Name- Misba Gajra
Student ID- 2117336

Module- Advanced Data Bases and Big Data
Topic – Assignment 2
Module Tutor- Mrs. Helen Martin

Table of Contents

Introduction

As the newly appointed Junior Data Scientist at Movie R Us, my role is to unlock potential insights buried within the rich Movie Ratings and Movie Extended Dataset. The purpose of this research is to provide light on the specifics of movie ratings by offering insightful analysis that goes beyond simple number evaluations. Furthermore, the research will go so far as to validate the budgetary allotments for these productions, providing Movies R Us with a more profound comprehension of the financial workings of the entertainment industry. The study will also examine more analysis, such as profit evaluations for various categories.

In the near future, Movies R Us hopes to obtain an updated dataset in the upcoming three months. This innovative strategy fits with the company's goal of increasing on-demand revenues by carefully selecting the most watched films, both from the past and the present, based on intelligent data analytics.

Portfolio Component 3: Solving Data Science Problem

Part 1- Exploring the Dataset

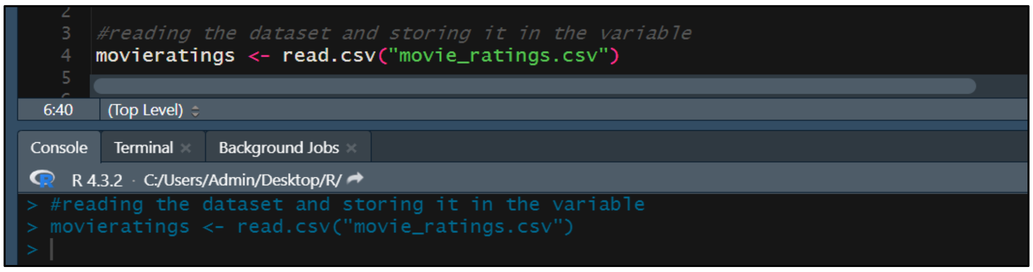
Explore your dataset (using str, nrow etc.) and explain your understanding.

1. **Reading the dataset and storing it in a variable**

   Input:

```
movieratings <- read.csv("movie_ratings.csv")
```

Output:



```
#reading the dataset and storing it in the variable
movieratings <- read.csv("movie_ratings.csv")
```

6:40 (Top Level)

Console Terminal Background Jobs

R 4.3.2 · C:/Users/Admin/Desktop/R/

```
> #reading the dataset and storing it in the variable
> movieratings <- read.csv("movie_ratings.csv")
> |
```

Purpose:

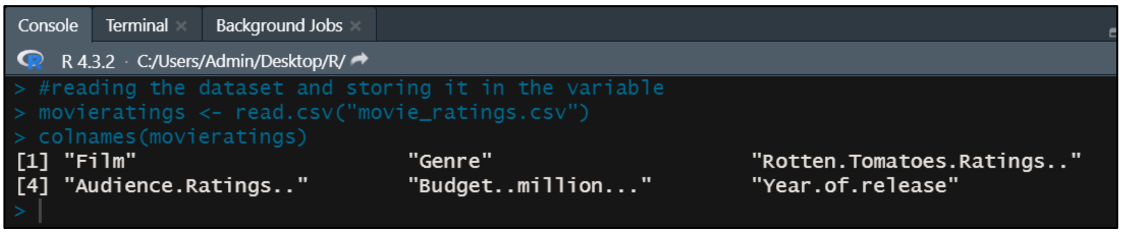
The R language's read.csv() function is used to read "comma separated value" files. It imports information as a data frame. The dataset is stored in the variable “movieratings”.

2. Viewing the column names

Input:

```
colnames(movieratings)
```

Output:



Console Terminal Background Jobs

R 4.3.2 · C:/Users/Admin/Desktop/R/

```
> #reading the dataset and storing it in the variable
> movieratings <- read.csv("movie_ratings.csv")
> colnames(movieratings)
[1] "Film"                  "Genre"                 "Rotten.Tomatoes.Ratings.."
[4] "Audience.Ratings.."    "Budget..million..."    "Year.of.release"
> |
```

Purpose:

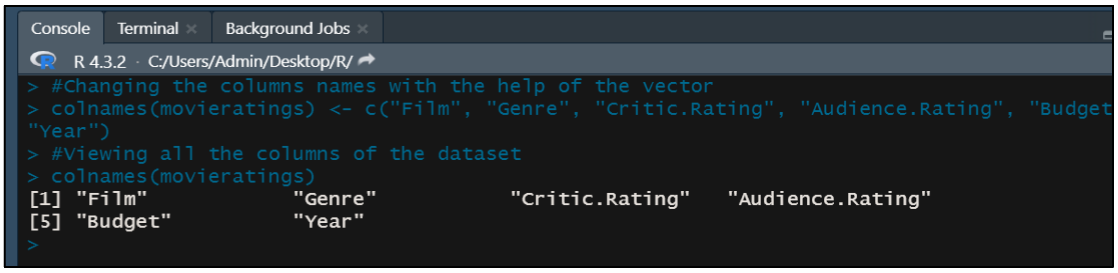
The colnames function is used to get or set an object that resembles a matrix by its row or column names. The colnames function in the following gets and shows all the columns of the movie ratings dataset.

3. Changing the column names

Input:

```
colnames(movieratings) <- c("Film", "Genre", "Critic.Rating", "Audience.Rating",

"Budget", "Year")
```

Output:



Console Terminal Background Jobs

R 4.3.2 · C:/Users/Admin/Desktop/R/

```
> #Changing the columns names with the help of the vector
> colnames(movieratings) <- c("Film", "Genre", "Critic.Rating", "Audience.Rating", "Budget
"Year")
> #Viewing all the columns of the dataset
> colnames(movieratings)
[1] "Film"            "Genre"           "Critic.Rating"   "Audience.Rating"
[5] "Budget"          "Year"
>
```

Purpose:

- By giving it a new vector of names, the function is used to modify a matrix's or data frame's column names.
- colnames(movieratings): This section gets the movieratings data frame's current column names.
- <-c("Film", "Genre", "Audience.Rating", "Critic.Rating", "Budget", "Year"): This section gives the movieratings column names a new vector of names. The new column names are represented by a vector composed of character strings that are created using the c() method.
- “Film” is assigned to the first column
- “Audience.Rating” is assigned to the second column.

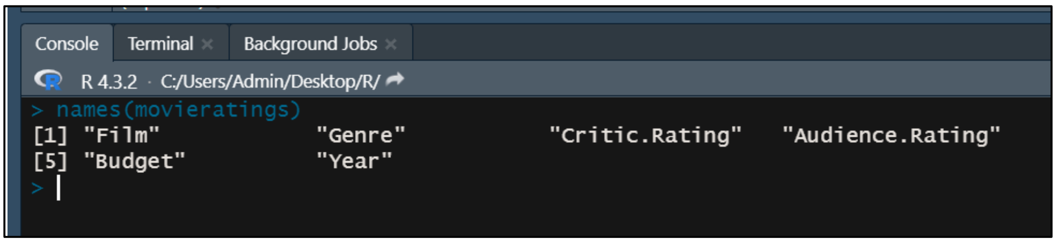
- "Critic.Rating" is assigned to the third column, which replaces the name of "Rotten Tomatoes Ratings%".
- "Budget" is assigned to the fourth column, which replaces the name of "Budget(million$)".
- "Year" is assigned to the fifth column, which replaces "year of release".
- With the new names, the dataset is more standardised and intuitive due to the consistent naming practice (e.g., using dots to separate words).
- The new titles continue to convey the key details about each column but are concise. When working with huge datasets or presenting data in reports and visualisations, this might be helpful.

4. Names Function

Input:

```
names(movieratings)
```

Output:



```
> names(movieratings)
[1] "Film"             "Genre"            "Critic.Rating"    "Audience.Rating"
[5] "Budget"           "Year"
> |
```

Purpose:

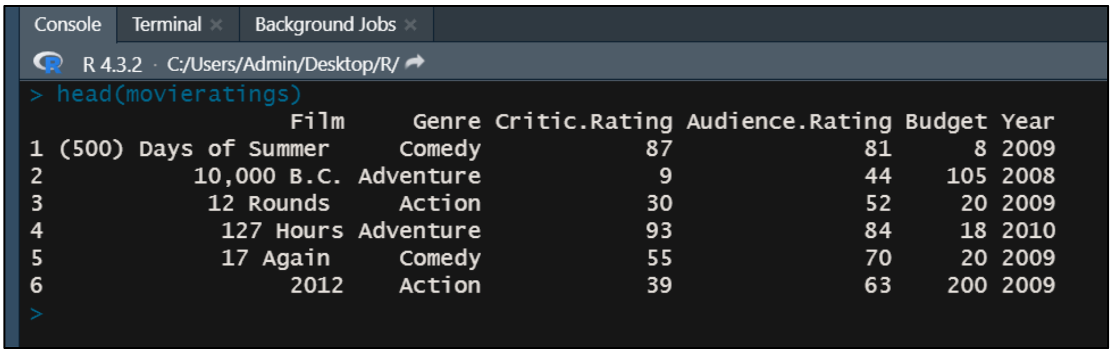
The names function is used to retrieve the elements of the vector or the columns of the dataset.

5. Head Function

Input:

```
head(movieratings)
```

Output:



```
> head(movieratings)
                  Film     Genre Critic.Rating Audience.Rating Budget Year
1 (500) Days of Summer    Comedy            87              81      8 2009
2          10,000 B.C. Adventure             9              44    105 2008
3            12 Rounds    Action            30              52     20 2009
4            127 Hours Adventure            93              84     18 2010
5             17 Again    Comedy            55              70     20 2009
6                 2012    Action            39              63    200 2009
>
```

Purpose:

To see the first six rows of a data frame or data set , the head() function is used. It provides a brief summary of the data by displaying the top rows of a data frame when applied.

6. Displaying the first 10 rows

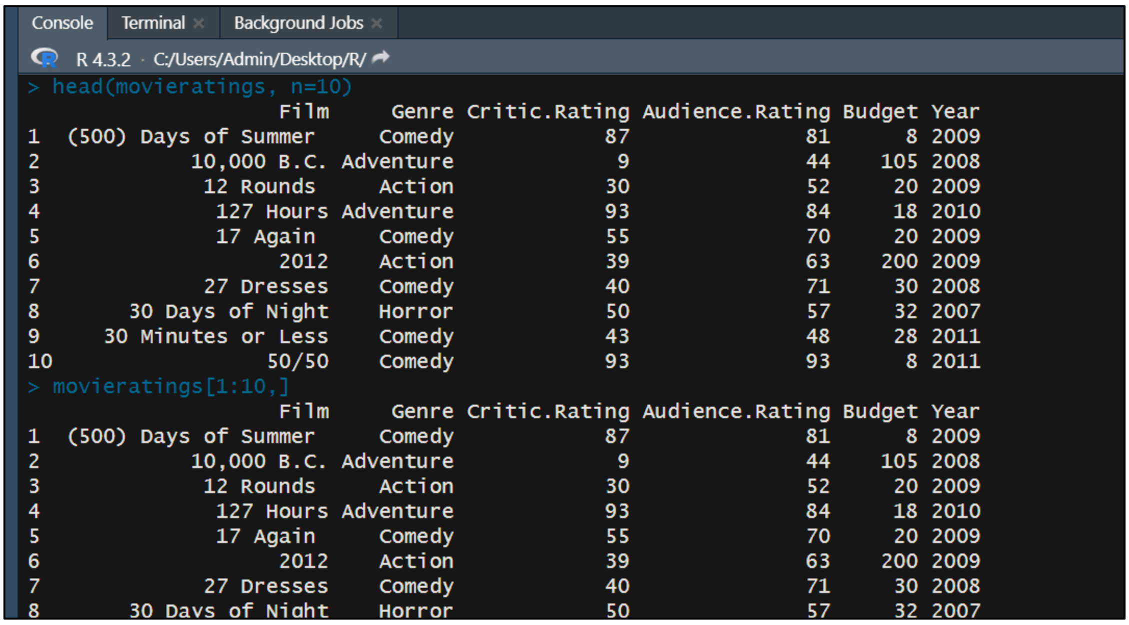
Input a:

```
head(movieratings, n=10)
```

Input b:

```
movieratings[1:10,]
```

Output:



```
> head(movieratings, n=10)
                  Film     Genre Critic.Rating Audience.Rating Budget Year
1  (500) Days of Summer    Comedy            87              81      8 2009
2          10,000 B.C. Adventure             9              44    105 2008
3            12 Rounds    Action            30              52     20 2009
4            127 Hours Adventure            93              84     18 2010
5             17 Again    Comedy            55              70     20 2009
6                 2012    Action            39              63    200 2009
7           27 Dresses    Comedy            40              71     30 2008
8     30 Days of Night    Horror            50              57     32 2007
9    30 Minutes or Less    Comedy            43              48     28 2011
10               50/50    Comedy            93              93      8 2011
> movieratings[1:10,]
                  Film     Genre Critic.Rating Audience.Rating Budget Year
1  (500) Days of Summer    Comedy            87              81      8 2009
2          10,000 B.C. Adventure             9              44    105 2008
3            12 Rounds    Action            30              52     20 2009
4            127 Hours Adventure            93              84     18 2010
5             17 Again    Comedy            55              70     20 2009
6                 2012    Action            39              63    200 2009
7           27 Dresses    Comedy            40              71     30 2008
8     30 Days of Night    Horror            50              57     32 2007
```

Purpose:

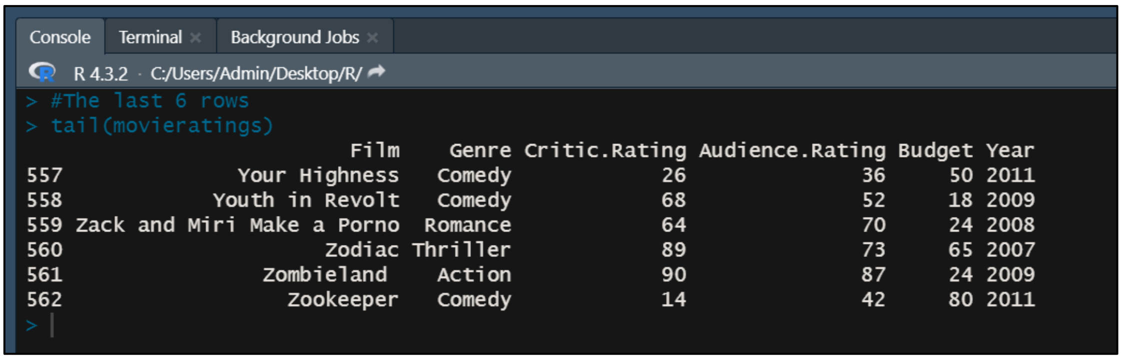
- The top 10 rows of the movieratings data frame are shown in R using both head and indexing function.
- In the head function, the expression n = 10 indicates the result should be the top ten rows.
- Whereas the rows of the movieratings data frame from 1 to 10 (inclusive) are chosen using indexing in movieratings[1:10,]. Thus, a subset of the data frame is extracted that only includes the rows from 1 to 10.

7. The tail function

Input:

```
tail(movieratings)
```

Output:



Console Terminal Background Jobs

R 4.3.2 · C:/Users/Admin/Desktop/R/

```
> #The last 6 rows
> tail(movieratings)
                        Film    Genre Critic.Rating Audience.Rating Budget Year
557             Your Highness   Comedy            26              36     50 2011
558           Youth in Revolt   Comedy            68              52     18 2009
559 Zack and Miri Make a Porno  Romance            64              70     24 2008
560                    Zodiac Thriller            89              73     65 2007
561                Zombieland   Action            90              87     24 2009
562                 Zookeeper   Comedy            14              42     80 2011
> 
```

Purpose:

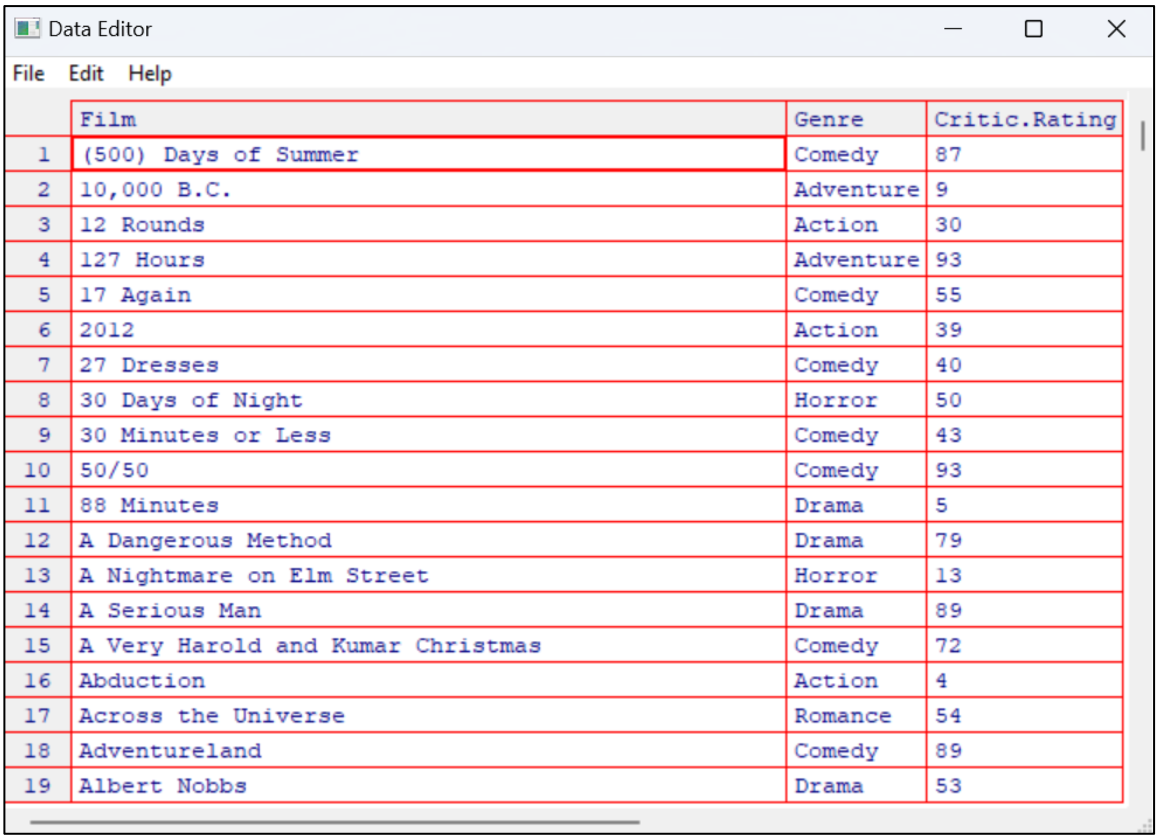
The last few rows of a data frame or matrix can be shown using R's tail() function. When used on a data frame, it displays the six rows, giving a brief summary of the data's conclusion.

8. The edit function

Input:

```
edit(movieratings)
```

Output:



	Film	Genre	Critic.Rating
1	(500) Days of Summer	Comedy	87
2	10,000 B.C.	Adventure	9
3	12 Rounds	Action	30
4	127 Hours	Adventure	93
5	17 Again	Comedy	55
6	2012	Action	39
7	27 Dresses	Comedy	40
8	30 Days of Night	Horror	50
9	30 Minutes or Less	Comedy	43
10	50/50	Comedy	93
11	88 Minutes	Drama	5
12	A Dangerous Method	Drama	79
13	A Nightmare on Elm Street	Horror	13
14	A Serious Man	Drama	89
15	A Very Harold and Kumar Christmas	Comedy	72
16	Abduction	Action	4
17	Across the Universe	Romance	54
18	Adventureland	Comedy	89
19	Albert Nobbs	Drama	53

Purpose:

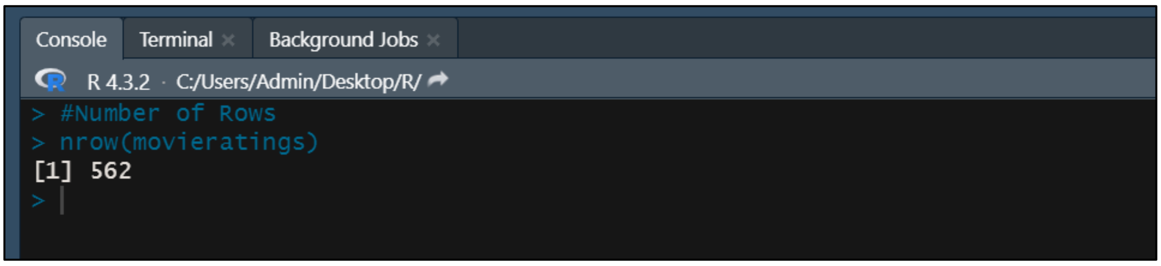
- The edit() function opens a data editor that allows to edit a data frame interactively.
- When used on a data frame, it launches an interface similar to a spreadsheet, allowing to examine and edit the data frame's contents.

9. The nrow function

Input:

```
nrow(movieratings)
```

Output:



```
> #Number of Rows
> nrow(movieratings)
[1] 562
>
```

Purpose:

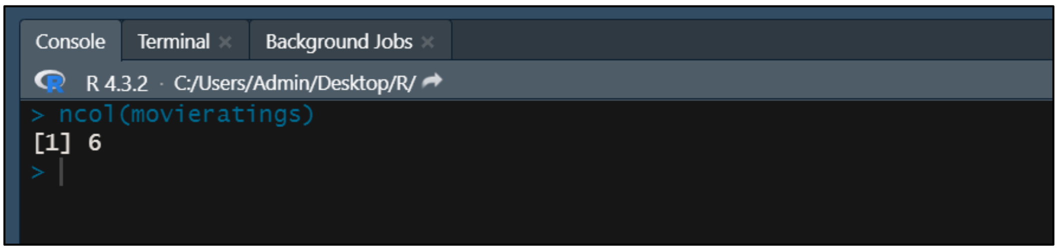
- To find out how many rows there are in a data frame, matrix, or other structure of a similar kind, the nrow() function is used.
- The function that yields the number of rows in the given data frame.
- The total number of rows in the movieratings data frame is provided in a single numerical figure as shown in the output above.

10. The ncol function

Input:

```
ncol(movieratings)
```

Output:



```
> ncol(movieratings)
[1] 6
>
```

Purpose:

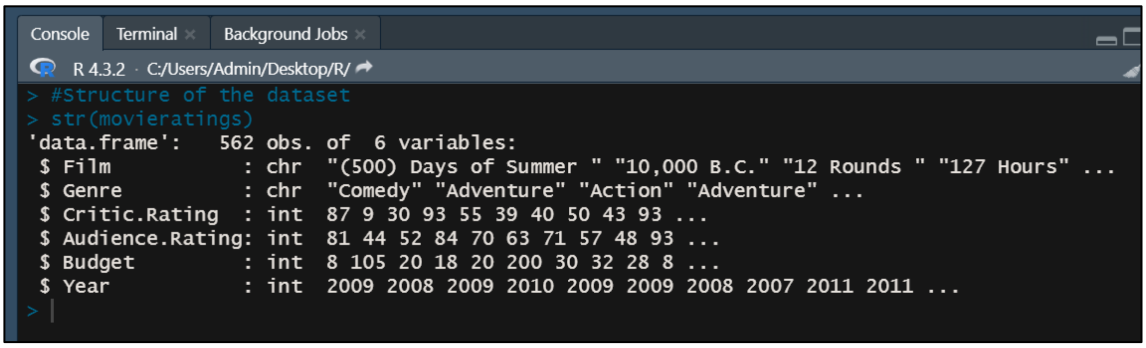
- To find out how many columns there are in a data frame, matrix, or other structure of a similar kind, the ncol() function is used.
- The function that yields the number of columns in the given data frame.
- The total number of columns in the movieratings data frame is provided in a single numerical figure as shown in the output above.

11. The str function

Input:

```
str(movieratings)
```

Output:



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

Purpose:

- The underlying structure of a R object may be concisely and systematically represented using the str() function in R.
- It shows details about the data types and the initial values of each variable (column).

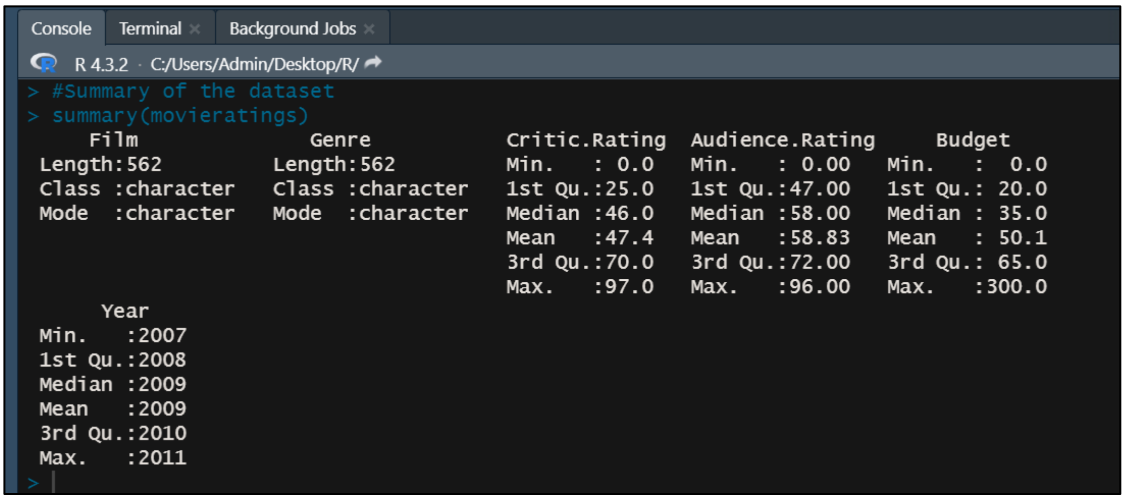
- Usually, this data consists of:
    1. The total number of rows, or observations.
    2. The total number of columns, or count.
    3. The variables' names.
    4. The data type of each variable.
- An overview of the first few values.
- As seen in the output, movieratings has six variables (columns) and 1,000 observations (rows).
- The data-type is an "int" for Critic.Rating, Audience.Rating, Budget and Year.
- The data-type in a "character" for Film and Genre.

12. The summary function

Input:

```
summary(movieratings)
```

Output:



```
> #Summary of the dataset
> summary(movieratings)
     Film                Genre            Critic.Rating  Audience.Rating     Budget     
 Length:562         Length:562         Min.   : 0.0   Min.   : 0.00   Min.   :  0.0  
 Class :character   Class :character   1st Qu.:25.0   1st Qu.:47.00   1st Qu.: 20.0  
 Mode  :character   Mode  :character   Median :46.0   Median :58.00   Median : 35.0  
                                       Mean   :47.4   Mean   :58.83   Mean   : 50.1  
                                       3rd Qu.:70.0   3rd Qu.:72.00   3rd Qu.: 65.0  
                                       Max.   :97.0   Max.   :96.00   Max.   :300.0  
      Year     
 Min.   :2007  
 1st Qu.:2008  
 Median :2009  
 Mean   :2009  
 3rd Qu.:2010  
 Max.   :2011  
> 
```

Purpose:

- To produce summary statistics and details about the variables (columns) in a data frame, the summary function is used.
- It gives a brief summary of the distribution, central tendency, and other important characteristics of the numerical variables.
- As seen in the output above, R will provide a summary for each numeric variable in the data. Usually, the summary contains the following details:
  1. Minimum: Each variable's lowest possible value.
  2. 1st Qu: The first 25 percentile or quartile.
  3. Median: The 50th percentile, also known as the median (or second quantile).
  4. Mean: The average or mean.
  5. 3rd Qu: The 75th percentile, or third quartile.
  6. Maximum: The highest number found in every variable.
- For numerical factors such as Audience.Rating and Critic.Rating , Year, and Budget, the summary statistics is shown.
- Whereas for character variables such as Film and Genre, the count of unique values is included in the result.

How Genre impacts the budget of the movie?

There are several benefits and reasons for using a boxplot to illustrate how genre affects the movie budgets:

- **Visualising Distribution**- As it is important to comprehend the distribution and central tendency of budgets within each genre when investigating how the genre affects the budget, boxplots offer an easily understood and transparent visual representation of the data distribution, incorporating essential summary statistics like the median, quartiles, and any anomalies (Chang, 2012).
- **Comparing Multiple Genres**- Boxplots make it possible to compare several genres at once on a single plot. Hence, seeing different genres side by side makes it simple to see how budget distributions vary and what parallels and differences exist between them (Belov & Nikulchev, 2019).
- **Managing anomalies**- According to (Healy, 2019), Boxplots provide information about extreme budget values by graphically highlighting potential differences. Henceforth, a comprehensive knowledge of the impact of genre may be impacted by notable outliers in movie expenses. Boxplots facilitate the identification and evaluation of outliers' impact on each genre.
- **Detecting Skewness**- Boxplots offer indicators of data distribution deviation visually. Uneven budget distributions across genres may point to trends or preferences within a certain genre. If budgets are dispersed or centered on a single figure, a boxplot can show this (Hadley, 2010).
- **Interpreting central tendency** - Boxplots provide information on the central trend of budgets by displaying the median and quartiles. Knowing the central tendency makes it easier to compare different genres and determines the normal budget for each (Chang, 2012).

Input for the R code:

```
b <- ggplot(data=movieratings, aes(x=Genre, y=Budget, colour= Genre))

g <- b + geom_boxplot(size=1.2) + geom_jitter(alpha=0.5, width = 0.2) +

  xlab("Genre") +

  ylab("Budget (in millions$)") +

  ggtitle("Distribution of Movie Budget by Genre") +

  theme_minimal() +

  theme(

    axis.title.x = element_text(colour="darkblue", size=12, face= "bold"),

    axis.title.y = element_text(colour="darkblue", size=12, face= "bold"),

    axis.text.x = element_text(colour="darkgreen", size=10),

    axis.text.y = element_text(colour="darkgreen", size=10),

    plot.title = element_text(colour= "Black", size=15, hjust=0.5, face= "bold"),

    panel.background = element_rect(fill = "white", colour = "white"),

    legend.position = c(1,1),

    legend.justification = c(1,1),

    legend.box.background = element_rect(color = "black"),

    axis.line = element_line(color = "black"),

    axis.ticks = element_line(color = "black"))
```

Explanation of the code:

1. **Creating the Data**

- Ggplot(): A ggplot object is initialised using the ggplot() method which is stored in object b (datacamp.com, 2023).
- movies = data: Indicates the dataset that will be utilised in the plot. The dataset in this instance is called "movies."

2. **Adding Aesthetics**

- aes(x = Genre, y = Budget, colour = Genre): The aesthetics function, specifies how the variables in the dataset should be translated to the plot's visual components.
- x = Genre: The x-axis is the mapping for the 'Genre' variable.
- y = Budget: The y-axis is the mapping for the 'Budget' variable.
- colour = genre: The plot's points are coloured using the 'Genre' variable. It will be simple to distinguish between different genres because each one will have a distinct colour (datacamp.com, 2023).

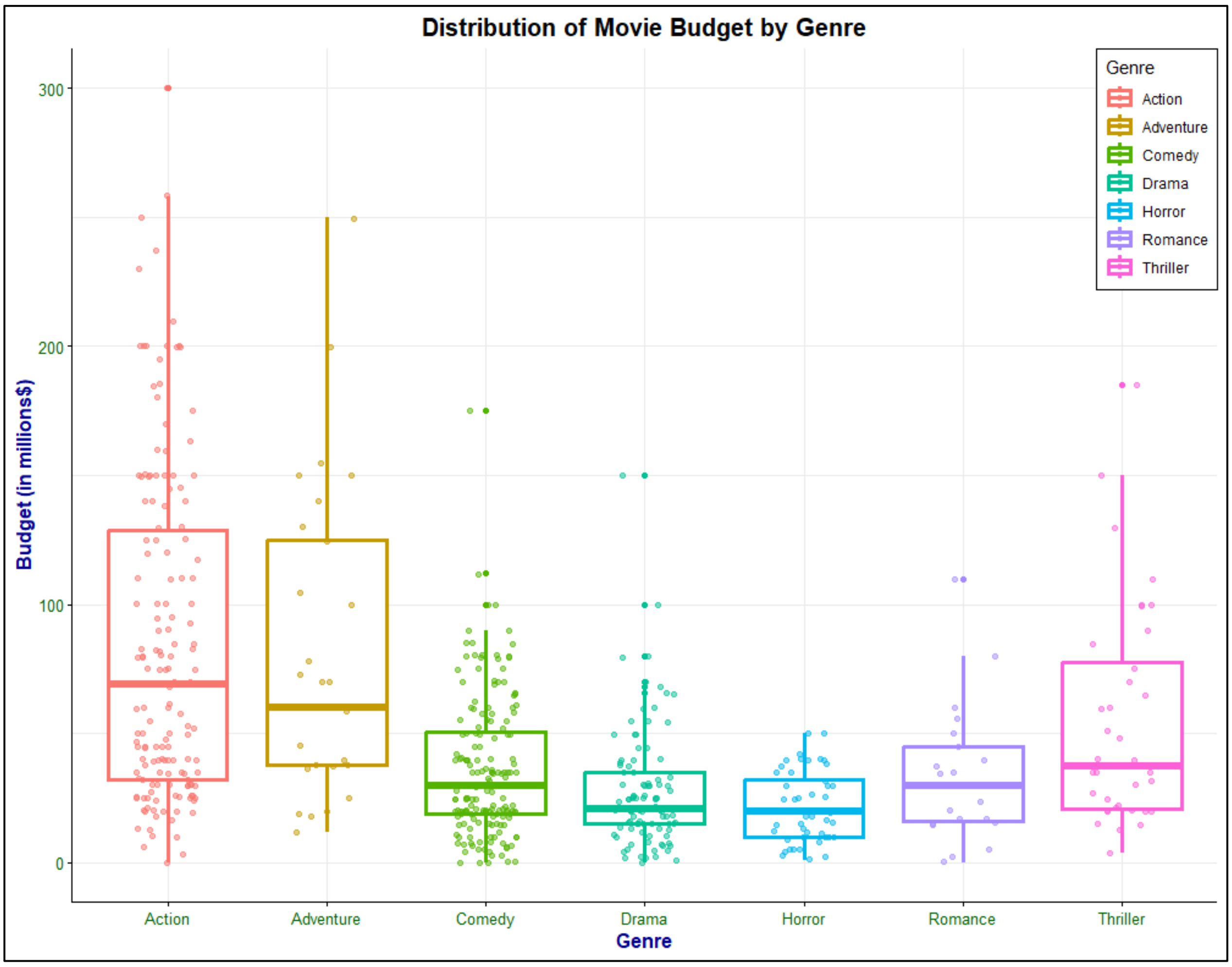
3. **Adding the Geometry**

- geom_boxplot(size = 1.2) : This function adds a boxplot layer to the plot with a given size for the boxplot aspects.
- geom_jitter(alpha = 0.5, width = 0.2): This function adds a jitter plot layer to the plot. Jittering improves in the visualisation of the point distribution; the variables "alpha" and "width" regulate the transparency and width of the jittered points, respectively (w3schools.com, 2023).

4. Adding the formatting effects

- xlab("Genre") + ylab("Budget (in millions$)"): Sets the labels for the x and y axis.
- (ggtitle("Distribution of Movie Budget by Genre") : Sets the primary plot title.
- theme_minimal() : Gives the story a minimalist theme for a clear, uncomplicated look.
- Theme(): The ggplot2 theme() function allows to change the way different plot components look. The following are the components within the theme() function:
- Axis.title.x and Axis.title.y : This function sets the colour of x and y axis to “dark blue”, sets the font size to 12 and sets the font weight to bold.
- Axis.text.x and Axis.text.y : This function sets the colour of x and y axis text to “dark green” and font size to 10.
- Plot.title : This functions sets the colour of the plot title to black, font size to 15, the horizontal justification to center and the font weight to bold.
- Panel.background : This function sets the background colour to white.
- Legend.position: This function sets the position of the legend and justification to the top right corner.
- Legend.box.background : This function sets the border colour of the legend box to black.
- Axis.line and Axis Ticks: This function sets the color of the axis line and ticks to black.

Output: Distribution of Movie Genre by Genre



Analysis for Distribution of Movie Genre by Genre:

- The graph demonstrates how genre affects the movie budgets.
- It's a boxplot that shows how film expenses are allocated to various genres.
- The median budget for each genre are displayed in each box where individual movies are represented by the dots.

- The action, adventure, comedy, drama, horror, romance, and thriller genres are all represented in the graph. It is evident from the graph that some genres, such as Action and Adventure, have a tendency to have bigger median costs as well as a broader range of budgets, indicating that these kinds of movies typically have larger budgets.
- However, the median budgets of films in genres like romance and horror are often lower and the range is more compressed meaning that these films usually have fewer budgets.
- The following are the analysis of how each genre impacts the budget:

  1. **Action Genre**
     - The middle of the box represents the median budget for action movies, indicating an approximate even dispersion of data around the median or suggesting a mix of budgets for the action genre.
     - There is a noticeable variance in the budgets for Action films, as shown by the size of the budget range inside the interquartile range (IQR), which is shown by the height of the box. Thus, action movies have a wide range of budget.
     - Dots above the upper part of the box represent a number of anomalies, which demonstrate that certain Action films have budgets that are noticeably larger than the others.
     - It can also be noticed that the highest budget movie was from action genre.

2. **The adventure genre**
   - Action movies and adventure movies both have a large range and a high median budget.
   - But there are many adventure movies over the upper part of box indicating a higher budget.

3. **The comedy genre**
   - Compared to Action and Adventure, Comedy has a lower median budget.
   - Less money is allocated to comedies than to action and adventure movies.
   - A few comedies have larger budgets than others.

4. **The drama genre**
   - Like comedy, drama movies has a low median budget.
   - It can be noticed that the budget for drama film are generally consistent and does not vary much.
   - A few drama films had larger budgets than others.

5. **The horror genre**
   - The genre with the lowest median budget is horror.
   - The costs for horror films are often small and relatively comparable, only a few are exceptions.

6. **The romance genre**
    - The romance movies also has a low median budget.
    - The finances for romance films are often smaller and don't vary greatly.
    - A few Romance films had larger budgets than others.

7. **The thriller genre**
    - The median budget of thriller films is somewhat greater than that of drama and romance films.
    - Budgets for thriller films tend to be a bit lower than those of action/adventure movies.
    - Certain thriller films have substantial budgets.

According to (Healy, 2019), the boxplot illustrates anomalies as well as the central trend, dispersion, and variance of film budgets across various genres. It demonstrates categorically that a film's genre may affect its budget, the action and adventure genres often have larger budgets than the horror and romance genres. But as seen in the graph, every genre has an exception. The analysis provides an insight on how different genres have an impact on the final movie production and its budget.

Is there any relation between the critic rating and the budget?

Input for R code 1:

```
c <- ggplot(data=movieratings, aes(x= Budget, y=Critic.Rating,

                              colour= Genre,

                              size= Budget))
```

Explanation for R code 1:

- **The data**: In the code, “data = movieratings”, specifies the data frame to retrieve the variable from and is stored in the object c.
- **The aesthetic mapping**: “aes(x=Budget, y=Critic.Rating, color=Genre, size=Budget)” defines the aesthetic mapping of the graph.
- **x=Budget**: This plots the x-axis as the movie budget.
- **y=Audience.Rating**: The y-axis is mapped to the audience rating.
- **color=Genre**: Aligns the colour of the points with the movie genre.
- **size=Budget**: Aligns the size of the points with the film's budget.

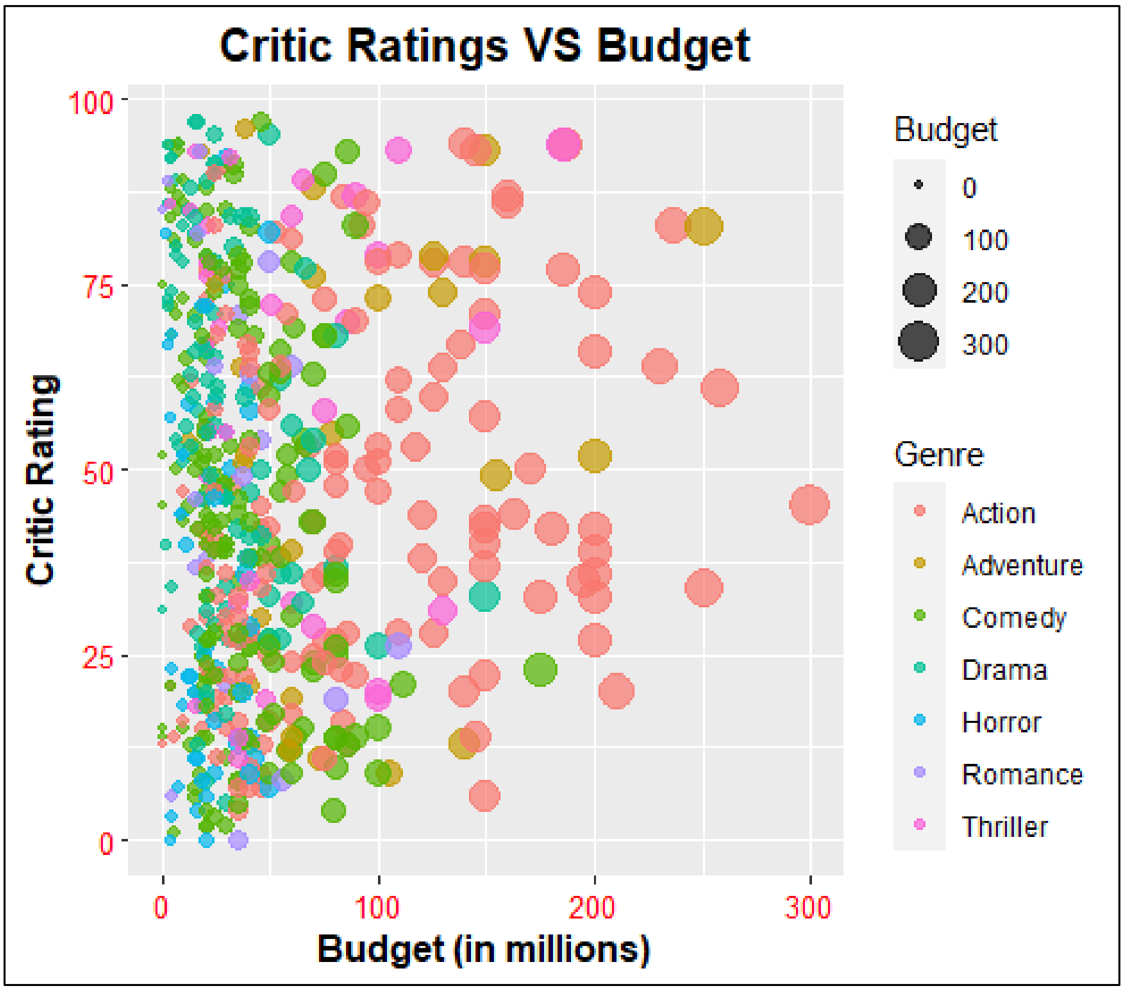
The code continued:

```
c +

 geom_point(alpha=0.7) +

 xlab("Budget (in millions)") +

 ylab("Critic Rating") +

 ggtitle("Critic Ratings VS Budget") +

 theme(

  axis.title.x = element_text(colour="Black", size=12, face= "bold"),

  axis.title.y = element_text(colour="Black", size=12, face= "bold"),

  axis.text.x = element_text(colour="Red", size=10),

  axis.text.y = element_text(colour="Red", size=10),

  plot.title = element_text(colour= "Black", size=15, hjust=0.5, face= "bold"))
```

- **geom_point()**: Produces a scatter plot by adding a layer of points to the plot.
- **Alpha=0.7**: The points' transparency is set to 0.7 using the alpha value, which results in a minor transparency.
- **Xlab("Budget (in millions)")** : The x-axis label is set to "Budget (in millions)".
- **Ylab("Critic Rating")**: The y-axis label is set to "Audience Rating".
- **ggtitle("Critic Ratings VS Budget")**: Sets "Critic Ratings VS Budget" as the plot's primary title.
- **theme()**: The theme() function is used to alter the plot's look.

- **axis.title.x and axis.title.y**: Set the x and y-axis titles to bold and black, respectively, as well as size and font weight.
- **axis.text.x and axis.text.y**: Change the x- and y-axis tick labels' colour and size to red and smaller, respectively.
- **plot.title**: Adjust the primary plot title's colour, size, and horizontal justification. This time, it's centered horizontally (hjust = 0.5), set to black, and bigger in size.

Output 3 : Relationship between Critic Rating and Budget



Analysis of the Graph:

- **X-axis (Budget)**: The horizontal axis shows the budget of each film, serving as a representation of the movie budget.
- **Y-axis: (Critic Rating)**: The critic ratings show how well each movie was accepted by the critics. This is a vertical axis.
- **Colour (Genre)**: The distribution of movies among genres can be examined by looking at the colour of each point, which is determined by the movie's genre.
- **Size (Budget)**: Each point's size corresponds to the movie's budget. Higher budgeted movies are represented with larger points.

Relationship between Critic Rating and Budget:

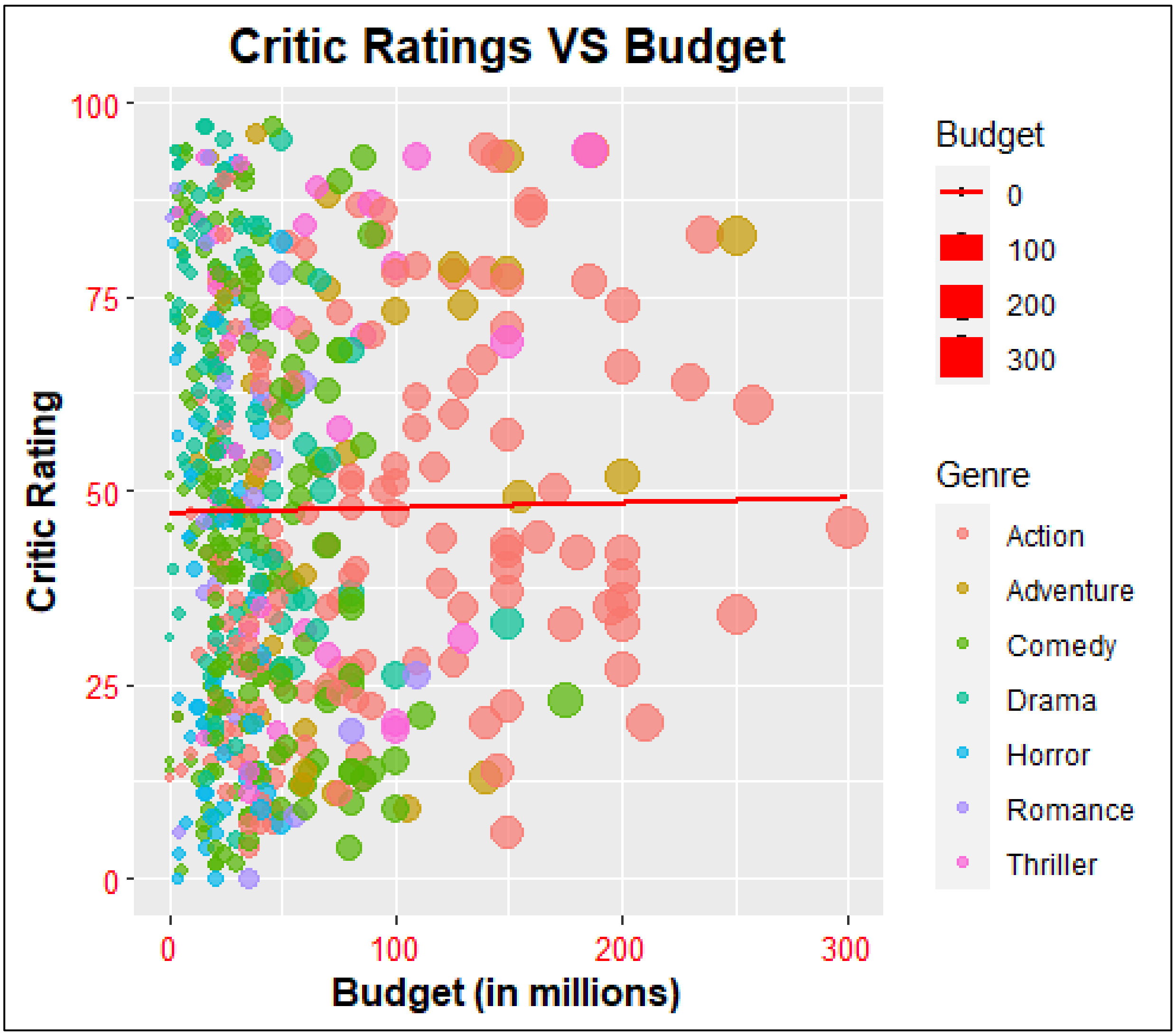
- The graph displays a scatter plot that depicts the correlation between the critic ratings and movie budget (measured in millions).
- The data points are color-coded by genre and the size of the dots indicates the budget amount.
- From the graph, it can be noticed that the points are scattered all over the graph.
- While there are high-budget movies that have gotten great reviews from critics, there are also low-budget movies that received mixed reviews, and vice versa for low-budget films.
- The below-mentioned graph is along with linear regressions that shows the trend.

Input for the R code 2:

```
c +

 geom_point(alpha=0.7) +

 geom_smooth(method=lm, se=FALSE, colour="Red") +

 xlab("Budget (in millions)") +

 ylab("Critic Rating") +

 ggtitle("Critic Ratings VS Budget") +

 theme(

  axis.title.x = element_text(colour="Black", size=12, face= "bold"),

  axis.title.y = element_text(colour="Black", size=12, face= "bold"),

  axis.text.x = element_text(colour="Red", size=10),

  axis.text.y = element_text(colour="Red", size=10),

  plot.title = element_text(colour= "Black", size=15, hjust=0.5, face= "bold"))
```

- In the function, “ geom_smooth(method=lm, se=FALSE, color="Red")”, the employment of a linear model, or linear regression, is indicated by the method=lm specification.
- When se=FALSE is used, the shaded zone surrounding the regression line is not displayed.

Output 3 : Relationship between Critic Rating and Budget with Linear regression



Analysis of Critic Rating VS Budget with Linear Regression:

- The red horizontal trend line in the updated graph above indicates that there isn't a clear upward or downward trend in the critic ratings when the budget rises.
- This suggests that elements other than the movie budget have a greater influence on a movie's critic ratings and shows that the rotten tomato ratings are generally independent of the budget.

- The wide range of geom points across all budget levels indicates that factors other than budget alone, such as acting performances, script quality, directorial talent, and individual critic preferences, are probably more important in determining the critics ratings for the movie.

Is there relationship between Audience Rating and Budget?

Input for the R code:

```
q <- ggplot(data=movieratings, aes(x= Budget, y=Audience.Rating,

                              colour= Genre,

                              size= Budget))
```

- **The data**: In the code, “data = movieratings”, specifies the data frame to retrieve the variable from and is stored in the object q.
- **The aesthetic mapping**: “aes(x=Budget, y=Audience.Rating, color=Genre, size=Budget)” defines the aesthetic mapping of the graph.
- **x=Budget**: This plots the x-axis as the movie budget.
- **y=Audience.Rating**: The y-axis is mapped to the audience rating.
- **color=Genre**: Aligns the colour of the points with the movie genre.
- **size=Budget**: Aligns the size of the points with the film's budget.

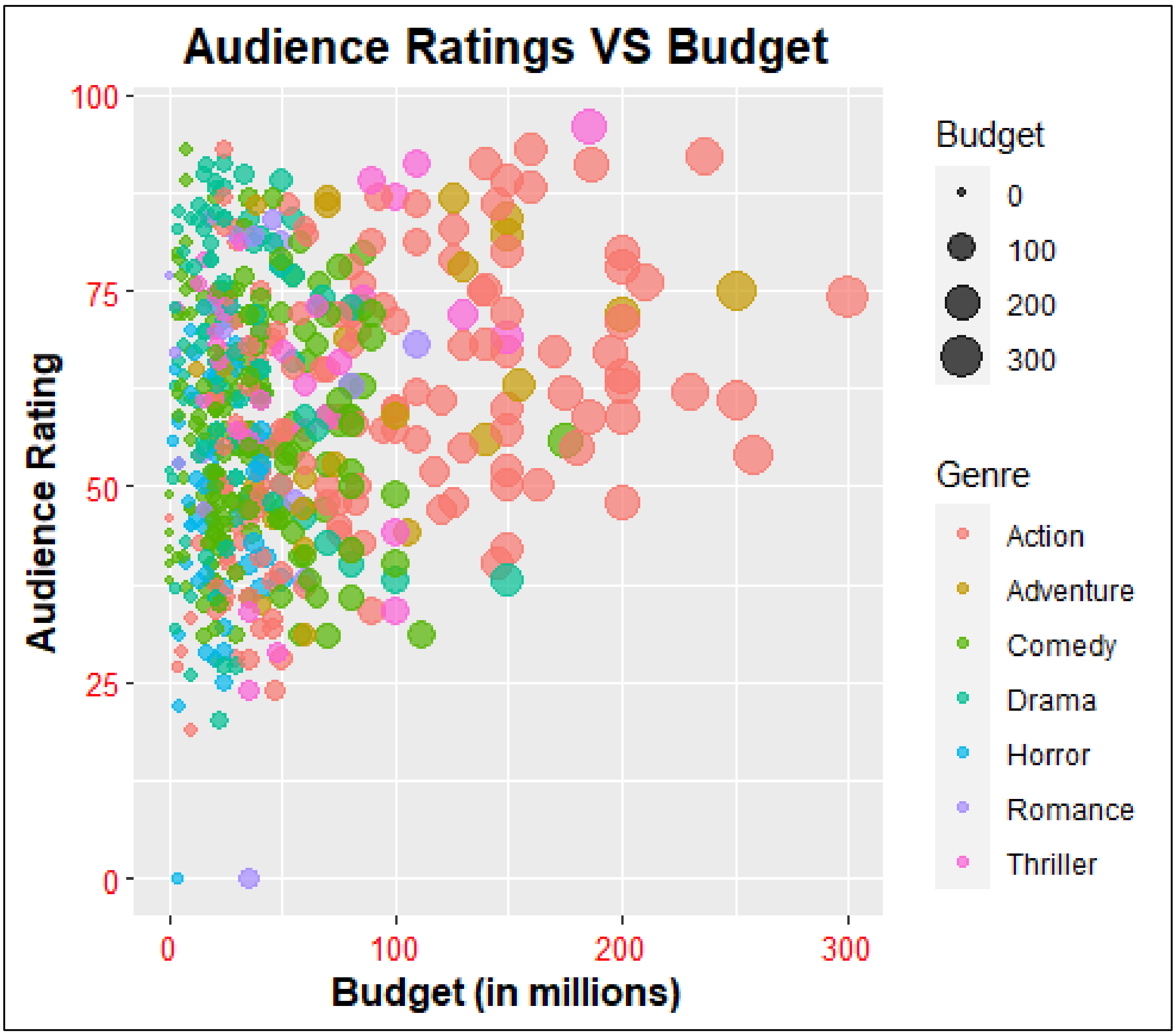
The code continued:

```
q +

 geom_point(alpha=0.7) +

 xlab("Budget (in millions)") +

 ylab("Audience Rating") +

 ggtitle("Audience Ratings VS Budget") +

 theme(

   axis.title.x = element_text(colour="Black", size=12, face= "bold"),

   axis.title.y = element_text(colour="Black", size=12, face= "bold"),

   axis.text.x = element_text(colour="Red", size=10),

   axis.text.y = element_text(colour="Red", size=10),

   plot.title = element_text(colour= "Black", size=15, hjust=0.5, face= "bold"))
```

- **geom_point()**: Produces a scatter plot by adding a layer of points to the plot.
- **Alpha=0.7**: The points' transparency is set to 0.7 using the alpha value, which results in a minor transparency.
- **Xlab("Budget (in millions)")** : The x-axis label is set to "Budget (in millions)".
- **Ylab("Audience Rating")**: The y-axis label is set to "Audience Rating".
- **ggtitle("Audience Ratings VS Budget")**: Sets "Audience Ratings VS Budget" as the plot's primary title.

- **theme()**: The theme() function is used to alter the plot's look.
- **axis.title.x and axis.title.y**: Set the x and y-axis titles to bold and black, respectively, as well as size and font weight.
- **axis.text.x and axis.text.y**: Change the x- and y-axis tick labels' colour and size to red and smaller, respectively.
- **plot.title**: Adjust the primary plot title's colour, size, and horizontal justification. This time, it's centered horizontally (hjust = 0.5), set to black, and bigger in size.

Output : Relationship between Audience Rating and Budget



Analysis of the graph:

- **X-axis (Budget)**: The horizontal axis shows the budget of each film, serving as a representation of the movie budget.
- **Y-axis: (Audience Rating)**: The audience ratings show how well each movie was accepted by the audience. This is a vertical axis.
- **Colour (Genre)**: The distribution of movies among genres can be examined by looking at the colour of each point, which is determined by the movie's genre.
- **Size (Budget)**: Each point's size corresponds to the movie's budget. Higher budgeted movies are represented with larger points.

Relationship between Audience Rating and Budget:

- The aforementioned graph shows the link between a movie's budget and audience rating.
- It appears that there is a relationship between the size of the dots and the budget, greater dots indicate larger budgets.
- The different movie's genre such as action, adventure, comedy, drama, horror, romance, and thriller is indicated by the colours of the dots.
- The data points are easier to see in the graph above. There doesn't seem to be any continuous pattern of bigger budget films routinely receiving higher audience ratings; instead, the distribution of points is somewhat varied.
- There are low-budget films with a broad spectrum of audience ratings, and there are high-budget films with both high and low audience ratings.
- The following graph shows a positive trend by adding a linear regression or trend line in the current plot.

- The trendline that has been included makes it easier to see the general pattern or connections. It shows the best-fit linear regression line in this instance.

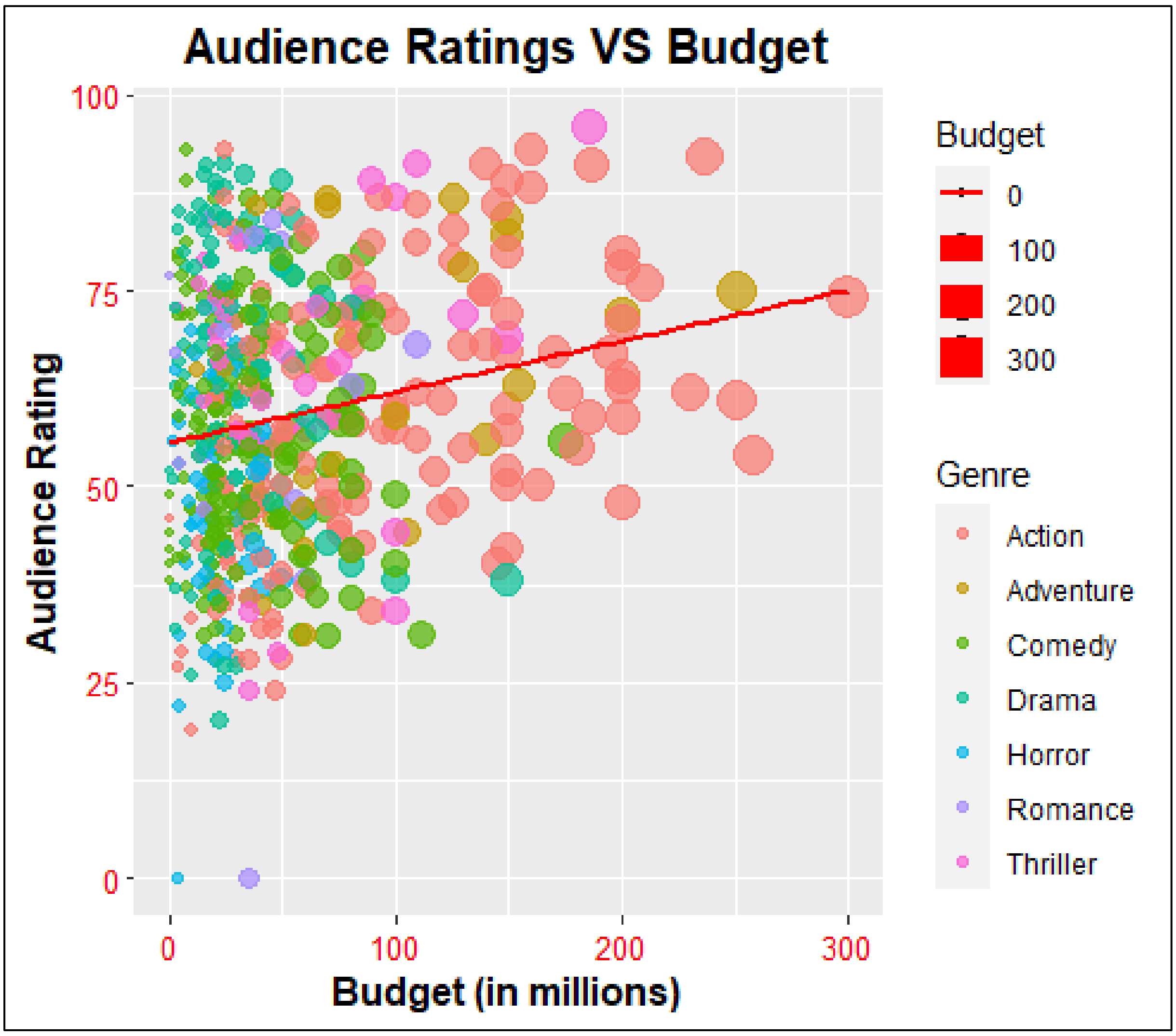
Audience Rating VS Budget with Linear Regression:

Input for the R code:

```
q +

 geom_point(alpha=0.7) +

 geom_smooth(method=lm, se=FALSE, colour=”Red”) +

 xlab(“Budget (in millions)”) +

 ylab(“Audience Rating”) +

 ggtitle(“Audience Ratings VS Budget”) +

 theme(

  axis.title.x = element_text(colour=”Black”, size=12, face= “bold”),

  axis.title.y = element_text(colour=”Black”, size=12, face= “bold”),

  axis.text.x = element_text(colour=”Red”, size=10),

  axis.text.y = element_text(colour=”Red”, size=10),

  plot.title = element_text(colour= “Black”, size=15, hjust=0.5, face= “bold”))
```

- In the function, “ geom_smooth(method=lm, se=FALSE, color="Red")”, the employment of a linear model, or linear regression, is indicated by the method=lm specification.
- When se=FALSE is used, the shaded zone surrounding the regression line is not displayed.

Output for Audience Rating VS Budget with Linear Regression:



Analysis of Audience Rating VS Budget with Linear Regression:

- Although there is a lot of fluctuation, the scatter plot shows a positive tendency, suggesting that when the budget rises, the audience rating may tend to rise.
- Given the large range of data points and several outliers, this shows that there may be a positive correlation between a movie's budget and audience reaction, but it is not a significant association as the data points are heavily scattered across the graph.
- In other words, bigger budgets may not always translates into higher audience ratings.
- This brief analysis takes into consideration that the budget of the movie does not necessarily influence the audience, there might be many other factors such as actors, script quality, directorial process, marketing that might impact the audience rating on the movie other than the budget (Chang, 2012).

Show the correlation between audience and critic ratings has evolved throughout the years by movie genre. (Request from the CEO)

Input for the R code:

```
p <- ggplot(data = movieratings, aes(x = Year, colour = Genre)) +

  geom_point(aes(y = Audience.Rating), size = 2, shape = 1, alpha = 0.7) +

  geom_point(aes(y = Critic.Rating), size = 2, shape = 5, alpha = 0.7) +

  geom_line(aes(y = Audience.Rating, linetype = "Audience"), size = 1) +

  geom_line(aes(y = Critic.Rating, linetype = "Critic"), size = 1) +

  facet_grid(Genre ~ ., scales = "free_y") +

  xlab("Year") +

  ylab("Audience Ratings VS Critic Ratings") +

  ggtitle("Co-relation between Audience and Critic Ratings throughout years by
Genre") +

  theme_minimal() +

  theme(

    axis.title.x = element_text(colour = "DarkGreen", size = 9, face = "bold"),

    axis.title.y = element_text(colour = "DarkGreen", size = 9, face = "bold"),

    axis.text.x = element_text(colour = "black", size = 7),
  axis.text.y = element_text(colour = "black", size = 7),

    plot.title = element_text(colour = "Black", size = 10, hjust = 0.5, face = "bold"),

    legend.position = "bottom",

    legend.box = "vertical",

    legend.box.background = element_rect(color = "black"),

    axis.line = element_line(color = "black"),

    axis.ticks = element_line(color = "black")

  ) +

  scale_linetype_manual(

    values = c("solid", "dotted"),

    labels = c("Audience", "Critic"),

    name = "Line Type",

    guide = "legend"

  )
```

Explanation of the R code:

1. **The data layer**

- ggplot(data = movieratings, aes(x = Year, colour = Genre)) :The movieratings data is used to initialise the plot, using Year on the x-axis and distinct colours for each Genre (w3schools.com, 2023).

2. **Geometry Layer**

- geom_point() : Adds points with varying transparencies and shapes for the audience and critic ratings.
- geom_line() : Adds lines with various line types for the audience and critic ratings (tidyverse.org, 2023).

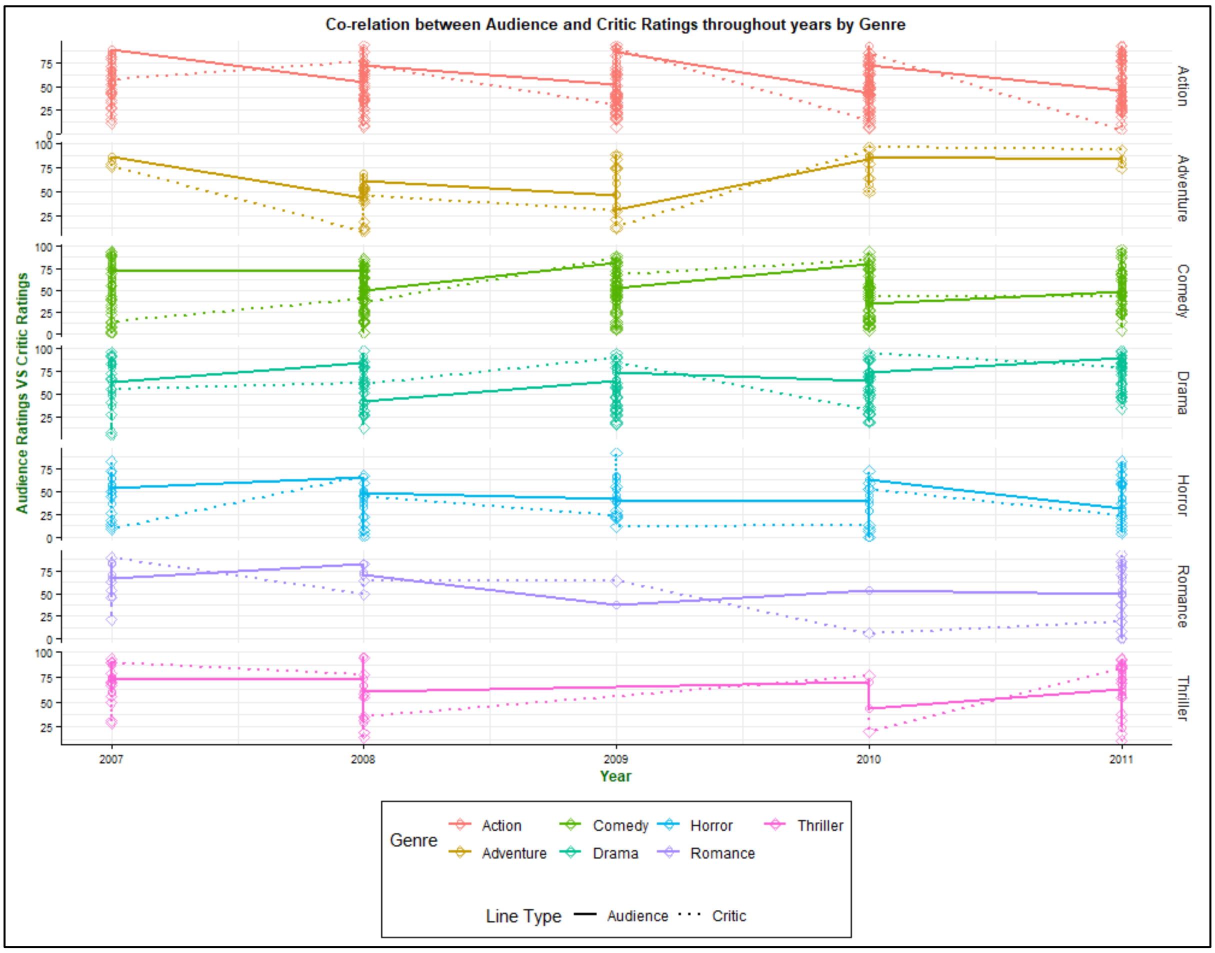
3. **Faceting**

- facet_grid(Genre~., scales = "free_y"): Partitions the plot vertically according to Genre, generating distinct boxes for every year. scales = "free_y" permits each facet's y-axis scale to be independent (tidyverse.org, 2023).

4. **Formatting**

- ggtitle: Sets the plot's primary title. xlab, ylab: Sets the labels for the x- and y-axes.
- values: Lists the various line types.
- labels: Legendary labelling.
- name: Title of a legend.
- guide: Indicates how the line types are supposed to appear in the tale.
- Theme(): Changes the theme's colours, sizes, and styles for the plot title, axis lines, ticks, legend, axis titles, and axis text (tidyverse.org, 2023).

Output of the Graph to show the correlation between audience and critic ratings has evolved throughout the years by movie genre.:



Analysis of the Graph:

The above graph is a line graph with the caption "Co-relation between Audience and Critic Ratings throughout years by Genre." It charts the scores given by critics and viewers to films in various categories from 2007 to 2011. Action, Adventure, Comedy, Drama, Horror, Romance, and Thriller are among the genres covered.

There are two lines, one for audience ratings (solid line) and the other for reviewer ratings (dotted line), each with a different colour for the respective genre.

The ratings, which seem to range from 0 to 100, are represented by the y-axis, while the years 2007 to 2011 are represented by the x-axis.

Important findings from the graph:

- **Action Genre**: Over time, both critic and audience reviews have been consistently strong, with audience reviews being marginally higher than critic reviews. Over time, there is a modest drop in both lines.
- **Adventure Genre**: Like Action, this genre has high scores from the audience and critics, with the public ranking somewhat higher than the critics.
- **Comedy Genre**: The ratings from the audience are consistently greater than those from critics. There is a little declining tendency in both ratings.
- **Drama Genre**: Unlike the other genres, the ratings given by critics for Drama are greater than those given by audiences. While viewership numbers indicate a tiny uptick, reviewer ratings remain mostly unchanged.
- **Horror Genre**: There's a big difference between the crowd and reviewer scores, with the audience scoring higher. Over time, ratings from critics and audiences both show a discernible drop.
- **Romance Genre**: The ratings given by critics and audience differ significantly, with audience ratings being substantially higher. As the years go by, the difference gets less, with an audience rating trending lower and a reviewer rating trending somewhat higher.
- **Thriller Genre**: The ratings for both audience and reviewers are the lowest among all genres. Both critic and audience ratings are gradually declining, with audience ratings consistently being higher.

With the exception of the drama genre, the graph appears to show that there is frequently differences in the ratings of films between the public and critics across all genres. The public typically gives films a higher rating than reviewers. Furthermore, it's clear that reviews from critics and audiences tend to differ from one other; over time and depending on the genre, they might diverge or converge.

To sum up, the graph displays the reviews from critics and viewers for various film genres during a five-year span. Nonetheless, patterns may be deduced, including the overall uniformity in evaluations for some genres and the divergence/convergence in others.

Create a graph to show the number of films from the dataset categorised by Genre.

- The genre variable is a categorical variable that represents several genres.
- Geom_histogram() is usually used for continuous data because it separates a continuous variable into bins, whereas Geom_bar() is intended for categorical data. Therefore, geom_bar() is used in the following graph to represent the data (geeksforgeeks.org, 2023).
- This task to count the occurrences of movies for each genre corresponds with the objective of geom_bar() and involves determining the frequencies of every genre.

- This function is useful for visualising the distribution of categorical data since it automatically computes the count for each category and generates a bar for each.

Input 1 for the R code:

```
s <- ggplot(movie_ratings, aes(x=Genre, fill= Genre))

t <- s +

  geom_bar(color = "Black", position = "dodge") +

  geom_text(

    stat = "count",

    aes(label = after_stat(count), y = after_stat(count)),

    vjust = -0.5,

    color = "Blue",

    size = 3.5

  ) +
labs(

    x = "Genre",

    y = "Number of Films",

    title = "Number of Movies categorized by Genre"

  ) +

  theme_light() +

  theme(

    axis.title = element_text(colour = "Black", size = 12, face = "bold"),

    axis.text = element_text(colour = "DarkGreen", size = 10),

    plot.title = element_text(colour = "Black", size = 15, hjust = 0.5, face = "bold"),

    legend.position = c(0.95, 0.95),

    legend.justification = c(1, 1),

    legend.box.background = element_rect(color = "Black"),

    axis.ticks = element_line(color = "DarkGreen")

  )
```

Explanation of the R code:

1. **Data and Aesthetic Mapping**
   - ggplot(movie_ratings, aes(x=Genre, fill= Genre)): This code initialises a ggplot object using the movie_ratings dataset, and it associates the x-axis with the 'Genre' variable and the aesthetic fill colour.

2. **Bar Plot Layer**
   - geom_bar(): Enables the plot to have a bar layer. The number of films in each category is shown by the bars.
   - colour = "Black": Changes the bars' border colour to black.
   - position = "dodge": This arranges the bars next to one another.

3. **Text Annotation Layer**
   - geom_text(): Enables the plot to have text labels.
   - stat = "count": Determines the number of films in each category.
   - Aes(y = after_stat(count), label = after_stat(count)): Maps the count to the label and y-axis positions.
   - vjust = -0.5: Modifies the text's vertical alignment above the bars.
   - colour = "Blue": This applies a blue hue to the text.
   - size = 3.5: Determines the text's size.

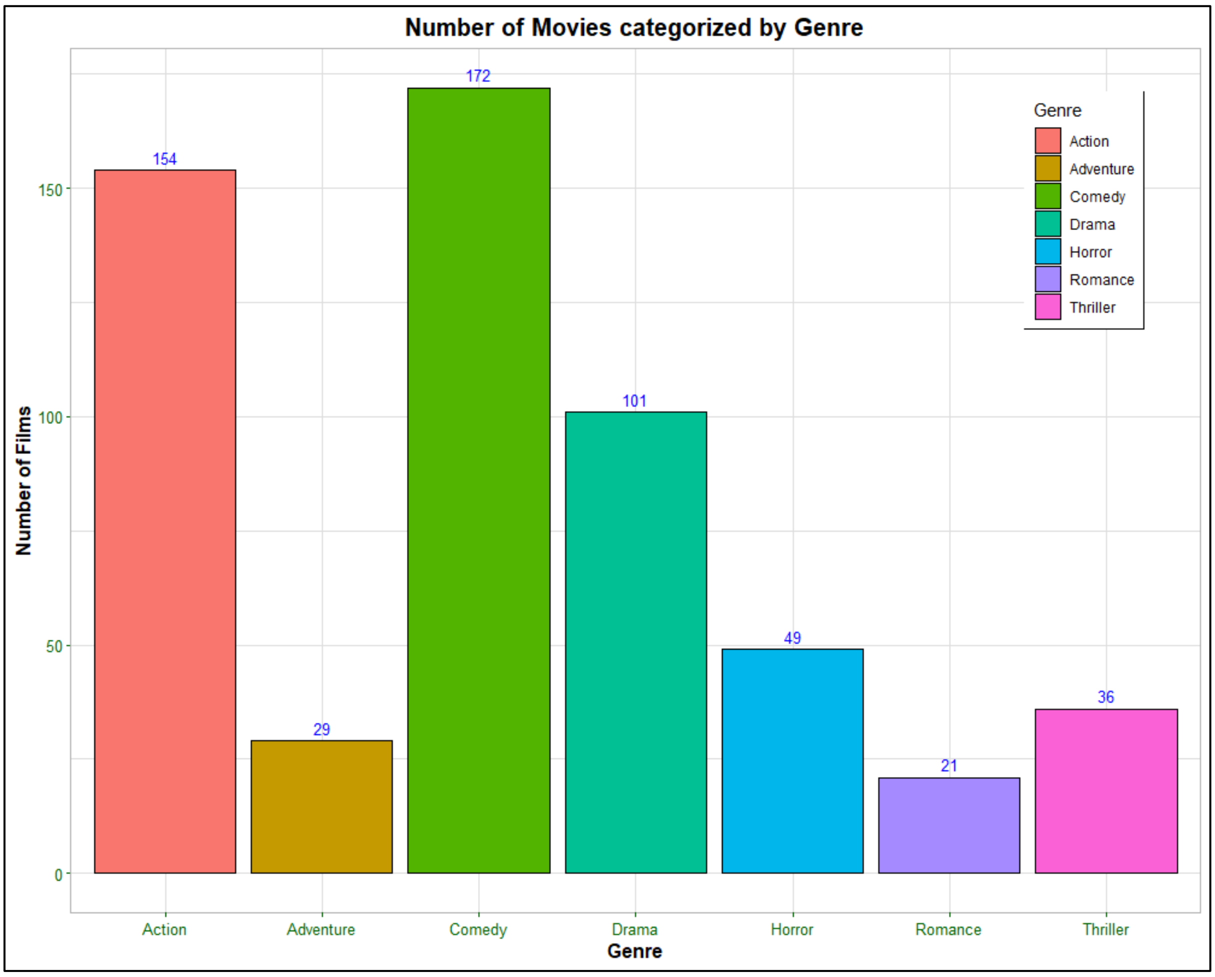
4. **Labels and Titles**

- labs(): Enhances the plot with labels and a title.
- The x-axis label is set with x = "Genre".
- The y-axis label is set with y = "Number of Films".
- Plot title is set with title = "Number of Movies categorised by Genre".

5. **Themes**

- **axis.title** : Set the x and y-axis titles to bold and black, respectively, as well as size 12 and face bold.
- **axis.text**: Change the x- and y-axis colour to dark green and size 10.
- **plot.title**: Adjust the primary plot title's colour, size, and horizontal justification and centered horizontally (hjust = 0.5), set to black, and bigger in size.
- Legend.position: This function sets the position of the legend and justification to the top right corner.
- Legend.box.background : This function sets the border colour of the legend box to black.
- Axis Ticks: This function sets the color of the axis ticks to dark green.

Output to show number of films categorised by Genre :



Analysis of the Graph :

- The displayed data is a vertical bar chart to show "Number of Movies categorised by Genre." The distribution of a specific number of films in each of the following seven categories is shown: action, adventure, comedy, drama, horror, romance, and thriller.
- A coloured bar matching to the legend on the right side of the graphic represents each genre. The genres are listed on the x-axis, while the "Number of Films" is shown on the y-axis.

1. **Action** : The category of action films comprises 154 films.
2. **Adventure** : The Adventure genre has 29 films.
3. **Comedy** : The Comedy genre comprises of 172 films that is highest among all genres.
4. **Drama** : The Drama genre has 101 films.
5. **Horror** : The Horror category has 49 films.
6. **Romance :** The Romance genre has the 21 movies which is lowest among all genres.
7. **Thriller** : The category of thrillers has 36 films.

The graph's main findings are as follows:

- Comedy is the most watched genre, with 172 films.
- With just 21 films, romance has the fewest amount of films.
- There are also a lot of films in the Action and Drama genres with 154 and 101 movies, respectively.
- Compared to the other genres, Adventure, Horror, and Thriller have the fewest number of films being 29, 49, and 36, respectively.
- The graph may be used to examine patterns in the creation of motion pictures, the level of popularity of particular genres, or to make judgements about marketing campaigns and distribution.

Part-2 Advanced Analytics

They give you the following graph image as the R code is not found and they would like to remove the spelling mistake in the Graph Title and rename the Graph Title "Gross Percentage By Genre". You need to recreate the graph by writing R code. You must use the Grammar of Graphics to recreate the following graph. You must also explain your code and display the output at each step.

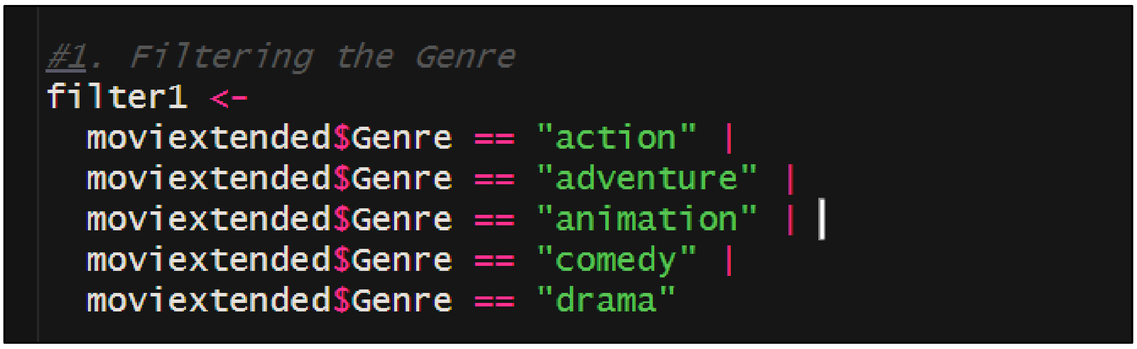
Input for the R code:

```
#1. Filtering the Genre

filter1 <- moviextended$Genre == "action" | moviextended$Genre == "adventure" |
moviextended$Genre == "animation" | moviextended$Genre == "comedy"
|moviextended$Genre == "drama"


#Allowing specific studios

filter2 <- moviextended$Studio %in% c("Buena Vista Studios", "WB", "Fox",
"Universal", "Paramount Pictures", "Sony")


#Combining filter1 and filter2

newmoviextended <- moviextended[filter1 & filter2,]
```

```
newmoviextended



#adding data-layer

p <- ggplot(data=newmoviextended, aes(x=Genre, y=Gross..US))

#adding geometries

g <- p + geom_jitter(aes(size=Budget, colour=Studio)) +

  geom_boxplot(alpha=0.7, outlier.color = NA)

#adding formatting

g <- g + xlab("Genre") +

  ylab("Gross % US") +

  ggtitle("Domestic Gross % by Genre") +

  theme(

    axis.title.x = element_text(colour="Blue", size=15),

    axis.title.y = element_text(colour="Blue", size=15),

    axis.text.x = element_text(colour="Black", size=10),

    axis.text.y = element_text(colour="Black", size=10),

    plot.title = element_text(colour="Black", size=20, hjust = 0.5, face = "bold"),

    legend.title=element_text(size=10),
legend.text = element_text(size=10),

    text = element_text(family="Comic Sans MS"))



g$labels$size <- "Budget $M"
```

Explanation of the code:

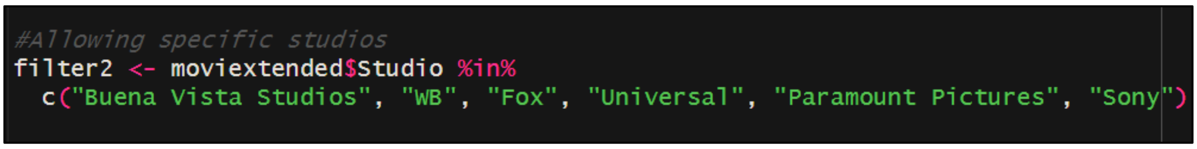
1. Filtering by Genre



```
#1. Filtering the Genre
filter1 <-
  moviextended$Genre == "action" |
  moviextended$Genre == "adventure" |
  moviextended$Genre == "animation" |
  moviextended$Genre == "comedy" |
  moviextended$Genre == "drama"
```

- One thing to be noticed in the graph is only specific genres are taken into account. Therefore, it is essential to filter genres first.
- This function establishes a logical vector (filter1) according to the genre, "action," "adventure," "animation," "comedy," or "drama."

2. Filtering by Studio

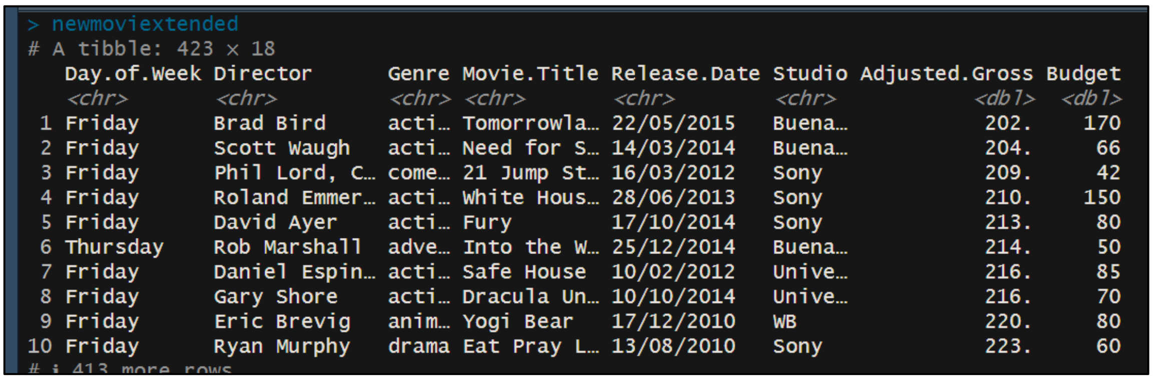


```
#Allowing specific studios
filter2 <- moviextended$Studio %in%
  c("Buena Vista Studios", "WB", "Fox", "Universal", "Paramount Pictures", "Sony")
```

- Similarly, only specific studios are included. So, studios are filtered using filter2 function (w3schools.com, 2023).

3. Combing the Filters

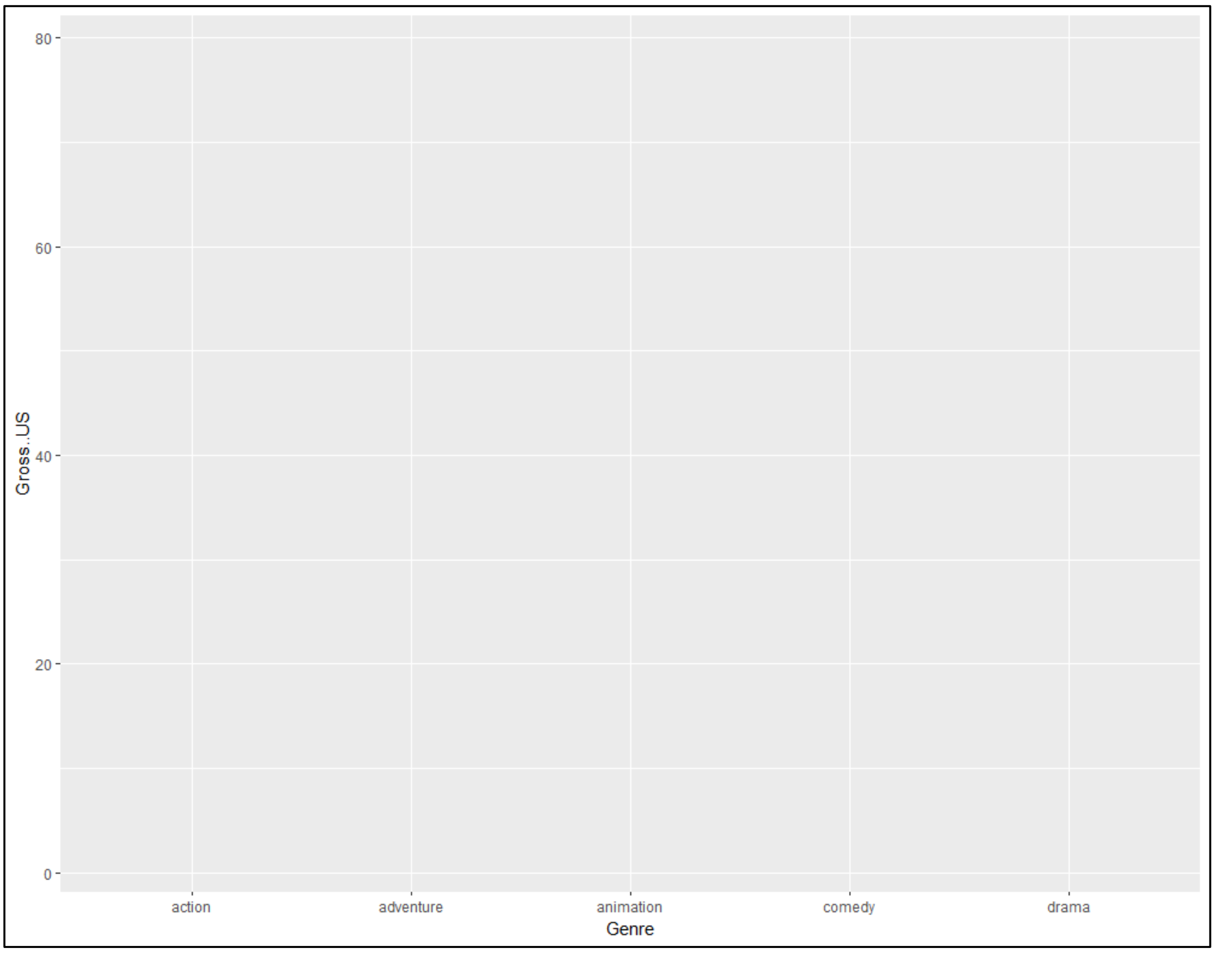
- newmoviextended <- moviextended[filter1 & filter2,] : This function uses the combined logical criteria from filter1 and filter2 to create a new data frame (newmoviextended) by subsetting the old data frame (moviextended) as shown below.



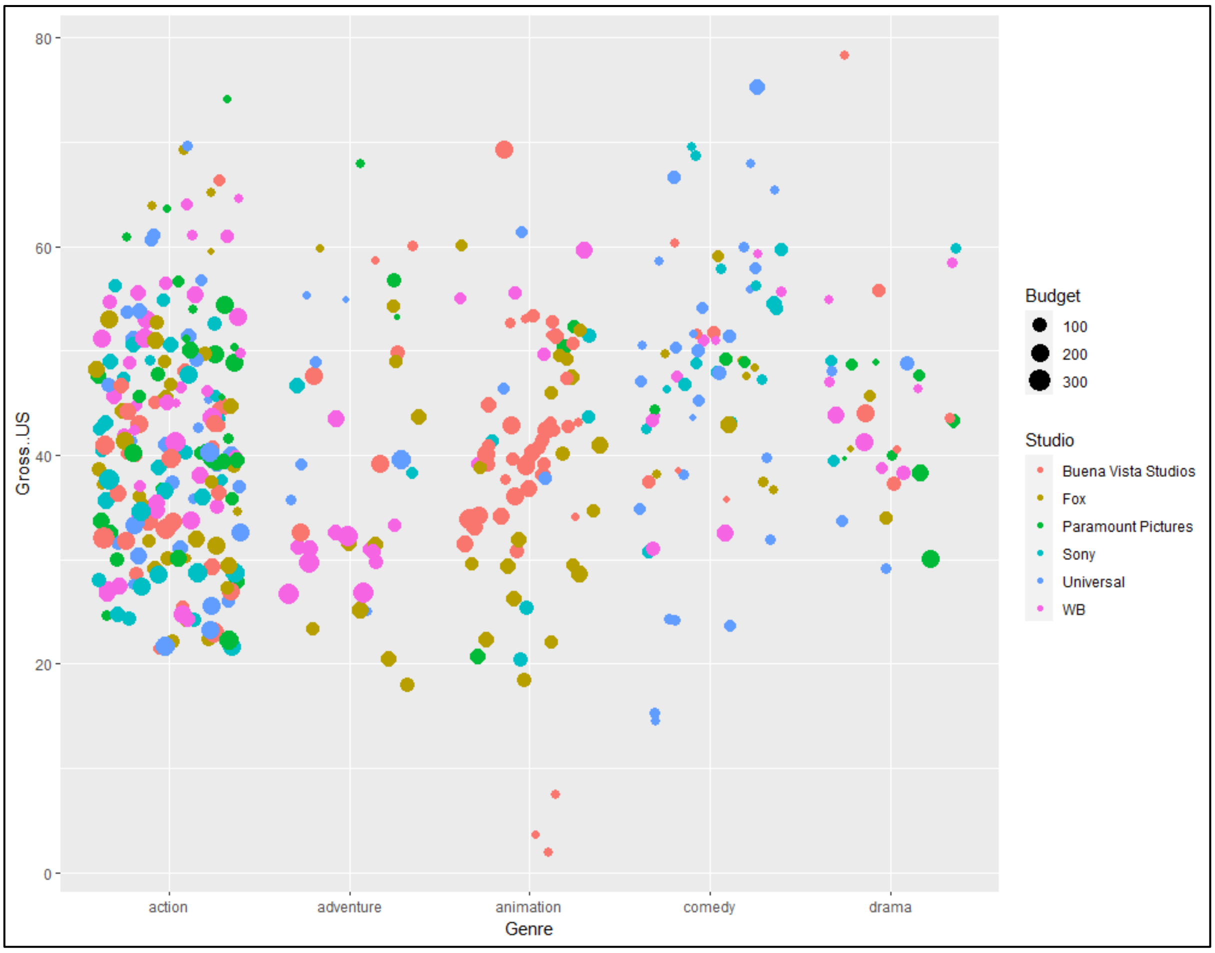
```
> newmoviextended
# A tibble: 423 × 18
   Day.of.Week Director       Genre Movie.Title Release.Date Studio Adjusted.Gross Budget
   <chr>       <chr>          <chr> <chr>       <chr>        <chr>           <dbl>  <dbl>
 1 Friday      Brad Bird      acti… Tomorrowla… 22/05/2015   Buena…           202.    170
 2 Friday      Scott Waugh    acti… Need for S… 14/03/2014   Buena…           204.     66
 3 Friday      Phil Lord, C… come… 21 Jump St… 16/03/2012   Sony             209.     42
 4 Friday      Roland Emmer… acti… White Hous… 28/06/2013   Sony             210.    150
 5 Friday      David Ayer     acti… Fury        17/10/2014   Sony             213.     80
 6 Thursday    Rob Marshall   adve… Into the W… 25/12/2014   Buena…           214.     50
 7 Friday      Daniel Espin… acti… Safe House  10/02/2012   Unive…           216.     85
 8 Friday      Gary Shore     acti… Dracula Un… 10/10/2014   Unive…           216.     70
 9 Friday      Eric Brevig    anim… Yogi Bear   17/12/2010   WB               220.     80
10 Friday      Ryan Murphy    drama Eat Pray L… 13/08/2010   Sony             223.     60
# ℹ 413 more rows
```

4. Creating a Scatter Plot with Jitter and Boxplot

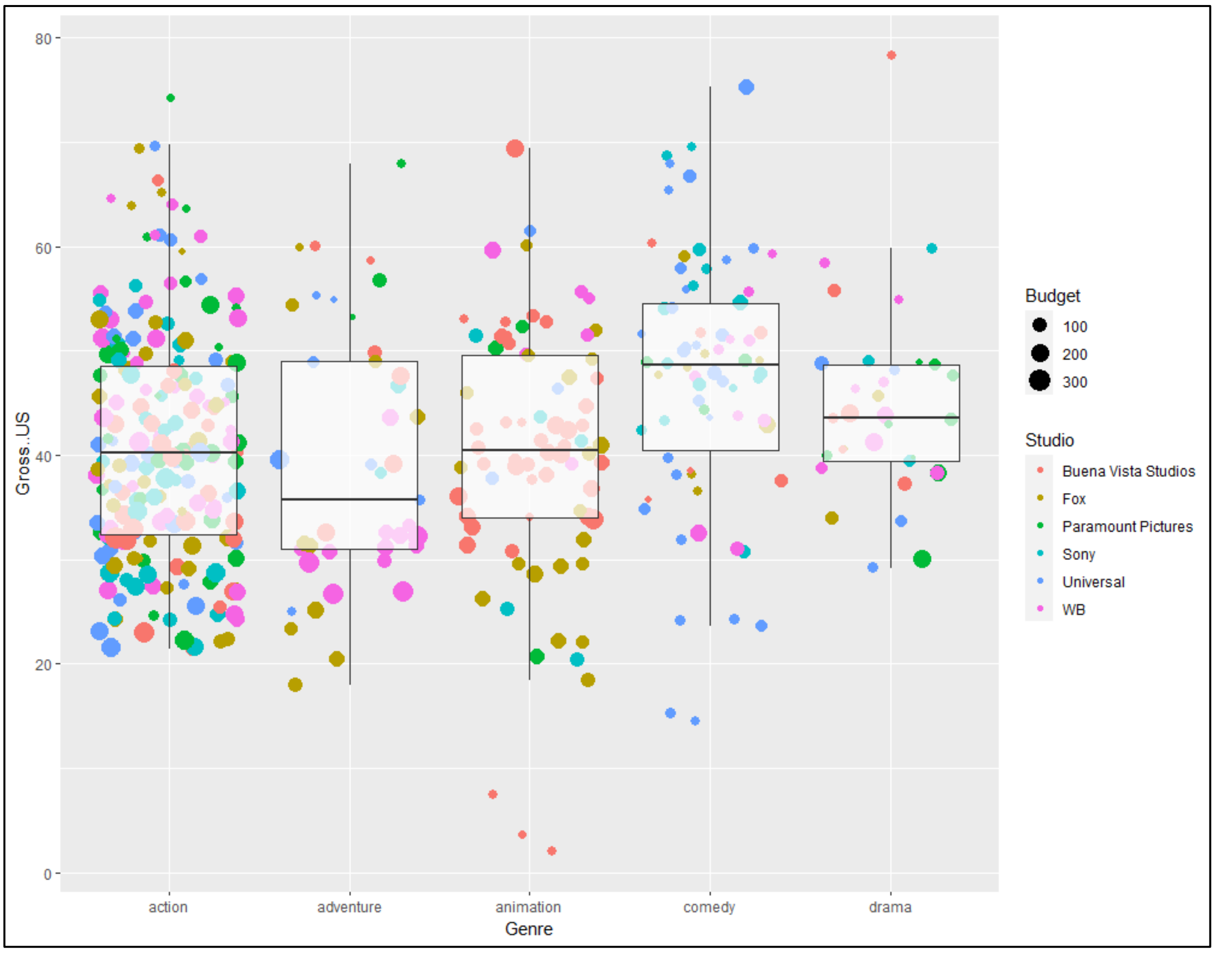
- p <- ggplot(data=newmoviextended, aes(x=Genre, y=Gross..US)): The object p is initialised with Gross % US on the y-axis and Genre on the x-axis as shown below.



- geom_jitter(aes(size=Budget, colour=Studio)) : Adds a jitter plot (geom_jitter) to display individual data points. Studio determines the colour of the dots, and the size of the points indicates the budget.



- geom_boxplot(alpha=0.7, outlier.color = NA): Adds a boxplot (geom_boxplot) with outliers coloured as not relevant (outlier.color = NA) and some transparency (alpha=0.7) as shown below.

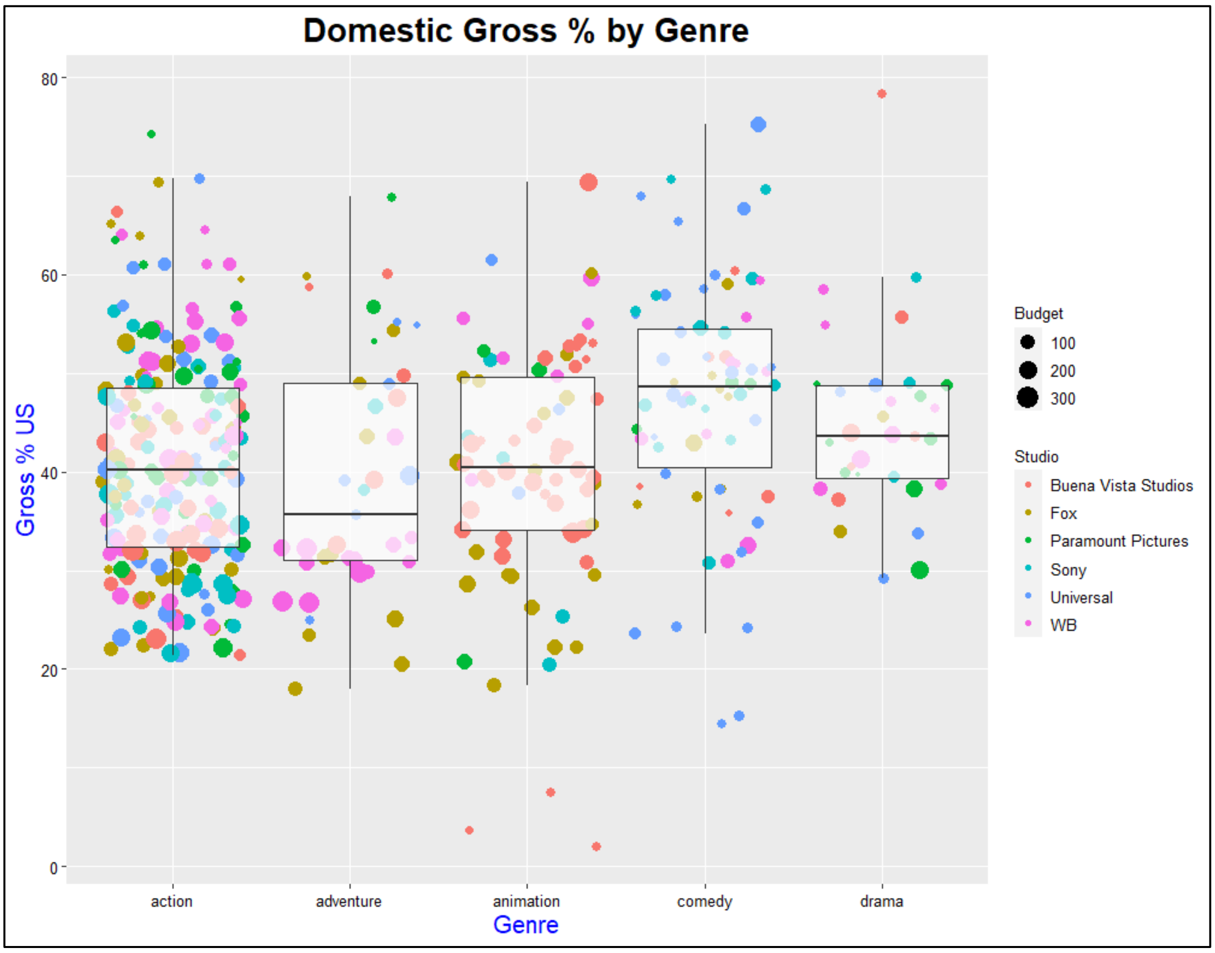


5. Formatting

- xlab("Genre"): Changes the label on the x-axis to "Genre."
- ylab("Gross % US"): y-axis label is set to "Gross % US."
- ggtitle("Domestic Gross% by Genre"): This function sets the title of the main plot to "Domestic Gross% by Genre."
- Axis.title.x and axis.title.y : Sets the colour and size of the x and y axis titles.
- Axis.text.x and axis.text.y : Sets the colour and size of the x and y axis ticks and texts.

- plot.title: Adjust the main plot title's colour, size, and boldness. legend.title and legend.text: Adjust the legend title and text's size. text: All of the plot's fonts should be set to "Comic Sans MS."
- legend.text and legend.title: Set the legend text's and title's font size.
- text: Change the plot's font family to "Comic Sans MS."
- g$labels$size <- "Budget $M": Adds label for the size aesthetic legend

Output for the Graph:



Write R code to find the trend of the Day of the week that most/least movies were released compared to other days.

- As the following graph will include working with count data and categorical factors (days of the week), a line plot could be useful to see patterns over time.
- The distribution of movie releases during the course of the week can also be shown using a bar chart as follows:

Input for the R code 1:

```
#Counting the number of movies released in each day of the week
day_of_week_counts <- table(moviextended$Day.of.Week)

#Finding the day with the most and least movie releases
most_released_day <- names(which.max(day_of_week_counts))
least_released_day <- names(which.min(day_of_week_counts))

#data/aesthetic layer
p <- ggplot(data = moviextended, aes(x = as.factor(Day.of.Week),fill =
Day.of.Week ))

#geometry layer
k <- p +
  geom_bar( color = "black")
```

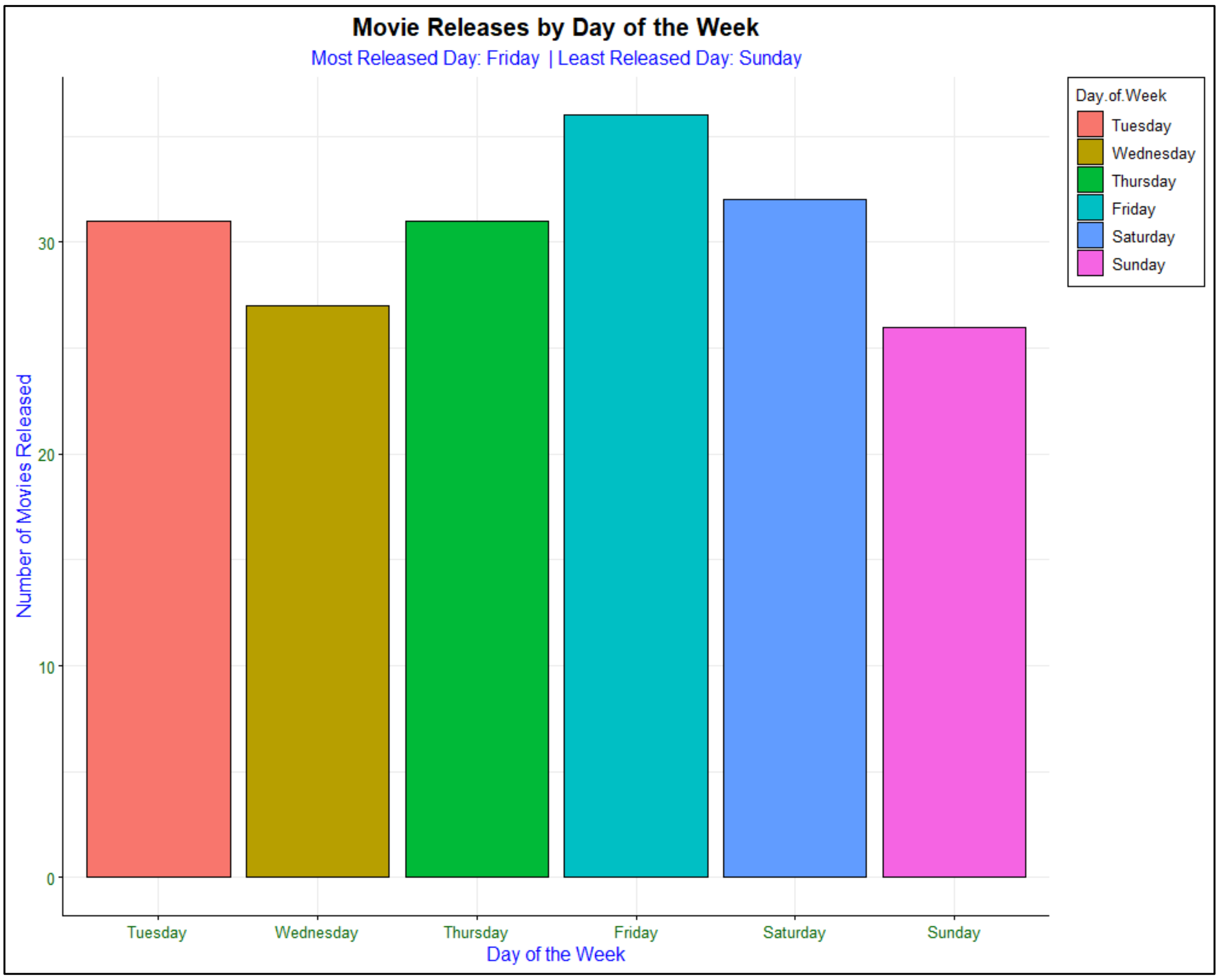
```
#formatting layer
k <- k +
  labs(title = "Movie Releases by Day of the Week",
       subtitle = paste("Most Released Day:", most_released_day, " | Least Released Day:", least_released_day),
       x = "Day of the Week",
       y = "Number of Movies Released") +
  theme_minimal() +
  theme(
    axis.title.x = element_text(colour="Blue", size=12),
    axis.title.y = element_text(colour="Blue", size=12),
    axis.text.x = element_text(colour="DarkGreen", size=10),
    axis.text.y = element_text(colour="DarkGreen", size=10),
    plot.title = element_text(colour="Black", size=15, hjust = 0.5, face = "bold"),
    plot.subtitle = element_text(colour="Blue", size=12, hjust = 0.5),
    legend.title=element_text(size=10),
    legend.text = element_text(size=10),
    legend.justification = c(1, 1),
    axis.line = element_line(color = "black"),
    axis.ticks = element_line(color = "black"),
    legend.box.background = element_rect(color = "Black"))
```

Explanation of the Code 1:

- moviextended$Day.of.Week: The days of the week for each movie release are stored in the "Day.of.Week" column of the dataframe moviextended is extracted.
- table(...): In this function, a table of counts for each distinct day is produced using the extracted column containing the days of the week and is stored in the variable named “day_of_week_counts”.
- which.max(day_of_week_counts): Gives back the day_of_week_counts table's maximum value's index.
- (which.max(day_of_week_counts)): This function finds the weekday that matches the maximum count, which is stored in most_released_day.
- (which.min(day_of_week_counts): This function finds the day of the week with the lowest number of movie releases and it is saved in least_released_day.
- p <- ggplot(data = moviextended, aes(x = as.factor(Day.of.Week),fill = Day.of.Week)) : In this section, the ggplot() method is used to initialise a ggplot object, p. It describes the data (moviextended) and defines the aesthetic mapping, where the fill colour is mapped to the day of the week and the x-axis indicates the days of the week (converted to a factor).
- k <- p + geom_bar(color = "black") : Then, a bar geometry (geom_bar()) is added to the already-existing ggplot object p to generate a new ggplot object k. The bar outlines' colour is set to black by using the colour = "black" option.

- Here, it's defining the axis labels, subtitle, and main title.
- theme_minimal(): This sets the plot's minimal theme.
- axis.text.x and axis.text.y: Set the colour and size of the x and y-axis text.
- plot.title: Set the colour, size, and alignment of the main plot.
- axis.title.x and axis.title.y: Set the colour and size of the x and y-axis titles.
- plot.subtitle: Configure the plot subtitle's colour, size, and alignment.
- legend.title and legend.text: Adjust the legend's title and text to the appropriate size.
- axis.line and axis.ticks: Choose the colour of the ticks and axis lines.
- Legend.box.background: Choose the background colour of the legend box.

Output 1 of the Graph to find movie release by day of the week :



Analysis in finding movie release by day of the week:

- The pattern shown in the bar chart verifies that, in comparison to other days of the week, Friday is the day when the most films are released, and Sunday has the fewest releases.
- Tuesday through Thursday are the mid-week days, and the numbers are rather stable with just little variations.

- The considerably lower number of releases on Saturday and the more substantial decline on Sunday can be attributed to industry practices in which movie openings are scheduled to take advantage of the maximum audience availability, which usually peaks over the weekend.

Graph 2:

Input of the R code:

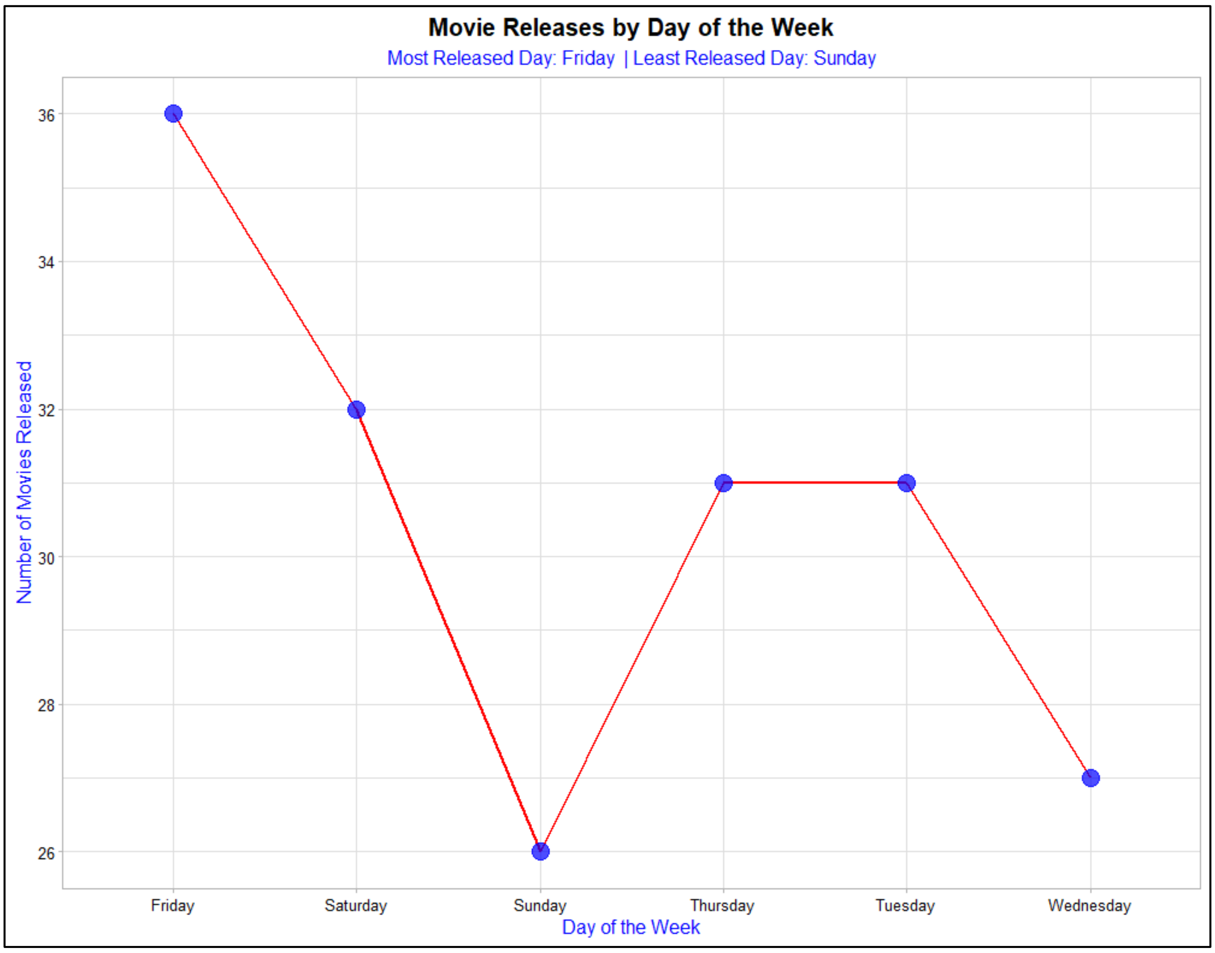
```
#Counting the number of movies released in each day of the week

day_of_week_counts <- table(moviextended$Day.of.Week)



#creating a dataframe

movie_df <- data.frame(Day.of.Week = names(day_of_week_counts),

                    Count = as.numeric(day_of_week_counts))



#removing any values with NA column

movie_df <- na.omit(movie_df)
```

```
#Convert the 'Day of Week' column to a factor for proper ordering

movie_df$Day.of.Week <- factor(movie_df$Day.of.Week,

                               levels = c("Monday", "Tuesday", "Wednesday", "Thursday",
"Friday", "Saturday"))



#data-layer

r <- ggplot(data = movie_df, aes(x = as.factor(Day.of.Week), y= Count, group = 1))

#adding geometry

m <- r +

  geom_line(color = "Red", size=1) +

  geom_point(color = "Blue", size=5, alpha=0.7)



#adding formatting

m <- m +

  labs(title = "Movie Releases by Day of the Week",

       subtitle = paste("Most Released Day:", most_released_day, " | Least Released
Day:", least_released_day),
x = "Day of the Week",

     y = "Number of Movies Released") +

 theme_light() +

 theme(

   axis.title.x = element_text(colour="Blue", size=12),

   axis.title.y = element_text(colour="Blue", size=12),

   axis.text.x = element_text(colour="Black", size=10),

   axis.text.y = element_text(colour="Black", size=10),

   plot.title = element_text(colour="Black", size=15, hjust = 0.5, face = "bold"),

   plot.subtitle = element_text(colour="Blue", size=12, hjust = 0.5),

   legend.title=element_text(size=10),

   legend.text = element_text(size=10))
```

- In this graph with geom line, after using the count function, a data frame is created.
- data.frame(...): Produces a data frame with "Day.of.Week" and "Count" as its two columns.
- names(counts_day_of_week): takes the day names and extracts them from the day_of_week_counts.

- as.numeric(counts_days_of_week): converts the counts into a number.
- na.omit(movie_df): eliminates all rows in the dataframe that have NA values.
- factor(...): To ensure correct ordering, converts the "Day.of.Week" column to a factor with predetermined levels. The levels are in this function are correctly arranged in relation to the days of the week.
- c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday") : The days of the week are listed in this instance in the proper chronological sequence, running from Monday through Saturday.
- ggplot(...): Uses the movie_df dataframe to initialise a ggplot object.
- aes(y = Count, group = 1, x = as.factor(Day.of.Week)): It utilises one group (1) for the whole plot and sets the x- and y-axes to "Day.of.Week" and "Count."
- geom_line(...): Appends a red-colored, precisely-sized line plot to the ggplot object.
- geom_point(...): Increases the size, colour, and transparency (alpha = 0.7) of the ggplot object by adding points.
- The formatting layer is same as the graph 1.

Output 2 of the Graph to find movie release by day of the week :



Analysis of the Graph 2:

- The number of films released on each day of the week is seen in the graph "Movie Releases by Day of the Week". The horizontal axis indicates the days of the week from Friday to Wednesday, while the vertical axis shows the total number of films released. A blue dot is placed on each data point, and a red line connects them all.
- The following is the detailed examination of the graph:

1. Friday: With a total of around 37, Friday is the day on the graph that displays the greatest number of films released. This implies that the most favoured day for movie releases is Friday.
2. Saturday: There are only around 27 releases on Saturday, a dramatic decrease from Friday. This is a considerable drop and suggests that Saturday is not as popular as Friday for movie launches.
3. Sunday: With about 26 films released, the number of releases falls even down to the lowest point on the graph. Sunday is the least popular day for movie releases, based on the graph's title.
4. Wednesday: There is a noticeable spike in releases starting on Sunday and continuing at a fairly steady rate through Tuesday, when about 29 films are released each day. On Wednesday, there is a minor uptick, bringing the total to about 30.

- Monday - Even though all the days of the week in chronological order from Monday to Sunday, is mentioned within the "movie_df$Day.of.Week" function , Monday is not visible in the graph. This proves that none of the movies in the extended movie dataset in released on Monday.

In conclusion, it is evident from the graph that Friday is the most popular day for movie releases, while Sunday is the least popular. There are no notable highs or lows in the number of movie releases during the midweek (kbroman.org, 2017).

Identify if the profit of a movie depends on any of the features in this data set i.e. genre, director, profit etc

There are different graphs showed below to visually identify if profit depends on any other variable of the movie dataset.

Input 1 : Code for Profit dependent upon directors

```
#Profit VS Directors

library(dplyr)

threshold <- 5

# Filter directors with movies above the threshold

new_d <- moviextended %>%

  group_by(Director) %>%

  filter(n() >= threshold) %>%

  mutate(Director_reorder = reorder(Director, Profit, FUN = mean))

# Data-layer

pd <- ggplot(new_d, aes(x = Director_reorder, y = Profit))
```

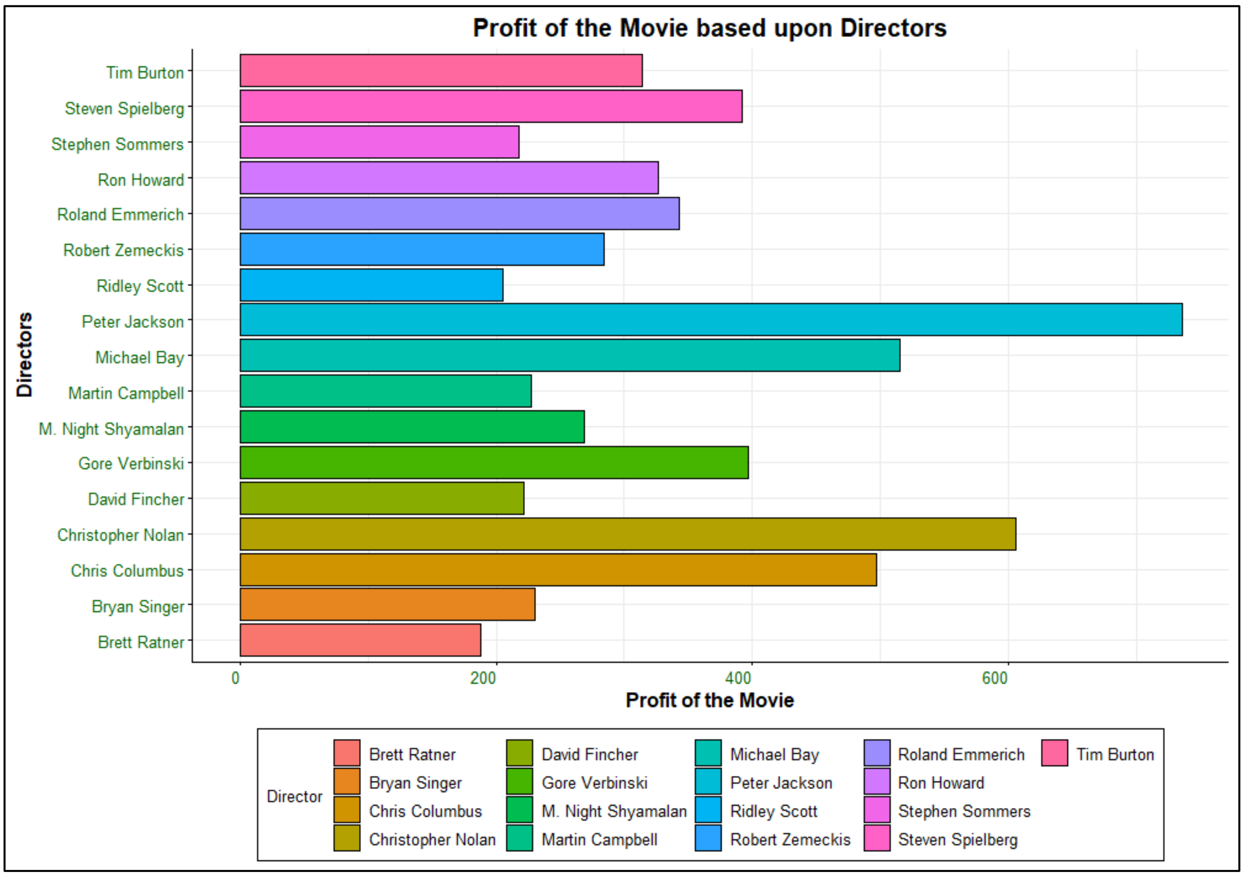
```
# Geometry layer

pd <- pd +

  stat_summary(fun = mean, geom = "bar", colour = "Black", aes(fill= Director))

# Formatting layer

pd <- pd +

  labs(

    title = "Profit of the Movie based upon Directors",

    x = "Directors",

    y = "Profit of the Movie"

  ) +

  theme_minimal() +

  theme(

    axis.title.x = element_text(colour = "Black", size = 12, face = "bold"),

    axis.title.y = element_text(colour = "Black", size = 12, face = "bold"),

    axis.text.x = element_text(colour = "DarkGreen", size = 10, hjust = 1),

    axis.text.y = element_text(colour = "DarkGreen", size = 10),

    plot.title = element_text(colour = "Black", size = 15, hjust = 0.5, face = "bold"),
legend.title = element_text(size = 10),

    legend.text = element_text(size = 10),

    axis.line = element_line(color = "black"),

    axis.ticks = element_line(color = "black"),

    legend.box.background = element_rect(color = "Black"),

    legend.position = "bottom",

    legend.justification = "center") +

  coord_flip()
```

Explanation of the code 1:

- The dplyr library is used for data manipulation and analysis.
- Chaining actions together is done with the %>% operator, also known as the pipe operator. It receives the output of the expression on its left and sends it to the function on its right as the first parameter.
- The Director variable is the variable that will be used to group the data using the dplyr function group_by().
- The number of rows in each group is returned by the filter() method's n() function. Consequently, groups (directors) with fewer films than threshold are filtered out by n() >= threshold.
- To add additional variables to the data frame, modify() function is used.

- The method reorder() rearranges a factor's levels according to a given variable. This time, it's Director, and Profit is the ordering variable.
- “FUN = mean” suggests that the order should be established using the profit mean. The mean yields an overall average profit for each director and is a meaningful measure of central tendency (Hadley, 2010).
- Then, a ggplot object pd is initialised with the profit variable represented on the y-axis and the reordered directors (Director_reorder) on the x-axis.
- The bars show each director's average profit and are highlighted in black (colour = "Black") and filled with various colours for each director (aes(fill = Director)).
- The formatting of the graph remains same as the previous graphs above.
- Coordinates are flipped using coord_flip() to produce a horizontal bar graphic.

Output for the Profit dependent upon Directors Graph:



Analysis of the Graph:

In this graph, there is a horizontal bar chart with the headline "Profit of the Movie based upon Directors." The length of each bar denotes a distinct director, and it also shows the profit margin for films the director has been involved with. The directors are filtered out with movies more than 5 to make the analysis look more efficient. Each director's bar is coloured differently for easy separation, and the directors are listed along the vertical axis.

This graph allows us to infer the following conclusions:

- Based on the graph, Steven Spielberg is the filmmaker whose films have brought in the most money. Among the filmmakers mentioned, his bar stretches the furthest on the scale, indicating that his films have made the greatest money.
- Compared to directors like Spielberg, Michael Bay, or Peter Jackson, filmmakers like Brett Ratner, Bryan Singer, and Chris Columbus have shorter bars, indicating lower profit levels linked with their films.
- Although this graph does not prove a link between variables the large range of revenues linked to various directors raises the possibility that movie profits are dependent on the director.

Input 2 : Code for Profit dependent upon IMDb.Rating

```
#Profit VS IMBD

#data-aesthetic layer

i <- ggplot(data = moviextended, aes(x = IMDb.Rating, y = Profit, group = 1))

#geometry

i <- i +

  geom_point(alpha=0.3, aes(group=1, colour=Genre, size=Budget)) +

  geom_smooth(method = "lm", se = FALSE, aes(group=1))
```

```
#adding formatting

i <- i +

  labs(

  title = "Profit vs IMDb Rating",

  x = "IMDb Rating",

  y = "Profit") +

  theme_minimal() +

  theme(

    axis.title.x = element_text(colour = "Black", size = 12),

    axis.title.y = element_text(colour = "Black", size = 12),

    axis.text.x = element_text(colour = "Blue", size = 10),

    axis.text.y = element_text(colour = "Blue", size = 10),

    plot.title = element_text(colour = "Black", size = 15, hjust = 0.5, face = "bold"),

    legend.title = element_text(size = 10),

    legend.text = element_text(size = 10),

    legend.justification = c(1, 1),

    axis.line = element_line(color = "black"),
axis.ticks = element_line(color = "black"),

    legend.box.background = element_rect(color = "Black")

  )
```

Explanation of the Code 2:

1. **Data and Aesthetics**

- A ggplot object is initialised with ggplot(). The data frame to be utilised is specified by the argument data = moviextended.
- aes(x = IMDb).Rating determines the aesthetic mappings (y = Profit, group = 1). IMDb is expressed on the x-axis.Rating, group = 1 guarantees that all points are regarded as belonging to the same group, and the y-axis shows profit.

2. **Geometry Layer**

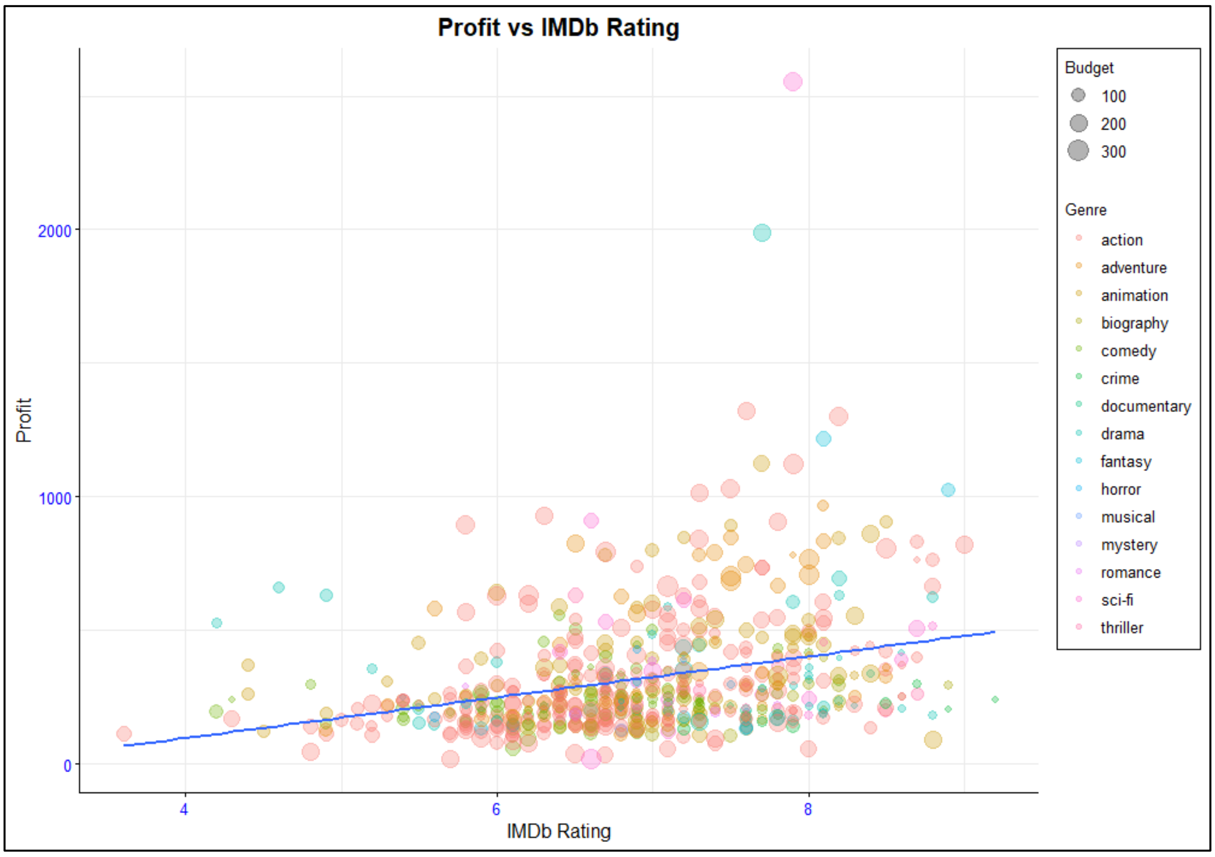
- Adds a scatter plot layer with points using geom_point().
- The points' transparency is set at alpha=0.3.
- The aesthetics for points are specified by aes(group=1, color=Genre, size=Budget). They are color-coded according to the Genre variable, sized according to the Budget variable, and in the same group (group=1).
- A smoothed line, in this case a linear regression line, is added to the graphic by the geom_smooth() function.
- Linear regression is specified with method = "lm".

- aes(group=1) guarantees that the line is drawn for the full dataset.

3. **The formatting Layer**

   The formatting structure remains similar to the graphs shown above.

Output for the Profit dependent upon IMDb Graph:



Analysis for the IMDb Rating VS Profit Graph:

The graph is a scatter plot demonstrating the correlation between the profit of films and their IMDb ratings, with extra data encoded about the budget size and genres of the films.

The graph allows for the following conclusions to be drawn:

- **Profit vs IMDb Rating**: A blue trendline indicates that there appears to be a positive relationship between profits and IMDb ratings. There appears to be a little increasing trend in profit as the IMDb rating rises, suggesting that higher-rated films often make larger earnings. The dispersion of points, however, is somewhat dispersed, particularly for films with mid-range ratings (about a 6-8), indicating that although there may be a trend for higher-rated films to make more money, the correlation is not particularly strong.
- **Budget Size**: The graph represents the various budget levels (100, 200, and 300) with dots of varying sizes. Although films with bigger budgets (shown by larger dots) appear to be more likely to make money, there are many of examples when films with lesser budgets make a lot of money. This suggests that although budget has a role in profit, it is not the only one.
- **Movie Genres**: Various hues correspond to distinct genres. The graph's points are widely distributed throughout genres, and no genre seems to dominate any one area, indicating that the potential for profit is not exclusive to any one kind of film.

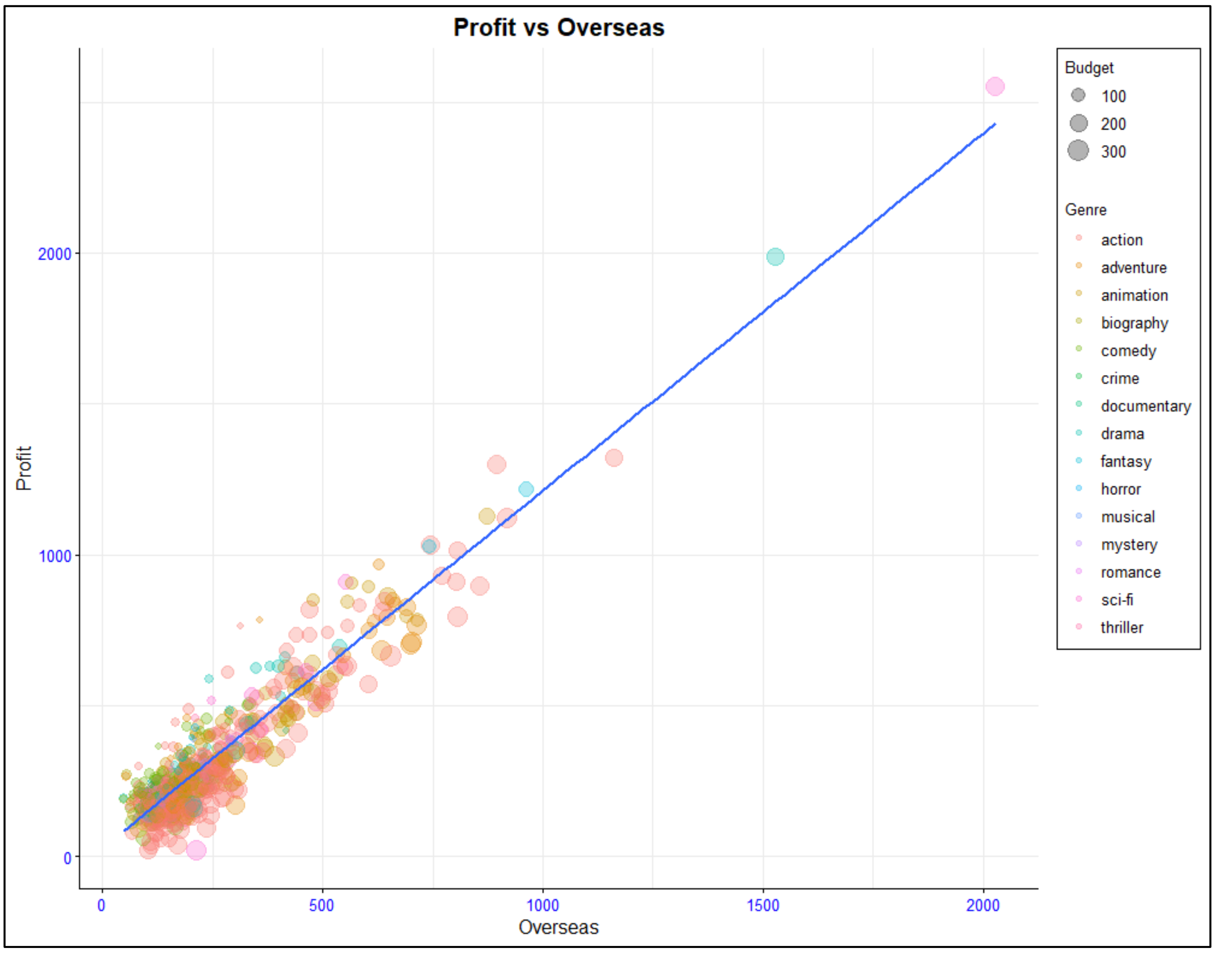
The graph suggests that there may be a non-deterministic link between profit and IMDb ratings. There are films with low to mid-range ratings that make a lot of money, and vice versa. High earnings are not only the result of these characteristics; other elements like genre and budget size also influence the profit.

Input 3: Code for Profit dependent upon Overseas

The code follows similar structure to above graph.

```
#Profit VS Overseas

#If profit depends upon Overseas

o <- ggplot(data = moviextended, aes(x = Overseas, y = Profit, group = 1))

#geometry

o <- o +

  geom_point(alpha=0.3, aes(group=1, colour=Genre, size=Budget)) +

  geom_smooth(method = "lm", se = FALSE, aes(group=1))

#adding formatting

o <- o +

  labs(

    title = "Profit vs Overseas",

    x = "Overseas",

    y = "Profit") +
theme_minimal() +

  theme(

    axis.title.x = element_text(colour = "Black", size = 12),

    axis.title.y = element_text(colour = "Black", size = 12),

    axis.text.x = element_text(colour = "Blue", size = 10),

    axis.text.y = element_text(colour = "Blue", size = 10),

    plot.title = element_text(colour = "Black", size = 15, hjust = 0.5, face = "bold"),

    legend.title = element_text(size = 10),

    legend.text = element_text(size = 10),

    legend.justification = c(1, 1),

    axis.line = element_line(color = "black"),

    axis.ticks = element_line(color = "black"),

    legend.box.background = element_rect(color = "Black")

  )
```

Output for Profit dependent upon Overseas Revenue:



Analysis for the Profit dependent upon Overseas Graph:

This scatter plot graph shows the correlation between a film's domestic profits and its foreign earnings. The colours signify the various genres, while the size of the dots represents the budget.

There is a clear positive association between profit and overseas profits, as seen by the graph's blue trendline. Profit tends to rise in parallel with increases in international earnings, indicating that profit is influenced by how well the film does in foreign markets.

A closer association between these two factors is indicated by the densely grouped points around the trendline, particularly for films with larger international profits than in the prior IMDb rating research. The graph indicates that a movie's profitability may be significantly predicted by its overseas profits.

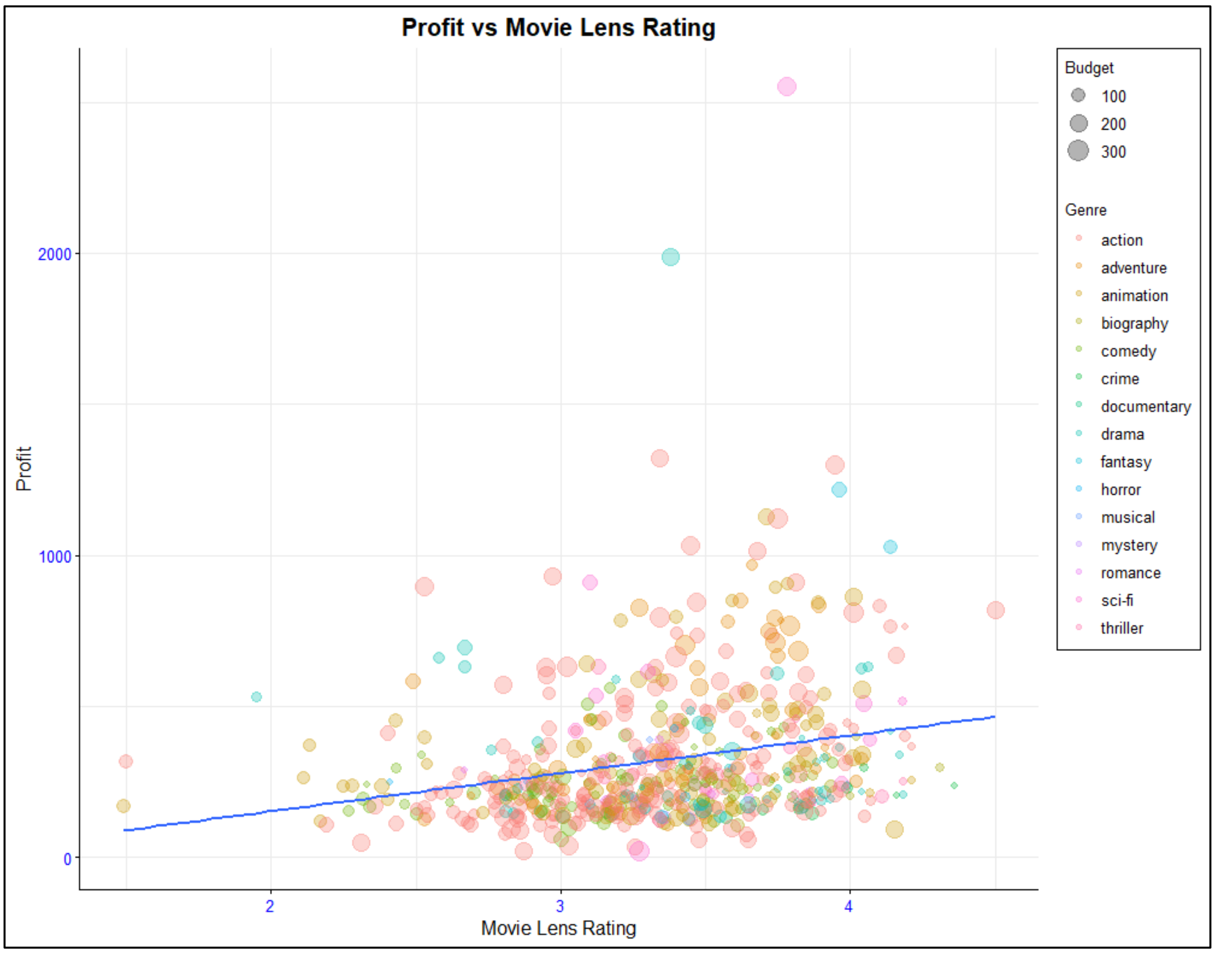
Input 4: Code for Profit dependent upon Movie Lens Rating

The code follows similar structure to graph above.

```
#Profit VS Movie Lens Rating

mr <- ggplot(data = moviextended, aes(x = MovieLens.Rating , y = Profit, group = 1))

#geometry

mr <- mr +

  geom_point(alpha=0.3, aes(group=1, colour=Genre, size=Budget)) +

  geom_smooth(method = "lm", se = FALSE, aes(group=1))
```

```
#adding formatting

mr <- mr +

  labs(

    title = "Profit vs Movie Lens Rating",

    x = "Movie Lens Rating",

    y = "Profit") +

  theme_minimal() +

  theme(

    axis.title.x = element_text(colour = "Black", size = 12),

    axis.title.y = element_text(colour = "Black", size = 12),

    axis.text.x = element_text(colour = "Blue", size = 10),

    axis.text.y = element_text(colour = "Blue", size = 10),

    plot.title = element_text(colour = "Black", size = 15, hjust = 0.5, face = "bold"),

    legend.title = element_text(size = 10),

    legend.text = element_text(size = 10),

    legend.justification = c(1, 1),
axis.line = element_line(color = "black"),

    axis.ticks = element_line(color = "black"),

    legend.box.background = element_rect(color = "Black")

  )
```

Output for the Profit dependent upon Movie Lens Rating Graph:



Analysis of the Profit dependent upon Movie Lens Rating Graph:

The association between Profit and Movie Lens Rating for different films is shown in a scatter plot on the graph. Every dot on the graph symbolises a movie, and its position is based on its profit on the vertical axis and its rating on the horizontal axis. The movie's budget is represented by the size of the dots; more dots denote a larger budget. The many movie genres are represented by the colours of the dots. The graph shows a wide dispersion of data points along the rating axis, indicating that the earnings of the films vary substantially throughout all Movie Lens Rating ranges.

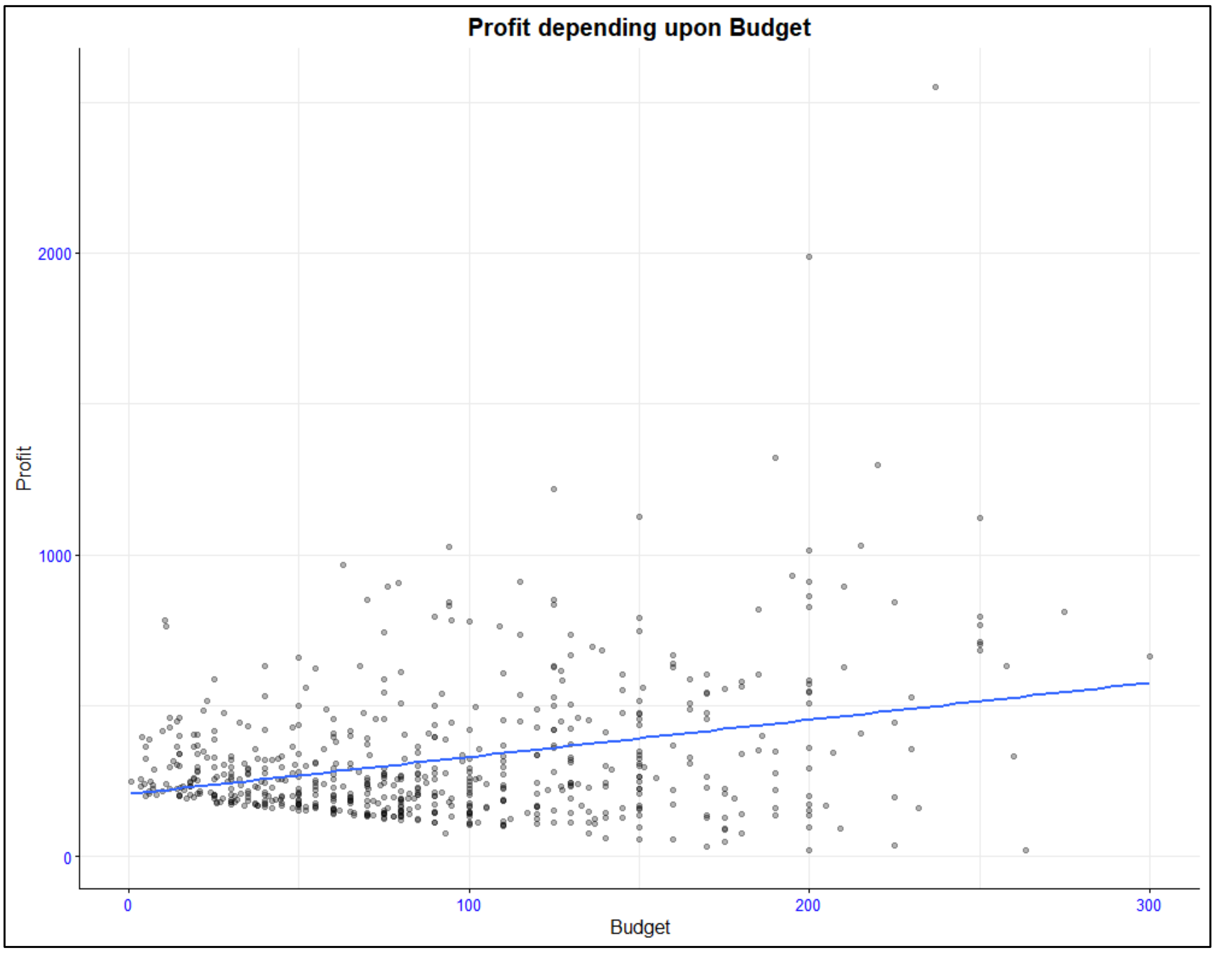
The small upward slope of the fitted line, which is most likely a linear regression line, may indicate a weakly positive relationship between a movie's profit and its Movie Lens Rating. Nevertheless, given that the data points are widely scattered and do not closely resemble the fitted line, the association does not seem to be very strong.

Input 5: Code for Profit dependent upon Budget.

The graph follows similar structure to the code above.

```
#Profit VS Budget

bb <- ggplot(data = moviextended, aes(x = Budget , y = Profit, group = 1))

#geometry

bb <- bb +

  geom_point(alpha=0.3, aes(group=1)) +

  geom_smooth(method = "lm", se = FALSE, aes(group=1))

#adding formatting

bb <- bb +

  labs(

    title = "Profit depending upon Budget",

    x = "Budget",

    y = "Profit") +

  theme_minimal() +

  theme(

    axis.title.x = element_text(colour = "Black", size = 12),

    axis.title.y = element_text(colour = "Black", size = 12),
    axis.text.y = element_text(colour = "Blue", size = 10),

    plot.title = element_text(colour = "Black", size = 15, hjust = 0.5, face = "bold"),

    legend.title = element_text(size = 10),

    legend.text = element_text(size = 10),

    legend.justification = c(1, 1),

    axis.line = element_line(color = "black"),

    axis.ticks = element_line(color = "black"),

    legend.box.background = element_rect(color = "Black")

  )
```

Output for the Profit dependent upon Budget Graph:



Analysis for the Profit dependent upon Budget Graph:

The graph, which shows the relationship between a movie's profit (on the vertical axis) and budget (on the horizontal axis), is a scatter plot. Every dot stands for a movie, and the positions of each dot indicate the associated profit and budget figures.

Upon examining the graph, one may observe a trend line with a positive slope, indicating a positive association between a film's budget and its profit margin.

The dispersion of the data points surrounding this line, however, shows that although there is a significant range in profit at almost all budget levels, greater budgets frequently translate into higher profits; the connection is not exactly proportionate.

There is a noticeable distribution of profit levels over all budget values, and a sizable number of data points are concentrated at the lower end of the budget spectrum. This distribution suggests that both high-budget films and low-budget films have produced notable financial gains and losses, respectively.

In summary, the graph suggests a degree of reliance between a film's profit and budget, showing a generally positive link. However, given the variety in the data, it appears that other elements are equally important for a movie's financial success, therefore the budget is not the only factor that predicts profit.

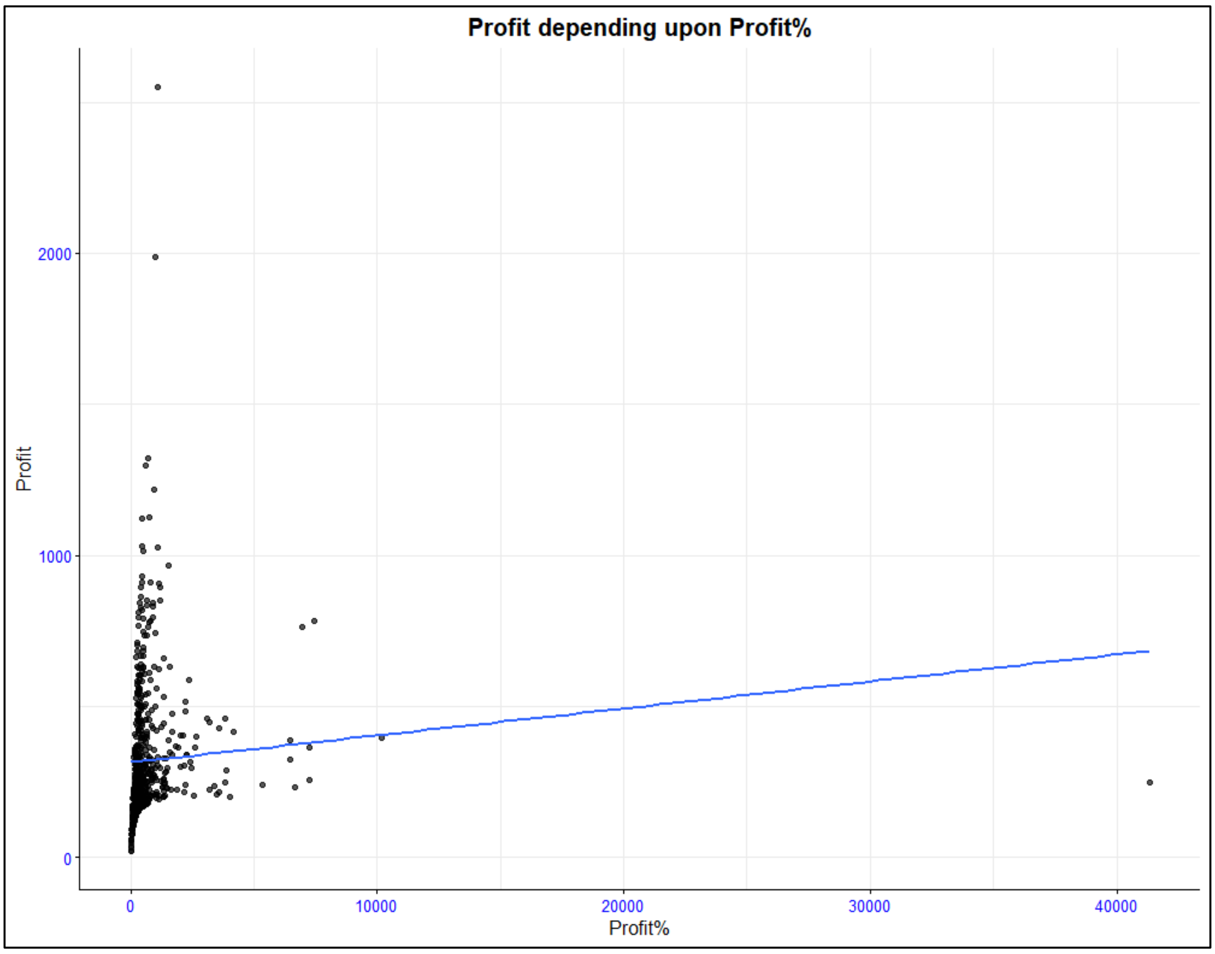
Input 6: Code for Profit dependent upon Profit%.

The code follows similar structure to the graph above.

```
#Profit VS Profit%

pp <- ggplot(data = moviextended, aes(x = Profit.. , y = Profit, group = 1))

#geometry

pp <- pp +

  geom_point(alpha=0.3, aes(group=1)) +

  geom_smooth(method = "lm", se = FALSE, aes(group=1))
```

```
pp <- pp +
  labs(
    title = "Profit depending upon Profit%",
    x = "Profit%",
    y = "Profit") +
  theme_minimal() +
  theme(
    axis.title.x = element_text(colour = "Black", size = 12),
    axis.title.y = element_text(colour = "Black", size = 12),
    axis.text.x = element_text(colour = "Blue", size = 10),
    axis.text.y = element_text(colour = "Blue", size = 10),
    plot.title = element_text(colour = "Black", size = 15, hjust = 0.5, face = "bold"),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.justification = c(1, 1),
axis.line = element_line(color = "black"),

    axis.ticks = element_line(color = "black"),

    legend.box.background = element_rect(color = "Black")

  )
```

Output for the Profit dependent upon Profit% Graph:



Analysis for the Profit dependent upon Profit% Graph:

There is a noticeable distribution of profit levels over all budget values, and a sizable number of data points are concentrated at the lower end of the budget spectrum. This distribution suggests that both high-budget films and low-budget films have produced notable financial gains and losses, respectively.

In summary, the graph suggests a degree of reliance between a film's profit and budget, showing a generally positive link. However, given the variety in the data, it appears that other elements are equally important for a movie's financial success, therefore the budget is not the only factor that predicts profit.

The graph indicates that a movie's profit and its profit % are positively correlated. This suggests that films with larger profit percentages typically make more money overall.

Still, a weak link is shown by the dispersion of the data points and the flatness of the trend line. This indicates that although there is a general pattern, there are also a lot of outliers, and a high profit margin does not equate to a big profit.

Input 7: Code for Profit dependent upon Studio

```
#Profit VS Studio

# Filter out studios with profit below 500

new_s <- moviextended %>%

  filter(Profit >= 500)



#data-aes

st <- ggplot(data = new_s , aes(x = Studio, y = Profit, fill = Studio))



#geometry layer

st <- st +

  geom_bar(stat = "identity", color = "black", width=0.4)
```

```
#formatting

st <- st +

  labs(

    title = "Total Profits by Studio",

    x = "Studio",

    y = "Total Profit"

  ) +

  theme_minimal() +

  theme(

    axis.title.x = element_text(colour = "Black", size = 12, face="bold"),

    axis.title.y = element_text(colour = "Black", size = 12, face="bold"),

    axis.text.x = element_text(colour = "DarkGreen", size = 10, angle= 0, hjust=1),

    axis.text.y = element_text(colour = "DarkGreen", size = 10, angle= 0, hjust=1),

    plot.title = element_text(colour = "Black", size = 15, hjust = 0.5, face = "bold"),

    axis.line = element_line(color = "black"),

    axis.ticks = element_line(color = "black"),

    legend.title = element_text(size = 8),

    legend.position = "bottom",
legend.box = "vertical",

    legend.box.background = element_rect(color = "Black")) +

    coord_flip()
```

Explanation of the Graph:

1. **Filtering**

   filter(Profit >= 500): Creates a new data frame called new_s by choosing only rows in which the profit is 500 or more.

2. **Data and Aesthetics**

   ggplot(data = new_s, aes(x = Studio, y = Profit, fill = Studio)): Initialises a ggplot object by utilising fill colour determined by studios and x-axis set to studios and profit.

3. **Geometry Layer**

   geom_bar(stat = "identity", colour = "black", width=0.4): Bars show the overall profit for each studio. stat = "identity" indicates that the bar heights match the data values. Bar borders are coloured black, and the bars' width may be adjusted with a width of 0.4.

4. **Formatting Layer**

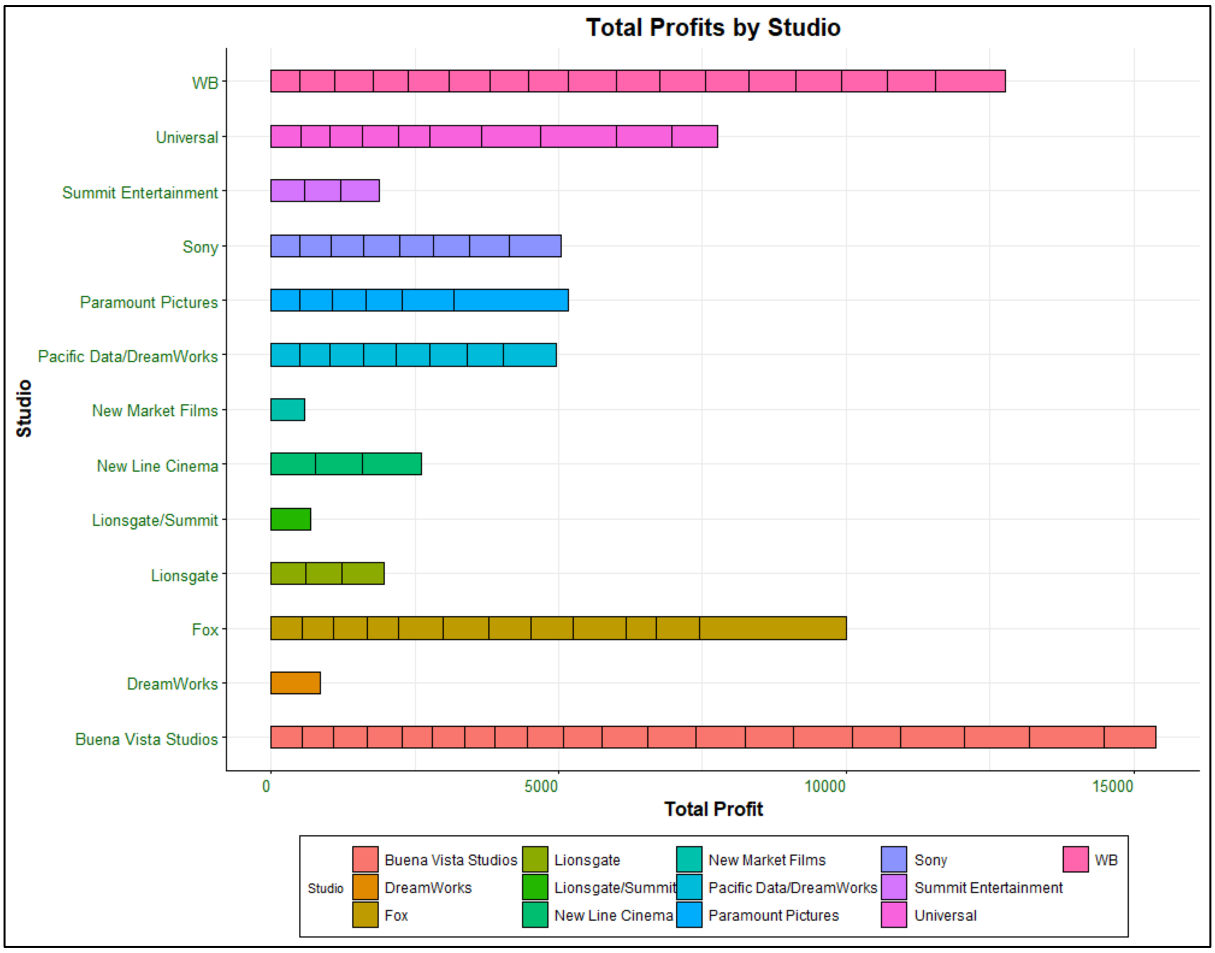
   labs(): Establishes the axis labels and title.

   theme_minimal(): Gives the story a simple theme.

   Theme(): Extends customisation of visual elements, such as coordinate flipping, axis titles, text, and plot titles.

   coord_flip(): To get a horizontal bar plot, flip the coordinate system.

Output for the Profit dependent upon Studio Graph:



Analysis for the Profit dependent upon Studio Graph:

The above graph, called "Total Profits by Studio," is a horizontal bar chart with the names of many film studios on the y-axis and their respective total earnings, expressed in 10,000s, on the x-axis. A caption is included at the bottom of the graph to assist in identifying which colour relates to which studio. Each studio is represented by a distinct colour.

The entire profit made by the respective studio is shown by the length of each bar. From top to bottom, the studios are organised in decreasing order of profit. Within the parameters of this graph, WB (shown in pink) looks to have the most overall earnings, while Buena Vista Studios (shown in red) appears to have the lowest. There may be a relationship between the studio and the profit made, as seen by the stark differences in profits across the studios. Given that "Studio" is a categorical variable, a bar chart makes sense as a visualisation method as it makes comparing several categories simple.

Bar charts offer a lucid graphical depiction of the variations in earnings among the studios, rendering the identification of the most profitable studios simple (Healy, 2019).

Unlike with numerical variables, there is no natural sequence or numerical progression because the studios are categorical in nature. This is respected by a bar graph, which treats each studio independently. It could be challenging to distinguish between studios with lower earnings if all studios are included in the graph, regardless of the magnitude of their profits.

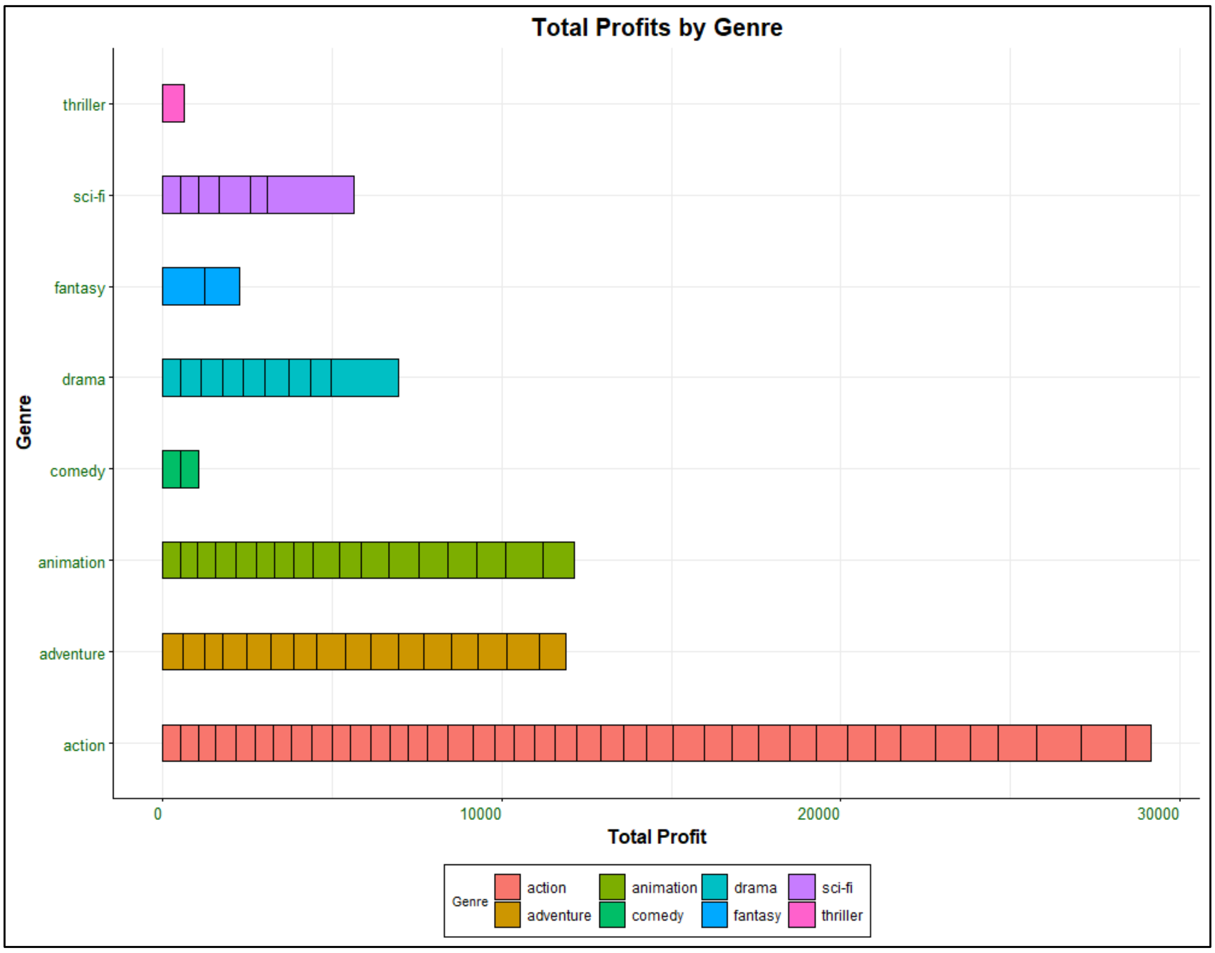
By establishing a cutoff point (profit higher than 500, for example), it concentrates the study on the best-performing studios and makes it more evident which studios are the most profitable. In conclusion, the bar chart clearly illustrates the correlation between studios and earnings, showing that there is, in fact, a profit variance that appears to be related to the studio.

Input 8: Code for Profit dependent upon Genre.

This graph is exactly similar to the above graph.

```
#If Profit depends upon the Genre

new_g <- moviextended %>%

  filter(Profit >= 500)

#data-aes

gg <- ggplot(data = new_g , aes(x = Genre, y = Profit, fill = Genre))

#geometry layer

gg <- gg +

  geom_bar(stat = "identity", color = "black", width=0.4)



#formatting

gg <- gg +

  labs(

    title = "Total Profits by Genre",

    x = "Genre",

    y = "Total Profit"

  ) +
theme_minimal() +

  theme(

    axis.title.x = element_text(colour = "Black", size = 12, face="bold"),

    axis.title.y = element_text(colour = "Black", size = 12, face="bold"),

    axis.text.x = element_text(colour = "DarkGreen", size = 10, angle= 0, hjust=1),

    axis.text.y = element_text(colour = "DarkGreen", size = 10, angle= 0, hjust=1),

    plot.title = element_text(colour = "Black", size = 15, hjust = 0.5, face = "bold"),

    axis.line = element_line(color = "black"),

    axis.ticks = element_line(color = "black"),

    legend.title = element_text(size = 8),

    legend.position = "bottom",

    legend.box = "vertical",

    legend.box.background = element_rect(color = "Black")) +

    coord_flip()
```

Output for the Profit dependent upon Genre Graph:



Analysis of Profit dependent upon the Genre Graph:

- The action genre (highlighted in red) has the longest bar, indicating the largest overall revenues.
- Out of all the genres covered, the thriller genre (in pink) has the lowest overall profits, as indicated by the shortest bar.
- The profitability of various genres may be seen to change visually, suggesting that the genre of the movie may have an impact on the overall profit.

Use ggplot and boxplot to identify if there is an anomaly / any anomalies in the data?

According to (Wickham & Grolemund, 2016), Boxplots are normally used to summarise a dataset's distribution and central tendency as well as to spot any outliers.

Box plots are useful for locating outliers or abnormalities in the data. Within a box plot:

- Outliers: Points that are outside of the "whiskers"—lines that extend beyond the box are regarded as outliers. These are distinct data points that stand out from the rest of the set.
- Box: The interquartile range (IQR), which includes the center, 50% of the data, is represented by the box itself. This core portion's spread is indicated by the box's length.
- Whiskers: The data range is shown by lines that extend from the box. Points that extend past the whiskers are regarded as possible anomalies.

Two types of graph can be shown to display the anomalies of the extended movie dataset.

The first graph is made up of several boxplots grouped in a grid format, each of which shows the distribution of a distinct numerical variable taken of the movie dataset.

Other graphs are shown with individual numerical variables showing skewness in their respective column.

Input for the R code:

```
    library(tidyr)

# Select only the numerical columns for the boxplots

    new_me <- moviextended %>%

    select(Adjusted.Gross, Budget, Gross, IMDb.Rating, MovieLens.Rating,
Overseas, `Overseas%`, Profit, `Profit..`, Runtime, US, `Gross..US`) %>%

  gather(key = "Variable", value = "Value")

# Creating the boxplot grid using facet_wrap

ggplot(data = new_me, aes(x = Variable, y = Value)) +

  geom_boxplot(outlier.color = "red", outlier.shape = 1, fill="yellow", size=1) +

  facet_wrap(~ Variable, scales = "free") +

  theme_minimal() +
theme(axis.text.y = element_text(colour = "Blue", size = 10 , angle= 90, hjust=0.5),

      axis.text.x = element_text(colour = "DarkGreen", size = 10),

      axis.title.x = element_text(colour = "Red", size = 12, face="bold"),

      axis.title.y = element_text(colour = "Red", size = 12, face="bold"),

      strip.text.x = element_text(size = 10 , face="bold", colour="Blue"),

      strip.background = element_rect(color = "black"),

      plot.title = element_text(colour = "RED", size = 15, hjust = 0.5, face = "bold"),

      axis.line = element_line(color = "black"),

      axis.ticks = element_line(color = "black")

      ) +

 labs(title = "Boxplot to identify Anomalies") + coord_flip()
```

Explanation for the R code:

1. **Loading the Library**

   library (Tidyr): The tidyr package, which offers tools for organising and restructuring data is loaded by this package.

2. **Loading and Gathering all the numerical variables from the dataset.**

- moviextended %>%: To make the code easier to read, chaining operations are done using the %>% operator (pipe).
- choose( ): A subset of the moviextended data frame's numerical columns are chosen by this function. The chosen columns consist of Modified. Runtime, US, Gross..US, Overseas, Overseas%, Budget, Gross, IMDb.Rating, MovieLens.Rating, Overseas, and Gross.
- gather( ): This function converts data from a wide format to a long format. The numerical columns that have been chosen are combined into two new columns called "Variable" that hold column names and "Value" that contain the associated numerical values.
- A variable and its associated numerical value are combined to form each row which is stored in the variable “new_me”(new movie_extended).

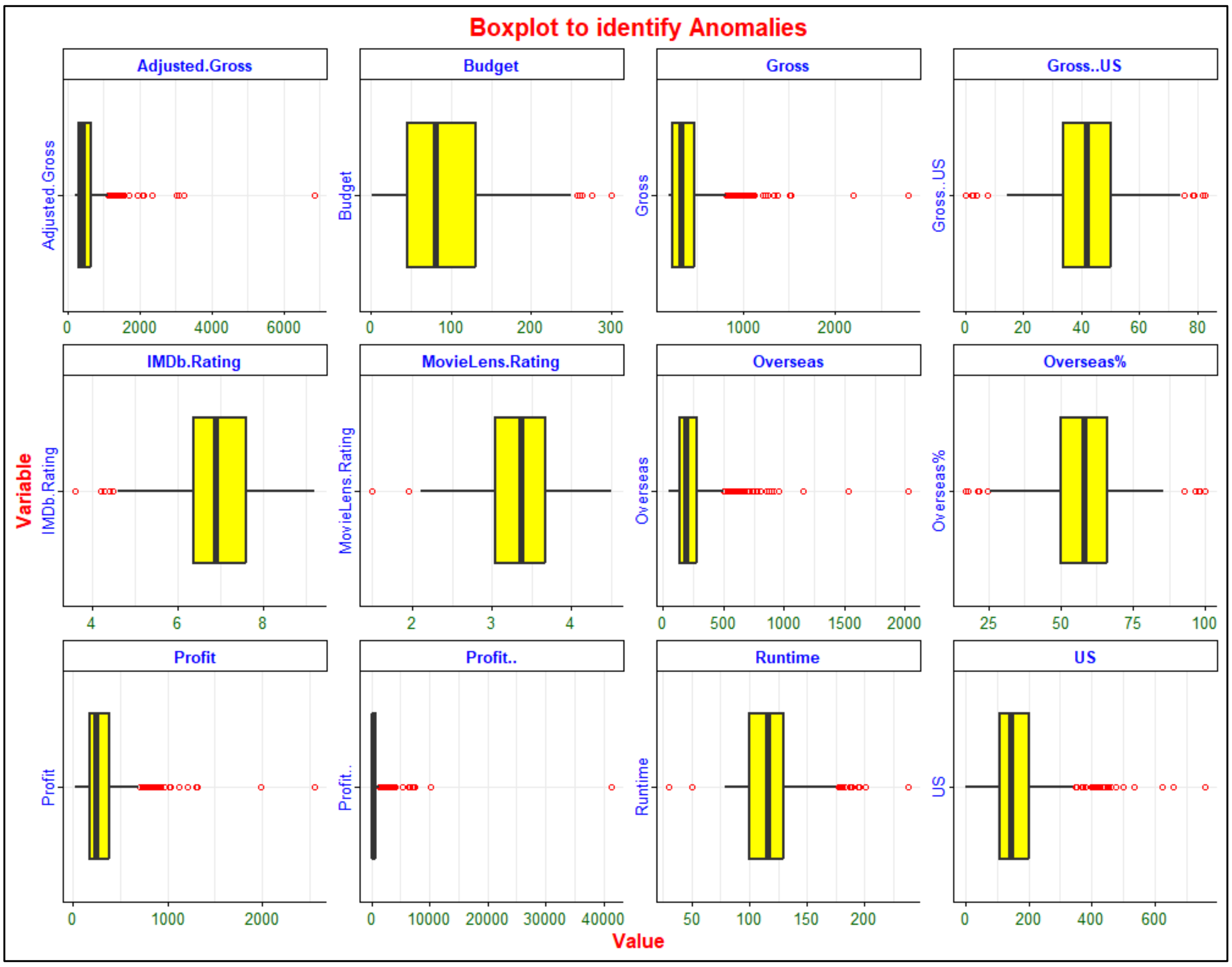
3. **Creating the Boxplot with facet wraps.**

- ggplot(data = new_me, aes(x = Variable, y = Value)): This uses the new_me data frame to start a ggplot object. It states that the y-axis displays the matching numerical values (Value) and the x-axis displays the variable names (Variable).
- geom_boxplot(outlier.color = "red", outlier.shape = 1, outlier.size = 5, size = 1) : The choices for highlighting outliers are described using this function, including the outlier colour, shape, size and the boxplot size.
- facet_wrap(~ Variable, scales = "free"): Produces a faceted boxplot grid with its facets representing distinct variables having scales free.

- theme_minimal(): Establishes a minimalistic theme for the story.
- theme(axis.text.x = element_text(hjust = 1), strip.text.x = element_text(size = 8))

  : This line modifies the look of facet labels (reduces size) and x-axis text does rotate to prevent overlapping.

Output of boxplot to identify if there is an anomaly / any anomalies in the data:



Analysis of the Graph:

The interquartile range (IQR, or the middle 50% of the data) is shown by a box in each boxplot, the median of the data is represented by a horizontal line inside the box, and the lowest and biggest values are represented by 'whiskers' that extend to 1.5*IQR of the first and third quartiles, respectively.

Dots are usually used to indicate outlier points, which are those that are not inside the whiskers.

The following are the analysis of each variable of the graph:

- **Adjusted Gross**: Displays the distribution of film industry adjusted gross receipts. Outliers, or films that have made a substantial amount more than the others, are shown above the top whisker.
- **Budget**: Shows how different movie budgets are distributed. Several outliers are located above the upper whisker, indicating that the production expenses of certain films are significantly greater.
- **Gross**: Shows the overall box office revenue from motion pictures. There are anomalies above the top whisker, much like with adjusted gross.
- **Gross US**: Shows the total revenue generated in the US market. Above the top whisker, there are outliers that represent films that have done extraordinarily well in the US.
- **IMDb Rating**: Displays the range of movie IMDb ratings. A few outliers show films with much higher and lower scores, respectively.
- **Movie Lens Rating**: Represents Movie Lens ratings. Like with IMDb ratings, there are outliers at both the top and bottom ends.

- **Overseas**: Displays how profits are distributed abroad. Above the upper whisker, there are outliers that represent highly successful foreign box office releases.
- **Overseas%**: Shows the portion of overall revenue that originates from international markets. Above and below the main distribution, there exist outliers.
- **Profit**: Shows how the films' profits were distributed. A few of the outliers are above the top whisker, indicating that certain films have made a much higher profit than others.
- **Profit %**: Seems to provide the same profit information, however on a different scale or maybe focusing on a distinct profit factor. Once more, a number of anomalies are over the upper whisker.
- **Runtime**: Denotes how movie runtimes are distributed. Both the lower and upper ends have outliers, which represent films with a typically short or extended runs.
- **US**: Displays the breakdown of the total gross profits allocated to the US. Above the top whisker, there are anomalies.

In conclusion, this grid of boxplots successfully draws attention to outliers while offering an extensive overview of the distribution of the dataset's numerous numerical variables. Either way, they might be indicators of data points that merit extra research. Thus, the usage of boxplot easily identified the abnormalities of the movie dataset that might be necessary for further research.

Find if there is any further insight you can find from this data set, marks will be awarded by the number of further useful graphs as well as their complexity.

The following bar graph shows average IMDb Rating by Genre

Input for the R code to show average IMDb Rating by Genre

```
genre_ratings <- moviextended %>%
  group_by(Genre) %>%
  summarise(Avg_IMDb_Rating = mean(IMDb.Rating, na.rm = TRUE))


#Data-Aesthetics
p <- bar_plot <- ggplot(genre_ratings, aes(x = Genre, y = Avg_IMDb_Rating, fill =
Genre))


#Geometry Layer
pg <- p +
  geom_bar(stat = "identity", position = "dodge", width = 0.7, color = "black")
```

```
#Formatting

pg <- pg +

  labs(title = "Average IMDb Ratings by Genre",

       x = "Genre",

       y = "Average IMDb Rating") +

  theme_minimal() +

  theme(

    axis.title.x = element_text(colour = "Black", size = 12),

    axis.title.y = element_text(colour = "Black", size = 12),

    axis.text.x = element_text(colour = "Darkgreen", size = 10, angle= 45, hjust=1),

    axis.text.y = element_text(colour = "DarkGreen", size = 10),

    plot.title = element_text(colour = "Black", size = 15, hjust = 0.5, face = "bold"),

    legend.title = element_text(size = 10),

    legend.text = element_text(size = 10),

    axis.line = element_line(color = "black"),

    axis.ticks = element_line(color = "black"),
    legend.position = "bottom",

    legend.justification = "center"

  )
```

Explanation of the Code:

1. **Data Preparation:**

   By classifying the dataset (moviextended) according to genre and averaging the IMDb rating, it calculates the average IMDb rating for each genre.

2. **Data-Aesthetics:**

   ggplot(): Sets the x- and y-axes to Genre and Avg_IMDb_Rating, and fills the colour according to Genre after initialising a ggplot object with the data aesthetics.

3. **Layer of Geometry:**

   geom_bar(): Using the condensed data, adds a bar overlay to the plot. The position = "dodge" guarantees that bars for various genres are presented side by side, while the stat = "identity" indicates that the heights of the bars are directly derived from the data.

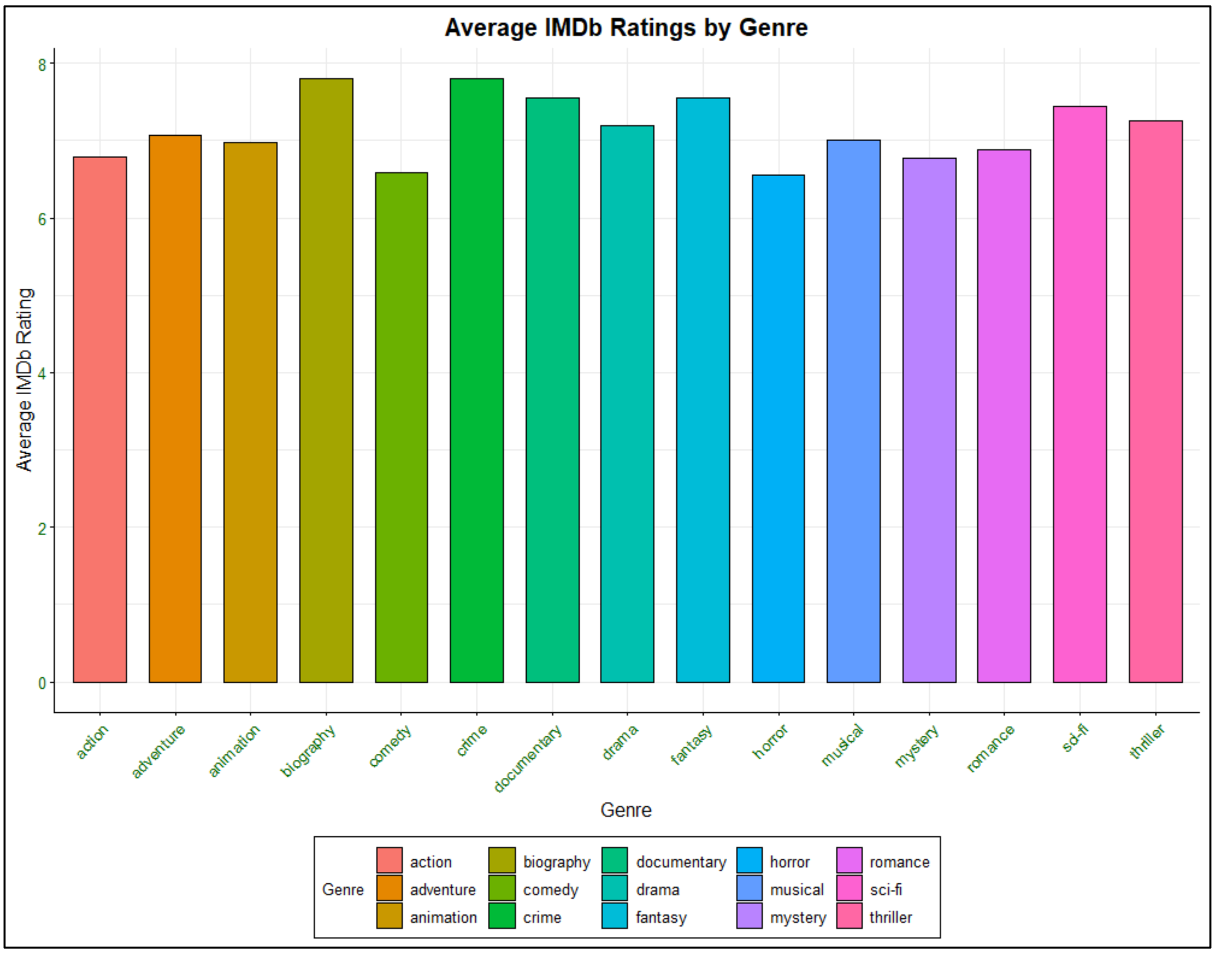
4. **Formatting:**

labs(): Configures the plot's title and axis labels.

theme_minimal(): Gives the story a simple theme.

theme(): Adds more customisation options, such as axis label colour, size, and orientation, as well as legend position.

Output for the average IMDb rating by Genre graph:



Analysis for the average IMDb rating by Genre graph:

- **Ratings Overview**: The vertical axis (y-axis) displays the average IMDb rating, ranging from 0 to 8. The genres are listed on the x-axis, the horizontal axis. According to IMDb users' ratings, every genre appears to have an average rating over 6, indicating that the overall calibre of films in these categories is generally rather good.
- **Highest and Lowest Scores**: Upon observation, the 'documentary' category has the highest average rating, just under 8, while the 'horror' genre has the lowest average rating, just over 6.
- **Comparison of Genres**: Based on average IMDb ratings, the "drama," "biography," and "documentary" genres rank highest. This might indicate that viewers usually enjoy these genres or that IMDb users tend to rate them higher for higher-quality material. However, genres like as "horror," "action," and "thriller" have lower average scores, which may indicate that the public has different expectations or has received them less well.
- **Genre Popularity vs. Rating**: It's crucial to understand that this chart just shows the average rating for each genre; it doesn't show how popular or how many films in each genre are. As a result, a genre with fewer but highly regarded films may have an average that is greater than a genre with numerous films with wider number of ratings.

Scatter Plot graph for Budget Vs Gross Earnings

Input for the R code:

```
# Scatter plot for Budget vs. Gross earnings

sp <- ggplot(moviextended, aes(x = Budget, y = Gross)) +

  geom_point(aes(color = Genre), alpha = 0.5) +

  geom_smooth(method = "lm", se = FALSE, aes(colour=Genre)) +

  labs(title = "Budget vs. Gross Earnings",

       x = "Budget",

       y = "Gross Earnings") +

  theme_minimal() +

  theme(

    axis.title.x = element_text(colour = "Black", size = 12),

    axis.title.y = element_text(colour = "Black", size = 12),

    axis.text.x = element_text(colour = "DarkGreen", size = 8),

    axis.text.y = element_text(colour = "DarkGreen", size = 8),

    plot.title = element_text(colour = "Black", size = 15, hjust = 0.5, face = "bold"),

    legend.title = element_text(size = 10),
legend.text = element_text(size = 10),

    legend.justification = c(1, 1),

    axis.line = element_line(color = "black"),

    axis.ticks = element_line(color = "black"),

    legend.box.background = element_rect(color = "Black"),

    strip.text.x = element_text(size = 9 , face="bold", colour="Brown"),

    strip.background = element_rect(color = "black")

  ) +  facet_wrap(~toupper(Genre))
```

Explanation of the R code:

1. **Data-Aesthetics**:

ggplot(): Using the moviextended dataset, initialises a ggplot object with the data aesthetics, defining the x- and y-axes as Budget and Gross, respectively.

2. Layer of Geometry:

The ggplot object gains a scatter plot layer with each point representing a movie thanks to the geom_point() function. To improve visibility of overlapping points, transparency is adjusted to 0.5 (alpha = 0.5) and points are coloured according to Genre (colour = Genre).

geom_smooth(): Adds a linear regression line to the plot (method = "lm"), with the line colour also distinguished by Genre (aes(colour=Genre)).

3. Formatting:

labs(): Configures the plot's title and axis labels.

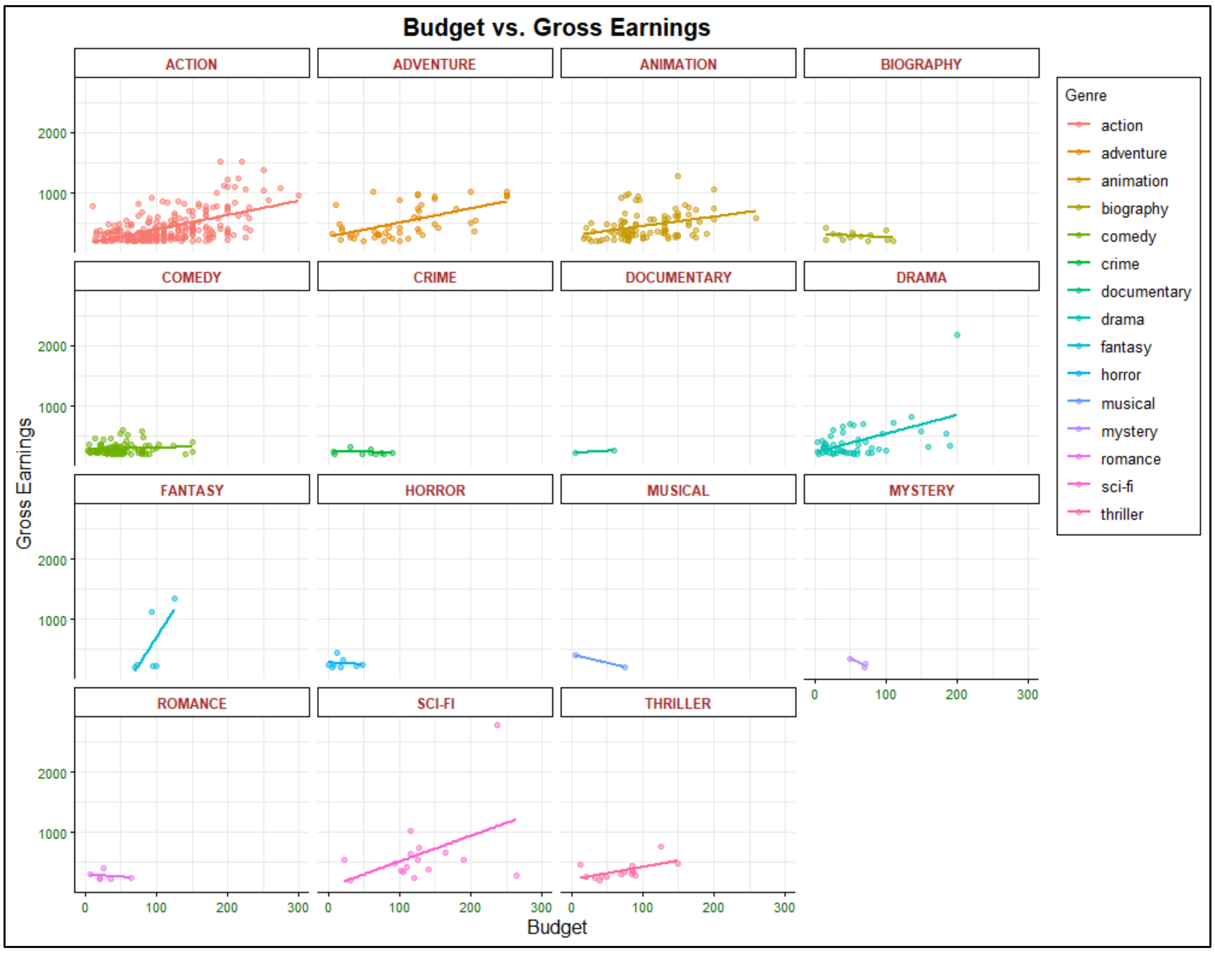
theme_minimal(): Gives the story a simple theme.

theme(): Further alters the appearance by changing the axis labels' colour, size, and orientation as well as the legend's placement and other visual components.

4. Faceting

facet_wrap(~toupper(Genre)): Uses toupper() to convert genre names to uppercase before splitting the plot into facets according to the Genre variable. Each genre's scatter plot is produced in this way.

Output for the Scatter Plot:



Analysis of the Graph:

The graph displays a scatter plot matrix with the headline "Budget vs. Gross Earnings," comparing the correlation between film budgets and gross earnings for various genres. A genre is represented by each cell in the matrix, and each dot plots the budget against the gross receipts of films in that genre.

- **Distribution of Genres**: As shown at the beginning of each plot, a distinct genre is the focus of each plot. The caption on the right illustrates which genres the colours and forms of the dots relate to.

- **Trend Observation**: There is a discernible pattern in some genres (e.g., Action, Adventure, Sci-Fi) where larger budgets are linked to greater gross revenues. Still, not every genre exhibits this tendency.
- **Outliers**: Compared to the other data points in the genre, certain genres have outliers with exceptionally high budgets or revenues.
- **Data Density**: Each genre has a different density of data points; the more data points in a given genre indicate that more films in that genre have been made and are being monitored.
- **Budget vs. Earnings connection**: The degree of genre-specific variation exists in the connection between budget and earnings. The Sci-Fi and Action genres, for instance, exhibit a stronger positive association, suggesting that films in these categories often have larger budgets and make more money.
- **Diversity**: There is substantial diversity within each genre in terms of the link between budget and earnings, demonstrating that a bigger budget does not necessarily ensure better earnings.

Reflective Piece

Embarking the journey in learning R programming has been a rewarding and demanding journey. I find learning the R programming language to be a really interesting and new experience that I have started. Discovering a language created especially for data analysis and statistical computing has opened up new opportunities. R is unique, particularly when it comes to data analysis because of its ability to express data graphically. R gives you the ability to visually examine and comprehend data, in contrast to SQL, where data is mostly observed in code. This graphic component facilitates understanding and makes it easier to see trends, patterns, and outliers. In the field of data analysis, the shift from simple coding to visual representation is innovative.

The most fascinating thing about data analysis, especially when using R, is how versatile ggplot graphs can be. An easily understood and adaptable framework for making a variety of visualisations is offered by the ggplot2 software. ggplot makes it easier to communicate complicated ideas visually, from simple scatter plots to elaborate graphs like box plots, bar graphs, and linear regression.

It's true that learning about all these graph kinds seemed overwhelming at first. Trying to grasp the complex details of each graph, understanding whether to utilise numerical or categorical variables, and navigating through a variety of possibilities was challenging. However, I am convinced that experience will greatly facilitate the analysis process, especially when combined with an in-depth knowledge .

Presenting the relationship between two variables is one area where I still struggle. Making meaningful connections between various parts is still an ability I want to develop via research and practice. I know there is room for improvement in this area, and I'm determined to do so.

Even though it seemed overwhelming at first, learning more about the topic has shown its underlying enticement. I've grown interested in data analysis because of its ability to reveal hidden insights and direct decision-making. My areas of concentration now include the complex characteristics of each graph, my comprehension of statistical ideas, and my capacity to convert unprocessed data into meaningful insight in form of graph.

An attempt was made to demonstrate how thorough and insightful the movie data analysis was. While I am aware that additional information still has to be obtained, I still made an effort to offer insightful information. My learning was reinforced by the project, which also pointed me areas in which I may improve.

Specifically, the Movie R Us initiative pushed me to integrate theory into practice by acting as an encouragement for practical application. Despite the project's high learning curve, its wide depth allowed me to use analytical tools and push limits. I find that learning the complexities of R and data analysis is both academically and professionally fascinating since I am driven to uncover increasingly challenging trends. The pursuit of mastery is ongoing, with every obstacle offering a chance to learn more about the intriguing data landscape.

In summary, learning R and data analysis has been an amazing learning curve. It has been a process of constant learning and development, from the first confusion to realising the innate interest in finding patterns in data. The difficulties faced are seen as stepping stones rather than as barriers to a deeper comprehension of the fascinating world of Data Analaysis.

Portfolio Component 4

Evaluating Big Data Technologies for Movie R U's : A Database Solution Report.

Introduction:

This research aims to give an overview of potential Big Data technologies that may be used to build a reliable and effective database system for Movies R Us. The firm wants to assess movie success and popularity on demand by using real-time, structured and unstructured data analysis. In order to satisfy the company's future demands, the report proposes three appropriate visual tools and methodologies that examines several Big Data analysis and visualisation techniques that might impact decision-making (Singh & Singh, 2012).

Big Data technologies are being used by enterprises in the age of digital transformation to obtain insightful data. The purpose for Movies R U's is to create a database system that can efficiently manage both structured and unstructured data, allowing for real-time analytics to improve decision-making.

Information about the Big Data Technologies:

2.1 Apache Hadoop

Overview:

The open-source Apache Hadoop framework is made for processing and storing massive datasets in a distributed manner. By offering a fault-tolerant and scalable solution, it enables enterprises to manage enormous amounts of data across computer clusters.

Suitability:

Processing and analysing large volumes of unstructured or semi-structured data is a specialisation of Hadoop. It is the best option for applications working with large and diverse information since it performs well in situations where standard relational database management systems could falter.

Advantages:

- Scalability: Organisations may expand the number of nodes in the Hadoop cluster as data quantities rise, guaranteeing that the system can manage growing data volumes without noticeably degrading performance.

- Fault-Tolerance When a node in a Hadoop cluster fails, the workload is automatically redistributed to other nodes in the cluster to keep processing going. This resilience helps the system be dependable while processing massive amounts of data.

- Cost-effectiveness: Companies may employ commodity hardware due to its scalability, which lowers infrastructure expenses as compared to traditional alternatives.

2.2 Apache Spark

Overview:

A robust, open-source distributed computing platform called Apache Spark is made for quick and adaptable handling of massive amounts of data. It offers an alternative to the conventional MapReduce processing paradigm and is renowned for its in-memory data processing capabilities, which lead to noticeably quicker data processing rates (amazon.com, 2022).

Suitability:

Spark is a well suitable for applications that need to process data quickly and with minimal latency. Because of its equal effectiveness for batch and stream processing, it enables businesses to manage a variety of workloads under one cohesive framework.

Advantages:

- Speed: Spark's speed is one of its biggest benefits. It does not require writing intermediate results to disc after each processing step since it processes data in-memory.
- Simplicity: Spark is more developer-friendly since it offers high-level APIs in Python, R, Java, Scala, and other languages. It simplifies application development by providing built-in libraries for a variety of tasks including stream processing, machine learning, graph processing, and SQL queries.
- Hadoop compatibility: Spark is made to work with Hadoop, allowing businesses to use Spark on pre-existing Hadoop clusters. It may read data straight from Hadoop's distributed file system and make use of the distributed storage system (HDFS).

2.2 Amazon Redshift

Overview:

Amazon Web Services (AWS) provides a cloud-based data warehouse solution called Amazon Redshift, which is completely managed. Its scalable and economical design makes it possible for enterprises to query and analyse massive datasets (amazon.com, 2022).

Suitability:

Because of its data warehousing-optimized design, it is the best option for companies needing high-performance analytics and reporting features. It can handle sophisticated aggregations and searches on large volumes of structured data.

Advantages:

- Scalability: Organisations may expand their analytical skills in parallel with their data requirements because to its ability to handle datasets of any size, from gigabytes to petabytes.
- Cost-effectiveness: Because of its scalability, organisations may optimise costs by modifying resources in response to demand.

Big Data Visualisation Tools

3.1 Tableau:

Overview:

Tableau is a potent and adaptable tool for data visualisation that is well-known for its ability to produce dashboards that are shared and interactive. It offers an intuitive user interface that enables users to transform unstructured data into insightful understandings via powerful visualisations.

Advantages:

- User-Friendly: Tableau's user-friendly interface is one of its best qualities. It is simple for users to explore and produce visualisations, even if they have no prior technical experience.
- Broad Data Connectivity: Spreadsheets, databases, cloud-based, and on-premises data are just a few of the many data sources that Tableau supports (tableau.com).

3.2 Power BI:

Overview:

Microsoft's business analytics solution for sharing insights and visualising data is called Power BI. Because of its smooth integration with the Microsoft ecosystem, it is a desirable option for businesses that currently use Microsoft products.

Advantages:

- Integration with Microsoft Products: Excel, Azure, and SQL Server are just a few of the Microsoft products that Power BI is closely linked with. This makes it easier to import data and share insights between Microsoft platforms.
- Cost-Effectiveness: Power BI is affordable for small and medium-sized businesses since it provides a free version with restricted features.

3.3 D3.js:

Overview:

D3.js is a well-known JavaScript toolkit that makes interactive and dynamic data visualisations for web browsers. It offers a great degree of flexibility and customizability.

Advantages:

- High Customisability: When it comes to designing unique visualisations, D3.js offers unmatched versatility.
- Flexibility: D3.js is an open-source toolkit that is adaptable and compatible with a variety of web technologies.

Summarising the big data visualisation tools:

According to (Somani & Deka, 2017), Tableau, Power BI, and D3.js are three tools that each have unique benefits and meet diverse demands for users. The decision between them is influenced by several elements, including the users' skill levels, the degree of customisation needed, and the organization's current technology stack.

Recommendations:

A suggested set of solutions takes into account Movies R Us particular requirements for real-time analytics on both structured and unstructured data, and includes the following recommendations:

Cloud based Big Data Analytics with AWS and Azure

The usage of platforms like AWS or Azure, along with services such as Amazon Redshift and Azure Synapse Analytics, gives Movies R Us scalability, cost-effectiveness, and access to a number of managed big data services.

Overview:

Amazon Web Services (AWS)

One of the top cloud platforms is Amazon Web Services (AWS), which provides a wide range of services with a particular emphasis on powerful big data analytics capabilities. The platform is the best option for businesses looking to leverage big data because of its scalability, affordability, and wide range of managed services ecosystem.

One of AWS's greatest advantages is its unmatched scalability, which enables companies to easily add resources in response to demand and maintain peak performance even under changing workloads. Because AWS has a pay-as-you-go approach that lets businesses only pay for the resources they use, it is notably cost-effective.

For businesses like Movies R Us, where sophisticated machine learning capabilities, effective data warehousing, and complicated analytics are essential, these characteristics make AWS a flexible and all-inclusive option.

Microsoft Azure:

One well-known cloud computing platform, Microsoft Azure, offers a range of services designed for powerful big data analytics. Azure is an appealing alternative for organisations with a variety of data analytics needs due to its scalability, wide tool set, and easy connection with Microsoft products.

Businesses who currently utilise Microsoft technology may be assured of compatibility and simplicity of use because to Azure's integration with Microsoft products, such as Windows Server and SQL Server. Because of the platform's scalability, businesses can effectively adjust to shifting workloads and maintain steady performance even during periods of high analytics demand.

Azure provides dependable data warehousing solutions and machine learning services in addition to handling sophisticated analytics queries. Because of this, Microsoft Azure is a flexible platform that Movies R Us may use to meet their analytics needs.

Advantages:

- Scalability: Scalable solutions are offered by AWS and Azure, giving Movies R Us the freedom to increase or decrease resources in response to demand. In order to handle changing workloads and maintain optimal performance during periods of high analytics demand, scalability is an essential component.

- Cost-Effectiveness: Pay-as-you-go cloud-based analytics solutions save Movies R Us money up front because of this. This cost-effectiveness is necessary to guarantee optimal resource utilisation and optimise budget allocation.

- Managed Services: AWS and Azure provide a range of managed big data services for access. By utilising these managed services, Movies R Us can greatly lessen its operational burden and concentrate on using data to generate insights rather than maintaining infrastructure.

Use Cases:

- Executing Complex Analytics Queries: - Large datasets may be queried using AWS Athena and Azure Data Explorer without requiring a complex infrastructure configuration. This makes it easier to analyse large datasets at a reasonable cost and supports a variety of query types that are essential for Movies R Us analytical requirements.

- Data Warehousing: Fully managed petabyte-scale data warehouses are offered by Amazon Redshift and Azure Synapse Analytics. These services meet Movies R Us needs for all-inclusive data warehousing solutions by facilitating the effective storage, retrieval, and analysis of structured and semi-structured data.

- Machine Learning Services: - The creation, instruction, and implementation of machine learning models are the exclusive purview of AWS Sage Maker and Azure Machine Learning. When these machine learning tools are included into analytics procedures, decision-making may be improved and Movies R with advanced predictive insight.

Conclusion:

With AWS and Azure's scalability, affordability, and managed services, Movies R Us should investigate using cloud-based big data analytics. However, before making a final choice, a thorough study should be carried out, taking into account unique needs, probable integration issues, and cost considerations. Movies R Us may achieve real-time analytics and make data-driven choices by utilising machine learning technologies in conjunction with the use of Amazon Redshift or Azure Synapse Analytics. It is vital for the organisation to do a complete examination and strategic planning to guarantee a seamless and efficient transition to cloud-based big data analytics.

Justification of using Cloud based Big Data Analytics:

Movies R Us can take strategic measures for optimal performance and resource utilisation to handle Apache Spark's resource-intensive nature. First and foremost, it is important to adjust the cluster configuration to the specific needs of the organisation, taking into account factors such as memory allocation and parallelism (Suhasini & Puli, 2021). By utilising Spark's dynamic resource allocation functionality, one may reduce needless resource allocation by automatically scaling based on workload needs. Reducing the requirement for precomputation by caching and storing frequently requested data in memory improves performance.

Because of the scalability of cloud infrastructure, especially on platforms like AWS or Azure, Movies R Us may optimise costs during peak workloads by dynamically adjusting resources as needed. Ensuring workload distribution and minimising pressure on individual resources are achieved through efficient data partitioning inside Spark RDDs and Data Frames (Sharma, 2020).

Data Lake with Apache Spark

Movies R Us has a strong big data analytics solution because of the use of a Data Lake and Apache Spark, which enables real-time processing, scalability, and effective management of both structured and unstructured data.

Apache Spark:

An open-source distributed computing platform called Apache Spark is made to handle big datasets quickly. Spark is appropriate for batch and stream processing, with an emphasis on real-time data processing. Its benefits include quick data processing, user-friendliness, and Hadoop interoperability. It may, however, be resource-intensive, necessitating that businesses provide adequate memory and effective cluster configurations (Shaikh & Mohiuddin, 2019).

Data Lake:

Movies R Us can store both structured and unstructured data at any size due to a centralised repository called a data lake. It gives the organisation agility and flexibility in processing and storing data, allowing it to get insights from a variety of data sources. With its robust analytics features, Apache Spark may be easily incorporated into a Data Lake context.

Advantages:

- Scalability: Movies R Us is able to grow resources dynamically because to the combination of Apache Spark and a Data Lake. This scalability meets the organization's demand for real-time analytics on a variety of datasets and guarantees peak performance under fluctuating workloads.

- Real-time Processing: Movies R Us' objective of doing data analysis on demand is in line with Apache Spark's capacity for real-time data processing. This is enhanced by the Data Lake architecture, which offers a centralised platform for real-time data processing and storing, enabling rapid and effective analytics.

- Cost-Effective Storage: When properly constructed, data lakes provide an affordable storage option. The company may save money by storing massive amounts of data without having to pay extra by utilising cloud-based storage alternatives in the Data Lake.

Use Cases:

- Real-time Analytics: Movies R Us is able to do analytics on streaming data thanks to Apache Spark's real-time processing capabilities when combined with a Data Lake. This is especially useful for tracking the success and popularity of films in real time.

- Unified Data Storage: For both structured and unstructured data, the Data Lake offers a unified storage solution. The Data Lake can be easily accessed and processed by Apache Spark, guaranteeing a unified analytics strategy.

- Ad Hoc Analysis: Because a Data Lake is flexible, it can be used for ad hoc analysis, which involves using Apache Spark to handle data as it comes in. This is useful for investigating novel patterns and trends without pre-formulated questions.

Conclusion:

A strong big data analytics solution is gained by Movies R Us through the use of a Data Lake with Apache Spark. Scalability, real-time processing, and affordable storage are all made possible by the combination. Strategic optimisations and ongoing monitoring can be used to solve resource-intensive processing-related issues. This strategy helps the organisation achieve its real-time analytics goal and puts it in a position to make data-driven decisions in the ever-changing film business.

Justification of using Data Lake with Apache Spark:

Adopting an Apache Spark Data Lake will provide Movies R Us with a number of benefits. The combined Data Lake functions as a single repository that effectively manages both structured and unstructured data.

Movies R Us can quickly gain insights from streaming data by integrating a Data Lake with Apache Spark, enabling prompt responses to audience preferences and market trends. Ad hoc analysis flexibility is a critical function of an Apache Spark-powered Data Lake. This combination allows Movies R Us to explore the full dataset using Apache Spark's in-memory processing, permitting data scientists to unearth significant insights without being constrained to predefined queries.

A Data Lake's capacity is increased by the inclusion of Apache Spark's built-in machine learning frameworks, which allows Movies R Us to carry out sophisticated analytics and predictive modelling. The company may now optimise business strategy and generate data-driven predictions thanks to this feature. Movies R Us may employ tactical solutions like dynamic resource allocation, effective data segmentation, and cluster configuration adjustments to handle Apache Spark's resource-intensive nature. By making these optimisations, real-time data processing advantages are obtained without putting undue strain on infrastructure resources. Essentially, Movies R Us particular requirements are well served by implementing a Data Lake with Apache Spark, which provides a complete solution for unified data storage, scalability, real-time analytics, economical processing, flexibility, seamless integration, and advanced analytics (Belov & Nikulchev, 2019).

Graph Databases with recommendation engines

The deployment of Graph Databases and Recommendation Engines may provide Movies R Us with targeted and efficient solutions for handling linked data and boosting user experience through personalised suggestions.

Neo4j

One of the best graph databases is Neo4j, which is excellent for organising and querying massively linked data. It is perfect for situations where knowing connections is important since it stores data in nodes and edges that represent entities and relationships. It is fit for applications requiring effective analysis of complicated interactions and linkages between entities.

Recommendations Engines:

Recommendation engines play a crucial role in offering consumers tailored content recommendations according to their interests and actions. Content-based filtering and collaborative filtering are two well-liked methods. Collaborative filtering suggests products by taking into account the likes and dislikes of related individuals. Content-based filtering makes recommendations for products by examining their attributes and comparing them to the preferences of the consumer.

Advantages:

- Graph databases: Movies R Us can model and analyse complex relationships between things like as directors, actors, and genres thanks to Neo4j. This offers profound understanding of how movie-related data is interrelated.

- Recommendation Engines: Movies R Us may provide customers with customised suggestions based on their viewing habits, preferences, and viewing history by utilising recommendation engines. This increases user happiness and engagement.

Use Cases:

- Collaborative Filtering for User Suggestions: Use collaborative filtering techniques to make movie recommendations based on the likes and dislikes of other users. This improves the recommendation system's accuracy for movies.

- Content-Based Sorting for Tailored Recommendations: Utilise content-based filtering to make movie recommendations to consumers based on the qualities and characteristics of films they have already liked. This improves personalisation and lessens the issue of cold starts.

- Graph-Based Relationship Analysis Database: Model and analyse the links between actors, directors, genres, and user interactions in movies using Neo4j. This makes thorough connection analysis easier to do and improves decision-making (neo4j.com, 2023).

Conclusion:

Movies R Us has a strong and user-centric strategy in the cutthroat film market thanks to the deep relationship insights and effective querying capabilities of graph databases and the precision of recommendation engines. To guarantee smooth integration and best outcomes, the organisation must properly plan and carry out the deployment of these technologies.

Justification of using Graph Databases:

Movies R Us has to use Graph Databases like Neo4j and Recommendation Engines in order to fully realise how interrelated movie-related data is. Comprehensive relationship analysis is made easier using Neo4j, which makes it possible to navigate intricate relationships between entities quickly. Understanding the complex web of connections between performers, directors, and genres requires knowledge of this. Recommendation engines also improve user experience by providing tailored movie recommendations based on individual tastes and habits. Together, these technologies enable Movies R Us to offer a customised and captivating platform that enhances user pleasure and content discovery in the ever-changing film business.

Conclusion:

In conclusion , Movies R Us assessment of Big Data technology offers a thorough examination of several options for creating a reliable database system. The paper examines Amazon Redshift, Apache Spark, Apache Hadoop, and a number of visualisation tools, providing an analysis of their benefits and applicability. Using cloud-based analytics with AWS or Azure and using tools like Amazon Redshift and Azure Synapse Analytics are among the recommendations. Furthermore, it is advised to strategically combine Apache Spark with Data Lakes for real-time processing and scalability. Relationship analysis and user engagement are improved by integrating recommendation engines with graph databases like Neo4j. Every technology is supported by its own features, giving Movies R Us customised solutions for real-time processing, personalised content, and structured and unstructured data analytics.

References

amazon.com. (2022). *aws.amazon.com.* Retrieved from https://aws.amazon.com/: https://aws.amazon.com/what-is/apache-spark/

amazon.com. (2022). *https://aws.amazon.com/.* Retrieved from aws.amazon.com: https://aws.amazon.com/redshift/

Belov, V., & Nikulchev, E. (2019). Analysis of Big Data Storage Tools for Data Lakes based on Apache Hadoop Platform. *ResearchGate.* ResearchGate.

Chang, W. (2012). R Graphics Cookbook:Practical Recipes for Visualizing Data. In W. Chang, *R Graphics Cookbook: Practical Recipes for Visualizing Data* (p. 416). O'Reilly Media.

datacamp.com. (2023). *https://www.datacamp.com/.* Retrieved from www.datacamp.com: https://www.datacamp.com/cheat-sheet/ggplot2-cheat-sheet

geeksforgeeks.org. (2023). *www.geeksforgeeks.org.* Retrieved from https://www.geeksforgeeks.org/: https://www.geeksforgeeks.org/histogram-in-r-using-ggplot2/

Hadley, W. (2010). Ggplot2 - Elegant graphics for data analysis. In W. Hadley, *Ggplot2 - Elegant graphics for data analysis* (p. 212). Springer.

Healy, K. (2019). Data Visualization: A practical solution. In K. Healy, *Data Visualization: A practical solution* (p. 272). Princeton University Press.

kbroman.org. (2017). *https://kbroman.org/.* Retrieved from kbroman.org: https://kbroman.org/datacarpentry_R_2017-01-10/03-ggplot2.html#:~:text=You%20can%20use%20both%20geom_line,points%20at%20the%20data%20values.&text=This%20brings%20up%20another%20important,on%20top%20of%20the%20other

neo4j.com. (2023). *neo4j.com.* Retrieved from https://neo4j.com/: https://neo4j.com/use-cases/?utm_source=google&utm_medium=PaidSearch&utm_campaign=GDB&utm_content=EMEA-X-Engagement-GDB-Text&utm_term=when%20to%20use%20graph%20database&gad_source=1&gclid=CjwKCAiA44OtBhAOEiwAj4gpOWyyiX7IOsCs23eg2piNd0TBhRLT37YR-C0RBZ_Ns1Q

Shaikh, E., & Mohiuddin, I. (2019). Apache Spark: A Big Data Processing Engine. *2019 2nd IEEE Middle East and North Africa COMMunications Conference (MENACOMM).* Manama: IEEE.

Sharma, G. (2020). Big Data & Cloud Computing: The Roles & Relationships. *Big Data & Cloud Computing: The Roles & Relationships*.

Singh, S., & Singh, N. (2012). Big Data analytics. *IEEE.* Mumbai: https://ieeexplore.ieee.org/document/6398180.

Somani, A. K., & Deka, G. C. (2017). Big Data Analytics. In A. K. Somani, & G. C. Deka, *Big Data Analytics.* CRC Press.

Suhasini, N. S., & Puli, S. (2021). Big Data Analytics in Cloud Computing. *IEEE.* Shimla: IEEE.

tableau.com. (n.d.). *https://www.tableau.com/.* Retrieved from https://www.tableau.com/: https://www.tableau.com/en-gb/trial/tableau-software?d=7013y000002aJFmAAM&utm_campaign=Prospecting-CORE-ALL-ALL-ALL-ALL&utm_medium=Paid+Search&utm_source=Google+Search&utm_campaign_id=2017049&utm_language=EN&utm_country=UKI&adgroup=&adused=STAT&creative=&

tidyverse.org. (2023). *https://ggplot2.tidyverse.org/.* Retrieved from ggplot2.tidyverse.org: https://ggplot2.tidyverse.org/reference/#facetting tidyverse.org. (2023). *https://ggplot2.tidyverse.org/.* Retrieved from ggplot2.tidyverse.org: https://ggplot2.tidyverse.org/reference/#themes

w3schools.com. (2023). *www.w3schools.com.* Retrieved from https://www.w3schools.com: https://www.w3schools.com/r/r_vectors.asp

Wickham, H. (2010). Ggplot2:Elegant Graphics for Data Analysis. In H. Wickham, *Ggplot2 :Elegant Graphics for Data Analysis* (p. 212). Springer.

Wickham, H., & Grolemund, G. (2016). R for Data Science : Import, Tidy, Transform, Visualize, and Model Data. In H. Wickham, & G. Grolemund, *R for Data Science : Import, Tidy, Transform, Visualize, and Model Data* (p. 520). O'Reilly Media.

Word Count:

Portfolio Component-3 : 13000

Portfolio Component-4: 3145

Reflective Piece: 500

Appendix (Entire Code):

```
#Portfolio Component 3: Solving Data Science Problem
#Student ID- 2117336

library(ggplot2)
library(tidyverse)
library(scales)
library(dplyr)
library(tidyr)



#Part 1- Exploring the Dataset

#reading the dataset and storing it in the variable
movieratings <- read.csv("movie_ratings.csv")

#Viewing all the columns of the dataset
colnames(movieratings)

#Changing the columns names with the help of the vector
colnames(movieratings) <- c("Film", "Genre", "Critic.Rating", "Audience.Rating",
                          "Budget", "Year")
#Retrieve the column names
names(movieratings)

#The first 6 rows of the dataset
head(movieratings)

#The first 10 rows
head(movieratings, n=10)
movieratings[1:10,]
```

```
#First 10 rows of the first 3 columns
movieratings[1:10, 1:3]

# rows of the first 3 columns
movieratings[5:10, 1:3]

#The last 6 rows
tail(movieratings)

#Opens the data editor
edit(movieratings)

#Number of Rows
nrow(movieratings)

#Number of Columns
ncol(movieratings)

#Structure of the dataset
str(movieratings)

#Summary of the dataset
summary(movieratings)



#2. How Genre impacts the Budget of the Movie ?
#data-aesthetic layer
b <- ggplot(data=movieratings, aes(x=Genre, y=Budget, colour= Genre))
#geometry layer
g <- b + geom_boxplot(size=1.2) + geom_jitter(alpha=0.5, width = 0.2)
```

```
#formatting layer
g <- g +  xlab("Genre") +
  ylab("Budget (in millions$)") +
  ggtitle("Distribution of Movie Budget by Genre") +
  theme_minimal() +
  theme(
    axis.title.x = element_text(colour="darkblue", size=12, face= "bold"),
    axis.title.y = element_text(colour="darkblue", size=12, face= "bold"),
    axis.text.x = element_text(colour="darkgreen", size=10),
    axis.text.y = element_text(colour="darkgreen", size=10),
    plot.title = element_text(colour= "Black", size=15, hjust=0.5, face= "bold"),
    panel.background = element_rect(fill = "white", colour = "white"),
    legend.position = c(1,1),
    legend.justification = c(1,1),
    legend.box.background = element_rect(color = "black"),
    axis.line = element_line(color = "black"),
    axis.ticks = element_line(color = "black"))



#3. Is there any relation between the critic rating and the budget?

#Relation between critics and budget
#data-aes layer
c <- ggplot(data=movieratings, aes(x= Budget, y=Critic.Rating,
                                  colour= Genre,
                                  size= Budget))
#geometry and formatting
c +
  geom_point(alpha=0.7) +
  xlab("Budget (in millions)") +
  ylab("Critic Rating") +
  ggtitle("Critic Ratings VS Budget") +
  theme(
    axis.title.x = element_text(colour="Black", size=12, face= "bold"),
    axis.title.y = element_text(colour="Black", size=12, face= "bold"),
    axis.text.x = element_text(colour="Red", size=10),
    axis.text.y = element_text(colour="Red", size=10),
    plot.title = element_text(colour= "Black", size=15, hjust=0.5, face= "bold"))

#Relation between critics and budget with linear regression
c +
  geom_point(alpha=0.7) +
  geom_smooth(method=lm, se=FALSE, colour="Red") +
  xlab("Budget (in millions)") +
  ylab("Critic Rating") +
  ggtitle("Critic Ratings VS Budget") +
  theme(
    axis.title.x = element_text(colour="Black", size=12, face= "bold"),
    axis.title.y = element_text(colour="Black", size=12, face= "bold"),
    axis.text.x = element_text(colour="Red", size=10),
    axis.text.y = element_text(colour="Red", size=10),
    plot.title = element_text(colour= "Black", size=15, hjust=0.5, face= "bold"))



#4. Is there relationship between Audience Rating and Budget?
#Audience Rating VS Budget
q <- ggplot(data=movieratings, aes(x= Budget, y=Audience.Rating,
                                  colour= Genre,
                                  size= Budget))
q +
  geom_point(alpha=0.7) +
  xlab("Budget (in millions)") +
  ylab("Audience Rating") +
  ggtitle("Audience Ratings VS Budget") +
  theme(
    axis.title.x = element_text(colour="Black", size=12, face= "bold"),
    axis.title.y = element_text(colour="Black", size=12, face= "bold"),
    axis.text.x = element_text(colour="Red", size=10),
    axis.text.y = element_text(colour="Red", size=10),
    plot.title = element_text(colour= "Black", size=15, hjust=0.5, face= "bold"))

#Audience rating vs budget with linear regression
q +
  geom_point(alpha=0.7) +
  geom_smooth(method=lm, se=FALSE, colour="Red") +
  xlab("Budget (in millions)") +
  ylab("Audience Rating") +
  ggtitle("Audience Ratings VS Budget") +
  theme(
    axis.title.x = element_text(colour="Black", size=12, face= "bold"),
    axis.title.y = element_text(colour="Black", size=12, face= "bold"),
    axis.text.x = element_text(colour="Red", size=10),
    axis.text.y = element_text(colour="Red", size=10),
    plot.title = element_text(colour= "Black", size=15, hjust=0.5, face= "bold"))




#5. Show corelation between critic ratings vs audience ratings by genre through out the years
p <- ggplot(data = movieratings, aes(x = Year, colour = Genre)) +
  geom_point(aes(y = Audience.Rating), size = 2, shape = 1, alpha = 0.7) +
  geom_point(aes(y = Critic.Rating), size = 2, shape = 5, alpha = 0.7) +
  geom_line(aes(y = Audience.Rating, linetype = "Audience"), size = 1) +
  geom_line(aes(y = Critic.Rating, linetype = "Critic"), size = 1) +
  facet_grid(Genre ~ ., scales = "free_y") +  # Separate box for each year
  xlab("Year") +
  ylab("Audience Ratings VS Critic Ratings") +
  ggtitle("Co-relation between Audience and Critic Ratings throughout years by Genre") +
  theme_minimal() +
  theme(
    axis.title.x = element_text(colour = "DarkGreen", size = 9, face = "bold"),
    axis.title.y = element_text(colour = "DarkGreen", size = 9, face = "bold"),
    axis.text.x = element_text(colour = "black", size = 7),
    axis.text.y = element_text(colour = "black", size = 7),
    plot.title = element_text(colour = "Black", size = 10, hjust = 0.5, face = "bold"),
    legend.position = "bottom",
    legend.box = "vertical",
    legend.box.background = element_rect(color = "black"),
    axis.line = element_line(color = "black"),
    axis.ticks = element_line(color = "black")
  ) +
  scale_linetype_manual(
    values = c("solid", "dotted"),
    labels = c("Audience", "Critic"),
    name = "Line Type",
    guide = "legend"
  )



#6. Create a graph to show the number of films from the dataset categorised by Genre.
#data-aes layer
s <- ggplot(movieratings, aes(x=Genre, fill= Genre))
#geometry & formatting layer
t <- s +
  geom_bar(color = "Black", position = "stack") +
  geom_text(
    stat = "count",
    aes(label = after_stat(count), y = after_stat(count)),
    vjust = -0.5,
    color = "Blue",
    size = 3.5
  ) +
  labs(
    x = "Genre",
    y = "Number of Films",
    title = "Number of Movies categorized by Genre"
  ) +
  theme_light() +
  theme(
    axis.title = element_text(colour = "Black", size = 12, face = "bold"),
    axis.text = element_text(colour = "DarkGreen", size = 10),
    plot.title = element_text(colour = "Black", size = 15, hjust = 0.5, face = "bold"),
    legend.position = c(0.95, 0.95),
    legend.justification = c(1, 1),
    legend.box.background = element_rect(color = "Black"),
    axis.ticks = element_line(color = "DarkGreen")
  )

#Part-2 Advanced Analytics        47
#Importing the Movie Extended Dataset
moviextended <- read_csv("movie_extended.csv")

#Converting the Column Names for simplicity through vectors
colnames(moviextended) <- c("Day.of.Week", "Director", "Genre",
"Movie.Title", "Release.Date", "Studio",
"Adjusted.Gross", "Budget", "Gross",
"IMDb.Rating", "MovieLens.Rating", "Overseas",
"Overseas%", "Profit", "Profit..",
"Runtime", "US", "Gross..US")


#Q-1 They give you the following graph image as the R code is not found and they would like to remove the spelling mistake in the Graph Title and rename the Graph Title “Gross Percentage By Genre”. You need to recreate the graph by writing R code. You must use the Grammar of Graphics to recreate the following graph. You must also explain your code and display the output at each step


#Filtering the Genre
filter1 <-
  moviextended$Genre == "action" |
  moviextended$Genre == "adventure" |
  moviextended$Genre == "animation" |
  moviextended$Genre == "comedy" |
  moviextended$Genre == "drama"

#Allowing specific studios
filter2 <- moviextended$Studio %in%
  c("Buena Vista Studios", "WB", "Fox", "Universal", "Paramount Pictures", "Sony")

#Combining filter1 and filter2
newmoviextended <- moviextended[filter1 & filter2,]

#printing the filter
print(newmoviextended)

#adding data-layer
p <- ggplot(data=newmoviextended, aes(x=Genre, y=Gross..US))

#adding geometries
g <- p + geom_jitter(aes(size=Budget, colour=Studio)) +
  geom_boxplot(alpha=0.7, outlier.color = NA)

#adding formatting
g <- g + xlab("Genre") +
  ylab("Gross % US") +
  ggtitle("Domestic Gross % by Genre") +
  theme(
    axis.title.x = element_text(colour="Blue", size=15),
    axis.title.y = element_text(colour="Blue", size=15),
    axis.text.x = element_text(colour="Black", size=10),
    axis.text.y = element_text(colour="Black", size=10),
    plot.title = element_text(colour="Black", size=20, hjust = 0.5, face = "bold"),
    legend.title=element_text(size=10),
    legend.text = element_text(size=10),
    text = element_text(family="Comic Sans MS"))

#adding label size
g$labels$size <- "Budget $M"



#2.Write R code to find the trend of the Day of the week that most/least movies were released.

#Graph 1 with Bar Plot
#Counting the number of movies released in each day of the week
day_of_week_counts <- table(moviextended$Day.of.Week)

# Find the day with the most and least movie releases
most_released_day <- names(which.max(day_of_week_counts))
least_released_day <- names(which.min(day_of_week_counts))

#data/aesthetic layer
p <- ggplot(data = moviextended, aes(x = as.factor(Day.of.Week),fill = Day.of.Week ))

#geometry layer
k <- p +
  geom_bar( color = "black")
```

```
#formatting layer
k <- k +
  labs(title = "Movie Releases by Day of the Week",
       subtitle = paste("Most Released Day:", most_released_day, " | Least Released Day:",
least_released_day),
       x = "Day of the Week",
       y = "Number of Movies Released") +
  theme_minimal() +
  theme(
    axis.title.x = element_text(colour="Blue", size=12),
    axis.title.y = element_text(colour="Blue", size=12),
    axis.text.x = element_text(colour="DarkGreen", size=10),
    axis.text.y = element_text(colour="DarkGreen", size=10),
    plot.title = element_text(colour="Black", size=15, hjust = 0.5, face = "bold"),
    plot.subtitle = element_text(colour="Blue", size=12, hjust = 0.5),
    legend.title=element_text(size=10),
    legend.text = element_text(size=10),
    legend.justification = c(1, 1),
    axis.line = element_line(color = "black"),
    axis.ticks = element_line(color = "black"),
    legend.box.background = element_rect(color = "Black"))



#Graph-2 with Geom Line
#Counting the number of movies released in each day of the week
day_of_week_counts <- table(moviextended$Day.of.Week)
print(day_of_week_counts)

#creating a dataframe
movie_df <- data.frame(Day.of.Week = names(day_of_week_counts),
                       Count = as.numeric(day_of_week_counts))
```

```
#removing any values with NA column
movie_df <- na.omit(movie_df)

#Converting the 'Day of Week' column to a factor for proper ordering
movie_df$Day.of.Week <- factor(movie_df$Day.of.Week,
                         levels = c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"))

#data-layer
r <- ggplot(data = movie_df, aes(x = as.factor(Day.of.Week), y= Count, group = 1))

#adding geometry
m <- r +
  geom_line(color = "Red", size=1) +
  geom_point(color = "Blue", size=5, alpha=0.7)

#adding formatting
m <- m +
  labs(title = "Movie Releases by Day of the Week",
       subtitle = paste("Most Released Day:", most_released_day, " | Least Released Day:",
least_released_day),
       x = "Day of the Week",
       y = "Number of Movies Released") +
  theme_light() +
  theme(
    axis.title.x = element_text(colour="Blue", size=12),
    axis.title.y = element_text(colour="Blue", size=12),
    axis.text.x = element_text(colour="Black", size=10),
    axis.text.y = element_text(colour="Black", size=10),
    plot.title = element_text(colour="Black", size=15, hjust = 0.5, face = "bold"),
    plot.subtitle = element_text(colour="Blue", size=12, hjust = 0.5),
    legend.title=element_text(size=10),
    legend.text = element_text(size=10))
```

```
#3. Identify  Identify if the profit of a movie depends on any of the features in this data set i.e. genre,
director, profit etc

#Graph-1 Profit VS Directors
#Threshold value 5 means including only those directors who directed more than 5 movies
threshold <- 5

# Filter directors with movies above the threshold
new_d <- moviextended %>%
  group_by(Director) %>%
  filter(n() >= threshold) %>%
  mutate(Director_reorder = reorder(Director, Profit, FUN = mean))

# Data-layer
pd <- ggplot(new_d, aes(x = Director_reorder, y = Profit))

# Geometry layer
pd <- pd +
  stat_summary(fun = mean, geom = "bar", colour = "Black", aes(fill= Director))

# Formatting layer
pd <- pd +
  labs(
    title = "Profit of the Movie based upon Directors",
    x = "Directors",
    y = "Profit of the Movie"
  ) +
  theme_minimal() +
  theme(
    axis.title.x = element_text(colour = "Black", size = 12, face = "bold"),
    axis.title.y = element_text(colour = "Black", size = 12, face = "bold"),
    axis.text.x = element_text(colour = "DarkGreen", size = 10, hjust = 1),
    axis.text.y = element_text(colour = "DarkGreen", size = 10),
    plot.title = element_text(colour = "Black", size = 15, hjust = 0.5, face = "bold"),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    axis.line = element_line(color = "black"),
    axis.ticks = element_line(color = "black"),
    legend.box.background = element_rect(color = "Black"),
    legend.position = "bottom",
    legend.justification = "center") +
  coord_flip()



#Graph-2 Profit VS IMBD
#data-aesthetic layer
a <- ggplot(data = moviextended, aes(x = IMDb.Rating, y = Profit, group = 1))
#geometry
i <- a +
  geom_point(alpha=0.3, aes(group=1, colour=Genre, size=Budget)) +
  geom_smooth(method = "lm", se = FALSE, aes(group=1))
#adding formatting
i <- i +
  labs(
  title = "Profit vs IMDb Rating",
  x = "IMDb Rating",
  y = "Profit") +
  theme_minimal() +
  theme(
    axis.title.x = element_text(colour = "Black", size = 12),
    axis.title.y = element_text(colour = "Black", size = 12),
    axis.text.x = element_text(colour = "Blue", size = 10),
    axis.text.y = element_text(colour = "Blue", size = 10),
    plot.title = element_text(colour = "Black", size = 15, hjust = 0.5, face = "bold"),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.justification = c(1, 1),
    axis.line = element_line(color = "black"),
    axis.ticks = element_line(color = "black"),
    legend.box.background = element_rect(color = "Black")
  )



#3. Profit VS Overseas
#If profit depends upon Overseas
a <- ggplot(data = moviextended, aes(x = Overseas, y = Profit, group = 1))
#geometry
o <- a +
  geom_point(alpha=0.3, aes(group=1, colour=Genre, size=Budget)) +
  geom_smooth(method = "lm", se = FALSE, aes(group=1))
#adding formatting
o <- o +
  labs(
    title = "Profit vs Overseas",
    x = "Overseas",
    y = "Profit") +
  theme_minimal() +
  theme(
    axis.title.x = element_text(colour = "Black", size = 12),
    axis.title.y = element_text(colour = "Black", size = 12),
    axis.text.x = element_text(colour = "Blue", size = 10),
    axis.text.y = element_text(colour = "Blue", size = 10),
    plot.title = element_text(colour = "Black", size = 15, hjust = 0.5, face = "bold"),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.justification = c(1, 1),
    axis.line = element_line(color = "black"),
    axis.ticks = element_line(color = "black"),
    legend.box.background = element_rect(color = "Black")
  )



#Graph4- Profit VS MovieLens Rating
a <- ggplot(data = moviextended, aes(x = MovieLens.Rating , y = Profit, group = 1))
#geometry
mr <- a +
  geom_point(alpha=0.3, aes(group=1, colour=Genre, size=Budget)) +
  geom_smooth(method = "lm", se = FALSE, aes(group=1))
#adding formatting
mr <- mr +
  labs(
    title = "Profit vs Movie Lens Rating",
    x = "Movie Lens Rating",
    y = "Profit") +
  theme_minimal() +
  theme(
    axis.title.x = element_text(colour = "Black", size = 12),
    axis.title.y = element_text(colour = "Black", size = 12),
    axis.text.x = element_text(colour = "Blue", size = 10),
    axis.text.y = element_text(colour = "Blue", size = 10),
    plot.title = element_text(colour = "Black", size = 15, hjust = 0.5, face = "bold"),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.justification = c(1, 1),
    axis.line = element_line(color = "black"),
    axis.ticks = element_line(color = "black"),
    legend.box.background = element_rect(color = "Black")
  )




#Graph 5- Profit VS Budget
a <- ggplot(data = moviextended, aes(x = Budget , y = Profit, group = 1))
#geometry
bb <- a +
  geom_point(alpha=0.3, aes(group=1)) +
  geom_smooth(method = "lm", se = FALSE, aes(group=1))
#adding formatting
bb <- bb +
  labs(
    title = "Profit depending upon Budget",
    x = "Budget",
    y = "Profit") +
  theme_minimal() +
  theme(
    axis.title.x = element_text(colour = "Black", size = 12),
    axis.title.y = element_text(colour = "Black", size = 12),
    axis.text.x = element_text(colour = "Blue", size = 10),
    axis.text.y = element_text(colour = "Blue", size = 10),
    plot.title = element_text(colour = "Black", size = 15, hjust = 0.5, face = "bold"),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.justification = c(1, 1),
    axis.line = element_line(color = "black"),
    axis.ticks = element_line(color = "black"),
    legend.box.background = element_rect(color = "Black")
  )
```

```
#Graph 6- Profit VS Profit%
a <- ggplot(data = moviextended, aes(x = Profit.. , y = Profit, group = 1))
#geometry
pp <- a +
  geom_point(alpha=0.3, aes(group=1)) +
  geom_smooth(method = "lm", se = FALSE, aes(group=1))
#adding formatting
pp <- pp +
  labs(
    title = "Profit depending upon Profit%",
    x = "Profit%",
    y = "Profit") +
  theme_minimal() +
  theme(
    axis.title.x = element_text(colour = "Black", size = 12),
    axis.title.y = element_text(colour = "Black", size = 12),
    axis.text.x = element_text(colour = "Blue", size = 10),
    axis.text.y = element_text(colour = "Blue", size = 10),
    plot.title = element_text(colour = "Black", size = 15, hjust = 0.5, face = "bold"),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.justification = c(1, 1),
    axis.line = element_line(color = "black"),
    axis.ticks = element_line(color = "black"),
    legend.box.background = element_rect(color = "Black")
  )




#Graph 7- Profit VS Studio
```

```
# Filter out studios with profit below 500
new_s <- moviextended %>%
  filter(Profit >= 500)

#data-aes
a <- ggplot(data = new_s , aes(x = Studio, y = Profit, fill = Studio))

#geometry layer
st <- a +
  geom_bar(stat = "identity", color = "black", width=0.4)

#formatting
st <- st +
  labs(
    title = "Total Profits by Studio",
    x = "Studio",
    y = "Total Profit"
  ) +
  theme_minimal() +
  theme(
    axis.title.x = element_text(colour = "Black", size = 12, face="bold"),
    axis.title.y = element_text(colour = "Black", size = 12, face="bold"),
    axis.text.x = element_text(colour = "DarkGreen", size = 10, angle= 0, hjust=1),
    axis.text.y = element_text(colour = "DarkGreen", size = 10, angle= 0, hjust=1),
    plot.title = element_text(colour = "Black", size = 15, hjust = 0.5, face = "bold"),
    axis.line = element_line(color = "black"),
    axis.ticks = element_line(color = "black"),
    legend.title = element_text(size = 8),
    legend.position = "bottom",
    legend.box = "vertical",
    legend.box.background = element_rect(color = "Black")) +
    coord_flip()
```

```
#count function to cross check with the graph
studio_counts <- new_s %>%
  count(Studio, wt = Profit, sort = TRUE)

#Graph 8- If Profit depends upon the Genre
new_g <- moviextended %>%
  filter(Profit >= 500)

#data-aes
gg <- ggplot(data = new_g , aes(x = Genre, y = Profit, fill = Genre))

#geometry layer
gg <- gg +
  geom_bar(stat = "identity", color = "black", width=0.4)

#formatting
gg <- gg +
  labs(
    title = "Total Profits by Genre",
    x = "Genre",
    y = "Total Profit"
  ) +
  theme_minimal() +
  theme(
    axis.title.x = element_text(colour = "Black", size = 12, face="bold"),
    axis.title.y = element_text(colour = "Black", size = 12, face="bold"),
    axis.text.x = element_text(colour = "DarkGreen", size = 10, angle= 0, hjust=1),
    axis.text.y = element_text(colour = "DarkGreen", size = 10, angle= 0, hjust=1),
    plot.title = element_text(colour = "Black", size = 15, hjust = 0.5, face = "bold"),
    axis.line = element_line(color = "black"),
    axis.ticks = element_line(color = "black"),
   legend.title = element_text(size = 8),
   legend.position = "bottom",
   legend.box = "vertical",
   legend.box.background = element_rect(color = "Black")) +
   coord_flip()

#Count function to count Genre
genre_counts <- filtered_data %>%
  count(Genre, wt = Profit, sort = TRUE)



#4. Use ggplot and boxplot to identify if there is an anomaly / any anomalies in the data?



# Select only the numerical columns for the boxplots
new_me <- moviextended %>%
  select(Adjusted.Gross, Budget, Gross, IMDb.Rating, MovieLens.Rating,
         Overseas, `Overseas%`, Profit, `Profit..`, Runtime, US, `Gross..US`) %>%
  gather(key = "Variable", value = "Value")



# Creating the boxplot grid using facet_wrap
p <- ggplot(data = new_me, aes(x = Variable, y = Value)) +
  geom_boxplot(outlier.color = "red", outlier.shape = 1, fill="yellow", size=1) +
  facet_wrap(~ Variable, scales = "free") +
  theme_minimal() +
  theme(axis.text.y = element_text(colour = "Blue", size = 10 , angle= 90, hjust=0.5),
        axis.text.x = element_text(colour = "DarkGreen", size = 10),
        axis.title.x = element_text(colour = "Red", size = 12, face="bold"),
        axis.title.y = element_text(colour = "Red", size = 12, face="bold"),
        strip.text.x = element_text(size = 10 , face="bold", colour="Blue"),
        strip.background = element_rect(color = "black"),
        plot.title = element_text(colour = "RED", size = 15, hjust = 0.5, face = "bold"),
        axis.line = element_line(color = "black"),
        axis.ticks = element_line(color = "black")
        ) +
  labs(title = "Boxplot to identify Anomalies") + coord_flip()



#Boxplot for Budget
ggplot(data = moviextended, aes(x="", y = Budget)) +
  geom_boxplot(outlier.color = "red", outlier.shape = 1, outlier.size= 5, size=1) +
  labs(title = "Boxplot for Budget Anomalies",
       y = "Budget ($millions)",
       x = "") +
  theme(
  axis.title.x = element_text(colour = "Blue", size = 12, face="bold"),
  axis.text.x = element_text(colour = "DarkGreen", size = 10, angle= 0, hjust=1),
  plot.title = element_text(colour = "Black", size = 15, hjust = 0.5, face = "bold"),
  axis.line = element_line(color = "black"),
  axis.ticks = element_line(color = "black"),
  legend.title = element_text(size = 8)) +
  coord_flip()

#Boxplot for Adjusted Gross
ggplot(data = moviextended, aes(x="", y = Adjusted.Gross)) +
  geom_boxplot(outlier.color = "red", outlier.shape = 1, outlier.size= 5, size=1) +
  labs(title = "Boxplot for  Adjusted Gross",
       y = "Adjusted Gross",
       x = "") +
  theme(
    axis.title.x = element_text(colour = "Blue", size = 12, face="bold"),
    axis.title.y = element_text(colour = "Blue", size = 12, face="bold"),
    axis.text.x = element_text(colour = "DarkGreen", size = 10, angle= 0, hjust=1),
    plot.title = element_text(colour = "Black", size = 15, hjust = 0.5, face = "bold"),
    axis.line = element_line(color = "black"),
    axis.ticks = element_line(color = "black"),
    legend.title = element_text(size = 8))  + coord_flip()

#Boxplot for Gross
ggplot(data = moviextended, aes(x="", y = Gross)) +
  geom_boxplot(outlier.color = "red", outlier.shape = 1, outlier.size= 5, size=1) +
  labs(title = "Boxplot for Gross",
       y = "Gross",
       x = "Value") +
  theme(
    axis.title.x = element_text(colour = "Blue", size = 12, face="bold"),
    axis.title.y = element_text(colour = "Blue", size = 12, face="bold"),
    axis.text.x = element_text(colour = "DarkGreen", size = 10, angle= 0, hjust=1),
    plot.title = element_text(colour = "Black", size = 15, hjust = 0.5, face = "bold"),
    axis.line = element_line(color = "black"),
    axis.ticks = element_line(color = "black"),
    legend.title = element_text(size = 8)) + coord_flip()

#Boxplot for Gross US
ggplot(data = moviextended, aes(x="", y = Gross..US)) +
  geom_boxplot(outlier.color = "red", outlier.shape = 1, outlier.size= 5, size=1) +
  labs(title = "Boxplot for Gross US",
       y = " Gross US",
       x = "Value") +
  theme(
    axis.title.x = element_text(colour = "Blue", size = 12, face="bold"),
    axis.title.y = element_text(colour = "Blue", size = 12, face="bold"),
    axis.text.x = element_text(colour = "DarkGreen", size = 10, angle= 0, hjust=1),
    plot.title = element_text(colour = "Black", size = 15, hjust = 0.5, face = "bold"),
    axis.line = element_line(color = "black"),
    axis.ticks = element_line(color = "black"),
    legend.title = element_text(size = 8))

#Boxplot for Overseas
ggplot(data = moviextended, aes(x="", y = Overseas)) +
  geom_boxplot(outlier.color = "red", outlier.shape = 1, outlier.size= 5, size=1) +
  labs(title = "Boxplot for Overseas",
       y = "Overseas",
       x = "Value") +
  theme(
    axis.title.x = element_text(colour = "Blue", size = 12, face="bold"),
    axis.title.y = element_text(colour = "Blue", size = 12, face="bold"),
    axis.text.x = element_text(colour = "DarkGreen", size = 10, angle= 0, hjust=1),
    plot.title = element_text(colour = "Black", size = 15, hjust = 0.5, face = "bold"),
    axis.line = element_line(color = "black"),
    axis.ticks = element_line(color = "black"),
    legend.title = element_text(size = 8)) + coord_flip()

#Boxplot for Profit
ggplot(data = moviextended, aes(x="", y = Profit)) +
  geom_boxplot(outlier.color = "red", outlier.shape = 1, outlier.size= 5, size=1) +
  labs(title = "Boxplot for Profit",
       y = "Profit",
       x = "") +
  theme(
    axis.title.x = element_text(colour = "Blue", size = 12, face="bold"),
    axis.title.y = element_text(colour = "Blue", size = 12, face="bold"),
    axis.text.x = element_text(colour = "DarkGreen", size = 10, angle= 0, hjust=1),
    plot.title = element_text(colour = "Black", size = 15, hjust = 0.5, face = "bold"),
    axis.line = element_line(color = "black"),
    axis.ticks = element_line(color = "black"),
    legend.title = element_text(size = 8)) + coord_flip()
```

```
#Boxplot for Runtime
ggplot(data = moviextended, aes(x="", y = Runtime)) +
 geom_boxplot(outlier.color = "red", outlier.shape = 1, outlier.size= 5, size=1) +
 labs(title = "Boxplot for Runtime",
      y = "US",
      x = "") +
 theme(
  axis.title.x = element_text(colour = "Blue", size = 12, face="bold"),
  axis.title.y = element_text(colour = "Blue", size = 12, face="bold"),
  axis.text.x = element_text(colour = "DarkGreen", size = 10, angle= 0, hjust=1),
  plot.title = element_text(colour = "Black", size = 15, hjust = 0.5, face = "bold"),
  axis.line = element_line(color = "black"),
  axis.ticks = element_line(color = "black"),
  legend.title = element_text(size = 8)) + coord_flip()



#Boxplot for US
ggplot(data = moviextended, aes(x="", y = US)) +
 geom_boxplot(outlier.color = "red", outlier.shape = 1, outlier.size= 5, size=1) +
 labs(title = "Boxplot for US",
      y = "US",
      x = "") +
 theme(
  axis.title.x = element_text(colour = "Blue", size = 12, face="bold"),
  axis.title.y = element_text(colour = "Blue", size = 12, face="bold"),
  axis.text.x = element_text(colour = "DarkGreen", size = 10, angle= 0, hjust=1),
  plot.title = element_text(colour = "Black", size = 15, hjust = 0.5, face = "bold"),
  axis.line = element_line(color = "black"),
  axis.ticks = element_line(color = "black"),
  legend.title = element_text(size = 8)) + coord_flip()
```

```
#6. Bar graph for average IMDb ratings by genre
genre_ratings <- moviextended %>%
  group_by(Genre) %>%
  summarise(Avg_IMDb_Rating = mean(IMDb.Rating, na.rm = TRUE))

#Data-Aesthetics
p <- bar_plot <- ggplot(genre_ratings, aes(x = Genre, y = Avg_IMDb_Rating, fill = Genre))

#Geometry Layer
pg <- p +
  geom_bar(stat = "identity", position = "dodge", width = 0.7, color = "black")

#Formatting
pg <- pg +
  labs(title = "Average IMDb Ratings by Genre",
       x = "Genre",
       y = "Average IMDb Rating") +
  theme_minimal() +
  theme(
    axis.title.x = element_text(colour = "Black", size = 12),
    axis.title.y = element_text(colour = "Black", size = 12),
    axis.text.x = element_text(colour = "Darkgreen", size = 10, angle= 45, hjust=1),
    axis.text.y = element_text(colour = "DarkGreen", size = 10),
    plot.title = element_text(colour = "Black", size = 15, hjust = 0.5, face = "bold"),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    axis.line = element_line(color = "black"),
    axis.ticks = element_line(color = "black"),
    legend.box.background = element_rect(color = "Black"),
    legend.position = "bottom",
    legend.justification = "center"
  )



# Scatter plot for Budget vs. Gross earnings
sp <- ggplot(moviextended, aes(x = Budget, y = Gross)) +
  geom_point(aes(color = Genre), alpha = 0.5) +
  geom_smooth(method = "lm", se = FALSE, aes(colour=Genre)) +
  labs(title = "Budget vs. Gross Earnings",
       x = "Budget",
       y = "Gross Earnings") +
  theme_minimal() +
  theme(
    axis.title.x = element_text(colour = "Black", size = 12),
    axis.title.y = element_text(colour = "Black", size = 12),
    axis.text.x = element_text(colour = "DarkGreen", size = 8),
    axis.text.y = element_text(colour = "DarkGreen", size = 8),
    plot.title = element_text(colour = "Black", size = 15, hjust = 0.5, face = "bold"),
    legend.title = element_text(size = 10),
    legend.text = element_text(size = 10),
    legend.justification = c(1, 1),
    axis.line = element_line(color = "black"),
    axis.ticks = element_line(color = "black"),
    legend.box.background = element_rect(color = "Black"),
    strip.text.x = element_text(size = 9 , face="bold", colour="Brown"),
    strip.background = element_rect(color = "black")
  ) +  facet_wrap(~toupper(Genre))
```

Thank-You