first created a list of all the unique keywords in the dataset. Then, we counted the top 10 keywords by frequency, and created a binary variable for each of these keywords.

## 2.13 Cinematographer and Production Company

The dataset contains movies with $\approx$ 800 unique cinematographers and production companies each. Since these variables exhibit high cardinality, which could lead to sparse data and overfitting, we decided to exclude them from the analysis.

# 3 Model building

## 3.1 Feature engineering

The steps taken to prepare the dataset for analysis are described below:

1. **Checking data types**: The data types of the variables were checked and corrected to ensure that they were classified as numeric or categorical.

2. **Standardizing numerical variables**: The numerical variables were standardized to ensure that they were on the same scale.

3. **One-hot encoding categorical variables**: The categorical variables (`"plot_keywords"` and `"distributors"`) were one-hot encoded to ensure that they could be used in the model.

4. **Removing irrelevant variables**: Irrelevant variables were removed from the dataset as they did not provide significant predictive power for the IMDb score.

## 3.2 Heteroskedasticity and linearity

Individual linear regressions were run for each of the numerical predictors, and then tests for heteroskedasticity and linearity were performed on each of the models. All the numerical predictors (except `actor_star_meter*`) were found to be significant at the 5% level.

The estimated coefficients of the significant variables (at 5% level) are shown in Figure 8.

### 3.2.1 Heteroskedasticity

Heteroskedasticity occurs when the variance of the error term is not constant across observations, which can lead to biased and inconsistent estimates of the standard errors of the regression coefficients.

Non-constant variance tests were performed on the individual linear regression models. Among the numerical predictors, `movie_budget`, `duration`, and `movie_meter_IMDBpro` were found to have a heteroskedasticity.

### 3.2.2 Linearity

Linearity tests (Tukey test) were performed on the individual linear regression models. Among the numerical predictors, `movie_budget`, and `nb_faces` have a linear relationship with the IMDb score, while the remaining predictors do not have a linear relationship with the IMDb score.

## 3.3 Multicollinearity

Multicollinearity occurs when two or more predictors are highly correlated with each other, which can lead to biased and inconsistent estimates of the regression coefficients.

The correlation matrix of the numerical predictors is shown in Figure 9. In addition to the above matrix, we looked at the variance inflation factor (VIF) of the numerical predictors.

From both the correlation matrix and the VIF, we did not find any evidence of multicollinearity among the numerical predictors.

### 3.4 Outliers

Outliers are observations that are significantly different from the other observations in the dataset.

We ran Bonferroni outlier tests on a simple linear regression model with all the relevant predictors, and identified $\approx 10$ outliers to be removed.

### 3.5 Predictors

The final features that were used in the model are listed in Table 1.

## 4 Results

After performing the above steps, we attempted three different models to predict the IMDb score:

1. **Linear regression**: A simple linear regression model was run with all the relevant predictors.
2. **Polynomial regression**: A polynomial regression model was run with non-linear predictors (identified in 3.2.2).
3. **Spline regression**: A spline regression model was run to try and capture the non-linear relationship more accurately.

The performance of the three models is shown in Table 2.

### 4.1 Linear regression

The linear regression model formula is shown below:

$$\begin{aligned}\text{imdb\_score} = &\text{movie\_budget} + \text{duration} + \text{release\_month}\\ &+ \text{maturity\_rating} + \text{aspect\_ratio} + \text{nb\_news\_articles} + \text{nb\_faces}\\ &+ \text{genre} + \text{movie\_meter\_IMDBpro} + \text{plot\_keywords} + \text{distributor}\end{aligned}$$

However, the model suffered from non-linear predictors.

## 4.2 Polynomial regression

For polynomial regression, we ran a grid search for the optimal degree of the non-linear predictors. After 125 iterations, we found the degrees that gave us the minimum MSE.

The polynomial regression model formula is shown below:

$$\begin{aligned}\text{imdb\_score} =& \text{movie\_budget} + \text{duration} + +\text{duration}^2 + \text{release\_month}\\ &+ \text{maturity\_rating} + \text{aspect\_ratio}\\ &+ \text{nb\_news\_articles} + \text{nb\_news\_articles}^2 + \text{nb\_news\_articles}^3 + \text{nb\_news\_articles}^4\\ &+ \text{nb\_faces} + \text{genre}+\\ &+ \text{movie\_meter\_IMDBpro} + \text{movie\_meter\_IMDBpro}^2 + \text{movie\_meter\_IMDBpro}^3\\ &+ \text{movie\_meter\_IMDBpro}^4 + \text{movie\_meter\_IMDBpro}^5\\ &+ \text{plot\_keywords} + \text{distributor}\end{aligned}$$

This model performed better than the linear regression model, with a 20-fold cross-validated MSE of 0.68.

## 4.3 Spline regression

For spline regression, we chose the number of knots to be 3, placing them at the 25th, 50th, and 75th percentiles of the data. For brevity, we omit the formula of the model here, but it can be found in the code.

This model performed marginally better than the polynomial regression model, with a 20-fold cross-validated MSE of 0.67.

## 4.4 Predictions

The predictions of all the three regression models on the test set are show in Table 3.

## 4.5 Model selection

Among the three models we tried, the spline regression model performed the best in terms of MSE. However, it presented significant issues with predictions outside the training set as it exceeded the range of IMDb scores (0-10). Further, the number of knots was chosen arbitrarily, and the model has the lowest interpretability among the three models.

Therefore, we elected to choose the **polynomial regression model** as our final model, as it performed marginally worse than the spline regression model, but had better interpretability.

## 4.6 Interpretation

The final model coefficients, after adjusting for heteroskedasticity, are presented in Table 4.

The following are the key takeaways from the model:

1. **Movie budget**: Contrary to popular belief, throwing money at a project doesn't necessarily make it better. Our model indicates that higher budgets are actually associated with slightly lower IMDb scores. This suggests a focus on resourceful filmmaking could be more beneficial.

2. **Duration**: Audiences appreciate a well-paced, substantial movie, but there is a point of diminishing returns. Longer movies initially receive higher IMDb scores, but this advantage tapers off for exceedingly long durations. Aim for a 'Goldilocks' duration that's just right.

3. **Number of news articles**: The number of news articles about a movie shows a quadratic relationship with IMDb scores. Initial media coverage boosts ratings, but the effect plateaus. A well-planned PR strategy that avoids overexposure could be the key.

4. **IMDB Pro Meter**: The IMDB Pro Meter, designed to gauge the star power of a movie, shows a complex relationship with IMDb scores. This implies that a star-studded cast alone won't guarantee a high rating. Focusing on script quality and production values should be a priority.

5. **Maturity rating**: Movies across all maturity ratings (G, PG, PG-13, R) are associated with lower IMDb scores. This suggests that aiming for a particular rating to attract a specific audience segment might not be as effective as making a film that naturally appeals to its intended audience.

6. **Aspect ratio**: Aspect ratios didn't show a significant impact on IMDb scores. This gives you the creative freedom to choose an aspect ratio that best serves your film's artistic needs without worrying about its market impact.

7. **Number of faces**: The model suggests that a cleaner, less cluttered poster with fewer faces is slightly more likely to be associated with a higher IMDb rating. This could mean that audiences prefer a focused promotional approach.

8. **Genre**: Different genres have varied impacts on IMDb scores. However, user preferences are highly subjective, and therefore, catering to any particular genre is not a good strategy.

9. **Plot keywords**: Themes like 'Murder' and 'Love' significantly influence IMDb scores. Leveraging these universal themes could resonate more with audiences.

10. **Distributor**: Interestingly, the choice of distributor did not significantly impact IMDb ratings. This implies that content remains king, irrespective of the distribution channel.

# Appendix A    Figures

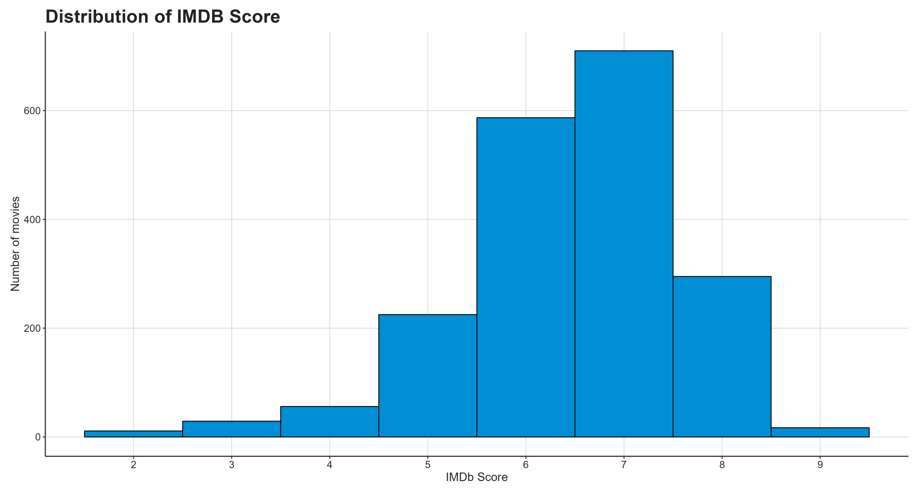
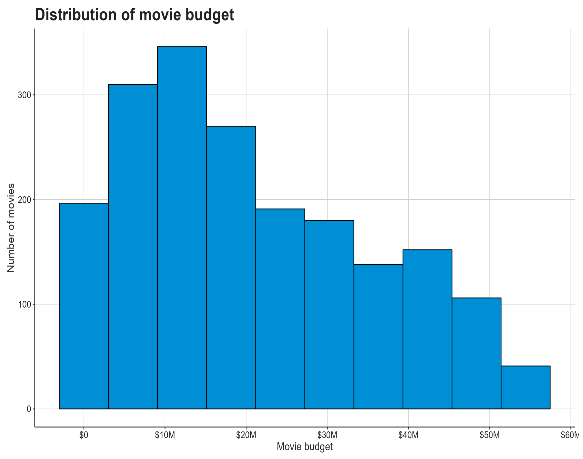
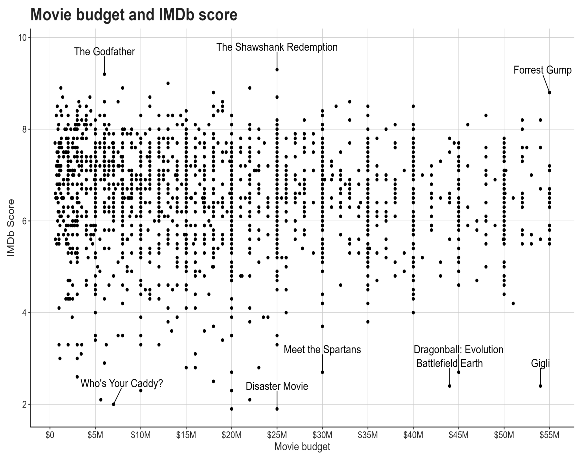


Figure 1: Distribution of IMDb score

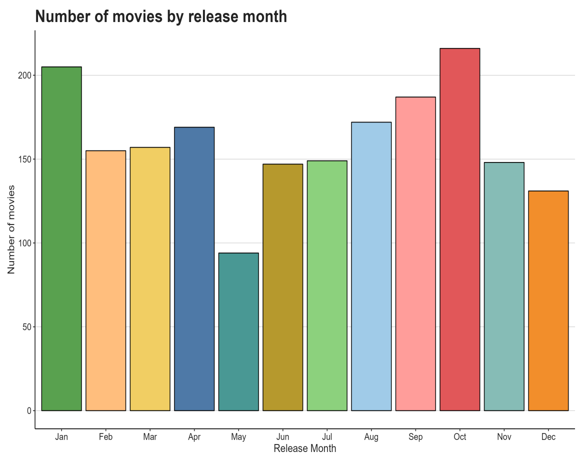


(a) Distribution of movie budget

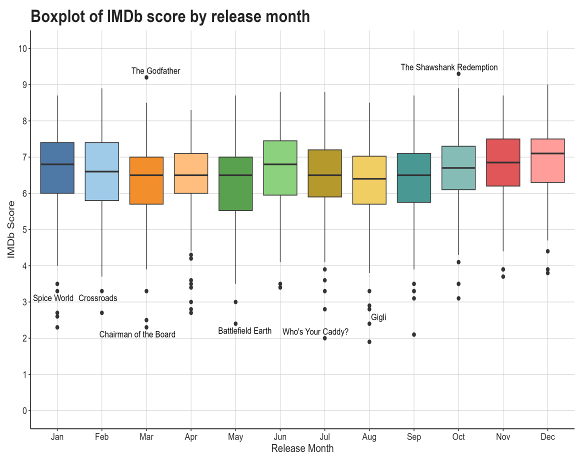


(b) IMDb score vs. movie budget

Figure 2: Movie budget



(a) Number of movies by release month



(b) IMDb score vs. release month

Figure 3: Release month

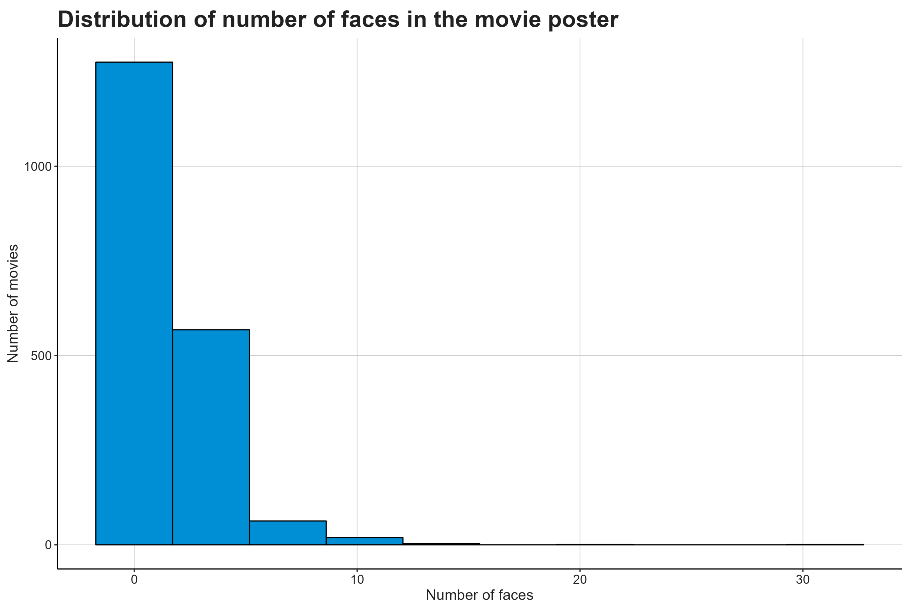
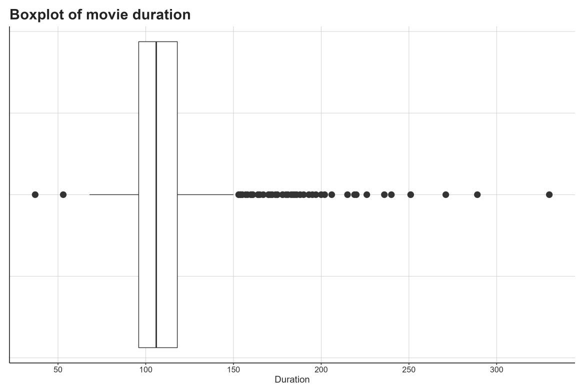
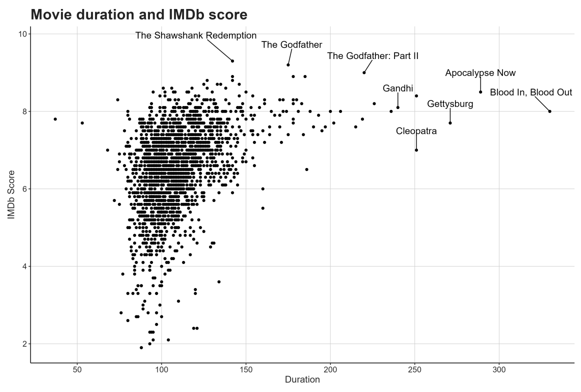


Figure 4: Distribution of number of faces



(a) Boxplot of duration



(b) IMDb score vs. duration

Figure 5: Duration

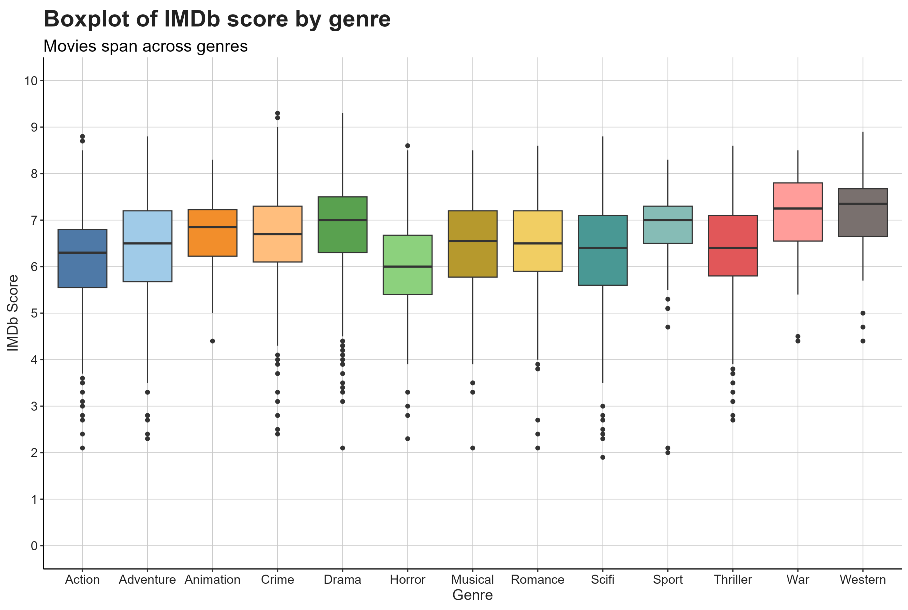


Figure 6: Distribution of ratings across genres

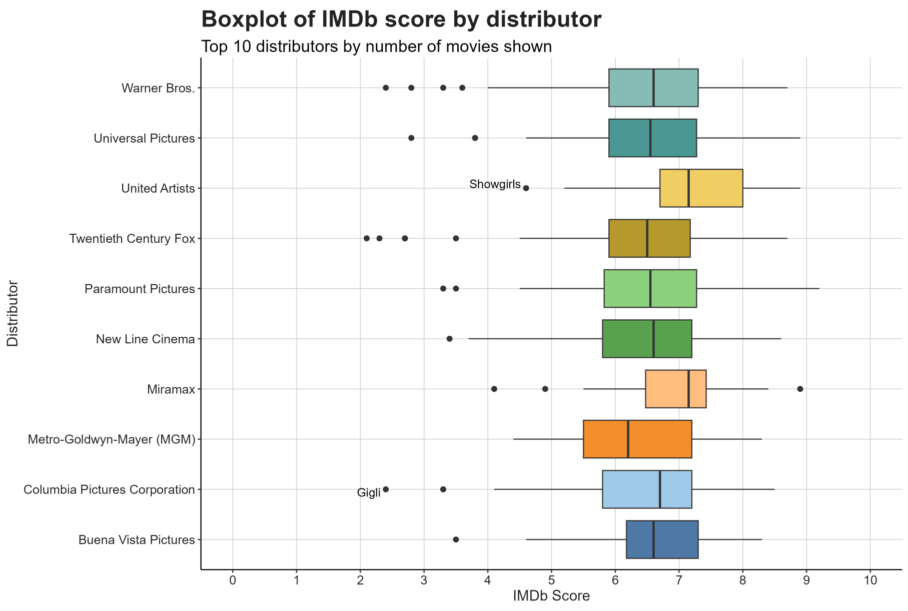


Figure 7: Distribution of movies across distributors

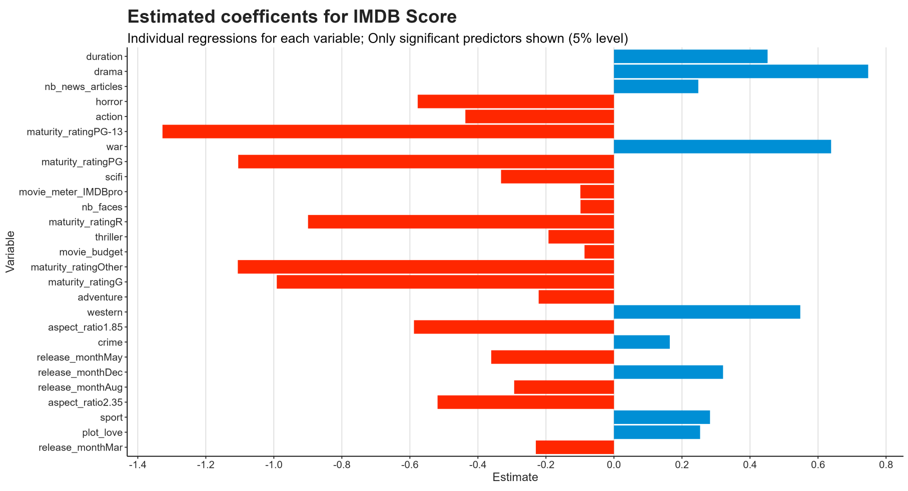


Figure 8: Individual regressions

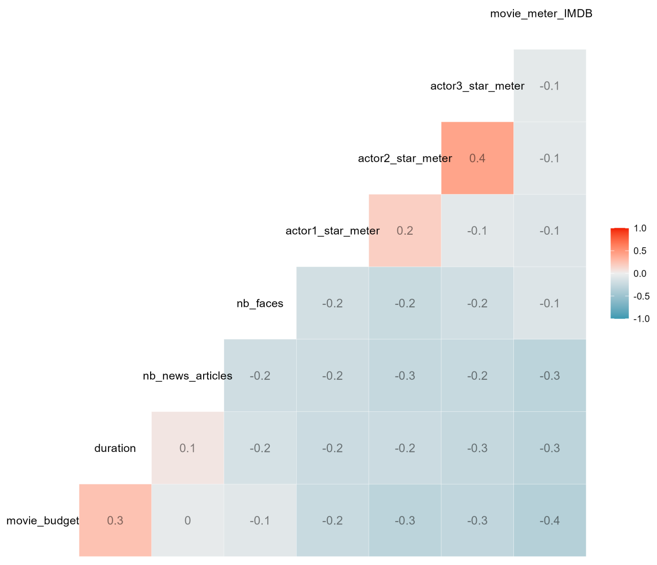


Figure 9: Correlation matrix

## Appendix B Tables

| **Numerical Predictors** | **Categorical Predictors** |
|---|---|
| `Movie Budget` | `Release month` |
| `Duration` | `Maturity rating` |
| `Number of faces` | `Aspect ratio` |
| `Number of news articles` | `Genre` |
| `Movie meter IMDBPro` | `Plot keywords` |

Table 1: List of Predictors

| **Metric** | **Linear** | **Polynomial** | **Spline** |
|---|---|---|---|
| MSE | 0.76 | 0.68 | 0.67 |
| RMSE | 0.87 | 0.83 | 0.82 |
| $R^2$ | 0.38 | 0.45 | 0.46 |
| Number of predictors | 52 | 60 | 61 |

Table 2: Model performance ($k = 20$)

| **Row ID** | **Movie Title** | **Release Date** | **Linear** | **Polynomial** | **Spline** |
|---|---|---|---|---|---|
| 1 | Pencils vs Pixels | 2023-11-07 | 6.19 | 5.87 | 3.44 |
| 2 | The Dirty South | 2023-11-10 | 6.45 | 5.41 | 3.40 |
| 3 | The Marvels | 2023-11-10 | 6.82 | 7.08 | 9.41 |
| 4 | The Holdovers | 2023-11-10 | 7.22 | 7.66 | 8.70 |
| 5 | Next Goal Wins | 2023-11-17 | 6.78 | 7.43 | 8.59 |
| 6 | Thanksgiving | 2023-11-17 | 6.31 | 7.13 | 9.66 |
| 7 | The Hunger Games: The Ballad of Songbirds and Snakes | 2023-11-17 | 7.02 | 7.90 | 10.38 |
| 8 | Trolls Band Together | 2023-11-17 | 7.01 | 7.71 | 8.88 |
| 9 | Leo | 2023-11-21 | 6.92 | 6.41 | 5.08 |
| 10 | Dream Scenario | 2023-11-22 | 6.43 | 7.10 | 8.28 |
| 11 | Wish | 2023-11-22 | 6.86 | 7.83 | 10.40 |
| 12 | Napoleon | 2023-11-22 | 7.57 | 8.32 | 10.52 |

Table 3: Model predictions

Table 4: Final model coefficients (heteroskedasticity adjusted)

| | *Dependent variable:* |
|---|---|
| | imdb_score |
| Movie Budget | $-0.18^{***}$ |
| | (0.02) |
| Duration | $12.38^{***}$ |
| | (0.98) |
| Duration$^2$ | $-4.39^{***}$ |
| | (0.86) |
| News Articles | $8.25^{***}$ |
| | (0.91) |
| News Articles$^2$ | $-3.30^{***}$ |
| | (0.86) |
| News Articles$^3$ | 1.37 |
| | (0.85) |
| News Articles$^4$ | $-0.46$ |
| | (0.84) |
| IMDB Pro Meter | $-3.25^{***}$ |
| | (0.85) |
| IMDB Pro Meter$^2$ | $6.13^{***}$ |
| | (0.89) |
| IMDB Pro Meter$^3$ | $-6.34^{***}$ |
| | (0.88) |
| IMDB Pro Meter$^4$ | $4.09^{***}$ |
| | (0.86) |
| IMDB Pro Meter$^5$ | $-6.86^{***}$ |
| | (0.85) |
| Released in February | 0.08 |
| | (0.09) |

*Continued on next page*

Table 4: Final model coefficients (heteroskedasticity adjusted) (Continued)

| | *Dependent variable:* |
|---|---|
| | imdb_score |
| Released in March | $-0.05$ |
| | (0.09) |
| Released in April | $-0.02$ |
| | (0.09) |
| Released in May | $-0.15$ |
| | (0.10) |
| Released in June | 0.14 |
| | (0.09) |
| Released in July | 0.07 |
| | (0.09) |
| Released in August | $-0.01$ |
| | (0.09) |
| Released in September | $-0.08$ |
| | (0.08) |
| Released in October | 0.05 |
| | (0.08) |
| Released in November | 0.12 |
| | (0.09) |
| Released in December | 0.07 |
| | (0.09) |
| G Rating | $-0.48^{**}$ |
| | (0.24) |
| PG Rating | $-0.48^{**}$ |
| | (0.20) |
| PG-13 Rating | $-0.62^{***}$ |
| | (0.19) |
| R Rating | $-0.40^{**}$ |

*Continued on next page*

Table 4: Final model coefficients (heteroskedasticity adjusted) (Continued)

| | *Dependent variable:* |
|---|---|
| | imdb_score |
| | (0.19) |
| Other Maturity Rating | $-0.68^{***}$ |
| | (0.24) |
| Aspect Ratio 1.85 | $-0.43^{***}$ |
| | (0.16) |
| Aspect Ratio 2.35 | $-0.42^{**}$ |
| | (0.16) |
| Other Aspect Ratio | $-0.41^{**}$ |
| | (0.18) |
| Number of Faces in Poster | $-0.09^{***}$ |
| | (0.02) |
| Action | $-0.28^{***}$ |
| | (0.06) |
| Adventure | $-0.05$ |
| | (0.07) |
| Sci-Fi | 0.04 |
| | (0.07) |
| Thriller | $-0.09^{*}$ |
| | (0.05) |
| Musical | $-0.13^{*}$ |
| | (0.08) |
| Romance | $-0.13^{***}$ |
| | (0.05) |
| Western | $0.30^{**}$ |
| | (0.14) |
| Sport | $0.25^{***}$ |
| | (0.09) |

*Continued on next page*

Table 4: Final model coefficients (heteroskedasticity adjusted) (Continued)

| | *Dependent variable:* |
|---|---|
| | imdb_score |
| Horror | $-0.46^{***}$ |
| | (0.07) |
| Drama | $0.33^{***}$ |
| | (0.05) |
| War | $0.26^{**}$ |
| | (0.11) |
| Animation | $0.87^{***}$ |
| | (0.20) |
| Crime | $0.14^{***}$ |
| | (0.05) |
| Plot = Murder | $-0.17^{*}$ |
| | (0.09) |
| Plot = Love | $0.19^{**}$ |
| | (0.08) |
| Plot = Friends | $-0.07$ |
| | (0.07) |
| Plot = Death | 0.06 |
| | (0.09) |
| Plot = High School | $-0.05$ |
| | (0.16) |
| Plot = Police | 0.10 |
| | (0.10) |
| Plot = New York City | $-0.12$ |
| | (0.12) |
| Plot = a Boy | 0.04 |
| | (0.10) |

*Continued on next page*

Table 4: Final model coefficients (heteroskedasticity adjusted) (Continued)

| | *Dependent variable:* |
|---|---|
| | imdb_score |
| Plot = Drugs | 0.09 |
| | (0.14) |
| Plot = School | −0.12 |
| | (0.13) |
| Distributor = Warner Bros. | −0.03 |
| | (0.07) |
| Distributor = Universal Pictures | −0.02 |
| | (0.07) |
| Distributor = Paramount Pictures | 0.03 |
| | (0.08) |
| Distributor = Twentieth Century Fox | −0.12 |
| | (0.08) |
| Distributor = Columbia Pictures Corporation | −0.03 |
| | (0.08) |
| Constant | $7.34^{***}$ |
| | (0.24) |
| Observations | 1,923 |
| Log Likelihood | −2,287.58 |
| Akaike Inf. Crit. | 4,697.15 |
| *Note:* | $^{*}$p<0.1; $^{**}$p<0.05; $^{***}$p<0.01 |

# Appendix C  Code

~~~
---
title: "MGSC 661 — Midterm Project"
output:
  pdf_document: default
  html_notebook: default
---

# Importing libraries and data

```{r include=FALSE}
Sys.setlocale("LC_ALL", "C")
require(visreg)
require(glue)
require(car)
require(lmtest)
require(splines)
require(psych)
require(stargazer)
require(plm)
require(ggplot2)
require(ggtext)
require(ggpubr)
require(ggrepel)
require(ggthemes)
require(scales)
require(caTools)
require(methods)
require(boot)
require(tidyverse)
require(dplyr)
require(glue)
require(GGally)
require(broom)


windowsFonts(Helvetica = "Product Sans")
~~~

```{r}
theme_lox <- function() {
  theme(
    panel.grid.major.x = element_line(linewidth = 0.3, colour = "#cbcbcb"),
    panel.grid.major.y = element_line(linewidth = 0.3, colour = "#cbcbcb"),
    plot.title = element_markdown(
      family = "Helvetica",
      size = 22,
```

```
      face = "bold",
      color = "#222222"
    ),
    plot.subtitle = element_text(
      family = "Helvetica",
      size = 16,
      margin = margin(2, 0, 2, 0)
    ),
    plot.caption = element_text(family = "Helvetica", face = "bold"),
    axis.text = element_text(
      family = "Helvetica",
      size = 12,
      color = "#222222"
    ),
    axis.title = element_text(
      family = "Helvetica",
      size = 14,
      color = "#222222"
    ),
    legend.text = element_text(family = "Helvetica", size = 12),
    legend.title = element_text(
      family = "Helvetica",
      size = 14,
      face = "bold"
    ),
    legend.position = "right",
    strip.text = element_text(
      family = "Helvetica",
      size = 12,
      hjust = 0.5
    )
  )
}
```
```{r}
data <- read.csv("./IMDB_data_Fall_2023.csv")
```

# Exploratory data analysis

```{r}
head(data)
summary(data)
```

### IMDB Score
```
```

~~~
```{r fig.width=15, fig.height=8}
ggplot(data, aes(x = imdb_score)) +
  geom_histogram(binwidth = 1, fill = '#008FD5', color='black') +
  scale_fill_fivethirtyeight() +
  scale_x_continuous(breaks = breaks_width(width = 1)) +
  labs(x = "IMDb Score", y = "Number of movies", title = "Distribution of I
  theme_pubr() +
  theme_lox()

ggsave(filename = "plots/imdb_score.png", device = "png")
~~~

### Movie budget

```{r fig.width=12, fig.height=8}
ggplot(data, aes(x = movie_budget)) +
  geom_histogram(bins = 10, fill = "#008FD5", color = "black") +
  scale_x_continuous(breaks = breaks_pretty(), labels = label_dollar(scale_
  labs(x = "Movie budget", y = "Number of movies", title = "Distribution of
  theme_pubr() +
  theme_lox()

ggsave(filename = "plots/movie_budget.png", device = "png")
```

### Release year

```{r fig.width=12, fig.height=8}
ggplot(data, aes(x = release_year)) +
  geom_histogram(bins = 10, fill = "#008FD5", color="black") +
  scale_x_continuous(breaks = breaks_pretty(n = 10)) +
  labs(x = "Release Year", y = "Number of movies", title = "Distribution of
  theme_pubr() +
  theme_lox()

ggsave(filename = "plots/release_year.png", device = "png")
```

### Release month

```{r fig.width=12, fig.height=8}
ggplot(data, aes(x = factor(release_month, levels = month.abb))) +
  geom_bar(aes(fill = release_month), color = "black") +
  guides(fill="none") +
  scale_fill_tableau(palette = "Tableau 20") +
  labs(x = "Release Month", y = "Number of movies", title = "Number of mov
  theme_pubr() +
  theme_lox() +
```

```
  theme(panel.grid.major.x = element_blank())

ggsave(filename = "plots/release_month.png", device = "png")
```

### Number of faces

```{r fig.width=12, fig.height=8}
ggplot(data, aes(x = nb_faces)) +
  geom_histogram(bins = 10, fill = "#008FD5", color = "black") +
  scale_x_continuous(breaks = breaks_pretty()) +
  labs(x = "Number of faces", y = "Number of movies", title = "Distribution
  theme_pubr() +
  theme_lox()

ggsave(filename = "plots/number_faces.png", device = "png")
```

### Duration

```{r fig.width=12, fig.height=8}
ggplot(data, aes(x = duration)) +
  geom_boxplot(outlier.size = 4) +
  scale_x_continuous(breaks = breaks_pretty()) +
  labs(x = "Duration", title = "Boxplot of movie duration") +
  theme_pubr() +
  theme_lox() +
  theme(axis.ticks.y = element_blank(), axis.text.y = element_blank())

ggsave(filename = "plots/duration.png", device = "png")
```

## Bivariate distributions

### Movie budget ~ IMDB Score

```{r fig.width=12, fig.height=8}
ggplot(data, aes(x = movie_budget, y = imdb_score)) +
  geom_point() +
  geom_text_repel(aes(label = movie_title), size = 5, max.overlaps = 2, nuc
  scale_x_continuous(breaks = breaks_pretty(n=10), labels = label_dollar(sc
  labs(x = "Movie budget", y = "IMDb Score", title = "Movie budget and IMDb
  theme_pubr() +
  theme_lox()

ggsave(filename = "plots/movie_budget_imdb_score.png", device = "png")
```

~~~
### Duration ~ IMDB Score

```{r fig.width=12, fig.height=8}
ggplot(data, aes(x = duration, y = imdb_score)) +
  geom_point() +
  geom_text_repel(aes(label = movie_title), size = 5, max.overlaps = 3, nuc
  scale_x_continuous(breaks = breaks_pretty(n=10), labels = label_number())
  labs(x = "Duration", y = "IMDb Score", title = "Movie duration and IMDb s
  theme_pubr() +
  theme_lox()

ggsave(filename = "plots/duration_imdb_score.png", device = "png")
~~~

### Genre ~ IMDB Score

```{r fig.width=12, fig.height=8}
data %>%
  select(movie_id, movie_title, imdb_score, action:crime) %>%
  pivot_longer(cols = c(-imdb_score, -movie_id, -movie_title), names_to = "
  mutate(genre = str_to_sentence(genre)) %>%
  group_by(genre) %>%
  filter(value == 1) %>%
  ggplot(aes(x = genre, y = imdb_score, fill = genre)) +
  geom_boxplot() +
  scale_fill_tableau(palette = "Tableau 20") +
  scale_y_continuous(breaks = breaks_pretty(n = 10), limits = c(0, 10)) +
  guides(fill = "none") +
  labs(x = "Genre", y = "IMDb Score", title = "Boxplot of IMDb score by ger
  theme_pubr() +
  theme_lox()

ggsave(filename = "plots/genre_imdb_score.png", device = "png")
```

### Release Month ~ IMDB Score

```{r fig.width=12, fig.height=8}
data %>%
  mutate(release_date = dmy(str_c(release_day, release_month, release_year,
  mutate(release_month = month(release_date, label = TRUE)) %>%
  ggplot(aes(x = release_month, y = imdb_score, fill = release_month)) +
  geom_boxplot(outlier.size = 2) +
  geom_text_repel(aes(label = movie_title), max.overlaps = 3, size = 4) +
  scale_fill_tableau(palette = "Tableau 20") +
  scale_y_continuous(breaks = breaks_pretty(n=10), limits = c(0, 10)) +
  guides(fill = "none") +
  labs(x = "Release Month", y = "IMDb Score", title = "Boxplot of IMDb scor
```

```
  theme_pubr() +
  theme_lox()

ggsave(filename = "plots/release_month_imdb_score.png", device = "png")
```

~~~
### Actor 1 ~ IMDB Score

```{r fig.width=12, fig.height=8}
data %>%
  filter(actor1 %in% (
    data %>%
      group_by(actor1) %>%
      count(sort = TRUE) %>%
      head(10)
  )$actor1) %>%
  ggplot(aes(
    x = fct_reorder(actor1, imdb_score, .fun = median),
    y = imdb_score,
    fill = actor1
  )) +
  geom_boxplot(outlier.size = 2) +
  geom_text_repel(aes(label = movie_title), max.overlaps = 2, size = 4) +
  scale_fill_tableau(palette = "Tableau 20") +
  scale_y_continuous(breaks = breaks_pretty(n=10), limits = c(0, 10)) +
  guides(fill = "none") +
  labs(
    x = "Actor 1 Name",
    y = "IMDb Score",
    title = "Boxplot of IMDb score by Actor 1",
    subtitle = "Top 10 actors by number of movies shown"
  ) +
  theme_pubr() +
  theme_lox()

ggsave(filename = "plots/actor1_imdb_score.png", device = "png")
~~~

### Distributor ~ IMDB Score

```{r fig.width=12, fig.height=8}
data %>%
  filter(distributor %in%
           (data %>%
              group_by(distributor) %>%
              count(sort = TRUE) %>%
              head(10))$distributor) %>%
  ggplot(aes(x = distributor, y = imdb_score, fill = distributor)) +
```

~~~
  geom_boxplot(outlier.size = 2) +
  geom_text_repel(aes(label = movie_title), max.overlaps = 2, size = 4) +
  scale_fill_tableau(palette = "Tableau 20") +
  scale_y_continuous(breaks = breaks_pretty(n=10), limits = c(0, 10)) +
  guides(fill = "none") +
  labs(
    x = "Distributor",
    y = "IMDb Score",
    title = "Boxplot of IMDb score by distributor",
    subtitle = "Top 10 distributors by number of movies shown"
  ) +
  theme_pubr() +
  theme_lox() +
  coord_flip()

ggsave(filename = "plots/distributor_imdb_score.png", device = "png")
```

# Data preprocessing

## Checking data types

```{r}
data %>% str()
```

```{r}
data_cleaned <- data %>%
  mutate(across(
    .cols = c(
      'language',
      'country',
      'maturity_rating',
      'aspect_ratio',
      'distributor',
      'director',
      'colour_film',
      'cinematographer',
      'production_company'
    ),
    ~ factor(.x)
  ))

data_cleaned <- data_cleaned %>%
  mutate(release_month = month(fast_strptime(release_month, format = "%b"),
                                              TRUE)) %>%
  mutate(release_month = factor(release_month, levels = month.abb, ordered
```
~~~

~~~
## Standardizing numerical columns
```{r}
data_cleaned <- data_cleaned %>%
  mutate(across(
    .cols = c(
      "movie_budget",
      'duration',
      'nb_news_articles',
      'actor1_star_meter',
      'actor2_star_meter',
      'actor3_star_meter',
      'nb_faces',
      'movie_meter_IMDBpro'
    ),
    .fns = ~ scale(.) %>% as.vector()
  ))
~~~

## Feature engineering

### Checking levels for categorical columns

```{r}
categorical_columns <- data_cleaned %>%
  summarize(across(where(is.factor), ~nlevels(.x))) %>%
  pivot_longer(everything(), names_to = "variable", values_to = "num_levels

categorical_columns
```

We see that director, cinematographer, and production company have a lot of

Let us check the counts for the other variables

```{r}
categorical_columns %>%
  filter(!variable %in% c("director", "cinematographer", "production_compar
  pull(variable) %>%
  walk(
    ~ data_cleaned %>%
      group_by_at(.x) %>%
      count(sort = TRUE) %>%
      print()
  )

```
```
```

~~~~
We see that:

-   The language is primarily "English"
-   The country is primarily "USA"
-   The films are primarily "Color"

These features can also be dropped when building the model.

Further:
-   The aspect ratios are primarily "2.35" or "1.85"
-   The maturity ratings are primarily "R", "PG-13", "PG" or "G"

For these features, rows containing values apart from the ones specified ab

For distributor and plot keywords, we can create binary features for the to

### Lumping factors into "Others"
```{r}
data_cleaned <- data_cleaned %>% mutate(
  aspect_ratio = fct_lump_n(aspect_ratio, n = 3),
  maturity_rating = fct_lump_n(maturity_rating, n = 5)
)
```



### Top 10 keywords

```{r}
top_10_keywords <- data_cleaned %>%
  select(plot_keywords) %>%
  separate_longer_delim(cols = "plot_keywords", delim = "|") %>%
  group_by(plot_keywords) %>%
  count(sort = TRUE) %>%
  head(10)

for (keyword in top_10_keywords$plot_keywords) {
  col_name <- glue("plot_{keyword}")
  data_cleaned[[col_name]] <-
    as.integer(lapply(data_cleaned$plot_keywords, function (x) {
      str_detect(x, keyword)
    }))
}
```

### Top 5 distributors

```{r}
~~~~

```
top_5_distributors <- data_cleaned %>%
  select(distributor) %>%
  separate_longer_delim(cols = "distributor", delim = "|") %>%
  group_by(distributor) %>%
  count(sort = TRUE) %>%
  head(5)

for (distributor in top_5_distributors$distributor) {
  col_name <- glue("distributor_{distributor}")
  data_cleaned[[col_name]] <-
    as.integer(lapply(data_cleaned$distributor, function (x) {
      str_detect(x, distributor)
    }))
}
```

## Checking heteroscedasticity and linearity
```{r fig.width=12, fig.height=8}
numerical_predictors <-
  c(
    "movie_budget",
    "duration",
    "nb_news_articles",
    "nb_faces",
    "actor1_star_meter",
    "actor2_star_meter",
    "actor3_star_meter",
    "movie_meter_IMDBpro"
  )

for (predictor in numerical_predictors) {

  print(glue("Running checks for: ", predictor))

  # Run linear model
  lin_reg_predictor <-
    lm(formula(paste("imdb_score ~", predictor)), data = data_cleaned)

  print(summary(lin_reg_predictor))
  # Run non-constant variance test
  print(ncvTest(lin_reg_predictor))


  # Generate a scatterplot
  print(
    ggplot(data, aes(
      x = !!sym(predictor), y = imdb_score
    )) +
      geom_point() +
      geom_smooth(method = lm) +
      scale_x_continuous(breaks = breaks_pretty()) +
      labs(y = "IMDb Score",
           title = paste("IMDb score ~ ", predictor)) +
      theme_pubr() +
      theme_lox()
  )

  # Check for linearity
  residualPlots(lin_reg_predictor)
}
```
~~~
We see that
- All predictors except the `actor_star_meters` are statistically significa
- `movie_budget`, `duration`, and `movie_meter_IMDBpro` have heteroskedasti
- `movie_budget` and `nb_faces` are linear, while the other numeric columns


## Checking for multicollinearity
```{r fig.width=12, fig.height=8}
data_cleaned %>%
  select(numerical_predictors) %>%
  cor() %>%
  round(2) %>%
  ggcorr(digits = 2, label = TRUE, label_alpha = 0.5)

ggsave(filename = "plots/corr_plot.png", device = "png")

vif_model <- lm(imdb_score ~ ., data = (data_cleaned %>% select(where(is.nu

vif(vif_model)
~~~

We see that there is no multicollinearity among the numeric variables in th

## Removing variables

```{r}
columns_to_remove <-
  c(
    "movie_id",
    "movie_title",
    "imdb_link",
    "actor1",
    "actor2",
    "actor3",
    "genres",
```

```
    "release_day",
    "release_year",
    "colour_film",
    "actor1_star_meter",
    "actor2_star_meter",
    "actor3_star_meter",
    "director",
    "cinematographer",
    "production_company",
    "language",
    "country",
    "distributor",
    "plot_keywords"
  )

data_cleaned <- data_cleaned %>% select(-all_of(columns_to_remove))
```

```{r}
data_cleaned %>% head()
```

# Model building
## Individual linear regression
```{r}
individual_regressions <- colnames(data_cleaned) %>%
  setdiff("imdb_score") %>%
  syms() %>%
  map(~expr(lm(imdb_score ~ !!.x, data = data_cleaned))) %>%
  map(eval.parent) %>%
  map(tidy) %>%
  bind_rows() %>%
  filter(term != "(Intercept)")
```

```{r}
individual_regressions %>%
  filter(!p.value < 0.05)
```

We see that individually, most of the non-significant predictors (at 5% lev

```{r fig.width = 15, fig.height = 8}
individual_regressions %>%
  filter(!p.value > 0.05) %>%
  arrange(p.value) %>%
  ggplot(aes(x = fct_reorder(term, -p.value), y = estimate)) +
  geom_col(aes(fill = estimate < 0)) +
  scale_fill_fivethirtyeight() +
  scale_y_continuous(breaks = pretty_breaks(n = 10)) +
  guides(fill = "none") +
  labs(
    x = "Variable",
    y = "Estimate",
    title = "Estimated coefficents for IMDB Score",
    subtitle = "Individual regressions for each variable; Only significant
  ) +
  theme_pubr() +
  theme_lox() +
  theme(panel.grid.major.y = element_blank()) +
  coord_flip()

ggsave(filename = "plots/individual_regressions.png", device = "png")
```

## Multiple linear regression
```{r fig.width=12, fig.height=8}
lin_reg_1 <- lm(imdb_score ~ ., data=data_cleaned)
summary(lin_reg_1)
residualPlot(lin_reg_1)
```
### Checking and removing outliers
```{r}
outlierTest(lin_reg_1)

outliers <- names(outlierTest(lin_reg_1)[[1]])
data_cleaned <- data_cleaned %>% filter(!row_number() %in% outliers)
```

### Running the model with outliers removed
```{r fig.width=12, fig.height=8}
lin_reg_2 <- lm(imdb_score ~ ., data=data_cleaned)
summary(lin_reg_2)
residualPlot(lin_reg_2)
```

After removing the outliers, we see that the non-linearity has reduced. Let

### Heteroskedasticity
```{r}
ncvTest(lin_reg_2)
```

Since the model contains heteroskedasticity, we can check the corrected co
```{r fig.width=12, fig.height=8}
coeftest(lin_reg_2, vcov = vcovHC(lin_reg_2, type = "HC1")) %>%
  tidy() %>%
  filter(term != "(Intercept)") %>%
  filter(!p.value > 0.05) %>%
  arrange(p.value) %>%
  ggplot(aes(x = fct_reorder(term, -p.value), y = estimate)) +
  geom_col(aes(fill = estimate < 0)) +
  scale_fill_fivethirtyeight() +
  scale_y_continuous(breaks = pretty_breaks(n = 10)) +
  guides(fill = "none") +
  labs(
    x = "Variable",
    y = "Estimate",
    title = "Heteroskedasticity corrected estimated coefficents for IMDB Sc
    subtitle = "Only significant predictors shown (5% level)"
  ) +
  theme_pubr() +
  theme_lox() +
  theme(panel.grid.major.y = element_blank()) +
  coord_flip()

ggsave(filename = "plots/multiple_linear_regression.png", device = "png")
```

### Baseline performance
```{r}
predicted_scores <- predict(lin_reg_2, data_cleaned)
actual_scores <- data_cleaned$imdb_score
mse <- mean((actual_scores - predicted_scores)^2)
rmse <- sqrt(mse)
print(glue("Linear regression MSE: {mse}"))
print(glue("Linear regression RMSE: {rmse}"))
```

## Linear regression - K-fold

To evaluate out-of-sample performance, we use K-fold cross-validation
```{r}
set.seed(420)

lin_reg_k <- glm(imdb_score ~ ., data = data_cleaned)
mse_lin_reg_k <- cv.glm(data_cleaned, lin_reg_k, K = 20)$delta[1]
rmse_lin_reg_k <- sqrt(mse_lin_reg_k)

print(glue("Linear regression MSE: {mse_lin_reg_k}"))
print(glue("Linear regression RMSE: {rmse_lin_reg_k}"))
```

## Polynomial regression

~~~
```{r}
set.seed(420)
poly_k_fold_mse <- function(d1, d2, d3, k) {
  poly_k_fold <-
    glm(
      imdb_score ~ movie_budget + poly(duration, d1, raw = d1 == 1) + poly(
      data = data_cleaned
    )

  mse_k_fold <- cv.glm(data_cleaned, poly_k_fold, K = k)$delta[1]
  print(glue("MSE for degrees ({d1}, {d2}, {d3}) on {k}-fold CV: {mse_k_fol
  mse_k_fold
}
~~~

### Finding the optimal degree for numeric features using K-fold CV
```{r warnings=FALSE}
mse_poly_k_fold_df = data.frame(
  i = NA,
  j = NA,
  k = NA,
  mse = NA
)
for (i in 1:5) {
  for (j in 1:5) {
    for (k in 1:5) {
      mse_poly_k_fold_df[nrow(mse_poly_k_fold_df) + 1,] <-
        c(i, j, k, poly_k_fold_mse(i, j, k, 5))
    }
  }
}
```
```{r}
mse_poly_k_fold_df[which.min(mse_poly_k_fold_df$mse), ]
```

### Polynomial regression model with optimal degrees
```{r}
set.seed(420)
poly_k_fold <- glm(imdb_score ~ movie_budget + poly(duration, 2) + poly(nb_

mse_poly_k_fold <- cv.glm(data_cleaned, poly_k_fold, K = 20)$delta[1]
rmse_poly_k_fold <- sqrt(mse_poly_k_fold)

print(glue("Polynomial MSE: {mse_poly_k_fold}"))
print(glue("Polynomial RMSE: {rmse_poly_k_fold}"))
```
```
```

~~~
## Spline regression
### Building the model
```{r}
spline_model_1 <- glm(
  imdb_score ~
    movie_budget +
    bs(
      duration,
      knots = quantile(data_cleaned$duration, c(0.25, 0.5, 0.75)),
      degree = 2
    ) +
    bs(
      nb_news_articles,
      knots = quantile(data_cleaned$nb_news_articles, c(0.25, 0.5, 0.75)),
      degree = 1
    ) +
    bs(
      movie_meter_IMDBpro,
      knots = quantile(data_cleaned$movie_meter_IMDBpro, c(0.25, 0.5, 0.75)
      degree = 1
    ) + . -movie_budget - duration - nb_news_articles - movie_meter_IMDBpro
  data = data_cleaned
)
~~~

### Running K-fold cross-validation
```{r warning=FALSE}
set.seed(420)
mse_spline_k_fold <- cv.glm(data_cleaned, spline_model_1, K = 20)$delta[1]
rmse_spline_k_fold <- sqrt(mse_spline_k_fold)
r_squared_spline_k_fold <- 1 - (spline_model_1$deviance / spline_model_1$nu

print(glue("Spline MSE: {mse_spline_k_fold}"))
print(glue("Spline RMSE: {rmse_spline_k_fold}"))
print(glue("R-squared: {r_squared_spline_k_fold}"))
```
## Reporting overall performance
```{r warning=FALSE, results='asis'}
performance <-
  data.frame(
    linear=c(
      mse_lin_reg_k,
      rmse_lin_reg_k,
      1 - lin_reg_k$deviance/lin_reg_k$null.deviance,
      length(lin_reg_k$coefficients) - 1
    ),
```

```
      poly=c(
        mse_poly_k_fold,
        rmse_poly_k_fold,
        1 - poly_k_fold$deviance/poly_k_fold$null.deviance,
        length(poly_k_fold$coefficients) - 1
      ),
      spline=c(
        mse_spline_k_fold,
        rmse_spline_k_fold,
        1 - spline_model_1$deviance/spline_model_1$null.deviance,
        length(spline_model_1$coefficients) - 1
      )
    )

names(performance) <- c("Linear", "Polynomial", "Spline")
rownames(performance) <- c("MSE", "RMSE", "R^2", "Number of predictors")

stargazer(performance, header = FALSE, digits = 2, type = "latex", out = "t
```

# Using the models
```{r}
test_data <- read.csv("./test_data_IMDB_Fall_2023.csv")
test_data
```
## Cleaning test data
```{r}
test_data_cleaned <- test_data %>%
  mutate(across(
    .cols = c(
      'language',
      'country',
      'maturity_rating',
      'aspect_ratio',
      'distributor',
      'director',
      'colour_film',
      'cinematographer',
      'production_company'
    ),
    ~ factor(.x)
  ))

test_data_cleaned <- test_data_cleaned %>%
  mutate(release_month = month(fast_strptime(release_month, format = "%b"),
                                            TRUE)) %>%
  mutate(release_month = factor(release_month, levels = month.abb, ordered
```

~~~
```{r}
test_data_cleaned <- test_data_cleaned %>%
  mutate(movie_budget = as.numeric(str_replace_all(movie_budget, ",", "")))
~~~

```{r}
test_data_cleaned <- test_data_cleaned %>%
  mutate(across(
    .cols = c(
      "movie_budget",
      'duration',
      'nb_news_articles',
      'actor1_star_meter',
      'actor2_star_meter',
      'actor3_star_meter',
      'nb_faces',
      'movie_meter_IMDBpro'
    ),
    .fns = ~ scale(.) %>% as.vector()
  ))
```

```{r}
for (keyword in top_10_keywords$plot_keywords) {
  col_name <- glue("plot_{keyword}")
  test_data_cleaned[[col_name]] <-
    as.integer(lapply(test_data_cleaned$plot_keywords, function (x) {
      str_detect(x, keyword)
    }))
}

for (distributor in top_5_distributors$distributor) {
  col_name <- glue("distributor_{distributor}")
  test_data_cleaned[[col_name]] <-
    as.integer(lapply(test_data_cleaned$distributor, function (x) {
      str_detect(x, distributor)
    }))
}
```

```{r}
columns_to_remove <-
  c(
    "movie_id",
    "movie_title",
    "imdb_link",
```

```
      "actor1",
      "actor2",
      "actor3",
      "genres",
      "release_year",
      "colour_film",
      'actor1_star_meter',
      'actor2_star_meter',
      'actor3_star_meter',
      "director",
      "cinematographer",
      "production_company",
      "language",
      "country",
      "distributor",
      "plot_keywords"
   )

test_data_cleaned <- test_data_cleaned %>% select(-all_of(columns_to_remove
```

```{r}
test_data_cleaned
```
## Predicting ratings of new movies
```{r warning=FALSE}
lin_reg_predict <- as.data.frame(predict(lin_reg_k, test_data_cleaned))
poly_reg_predict <- as.data.frame(predict(poly_k_fold, test_data_cleaned))
spline_reg_predict <- as.data.frame(predict(spline_model_1, test_data_clean

predictions <-
   data.frame(
      test_data$movie_title,
      dmy(
         str_c(
            test_data$release_day,
            test_data$release_month,
            test_data$release_year,
            sep = "-"
         )
      ),
      lin_reg_predict,
      poly_reg_predict,
      spline_reg_predict
   )


names(predictions) <- c("Movie-Title", "Release-Date" ,"Linear", "Polynomia
predictions
```
```{r}
stargazer(predictions, header = FALSE, digits = 2, type = "latex", out = "t
```

## Running ANOVA
```{r}
anova(lin_reg_k, poly_k_fold, spline_model_1, test = "F")
```
From the ANOVA, we see that the **spline** regression **model** has the lowest **resi**

# Presenting the final model
```{r warning=FALSE}
names(poly_k_fold$coefficients) <- c(
  "Intercept",
  "Movie Budget",
  "Duration",
  "Duration^2",
  "News Articles",
  "News Articles^2",
  "News Articles^3",
  "News Articles^4",
  "IMDB Pro Meter",
  "IMDB Pro Meter^2",
  "IMDB Pro Meter^3",
  "IMDB Pro Meter^4",
  "IMDB Pro Meter^5",
  "Released in February",
  "Released in March",
  "Released in April",
  "Released in May",
  "Released in June",
  "Released in July",
  "Released in August",
  "Released in September",
  "Released in October",
  "Released in November",
  "Released in December",
  "G Rating",
  "PG Rating",
  "PG-13 Rating",
  "R Rating",
  "Other Maturity Rating",
  "Aspect Ratio 1.85",
  "Aspect Ratio 2.35",
  "Other Aspect Ratio",
  "Number of Faces in Poster",
  "Action",
  "Adventure",
  "Sci-Fi",
  "Thriller",
  "Musical",
  "Romance",
  "Western",
  "Sport",
  "Horror",
  "Drama",
  "War",
  "Animation",
  "Crime",
  "Plot = Murder",
  "Plot = Love",
  "Plot = Friends",
  "Plot = Death",
  "Plot = High School",
  "Plot = Police",
  "Plot = New York City",
  "Plot = a Boy",
  "Plot = Drugs",
  "Plot = School",
  "Distributor = Warner Bros.",
  "Distributor = Universal Pictures",
  "Distributor = Paramount Pictures",
  "Distributor = Twentieth Century Fox",
  "Distributor = Columbia Pictures Corporation"
)

coeftest(poly_k_fold, vcov = vcovHC(poly_k_fold, type = "HC1")) %>%
  stargazer(
    header = FALSE,
    digits = 2,
    type = "latex",
    out = "tables/polynomial_regression_summary.html",
    title = "Polynomial regression"
  )
```