General Sir John Kotelawala Defence University.

Faculty of Management, Social Sciences and Humanities

Department of Languages.

BSc in Applied Data Science Communication.

Fundamentals of Data Mining /LB2114

Year 2: Semester 3

Assignment number 1

04.03.2024

P.D.S.S Pitiyage - D/ADC/23/0015

W.P.T.M Pathirana - D/ADC/23/0021

M.G Saranga - D/ADC/23/0020

P.R Withanage - D/ADC/23/0041

Table of Content

Task 02 - Apply Clustering to the Blockbuster Top Movie dataset using R.

1. Introduction
2. Dataset
3. Explanation and Preparation of Dataset
    a. Data pre-processing
    b. Data Explanation
4. Data Mining
    a. Clustering
    b. K – means Clustering
    c. Elbow Method
    d. Visualization tools available in R for clustering
5. Implementation in R
    a. R Packages
6. Results analysis and Discussion
7. Conclusion

Task 03 - Build a Dynamic Dashboard in Power BI

1.Introduction

2.Exploration of Data Set

3.Dashboard Design & Implementation

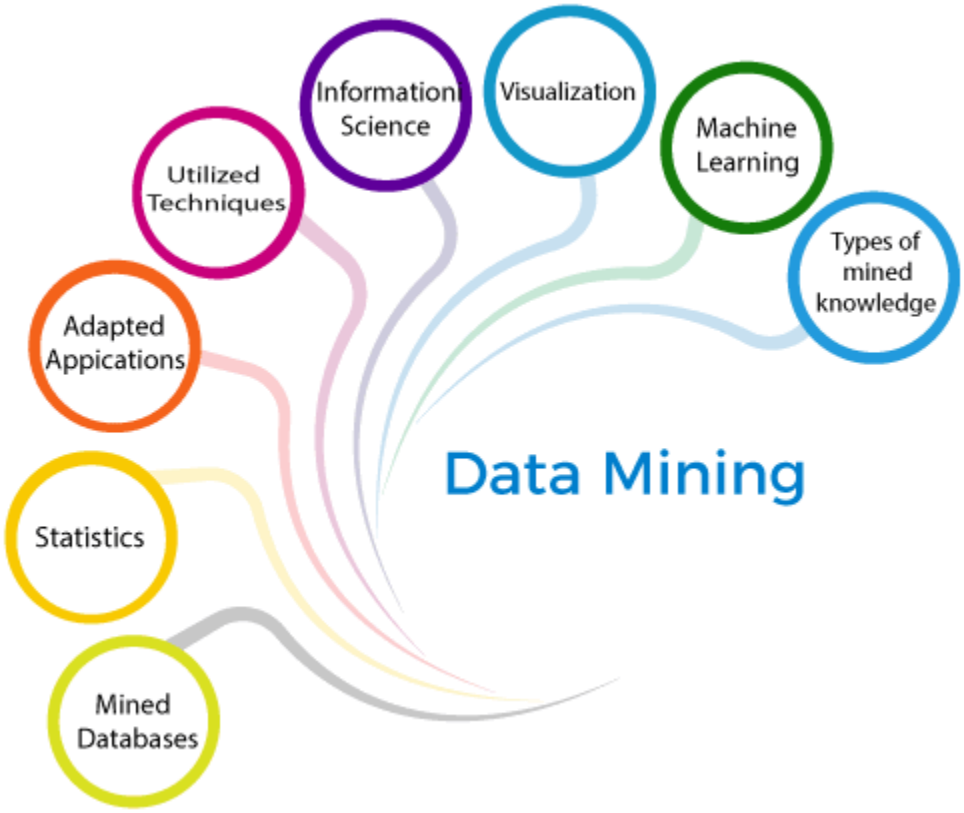
4.Conclusion.

Introduction to data mining.

- **What is Data mining?**

The systematic process of finding significant patterns, trends, and insights within large databases is known as data mining. It attempts to extract useful knowledge from big data sources, such as databases, data warehouses, and the Internet, by combining statistical and computational approaches. Important phases including data cleansing, integration, selection, transformation, mining, pattern evaluation, and knowledge representation are all part of the extensive process.

Conventional machine learning is essential to data mining because it can differentiate between pre-classified data for supervised learning (regression and classification) and unlabeled data for unsupervised learning (clustering, association rule mining, and dimensionality reduction). Applications for these strategies may be found in a wide range of sectors, including marketing, banking, and healthcare. They enable firms to identify trends, understand consumer behavior, and make data-driven, purposeful decisions. Data mining is essentially an essential tool that helps strategic decision-making across a range of sectors by revealing insightful information.



- **Importance of Data Mining in Software Engineering Teamwork Assessment.**

When it comes to evaluating software engineering cooperation in educational contexts, data mining is quite helpful. Its importance stems from the careful examination of large-scale datasets, which illuminates the dynamics of collaboration.

In this project, data mining plays a crucial role in identifying patterns and trends, providing educators with a comprehensive grasp of team dynamics, work distribution, and individual contributions. By using predictive modeling, data mining gives teachers the ability to proactively identify possible problems and achievements based on available data, allowing them to customize their interventions and teaching methods.

In addition to encouraging continuous skill improvement in collaboration, this evidence-based approach also helps to improve collaborative learning environments in software engineering education as a whole. Data mining is a powerful tool that may be used to maximize the quality of software engineering education by uncovering hidden insights, forecasting future events, and directing interventions thanks to its ability to provide a basis for informed decision-making.

Overview of tasks

Using R Studio, Python, and Microsoft Power BI, this project uses advanced data analysis methods to extensively examine datasets and generate predictions based on the results. Classification and clustering are two core machine learning techniques that are implemented with the main emphasis. These methods are carefully applied to specific datasets in order to derive significant insights and predict results.
The primary tasks in this project are:

- Classification: Data is categorized using machine learning algorithms, which makes it easier to spot underlying patterns and trends.

- Clustering: grouping related data points using clustering techniques, which facilitates the investigation of underlying patterns and relationships in the datasets.

Apart from R Studio, Python is also utilized as an additional method. Microsoft Power BI is used to graphically exhibit the results, guaranteeing a thorough and easy-to-understand comprehension of the data. With the help of this multifaceted methodology, the datasets will be robustly explored, producing informative insights and useful forecasts.

TASK 01

Creating a classification model for software engineering teamwork assessment in education setting.

1.Introduction

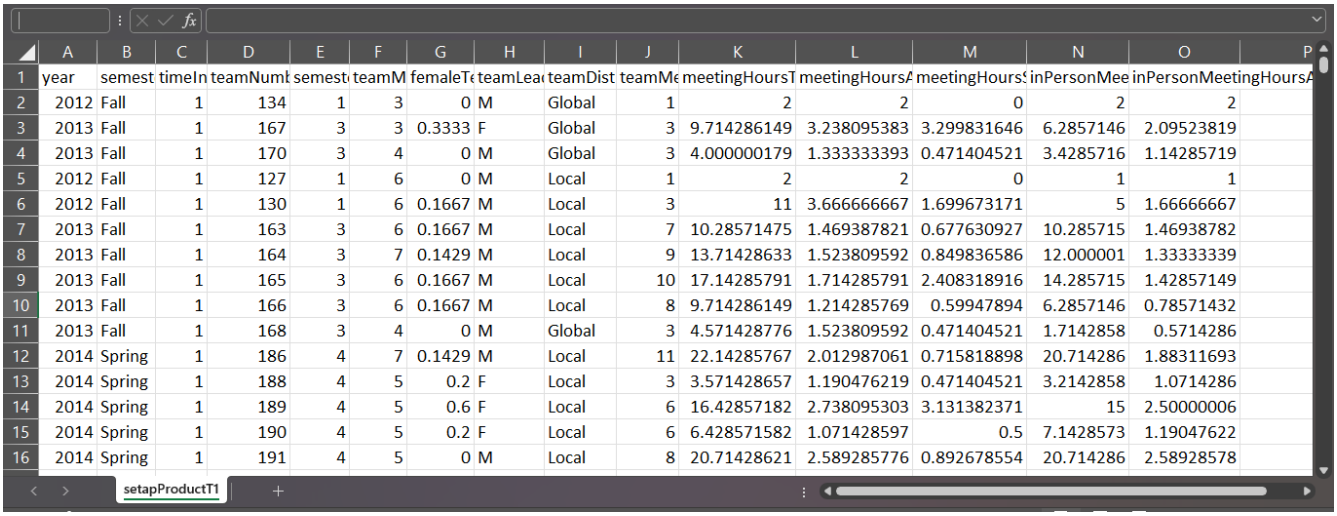
The use of the R programming language and the K-Nearest Neighbor (KNN) machine learning algorithm on an extensive dataset collected from San Francisco State University's software engineering classes is the main focus of this study. The collection, which spans several semesters, provides a wealth of information for researchers studying software engineering education by capturing a variety of facts about students' activities.

Our main goal is to use the KNN classifier to use available characteristics to estimate if a team is local or global. By utilizing real data to demonstrate how KNN is used, we hope to demystify the process of creating and gaining access to machine learning models.

This study explores how collaboration is assessed in learning environments using the software engineering paradigm. It specifically aims to evaluate the dynamics of collaboration, providing insightful information to teachers and students alike. Through an analysis of temporal, structural, and behavioral components, we want to fully comprehend software engineering team performance by comparing it to agricultural environmental conditions. For an unbiased assessment, parameters like "time Interval", "semester Id", "team Number" and "team Distribution" offer crucial organizational and chronological context.

The use of quantitative indicators such as 'meetingHoursTotal','meetingHoursAverage','meetingHoursStandardDeviation' and 'inPersonMeetingHoursTotal' provides a comprehensive understanding of cooperative endeavors. The algorithm created for this study aims to improve software engineering students' overall cooperation experience, which is in line with the larger goal of educating students for productive teamwork in professional settings.

2.Dataset



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	year	semest	timeIn	teamNumb	semest	teamM	femaleTe	teamLea	teamDist	teamMe	meetingHoursT	meetingHoursA	meetingHours	inPersonMee	inPersonMeetingHoursA	
2	2012	Fall	1	134	1	3	0	M	Global	1	2	2	0	2	2	
3	2013	Fall	1	167	3	3	0.3333	F	Global	3	9.714286149	3.238095383	3.299831646	6.2857146	2.09523819	
4	2013	Fall	1	170	3	4	0	M	Global	3	4.000000179	1.333333393	0.471404521	3.4285716	1.14285719	
5	2012	Fall	1	127	1	6	0	M	Local	1	2	2	0	1	1	
6	2012	Fall	1	130	1	6	0.1667	M	Local	3	11	3.666666667	1.699673171	5	1.66666667	
7	2013	Fall	1	163	3	6	0.1667	M	Local	7	10.28571475	1.469387821	0.677630927	10.285715	1.46938782	
8	2013	Fall	1	164	3	7	0.1429	M	Local	9	13.71428633	1.523809592	0.849836586	12.000001	1.33333339	
9	2013	Fall	1	165	3	6	0.1667	M	Local	10	17.14285791	1.714285791	2.408318916	14.285715	1.42857149	
10	2013	Fall	1	166	3	6	0.1667	M	Local	8	9.714286149	1.214285769	0.59947894	6.2857146	0.78571432	
11	2013	Fall	1	168	3	4	0	M	Global	3	4.571428776	1.523809592	0.471404521	1.7142858	0.5714286	
12	2014	Spring	1	186	4	7	0.1429	M	Local	11	22.14285767	2.012987061	0.715818898	20.714286	1.88311693	
13	2014	Spring	1	188	4	5	0.2	F	Local	3	3.571428657	1.190476219	0.471404521	3.2142858	1.0714286	
14	2014	Spring	1	189	4	5	0.6	F	Local	6	16.42857182	2.738095303	3.131382371	15	2.50000006	
15	2014	Spring	1	190	4	5	0.2	F	Local	6	6.428571582	1.071428597	0.5	7.1428573	1.19047622	
16	2014	Spring	1	191	4	5	0	M	Local	8	20.71428621	2.589285776	0.892678554	20.714286	2.58928578	

setapProductT1

This dataset comes from the UC Irvine Machine Learning Repository, which was created especially to capture the subtleties of technical evaluations used in San Francisco State University's software engineering curriculum. The data, which was gathered across several semesters, consists of student timesheets, tool logs, and teacher observations from software engineering classes. Ensuring permission and privacy, all data is collected with student agreement and anonymized.

The dataset may be accessed at .***https://archive.ics.uci.edu/dataset/393/data+for+software+engineering+teamwork+assessment+in+education+setting*** It is a valuable tool for examining how teaching approaches are implemented in the field of software engineering, as well as how well they engage students. It offers a priceless starting point for studies intended to improve academic results and comprehend the mechanics of learning in technical fields.

3. Explanation and Preparation of the Dataset

The assessment, analysis, and categorization of collaborative dynamics in software engineering education are the main foci. Student Activity Measures (SAMs) combined into Team Activity Measures (TAMs) for both local and international teams are included in the dataset. Results represent assessments of the software engineering process and products and are categorized as A or F. The data includes 383 students from 18 different class sections over the course of seven semesters.

Overall Dataset Statistics

Team Composition	
Local Teams	59
Global Teams	15
Total Teams	74

Semesters Covered	
Number of Semesters	7
First Semester	Fall 2012
Last Semester	Fall 2015

Student Information	
Number of Students	383
Class Sections	18

Dataset Details	
Number of TAM Features	115
Number of Class Labels (Outcomes)	2

Outcome Statistics	
Total Issues	255
Team Composition	59 Local Teams, 15 Global Teams (Total: 74 Teams)
Outcome Instances	74 (Process: A/F, Product: A/F)

The final dataset is an aggregation of data from eleven Excel files, merging pertinent data for a more focused and effective study. Some data columns were removed for the project since it was thought they weren't important for the analysis. The columns that were chosen had as their focus the qualities that are crucial for assessing the dynamics of collaboration in the context of software engineering education. This condensed dataset guarantees relevance to the project's goals by enabling a targeted analysis.

Description of Variables:

- Year-(removed) Stands for the academic year that the software engineering course was held.
- Semester-(removed)Indicates the particular semester (Fall or Spring) of the school year.
- timeInterval - Refers to set times throughout class that stand in for significant accomplishments made by each student team.
- teamNumber - Distinctly identifies every student team taking the software engineering course.
- semesterId - Helps with data structure and classification by acting as the semester's unique identifier.
- teamMemberCount - Indicates how many students are working on the final class project on each team.
- femaleTeamMembersPercent - Indicates the proportion of female team members in every student group.
- teamLeadGender - Classifies the team lead's gender as either male or female.
- teamDistribution - Gives information on the makeup of teams and makes a distinction between local and global team structures.
- teamMemberResponseCount - Shows the number of answers from the team, which helps with the assessment of the dynamics within the team.
- meetingHoursTotal - Calculates how many hours a team meets all semester long.
- meetingHoursAverage - Computes the average number of hours spent in team meetings and provides a measure specific to each team.

- meetingHoursStandardDeviation - Calculates the variation or dispersion of the total number of meeting hours during the semester.
- inPersonMeetingHoursTotal - Represents the total number of hours spent over the semester attending in-person team meetings.
- inPersonMeetingHoursAverage - Gives a measure specific to each team that indicates the average number of hours spent on in-person team meetings.

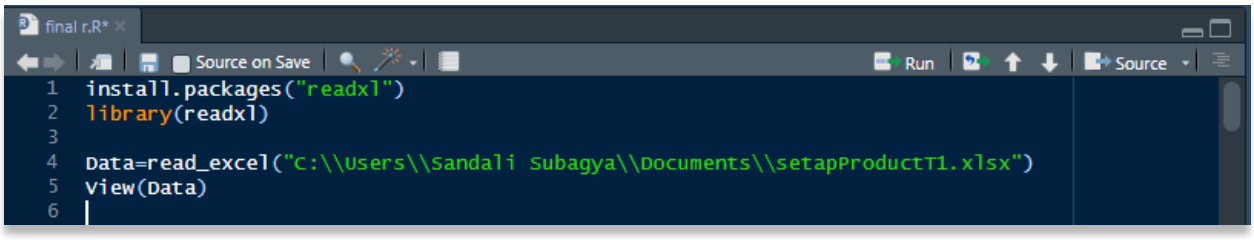
	timeInterval	teamNumber	semesterId	teamMemberCount	femaleTeamMembersPercent	teamLeadGender	teamDistribution
1	1	134	1	3	0.0000	M	Global
2	1	167	3	3	0.3333	F	Global
3	1	170	3	4	0.0000	M	Global
4	1	127	1	6	0.0000	M	Local
5	1	130	1	6	0.1667	M	Local
6	1	163	3	6	0.1667	M	Local
7	1	164	3	7	0.1429	M	Local
8	1	165	3	6	0.1667	M	Local
9	1	166	3	6	0.1667	M	Local

teamMemberResponseCount	meetingHoursTotal	meetingHoursAverage	meetingHoursStandardDeviation	inPersonMeetingHoursTotal	inPersonMeetingHoursAverage
1	2.000000	2.0000000	0.0000000	2.000000	2.0000000
3	9.714286	3.2380954	3.2998316	6.285715	2.0952382
3	4.000000	1.3333334	0.4714045	3.428572	1.1428572
1	2.000000	2.0000000	0.0000000	1.000000	1.0000000
3	11.000000	3.6666667	1.6996732	5.000000	1.6666667
7	10.285715	1.4693878	0.6776309	10.285715	1.4693878
9	13.714286	1.5238096	0.8498366	12.000001	1.3333334
10	17.142858	1.7142858	2.4083189	14.285715	1.4285715

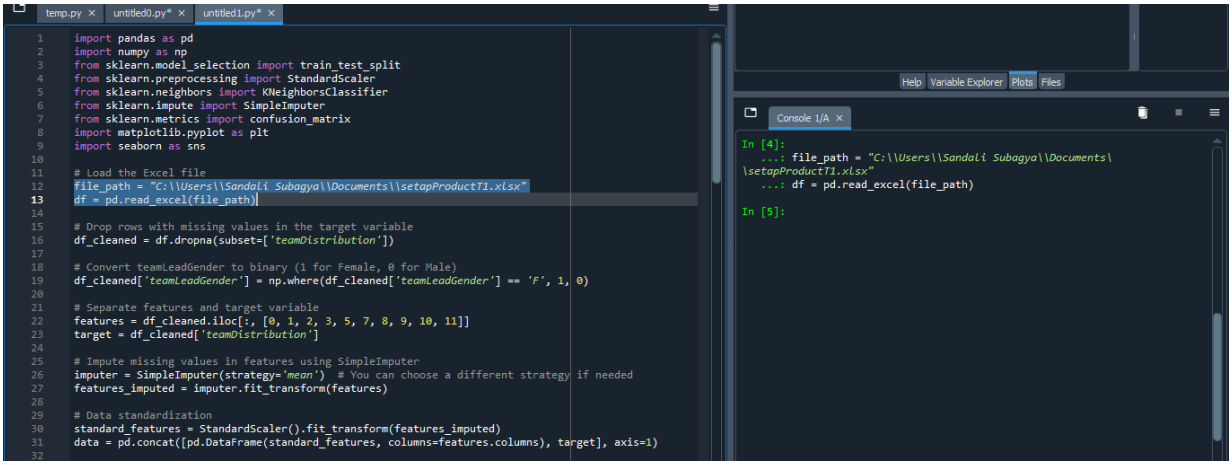
4.Data Preprocessing and Implementation in R.

4.1 Import excel.

- The dataset was imported into R Studio in the first stages of data preparation and implementation in R. To handle Excel files efficiently, the'readxl' package was utilized.



(Load the Excel file using R)



(Using Python)

4.2 Preparation of data

- Afterwards, a visual inspection of the dataset was conducted to learn more about its composition and organization. Null entries were eliminated from the dataset to guarantee a solid and accurate analysis, improving the dataset's overall quality.

```r
df_cleaned <- na.omit(Data)
view(df_cleaned)
```

(Drop null values using R)

```python
# Drop rows with missing values in the target variable
df_cleaned = df.dropna(subset=['teamDistribution'])
```

(Using Python)

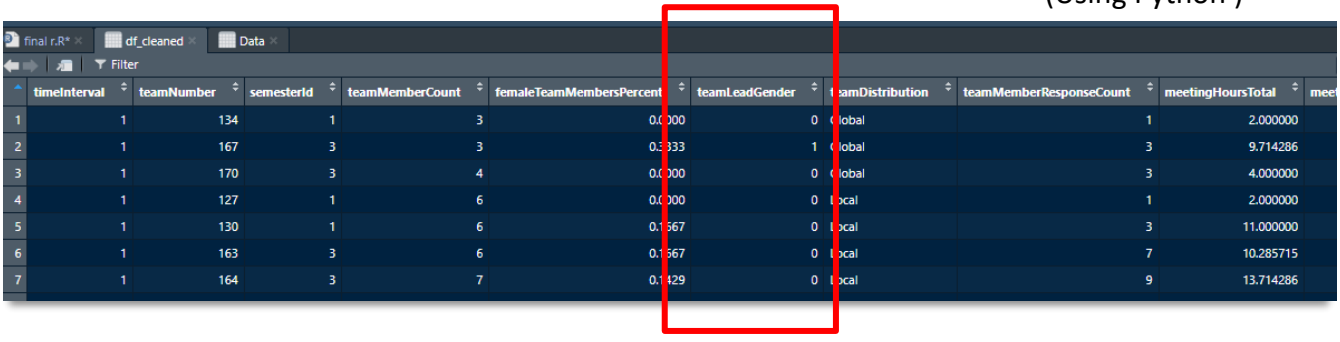
- Subsequently, to ensure that the dataset was now free of null values and provide a strong basis for additional analysis, a check was done. Using the 'cbind' function, extra numerical columns were concatenated to convert the categorical columns into numerical representations.

```r
names(df_cleaned)
table(df_cleaned$teamLeadGender)
df_cleaned$teamLeadGender <- ifelse(df_cleaned$teamLeadGender=="F",1,0)
table(df_cleaned$teamLeadGender)
view(df_cleaned)
```

(Convert categorical to numerical using R)

```python
# Convert teamLeadGender to binary (1 for Female, 0 for Male)
df_cleaned['teamLeadGender'] = np.where(df_cleaned['teamLeadGender'] == 'F', 1, 0)
```

(Using Python)



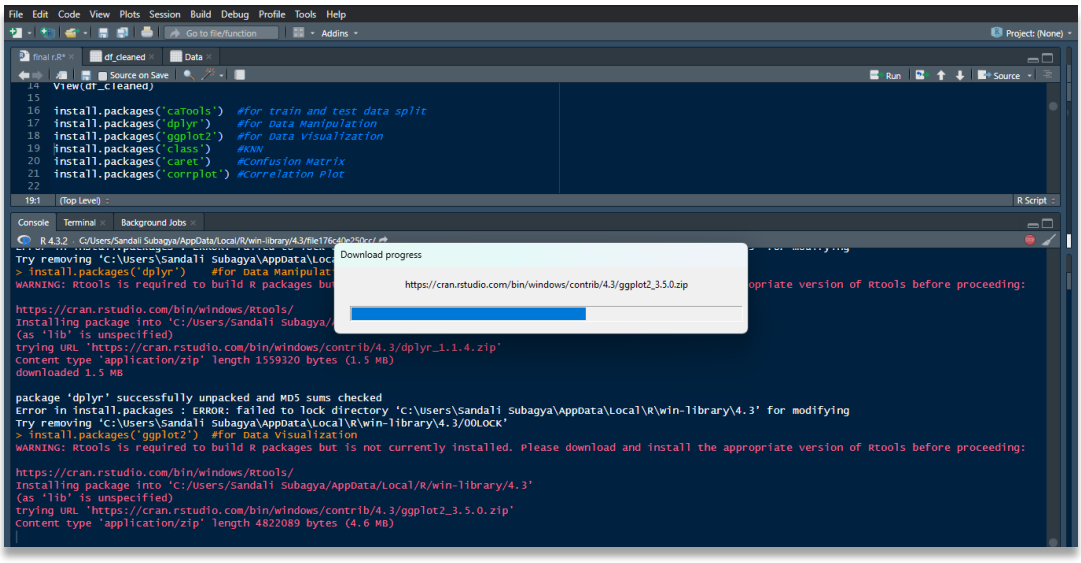
	timeInterval	teamNumber	semesterId	teamMemberCount	femaleTeamMembersPercent	teamLeadGender	amDistribution	teamMemberResponseCount	meetingHoursTotal
1	1	134	1	3	0.0000	0	Global	1	2.000000
2	1	167	3	3	0.3333	1	Global	3	9.714286
3	1	170	3	4	0.0000	0	Global	3	4.000000
4	1	127	1	6	0.0000	0	Local	1	2.000000
5	1	130	1	6	0.1667	0	Local	3	11.000000
6	1	163	3	6	0.1667	0	Local	7	10.285715
7	1	164	3	7	0.1429	0	Local	9	13.714286

- By following a step-by-step procedure, the dataset was made ready for statistical modeling and machine learning applications, meeting the standards needed for precise and perceptive analysis.

4.3 Install packages.

Load Required Libraries:
Add necessary libraries to the program, such corrplot, ggplot2, caret, dplyr, caTools, and class. These libraries include resources for machine learning, correlation analysis, data processing, and visualization.

- caTools: Offers data splitting methods, which are crucial for generating training and testing sets for machine learning.
- dplyr: Provides a range of effective data manipulation capabilities, such as data frame transformation, filtering, and summarization.
- ggplot2: Makes it possible to create complex, adaptable data visualizations for exploratory data research.
- class: Uses classification methods for predictive modeling, such as the K-Nearest Neighbors (KNN) algorithm.
- tender: Offers a uniform interface for diverse methods, simplifying the training and testing of machine learning models.
- corrplot: Makes it easier to see correlation matrices and helps interpret how the variables in the dataset relate to one another.



(Install packages using R)

```python
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.impute import SimpleImputer
from sklearn.metrics import confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns
```

(Using Python)

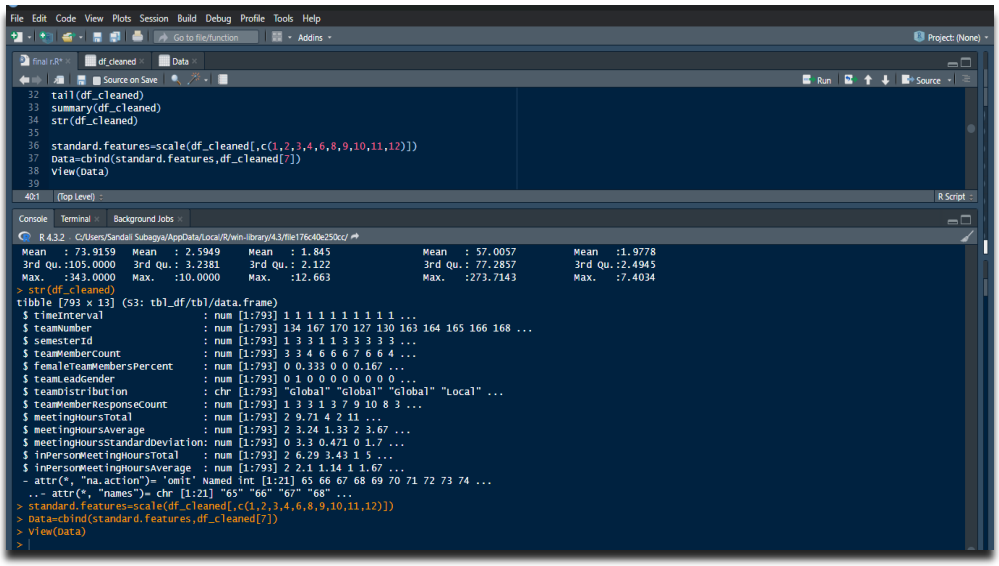
- The main goal of this stage is to fully comprehend the dataset's structure. An extensive investigation is carried out by looking at the column names ({names}), first rows ({head}), last rows ({tail}), summary statistics ({summary}), and data types ({str}). Finding important characteristics, trends, and possible problems in the dataset is made possible by this procedure. When these exploratory approaches are combined, they offer insightful information about the properties of the dataset, which facilitates later decision-making regarding data pretreatment and analysis.

```
> names(df_cleaned)
 [1] "timeInterval"                "teamNumber"                  "semesterId"                  "teamMemberCount"
 [5] "femaleTeamMembersPercent"    "teamLeadGender"              "teamDistribution"            "teamMemberResponseCount"
 [9] "meetingHoursTotal"           "meetingHoursAverage"         "meetingHoursStandardDeviation" "inPersonMeetingHoursTotal"
[13] "inPersonMeetingHoursAverage"
> head(df_cleaned)
# A tibble: 6 x 13
  timeInterval teamNumber semesterId teamMemberCount femaleTeamMembersPercent teamLeadGender teamDistribution teamMemberResponseCount meetingHoursTotal
         <dbl>      <dbl>      <dbl>           <dbl>                   <dbl>          <dbl> <chr>                             <dbl>            <dbl>
1            1        134          1               3                   0                  0 Global                                1             2
2            1        167          3               3                   0.333              1 Global                                3             9.71
3            1        170          3               4                   0                  0 Global                                3             4.00
4            1        127          1               6                   0                  0 Local                                 1             2
5            1        130          1               6                   0.167              0 Local                                 3            11
6            1        163          3               6                   0.167              0 Local                                 7            10.3
# i 4 more variables: meetingHoursAverage <dbl>, meetingHoursStandardDeviation <dbl>, inPersonMeetingHoursTotal <dbl>, inPersonMeetingHoursAverage <dbl>
> tail(df_cleaned)
# A tibble: 6 x 13
  timeInterval teamNumber semesterId teamMemberCount femaleTeamMembersPercent teamLeadGender teamDistribution teamMemberResponseCount meetingHoursTotal
         <dbl>      <dbl>      <dbl>           <dbl>                   <dbl>          <dbl> <chr>                             <dbl>            <dbl>
1           11        217          6               5                   0                  0 Local                                35             74.2
2           11        220          6               5                   0                  0 Local                                35             81.8
3           11        227          7               6                   0.167              0 Local                                42            226.
4           11        235          7               5                   0                  0 Local                                28            109.
5           11        236          7               6                   0.333              1 Local                                43            105.
6           11        240          7               6                   0                  0 Local                                42             73.6
# i 4 more variables: meetingHoursAverage <dbl>, meetingHoursStandardDeviation <dbl>, inPersonMeetingHoursTotal <dbl>, inPersonMeetingHoursAverage <dbl>
```

```
> summary(df_cleaned)
  timeInterval     teamNumber      semesterId    teamMemberCount femaleTeamMembersPercent teamLeadGender   teamDistribution   teamMemberResp
 Min.   : 1.000   Min.   :124.0   Min.   :1.000   Min.   :3.000   Min.   :0.0000           Min.   :0.0000   Length:793         Min.   : 1.00
 1st Qu.: 3.000   1st Qu.:162.0   1st Qu.:3.000   1st Qu.:5.000   1st Qu.:0.0000           1st Qu.:0.0000   Class :character   1st Qu.:13.00
 Median : 6.000   Median :196.0   Median :5.000   Median :5.000   Median :0.1667           Median :0.0000   Mode  :character   Median :24.00
 Mean   : 6.091   Mean   :191.3   Mean   :4.433   Mean   :5.189   Mean   :0.1761           Mean   :0.1929                      Mean   :27.12
 3rd Qu.: 9.000   3rd Qu.:223.0   3rd Qu.:6.000   3rd Qu.:6.000   3rd Qu.:0.2000           3rd Qu.:0.0000                      3rd Qu.:36.00
 Max.   :11.000   Max.   :241.0   Max.   :7.000   Max.   :7.000   Max.   :0.8333           Max.   :1.0000                      Max.   :84.00
 meetingHoursTotal  meetingHoursAverage meetingHoursStandardDeviation inPersonMeetingHoursTotal inPersonMeetingHoursAverage
 Min.   :  0.8571   Min.   : 0.2143     Min.   : 0.000                Min.   :  0.5714          Min.   :0.1429
 1st Qu.: 27.8571   1st Qu.: 1.7931     1st Qu.: 1.003                1st Qu.: 20.7857          1st Qu.:1.3367
 Median : 56.6429   Median : 2.4000     Median : 1.440                Median : 43.5714          Median :1.8117
 Mean   : 73.9159   Mean   : 2.5949     Mean   : 1.845                Mean   : 57.0057          Mean   :1.9778
 3rd Qu.:105.0000   3rd Qu.: 3.2381     3rd Qu.: 2.122                3rd Qu.: 77.2857          3rd Qu.:2.4945
 Max.   :343.0000   Max.   :10.0000     Max.   :12.663                Max.   :273.7143          Max.   :7.4034
> str(df_cleaned)
tibble [793 x 13] (S3: tbl_df/tbl/data.frame)
 $ timeInterval               : num [1:793] 1 1 1 1 1 1 1 1 1 1 ...
 $ teamNumber                 : num [1:793] 134 167 170 127 130 163 164 165 166 168 ...
 $ semesterId                 : num [1:793] 1 3 3 1 1 3 3 3 3 3 ...
 $ teamMemberCount            : num [1:793] 3 3 4 6 6 6 7 6 6 4 ...
 $ femaleTeamMembersPercent   : num [1:793] 0 0.333 0 0 0.167 ...
 $ teamLeadGender             : num [1:793] 0 1 0 0 0 0 0 0 0 0 ...
 $ teamDistribution           : chr [1:793] "Global" "Global" "Global" "Local" ...
 $ teamMemberResponseCount    : num [1:793] 1 3 3 1 3 7 9 10 8 3 ...
 $ meetingHoursTotal          : num [1:793] 2 9.71 4 2 11 ...
 $ meetingHoursAverage        : num [1:793] 2 3.24 1.33 2 3.67 ...
 $ meetingHoursStandardDeviation: num [1:793] 0 3.3 0.471 0 1.7 ...
 $ inPersonMeetingHoursTotal  : num [1:793] 2 6.29 3.43 1 5 ...
 $ inPersonMeetingHoursAverage: num [1:793] 2 2.1 1.14 1 1.67 ...
 - attr(*, "na.action")= 'omit' Named int [1:21] 65 66 67 68 69 70 71 72 73 74 ...
  ..- attr(*, "names")= chr [1:21] "65" "66" "67" "68" ...
>
```

4.4 Data Standardization

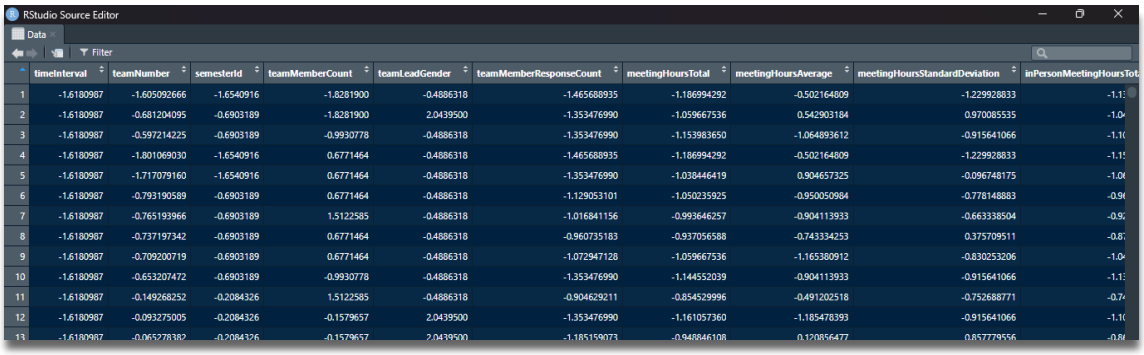
One important preprocessing step in machine learning is standardizing features. A mean of 0 and a standard deviation of 1 are achieved by converting numerical characteristics in this way. As a result, the analysis is kept free of variables with bigger scales by guaranteeing that all characteristics are on the same scale. The scale function is utilized in this particular code to standardize a subset of the dataset ('df_cleaned'). The 7th column of the original dataset is left intact when the outcome is merged with it. Model performance and interpretability are enhanced by the fact that this standardized dataset ({Data}) is more suited to machine learning methods.



(Data Standardization using R)

```
# Data standardization
standard_features = StandardScaler().fit_transform(features_imputed)
data = pd.concat([pd.DataFrame(standard_features, columns=features.columns), target], axis=1)
```

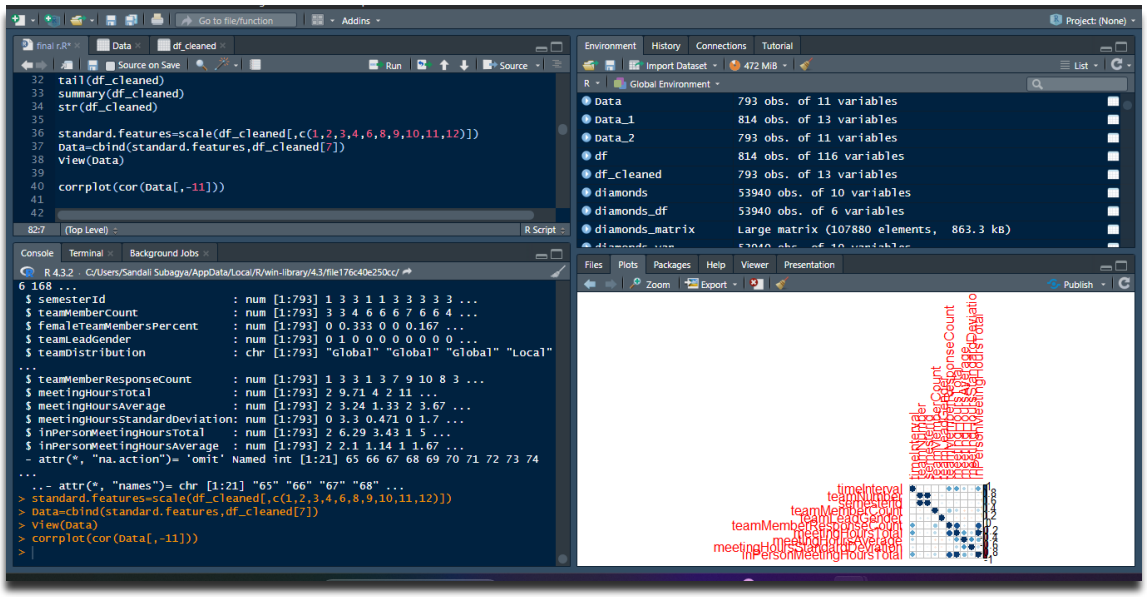
(Using Python)



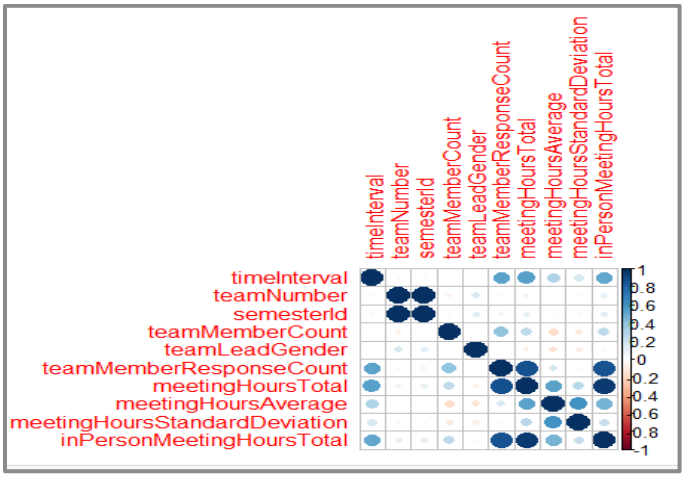
(Dataset after standardization)

4.5 Data Visualization

Verify whether there are any correlations between the data columns in the dataset. The `corrplot` function is used in this phase to visually show the correlations between the dataset's various columns. The first steps in the procedure involve importing the dataset, addressing missing values, and converting category variables into numerical representations. The data is ready for correlation analysis by normalizing numerical characteristics. A correlation matrix plot, produced by the `corrplot` function, shows the correlations between the variables graphically. In order to help comprehend inter-column interactions, this exploration is crucial for spotting possible dependencies or linkages between various dataset components.

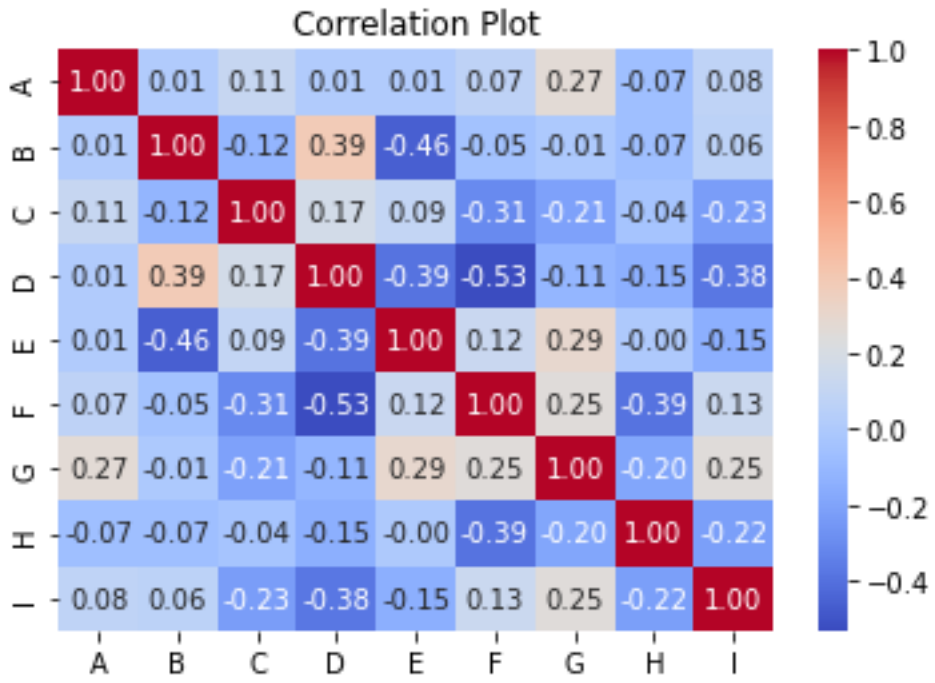


(Data Visualization by Using R)



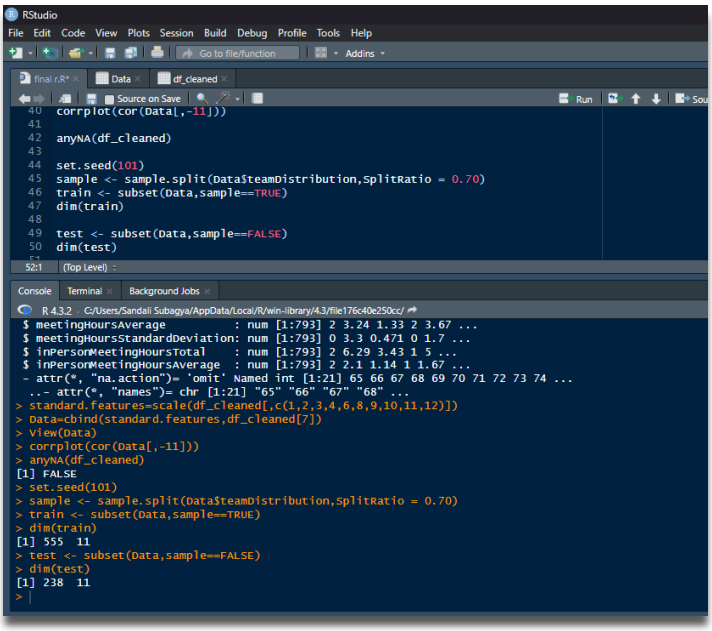
```python
# Correlation plot
numeric_data = data.select_dtypes(include=[np.number])
corr_matrix = numeric_data.corr()

# Plot correlation matrix using seaborn for better visualization
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', fmt=".2f")
# Set the title
plt.title('Correlation Plot')
plt.show()
```

(Using Python)

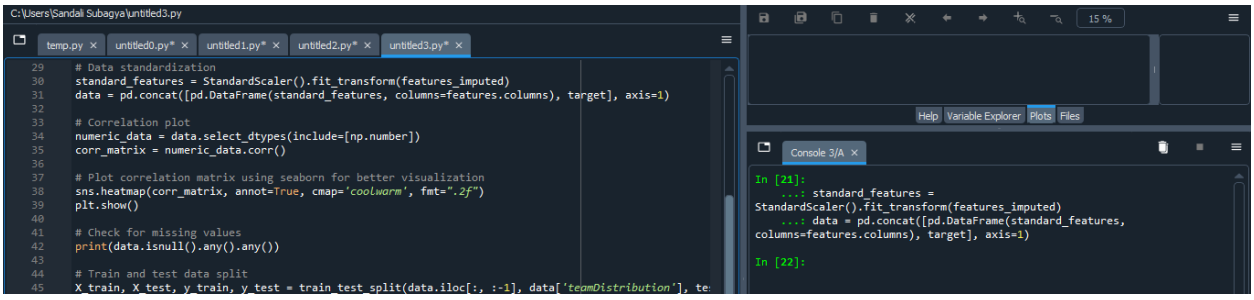


4.4 Test and Train Data Split

The next critical step is to separate the data into training and testing sets after the dataset has been standardized. The `sample.split` function is used to do this, and a given split ratio—in this example, 70% for training and 30% for testing—is used. Reproducibility is guaranteed via `set.seed(101)}. 70% of the standardized data are in the resultant training set, designated as "train," while the remaining 30% are in the testing set, designated as "test." To ensure proper partitioning, each set's dimensions are shown. In order to ensure that machine learning models may be applied to new data, it is essential to separate the training of the models on one subset and the evaluation of their performance on a different subset.



(Data Visualization by Using R)



(Using Python)

5.Data Mining

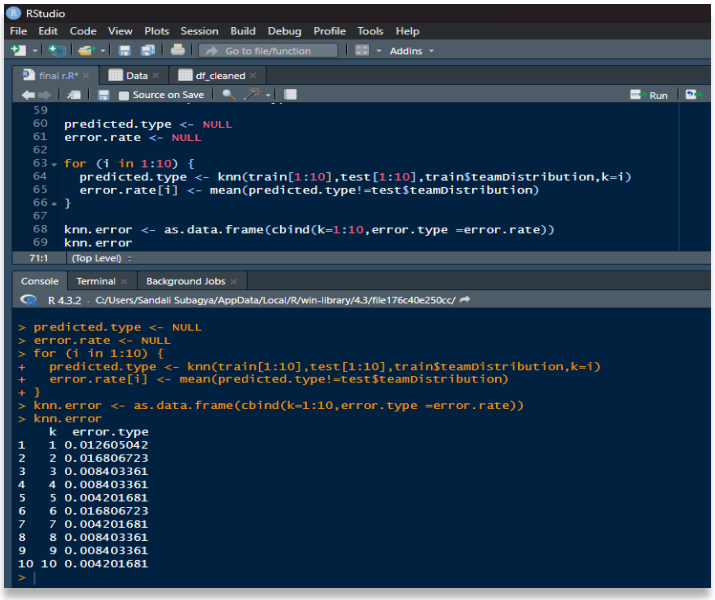
5.1 K-Nearest Neighbor (KNN) Model

Created a confusion matrix by running the k-Nearest Neighbors (KNN) algorithm on the test set. This matrix contrasts the actual team distribution values with the projected classifications. The following is the confusion matrix:

	Reference	
Prediction	Global	Local
Global	47	2
Local	1	188

The model's performance was evaluated by computing many statistics from this confusion matrix. With respect to the percentage of accurately predicted cases out of the total, the model's overall accuracy is 98.74%. A strong agreement beyond chance is shown by the kappa coefficient of 0.9612, which measures the agreement between the actual and anticipated classifications.

The model's sensitivity (true positive rate) of 97.92% suggests that it can accurately identify international teams, while its specificity (true negative rate) of 98.95% suggests that it can identify local teams well. The precision, or positive predictive value, is 95.92% for positive forecasts and 99.47% for negative predictions, indicating the accuracy of negative predictions. All of these statistics support the KNN model's efficacy in correctly predicting the team distribution in the provided dataset. As a useful tool for this classification assignment, the model shows great accuracy and reliability in differentiating between global and local teams.



(Use KNN model in R)

```
Console   Terminal   Background Jobs
R 4.3.2 · C:/Users/Sandali Subagya/AppData/Local/R/win-library/4.3/file176c40e250cc/
> predicted.type <- knn(train[1:10],test[1:10], train$teamDistribution,k=3)
> #Error in prediction
> error <- mean(predicted.type!=test$teamDistribution)
> error
[1] 0.008403361
> #Confusion Matrix
> confusionMatrix(predicted.type,as.factor(test$teamDistribution))
Confusion Matrix and Statistics

          Reference
Prediction Global Local
    Global     47     1
    Local       1   189

               Accuracy : 0.9916
                 95% CI : (0.97, 0.999)
    No Information Rate : 0.7983
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9739

 Mcnemar's Test P-Value : 1

            Sensitivity : 0.9792
            Specificity : 0.9947
         Pos Pred Value : 0.9792
         Neg Pred Value : 0.9947
             Prevalence : 0.2017
         Detection Rate : 0.1975
   Detection Prevalence : 0.2017
      Balanced Accuracy : 0.9870

       'Positive' Class : Global

>
```

```python
    # KNN classification with k=1
    knn = KNeighborsClassifier(n_neighbors=1)
    knn.fit(X_train, y_train)
    y_pred = knn.predict(X_test)
    # Error rate
    error = np.mean(y_pred != y_test)
    print("Error rate:", error)
    # Confusion matrix
    conf_matrix = confusion_matrix(y_test, y_pred)
    print("Confusion Matrix:")
    print(conf_matrix)
    # Plot confusion matrix
    sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues')
    plt.xlabel('Predicted')
    plt.ylabel('Actual')
    plt.title('Confusion Matrix')
    plt.show()
    # Error rate for different values of k
    error_rates = []
    for k in range(1, 11):
        knn = KNeighborsClassifier(n_neighbors=k)
        knn.fit(X_train, y_train)
        y_pred = knn.predict(X_test)
        error_rates.append(np.mean(y_pred != y_test))
    plt.plot(range(1, 11), error_rates, marker='o')
    plt.xlabel('Value of K')
    plt.ylabel('Error')
    plt.title('Error Rate vs. K Value')
    plt.show()
    # KNN with optimal k (e.g., k=3)
    k_optimal = 3
    knn_optimal = KNeighborsClassifier(n_neighbors=k_optimal)
    knn_optimal.fit(X_train, y_train)
    y_pred_optimal = knn_optimal.predict(X_test)
```

```python
# Error rate with optimal k
error_rate_optimal = np.mean(y_pred_optimal != y_test)
print("Error rate with optimal k (", k_optimal, "):", error_rate_optimal)

# Confusion matrix with optimal k
conf_matrix_optimal = confusion_matrix(y_test, y_pred_optimal)
print("Confusion Matrix with optimal k:")
print(conf_matrix_optimal)

# Plot the error rates
error_data = pd.DataFrame({'k': range(1, 11), 'error_rate': error_rates})
plt.plot(error_data['k'], error_data['error_rate'], marker='o')
plt.xlabel('Value of K')
plt.ylabel('Error Rate')
plt.title('Error Rate vs. K Value')
plt.show()
```

(Using Python)

These are the salient features that should be noted.

- Accuracy: How well the model predicts global or local events overall.
- The range within which the true accuracy is most likely to fall is known as the 95% Confidence Interval (CI).
- Kappa: A chance-adjusted measure of the degree of agreement between the actual and anticipated categories.
- Sensitivity (True Positive Rate): The percentage of real cases worldwide that are successfully recognized.
- Specificity, also known as True Negative Rate, is the percentage of real local instances that are properly recognized.
- Balanced Accuracy: A metric that strikes a balance between specificity and sensitivity on average.

5.2 Test and Train Data Split

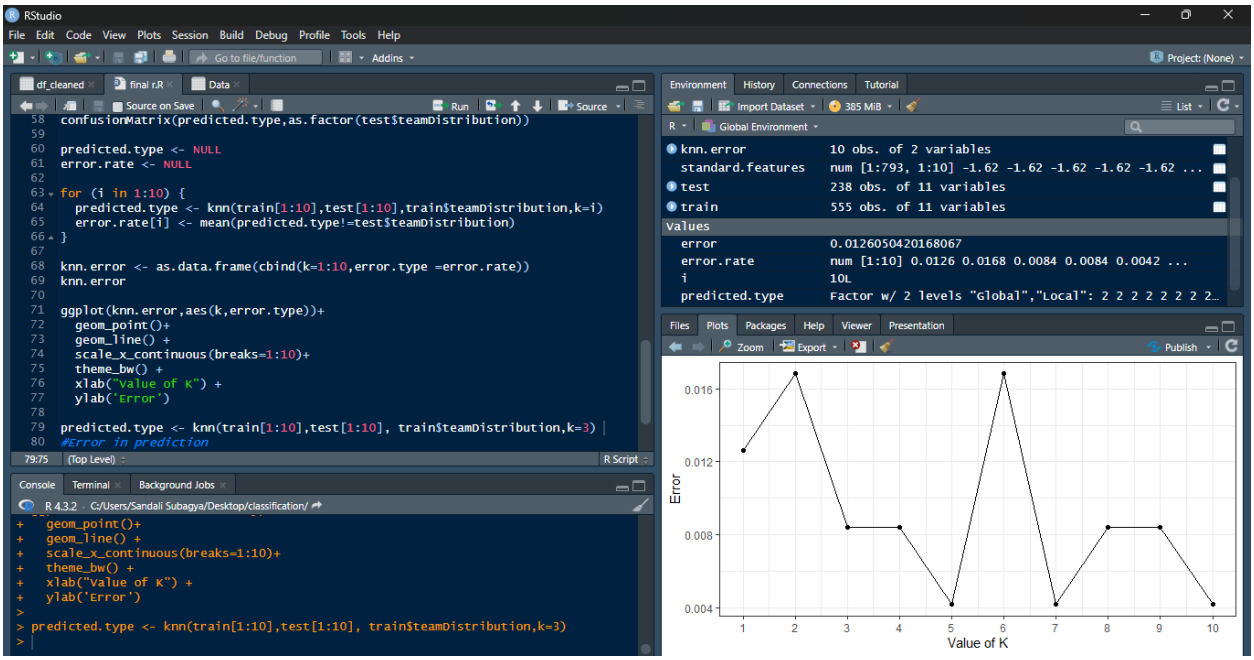
We use the set.seed() method to prioritize repeatability in our analysis. We make sure that random operations, such data splitting, provide consistent results between runs by assigning the seed to a certain number. As an example, we apply this strategy to divide the dataset according to the 'teamDistribution' variable. To be more precise, we give the training set 70% of the data and the testing set 30%. The 'train' and 'test' vectors hold the subsets that are produced as a consequence of using the sample.split() procedure. Next, the dim() method is used to analyze the training set's dimensions.

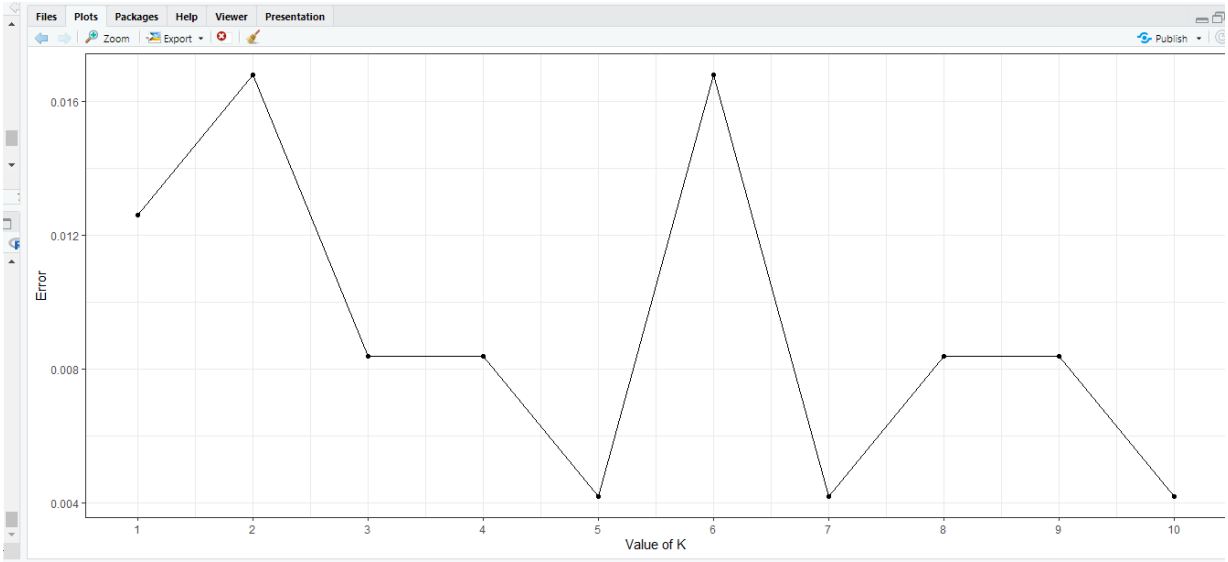
(Data Visualization by Using R)

Our findings are consistent because of this methodical methodology, and the correlation matrix that is produced as a result sheds light on the connections between different variables. Correlations, for example, can reveal relationships between various components of collaboration in a dataset pertaining to Software Engineering collaboration Assessment in an Educational Setting, which advances our comprehension of the underlying dynamics. Strong correlations are indicated by variables with correlation coefficients closer to 1 or -1, whilst low correlation is suggested by values close to 0. In order to get insight into the dynamics of cooperation in an educational setting, one may, for instance, investigate connections between assessment results and collaboration indicators. Notably, this technique contributes to a thorough evaluation strategy by enabling a detailed investigation of elements impacting collaborative success.

5.3 Choosing K Value by Visualization

A ggplot visualization illustrates the correlation between various K values and the associated prediction errors. The K value at which the error is minimized is found using the figure. Based on the visualization in this particular instance, K=3 is selected. The performance of the K-Nearest Neighbor (KNN) model is then assessed after it has been trained using K=3.






```
# KNN with optimal k (e.g., k=3)
k_optimal = 3
knn_optimal = KNeighborsClassifier(n_neighbors=k_optimal)
knn_optimal.fit(X_train, y_train)
y_pred_optimal = knn_optimal.predict(X_test)

# Error rate with optimal k
error_rate_optimal = np.mean(y_pred_optimal != y_test)
print("Error rate with optimal k (", k_optimal, "):", error_rate_optimal)

# Confusion matrix with optimal k
conf_matrix_optimal = confusion_matrix(y_test, y_pred_optimal)
print("Confusion Matrix with optimal k:")
print(conf_matrix_optimal)

# Plot the error rates
error_data = pd.DataFrame({'k': range(1, 11), 'error_rate': error_rates})
plt.plot(error_data['k'], error_data['error_rate'], marker='o')
plt.xlabel('Value of K')
plt.ylabel('Error Rate')
plt.title('Error Rate vs. K Value')
plt.show()
```

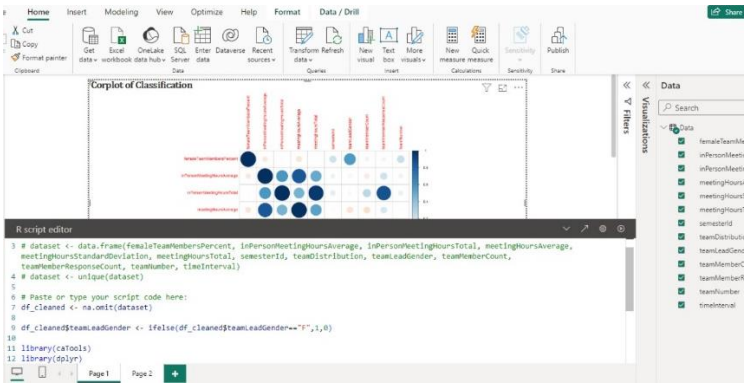
(Using Python)

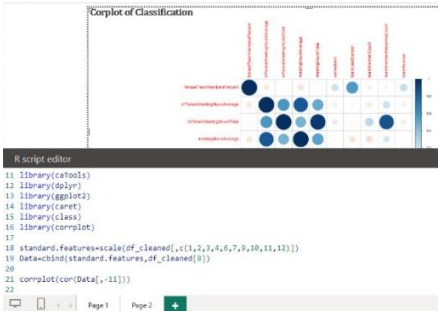
7. Result Analysis and Discussion

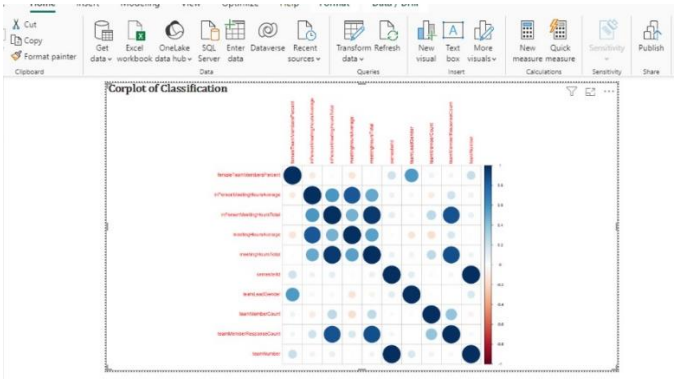
R Codes in Power BI: We combined Power BI's visualization skills with statistical computation capabilities by integrating R codes into the platform.

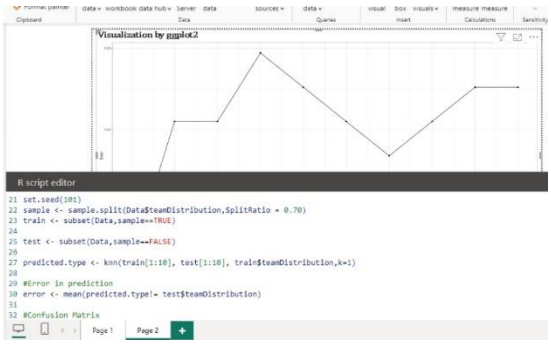
Successful Plot Replication: The identical plot was successfully reproduced when we executed the R script in Power BI. This guarantees consistency and accuracy while viewing data.

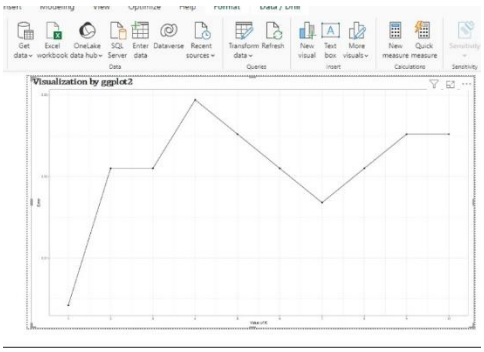
Visual Results: Visual representations of the plots produced by Power BI and R script are included, allowing for an easy comparison.











Illustrative Purpose: These images provide stakeholders with a point of reference by demonstrating the seamless transfer and precise replication of outcomes between R and Power BI.

Exploratory Phase: We only integrated R and Power BI for exploratory purposes, which allowed us to evaluate and comprehend the possible advantages of doing so.

Focused on Clustering: Our analysis's clustering section was especially enhanced by the Power BI connection, demonstrating the tool's versatility for complex data jobs. This was only one extra step.

8. Conclusion

This investigates the use of the software engineering paradigm for collaborative assessment in learning environments, based on the information supplied. Based on the ggplot visualization, scatter plot, confusion matrix, k-Nearest Neighbors (KNN) model predictions, and other pertinent data, the following conclusion has been reached:

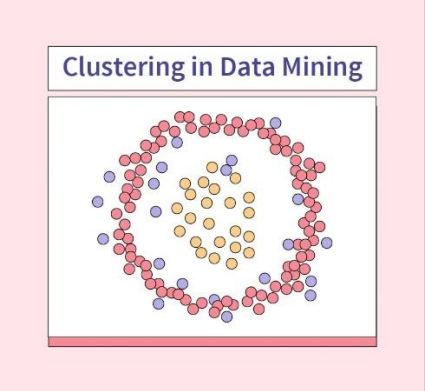
The link between a KNN model's number of neighbors (k) and the error rate that results shows a pattern in the error rate as k varies, according to the ggplot visualization. The scatter figure indicates that there is a positive association between the total number of meeting hours and the number of team members, indicating that larger teams often have more meeting hours. It is unclear from this figure alone how the average number of team members and meeting hours spent by each team member relate to one another.
The confusion matrix shows that the KNN model predictions have a high accuracy rate of 99.16%. The model appears to perform well in accurately categorizing both local and global cooperation types, based on the sensitivity and specificity values. The model's balanced accuracy is 98.70%, while the total prevalence of worldwide collaboration is 20.17%.

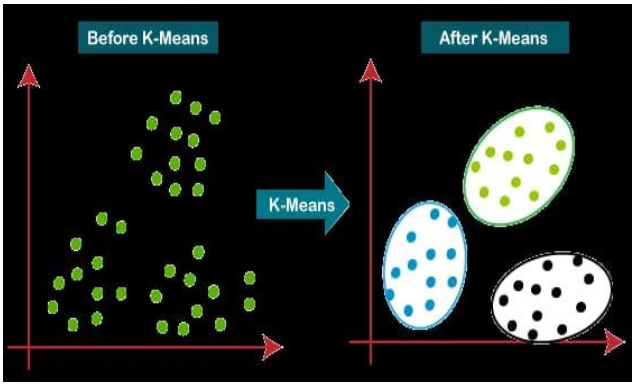
It is advised to employ additional visualizations, such as a corrplot to evaluate correlations between various variables, in order to better examine linkages within your dataset. This can offer insightful information on possible variables affecting the dynamics of collaboration.

In fact, this effectively uses a combination of visualizations and predictive models to examine the dynamics of cooperation in learning settings. The great accuracy of the KNN model suggests that it has the ability to anticipate different sorts of collaboration. Nevertheless, more investigation, such correlation analyses with a corrplot, might improve our comprehension of the variables affecting cooperation. Furthermore, given that the number of team members and meeting hours are positively correlated, it might be advantageous to investigate methods for increasing the effectiveness of cooperation in bigger teams. This could involve making modifications to team structures or putting specific interventions into place. All things considered, this study establishes a strong basis for comprehending and enhancing teamwork in software engineering classroom settings.

TASK 02

Apply Clustering to the Blockbuster Top Movie dataset using R.





1. Introduction

The film industry is a dynamic and multifaceted field encompassing the creation, distribution, and exhibition of movies. It combines creativity, technology, and commerce to produce visual storytelling experiences that captivate audiences worldwide. From the early days of silent films to the modern era of blockbusters and streaming services, the movie industry has evolved significantly, influencing culture, entertainment, and even societal norms. It involves a diverse range of professionals, including filmmakers, actors, directors, producers, distributors, and exhibitors, all working together to bring stories to life on the big screen and beyond.

Our dataset encapsulates a comprehensive overview of the cinematic landscape spanning four decades, from 1975 to 2015, focusing on the top ten most popular movies each year. This meticulously curated collection provides a wealth of information essential for understanding the evolution of the film industry and its impact on popular culture.

With a diverse array of attributes meticulously gathered for each movie, our dataset offers a multifaceted view into the dynamics of box office success, critical acclaim, and audience reception. Key features include audience freshness, (RT) audience score, RT freshness, RT score, inflation-adjusted figures for 2015, IMDb rating, movie length, ranking within each year, studio affiliation, worldwide gross earnings, and the respective release year.

These attributes collectively offer a nuanced understanding of each film's performance and reception, enabling researchers, analysts, and enthusiasts to delve deep into various aspects of cinematic trends and preferences over the past four decades. With IMDb ratings offering insights into user-generated reviews and movie length providing context on storytelling formats, our dataset encompasses both qualitative and quantitative dimensions crucial for holistic analysis.

In this analysis, we delve into the world of blockbuster cinema spanning four decades, aiming to uncover patterns, similarities, and differences among the top-grossing films of each era. By employing clustering techniques, we seek to identify distinct groups or clusters of blockbuster movies based on various attributes such as genre, critical acclaim, audience reception, and financial performance. Through this exploration, we aim to gain a deeper understanding of the underlying dynamics driving the success and cultural significance of blockbuster movies across different time periods.Our analysis not only sheds light on the cinematic landscape of each decade but also offers valuable insights for filmmakers, studios, and industry stakeholders seeking to anticipate audience preferences, trends, and market opportunities. By discerning the defining characteristics of blockbuster movies within each cluster, we can glean actionable insights to inform strategic decision-making, marketing strategies, and creative endeavors in the dynamic and competitive world of contemporary cinema.

2. Dataset

We used the “Blockbuster top ten movie per year” data set which was provided by Data.world .

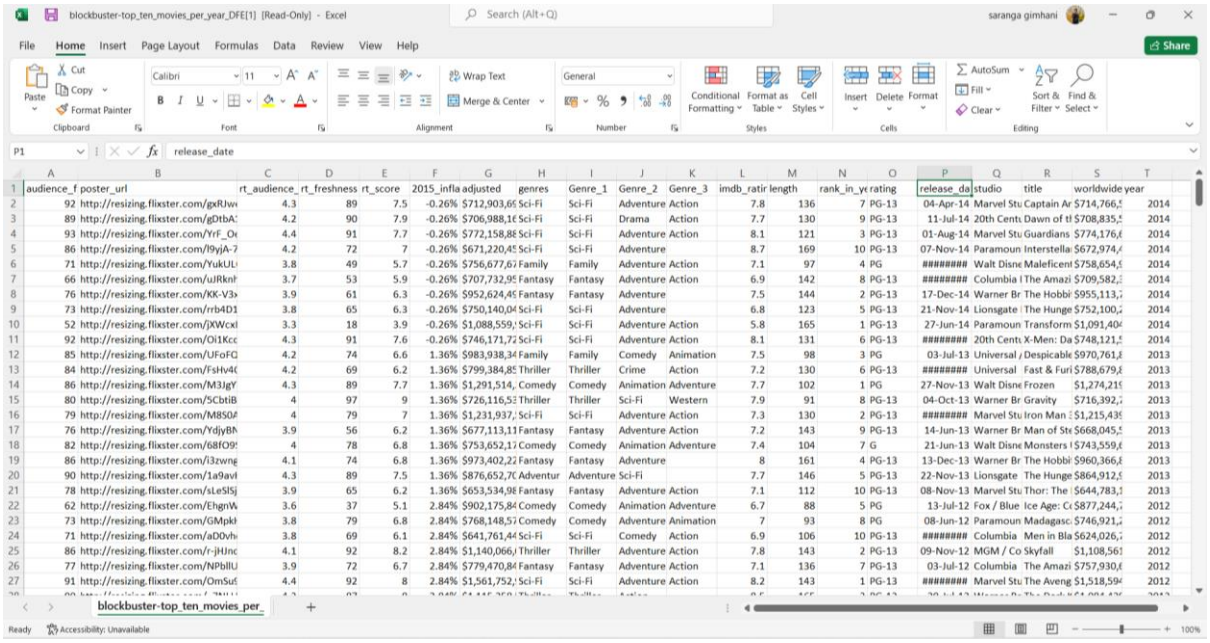
Our dataset encapsulates a comprehensive overview of the cinematic landscape spanning four decades, from 1975 to 2015, focusing on the top ten most popular movies each year. This meticulously curated collection provides a wealth of information essential for understanding the evolution of the film industry and its impact on popular culture. It includes 399 rows & 20 columns.

https://data.world/crowdflower/blockbuster-database/workspace/file?filename=blockbuster-top_ten_movies_per_year_DFE.csv

Attributes of the dataset:

audience_freshness	integer
poster_url	url
rt_audience_score	decimal
rt_freshness	integer
rt_score	decimal
2015_inflation	decimal
Adjusted	decimal
Genres	string
genre_1	String
genre_2	String
genre_3	string
imdb_rating	decimal
Length	integer
rank_in_year	integer
rating	string
release_date	string
Studio	String
Title	String
worldwide_gross	decimal
Year	year

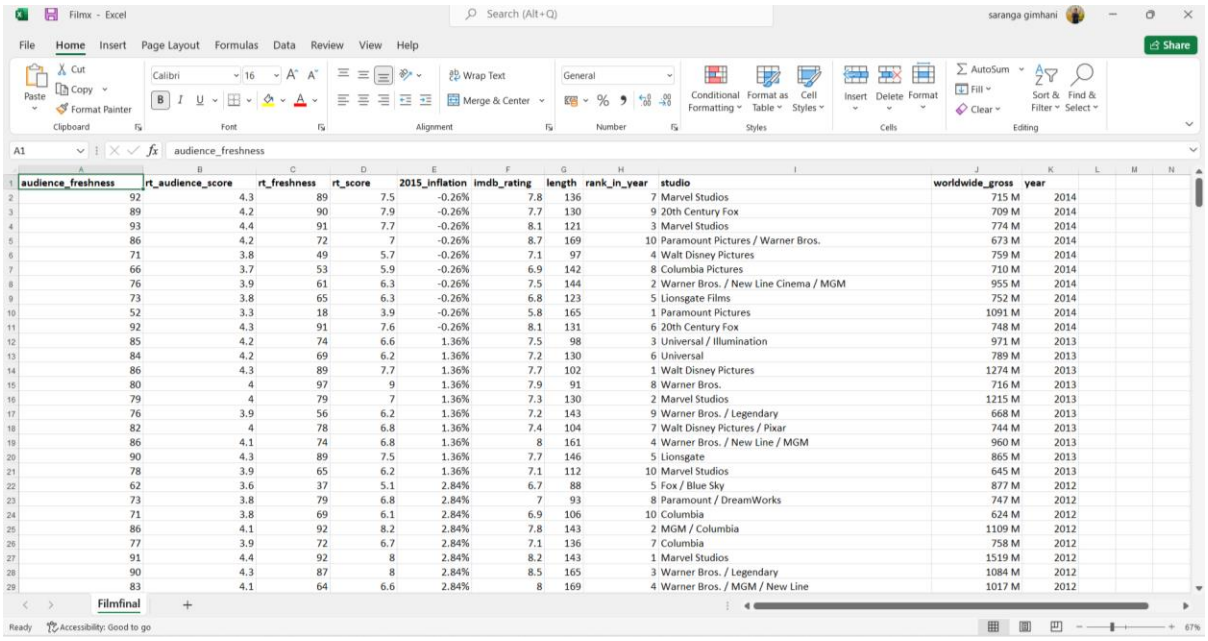
In attributes **rt** means Rotten Tomato audience/critic rating.



3. Explanation and Preparation of Dataset.

a. Data pre-processing

In the dataset preprocessing phase, we made several adjustments to enhance its suitability for analysis. Firstly, we removed several attributes that were deemed non-essential or redundant for our purposes. This included the poster_url, adjusted, genres, genre_1, genre_2, genre_3, release_date, and rating attributes. These attributes either contained URLs linking to movie posters, represented adjusted box office revenue figures, or denoted movie genres and release dates, none of which were utilized in our subsequent clustering analysis. Additionally, certain attributes were converted to numeric data type to facilitate numerical analysis and computations, while others were retained as text data type to preserve their categorical nature and facilitate qualitative analysis. These adjustments were made to streamline the dataset and ensure its suitability for our clustering analysis, allowing us to focus on the most relevant features and attributes for identifying patterns and trends among blockbuster movies.



audience_freshness	rt_audience_score	rt_freshness	rt_score	2015_inflation	imdb_rating	length	rank_in_year	studio	worldwide_gross	year
92	4.3	89	7.5	-0.26%	7.8	136	7	Marvel Studios	715 M	2014
89	4.2	90	7.9	-0.26%	7.7	130	9	20th Century Fox	709 M	2014
93	4.4	91	7.7	-0.26%	8.1	121	3	Marvel Studios	774 M	2014
86	4.2	72	7	-0.26%	8.7	169	10	Paramount Pictures / Warner Bros.	673 M	2014
71	3.8	49	5.7	-0.26%	7.1	97	4	Walt Disney Pictures	759 M	2014
66	3.7	53	5.9	-0.26%	6.9	142	8	Columbia Pictures	710 M	2014
76	3.9	61	6.3	-0.26%	7.5	144	2	Warner Bros. / New Line Cinema / MGM	955 M	2014
73	3.8	65	6.3	-0.26%	6.8	123	5	Lionsgate Films	752 M	2014
52	3.3	18	3.9	-0.26%	5.8	165	1	Paramount Pictures	1091 M	2014
92	4.3	91	7.6	-0.26%	8.1	131	6	20th Century Fox	748 M	2014
85	4.2	74	6.6	1.36%	7.5	98	3	Universal / Illumination	971 M	2013
84	4.2	69	6.2	1.36%	7.2	130	6	Universal	789 M	2013
86	4.3	89	7.7	1.36%	7.7	102	1	Walt Disney Pictures	1274 M	2013
80	4	97	9	1.36%	7.9	91	8	Warner Bros.	716 M	2013
79	4	79	7	1.36%	7.3	130	2	Marvel Studios	1215 M	2013
76	3.9	56	6.2	1.36%	7.2	143	9	Warner Bros. / Legendary	668 M	2013
82	4	78	6.8	1.36%	7.4	104	7	Walt Disney Pictures / Pixar	744 M	2013
86	4.1	74	6.8	1.36%	8	161	4	Warner Bros. / New Line / MGM	960 M	2013
90	4.3	89	7.5	1.36%	7.7	146	5	Lionsgate	865 M	2013
78	3.9	65	6.2	1.36%	7.1	112	10	Marvel Studios	645 M	2013
62	3.6	37	5.1	2.84%	6.7	88	5	Fox / Blue Sky	877 M	2012
73	3.8	79	6.8	2.84%	7	93	8	Paramount / DreamWorks	747 M	2012
71	3.8	69	6.1	2.84%	6.9	106	10	Columbia	624 M	2012
86	4.1	92	8.2	2.84%	7.8	143	2	MGM / Columbia	1109 M	2012
77	3.9	72	6.7	2.84%	7.1	136	7	Columbia	758 M	2012
91	4.4	92	8	2.84%	8.2	143	1	Marvel Studios	1519 M	2012
90	4.3	87	8	2.84%	8.5	165	3	Warner Bros. / Legendary	1084 M	2012
83	4.1	64	6.6	2.84%	8	169	4	Warner Bros. / MGM / New Line	1017 M	2012

3.b Data Explanation

audience_freshness	Audience who rated a movie positively and recommend to others.
rt_audience_score	The overall satisfaction level of the move's audience. A higher audience score suggest that a larger proportion of viewers enjoyed movie.
rt_freshness	The proportion of professional critics who gave the movie positive review. Higher freshness indicates positive opinion of the movie.
rt_score	It represent the overall reception of the movie, taking into account both critical and audience opinions.
2015_inflation	Refers to the adjustment of monetary values from previous years to their equivalent purchasing power in 2015.
imdb_rating	The rating of the movie on the IMDb platform
length	Duration of the movies in minutes
rank_in_year	The rank in year indicates the position of a movie relative to others released in same year .

studio	Production company responsible for producing and telecast movie.
worldwide gross	The gross revenue generated by the movie at the box office in Millions
year	The year of the movie release

4.Data mining

The practice of evaluating a huge batch of data to find trends and patterns is known as data mining. Corporations may employ data mining for a variety of purposes, including learning about what customers want to buy and detecting fraud and spam. This type of data mining has recently come under fire, as consumers are often unaware that their personal information is being mined, especially when it is used to influence 12 preferences. A collection of algorithms is utilized to extract the important correlations in the data for each sort of data mining application. These approaches differ based on the problem that has to be solved. Clustering is most prominent approach that can used in Data mining.

4.a Clustering

Clustering is a machine learning technique that groups data items based on their similarity. K-means clustering, and hierarchical clustering are two popular data mining clustering techniques. This is a prominent methodology for statistical data analysis and an unsupervised learning method. As an unsupervised learning method, clustering doesn't rely on labeled data or predetermined groups. Instead, the program uses the similarity between data points to automatically identify clusters. Applications for clustering algorithms include customer segmentation, picture analysis, anomaly detection, and more.

4.b k- means clustering

In observations can be divided into K clusters using the K-means clustering technique. Each obs ervation is to be assigned to the cluster with the closest mean or centroid, which acts as a proto type for the cluster, using vector quantization. Typically, unsupervised algorithms make inferen ces from datasets using only input vectors without referring to known, or labelled, outcomes. K-means clustering, which was initially created for signal processing, is now frequently used in ma chine learning to divide data points into K clusters based on their similarity. The objective is to p roduce internal homogeneous and distinguishable clusters by minimizing the sum of squared di stances between the data points and their respective cluster centroids.

How the K-means algorithm works:

The K-means technique in data mining starts with a first group of randomly picked centroids, w hich serve as the starting points for each cluster, and then performs iterative (repetitive) compu tations to optimize the centroids' placements.

- Step 1: Select k randomly selected spots to serve as cluster centers.
- Step 2: Calculate the Euclidean distance between each datapoint and each cluster center.
- Step 3: Assign the datapoint to the cluster with the shortest Euclidean distance.
- Step 4: Recalculate the cluster centroids using the datapoints given in step 3.
- Step-5: Repeat steps 2–4 until the centroids stop updating or until the maximum number of iterations is achieved.

4.c Elbow Method

For a range of k values, the elbow approach uses the k-means clustering algorithm.

1.It calculates a score for each value of k.

2.It calculates the distortion score by default, which is the sum of the squared distances between each datapoint and its assigned center.

3.When we plot the value of k against this score, we get an elbow-shaped plot, thus the name.

4.The elbow point is where the distortion decreases the maximum, and here is where the ideal value of k is found.

4.d. Visualization tools available in R for clustering

By writing a few lines of code in R, we can build visually appealing data visualizations. We utilize R's many features to do this. Data visualization is a useful approach for visualizing data and obtaining insight into it.

- **Cluster plot** - Clusters are small groups that can be found in scatter plots. K- means clustering is the most often used unsupervised machine learning approach splitting given data set into a collection of k groups where k is the number of groups that the analyst has pre- specified. Cluster analysis and fact analysis are methods for categorizing objects into fewer components in Clusters.
- **Scatterplot Matrix** - A scatterplot matrix can reveal the relationships between numerous variables. The matrix can reveal relationships between variables after charting two-way combinations of the variables to emphasize which associations are likely to be essential. Outliers several scatter plots can also be detected using the matrix.

- **Distance Matrix** - This plot displays the distances between rows of a matrix or data frame. Factoextra R package is used in this process and that package facilitating the extraction and visualization of exploratory multivariable data analysis output.
- **Dot plot** - A dot plot or a dot chart is comparable to scatter plot. but the main difference is that in R the dot plot shows the index (each category) on the vertical axis, allowing to examine the value of each observation.

5. Implementation in R

In R, clustering refers to the grouping of similar data into groups or clusters in order to differentiate one group from the others. R can also be used to express this in a graphical way.

5.1 R packages

- **Cluster package** -The cluster package in R is a powerful tool for performing clustering analysis, a technique used to group similar data points together based on their characteristics. It offers functions for popular clustering algorithms such as k-means, hierarchical clustering. With cluster, users can explore their data, identify underlying patterns, and gain insights into the structure and relationships within their datasets.

- **ggplot package -** The ggplot2 package in R is a popular data visualization package that allows users to create high-quality, customizable plots with ease. With ggplot2, users can quickly add aspects for investigating relationships within their data, generate a wide range of plots, and customise themes and aesthetics. It's frequently used for producing publication-quality visuals for reports and presentations, as well as for exploratory data analysis.

- **Factoextra package** - The factoextra package in R is a versatile tool for enhancing the interpretation and visualization of multivariate analysis results. It offers a wide range of visualization functions, including biplots, scree plots, and cluster dendrograms, to help users gain insights into their data and communicate findings effectively.

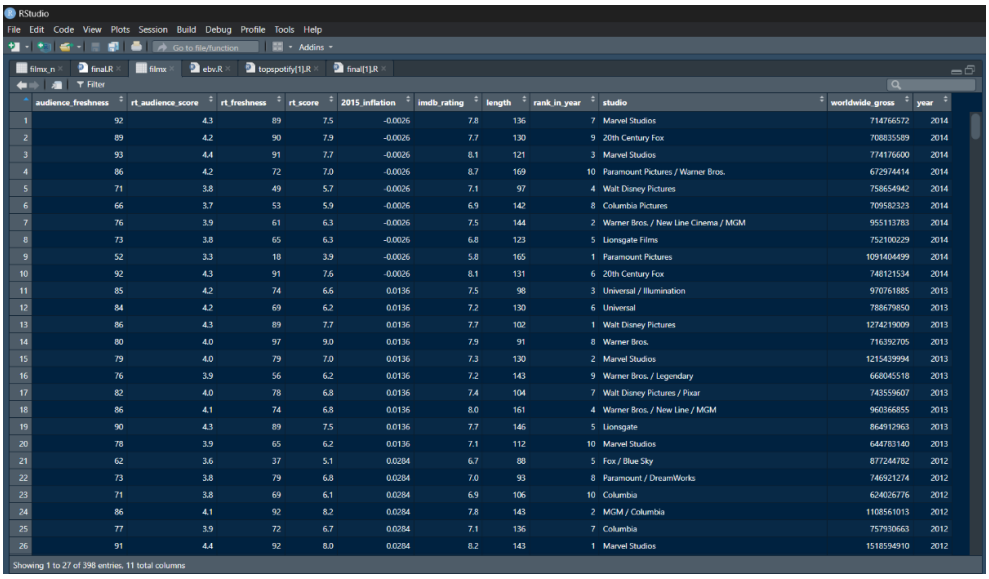
5.2 Implementation

Import dataset from Excel to R Studio.

Read the data set in R for that use the following r code for it.

```
> library(readxl)
> filmx <- read_excel("C:/Users/ASUS/Desktop/Filmx.xlsx")
> View(filmx)
```

(Import dataset by Using R)



	audience_freshness	rt_audience_score	rt_freshness	rt_score	2015_inflation	imdb_rating	length	rank_in_year	studio	worldwide_gross	year
1	92	4.3	89	7.5	-0.0026	7.8	136	7	Marvel Studios	714766572	2014
2	89	4.2	90	7.9	-0.0026	7.7	130	9	20th Century Fox	708835589	2014
3	93	4.4	91	7.7	-0.0026	8.1	121	3	Marvel Studios	774176600	2014
4	86	4.2	72	7.0	-0.0026	8.7	169	10	Paramount Pictures / Warner Bros.	672974414	2014
5	71	3.8	49	5.7	-0.0026	7.1	97	4	Walt Disney Pictures	758654942	2014
6	66	3.7	53	5.9	-0.0026	6.9	142	8	Columbia Pictures	709582323	2014
7	76	3.9	61	6.3	-0.0026	7.5	144	2	Warner Bros. / New Line Cinema / MGM	955113783	2014
8	73	3.8	65	6.3	-0.0026	6.8	123	5	Lionsgate Films	752100229	2014
9	52	3.3	18	3.9	-0.0026	5.8	165	1	Paramount Pictures	1091404499	2014
10	92	4.3	91	7.6	-0.0026	8.1	131	6	20th Century Fox	748121534	2014
11	85	4.2	74	6.6	0.0136	7.5	98	3	Universal / Illumination	970761885	2013
12	84	4.2	69	6.2	0.0136	7.2	130	6	Universal	788679850	2013
13	86	4.3	89	7.7	0.0136	7.7	102	1	Walt Disney Pictures	1274219009	2013
14	80	4.0	97	9.0	0.0136	7.9	91	8	Warner Bros.	716392705	2013
15	79	4.0	79	7.0	0.0136	7.3	130	2	Marvel Studios	1215439994	2013
16	76	3.9	56	6.2	0.0136	7.2	143	9	Warner Bros. / Legendary	668045518	2013
17	82	4.0	78	6.8	0.0136	7.4	104	7	Walt Disney Pictures / Pixar	743559607	2013
18	86	4.1	74	6.8	0.0136	8.0	161	4	Warner Bros. / New Line / MGM	960366855	2013
19	90	4.3	89	7.5	0.0136	7.7	146	5	Lionsgate	864912963	2013
20	78	3.9	65	6.2	0.0136	7.1	112	10	Marvel Studios	644783140	2013
21	62	3.6	37	5.1	0.0284	6.7	88	5	Fox / Blue Sky	877244782	2012
22	73	3.8	79	6.8	0.0284	7.0	93	8	Paramount / DreamWorks	746921274	2012
23	71	3.8	69	6.1	0.0284	6.9	106	10	Columbia	624026776	2012
24	86	4.1	92	8.2	0.0284	7.8	143	2	MGM / Columbia	1108561013	2012
25	77	3.9	72	6.7	0.0284	7.1	136	7	Columbia	757930663	2012
26	91	4.4	92	8.0	0.0284	8.2	143	1	Marvel Studios	1518594910	2012

Showing 1 to 27 of 398 entries, 11 total columns

```
# Importing the dataset
filmx = pd.read_excel("C:\\Users\\Sandali Subagya\\Desktop\\ttt\\Filmx.xlsx")
```

(Using Python)

- Explore the Data set

We used ‘head'() function is used to collect the first 6 rows in this dataset. So we could figure o ut what kind of data is in it.

```
> #Explore the Data set
> head(filmx)
# A tibble: 6 × 11
  audience_freshness rt_audience_score rt_freshness rt_score `2015_inflation` imdb_rating length rank_in_year studio
               <dbl>             <dbl>        <dbl>    <dbl>           <dbl>       <dbl>  <dbl>        <dbl> <chr>
1                 92               4.3           89      7.5         -0.0026         7.8    136            7 Marvel Studi…
2                 89               4.2           90      7.9         -0.0026         7.7    130            9 20th Century…
3                 93               4.4           91      7.7         -0.0026         8.1    121            3 Marvel Studi…
4                 86               4.2           72      7           -0.0026         8.7    169           10 Paramount Pi…
5                 71               3.8           49      5.7         -0.0026         7.1     97            4 Walt Disney …
6                 66               3.7           53      5.9         -0.0026         6.9    142            8 Columbia Pic…
# ℹ 2 more variables: worldwide_gross <dbl>, year <dbl>
>
```

We used ‘tail ()' function is used to collect the last 6 rows in this dataset.

```
> tail(filmx)
# A tibble: 6 × 11
  audience_freshness rt_audience_score rt_freshness rt_score `2015_inflation` imdb_rating length rank_in_year
               <dbl>             <dbl>        <dbl>    <dbl>            <dbl>       <dbl>  <dbl>        <dbl>
1                 56               3.2           38      4.6             3.39         6.4    136            7
2                 53               3.2           63      6.2             3.39         6.3    109            5
3                 60               3.2           56      6.2             3.39         6.2    100            8
4                 77               3.6            0      0               3.39         6.4    103           10
5                 82               3.4           89      6.7             3.39         7.1    113            6
6                 67               3.2           74      6.8             0.313        6.7    106            7
# ℹ 3 more variables: studio <chr>, worldwide_gross <dbl>, year <dbl>
>
```

From the code “names ()” inspect the names of the columns.

```
> names(filmx)
 [1] "audience_freshness" "rt_audience_score"  "rt_freshness"       "rt_score"           "2015_inflation"
 [6] "imdb_rating"        "length"             "rank_in_year"       "studio"             "worldwide_gross"
[11] "year"
>
```

From the function“summary ()” inspect the summary of the dataset.

```
> summary(filmx)
 audience_freshness rt_audience_score  rt_freshness       rt_score
 Min.   : 24.00     Min.   :1.700     Min.   :  0.00   Min.   :0.000
 1st Qu.: 61.00     1st Qu.:3.200     1st Qu.: 54.00   1st Qu.:5.700
 Median : 75.00     Median :3.500     Median : 75.50   Median :6.600
 Mean   : 72.27     Mean   :3.479     Mean   : 69.38   Mean   :6.503
 3rd Qu.: 85.00     3rd Qu.:3.700     3rd Qu.: 88.00   3rd Qu.:7.500
 Max.   :100.00     Max.   :4.500     Max.   :100.00   Max.   :9.100
 2015_inflation     imdb_rating       length       rank_in_year      studio
 Min.   :-0.0026   Min.   :4.400   Min.   : 27.0   Min.   : 1.00   Length:398
 1st Qu.: 0.2192   1st Qu.:6.500   1st Qu.:102.2   1st Qu.: 3.00   Class :character
 Median : 0.5493   Median :7.000   Median :117.0   Median : 6.00   Mode  :character
 Mean   : 0.8664   Mean   :7.053   Mean   :119.3   Mean   : 5.53
 3rd Qu.: 1.1944   3rd Qu.:7.675   3rd Qu.:131.8   3rd Qu.: 8.00
 Max.   : 3.3888   Max.   :9.000   Max.   :201.0   Max.   :10.00
 worldwide_gross         year
 Min.   :3.467e+07   Min.   :1975
 1st Qu.:1.086e+08   1st Qu.:1985
 Median :3.312e+08   Median :1995
 Mean   :3.909e+08   Mean   :1995
 3rd Qu.:5.825e+08   3rd Qu.:2005
 Max.   :2.749e+09   Max.   :2014
```

We used “str ()” function to obtain the result, which compactly displays the internal structure.

```
> str(filmx)
tibble [398 × 11] (S3: tbl_df/tbl/data.frame)
 $ audience_freshness: num [1:398] 92 89 93 86 71 66 76 73 52 92 ...
 $ rt_audience_score : num [1:398] 4.3 4.2 4.4 4.2 3.8 3.7 3.9 3.8 3.3 4.3 ...
 $ rt_freshness      : num [1:398] 89 90 91 72 49 53 61 65 18 91 ...
 $ rt_score          : num [1:398] 7.5 7.9 7.7 7 5.7 5.9 6.3 6.3 3.9 7.6 ...
 $ 2015_inflation    : num [1:398] -0.0026 -0.0026 -0.0026 -0.0026 -0.0026 -0.0026 -0.0026 -0.0026 -0.0026 -0.0026 ...
 $ imdb_rating       : num [1:398] 7.8 7.7 8.1 8.7 7.1 6.9 7.5 6.8 5.8 8.1 ...
 $ length            : num [1:398] 136 130 121 169 97 142 144 123 165 131 ...
 $ rank_in_year      : num [1:398] 7 9 3 10 4 8 2 5 1 6 ...
 $ studio            : chr [1:398] "Marvel Studios" "20th Century Fox" "Marvel Studios" "Paramount Pictures / Warner Bros." ...
 $ worldwide_gross   : num [1:398] 7.15e+08 7.09e+08 7.74e+08 6.73e+08 7.59e+08 ...
 $ year              : num [1:398] 2014 2014 2014 2014 2014 ...
```

- Check the dimensions of the data set.

We can get the number of rows from “nrow ()” function, the number of columns from “ncol ()”, and from “dim ()” function the dimension of the dataset.

```
> nrow(filmx)
[1] 398
> ncol(filmx)
[1] 11
> dim(filmx)
[1] 398  11
```

Install require packages These were the r codes use for installing and activate “cluster package”.

```
> library(cluster)
```

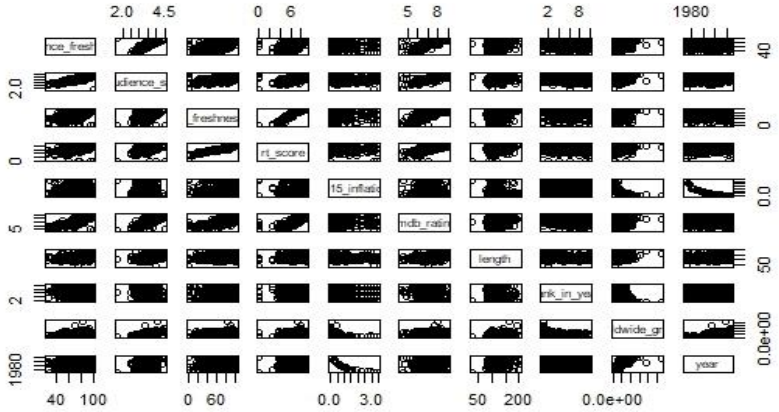
```python
import pandas as pd
import numpy as np
import seaborn as sns
from sklearn.preprocessing import MinMaxScaler
from sklearn.cluster import KMeans
from sklearn.metrics import pairwise_distances
import matplotlib.pyplot as plt

# Importing the dataset
filmx = pd.read_excel("C:\\Users\\Sandali Subagya\\Desktop\\ttt\\Filmx.xlsx")

# Exploratory Data Analysis
print(filmx.info())
print(filmx.describe())
print(filmx.head())

# Visualization - pairs plot
sns.pairplot(filmx.iloc[:, :-1])
plt.show()

# Normalization
def normalize(df):
    if pd.api.types.is_numeric_dtype(df):
        return (df - df.min()) / (df.max() - df.min())
    return df

filmx_n = filmx.copy()
filmx_n.iloc[:, :-1] = filmx_n.iloc[:, :-1].applymap(normalize)

# Rearrange dataset
selected_columns = [filmx_n.columns[-2]] + list(filmx_n.columns[:-2]) + [filmx_n.columns[-1]]
filmx_n = filmx_n[selected_columns]
```

```python
# Exclude non-numeric 'studio' column from distance calculation
numeric_columns = filmx_n.columns[:-2]
distance = pairwise_distances(filmx_n[numeric_columns], metric='euclidean')

# Visualize distance matrix using matplotlib imshow
plt.imshow(distance, cmap='viridis', interpolation='nearest')
plt.title('Distance Matrix')
plt.colorbar(label='Euclidean Distance')
plt.xticks(np.arange(len(filmx_n)), filmx_n.iloc[:, -1], rotation='vertical')
plt.yticks(np.arange(len(filmx_n)), filmx_n.iloc[:, -1])
plt.show()

# K-Means Clustering
kmeans = KMeans(n_clusters=3, random_state=123)
filmx_n['cluster'] = kmeans.fit_predict(filmx_n[numeric_columns])

# Cluster Tendency
tendency = kmeans.inertia_

# Elbow Method
wss = []
for k in range(1, 11):
    kmeans = KMeans(n_clusters=k, random_state=123)
    kmeans.fit(filmx_n[numeric_columns])
    wss.append(kmeans.inertia_)

# Visualize Elbow Method
plt.plot(range(1, 11), wss, marker='o')
plt.xlabel('Number of Clusters')
plt.ylabel('Within Sum of Squares (WSS)')
plt.title('Elbow Method')
plt.show()
```

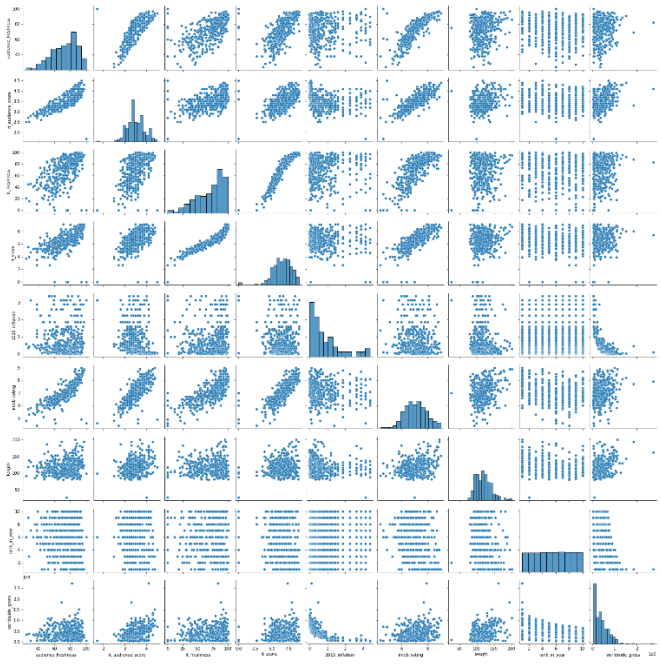
(Python code for that part)

Then we have to choose the columns that we need for clustering. In this step, we removed the "studio" column which does not impact the K means clustering method. Following that, the pair function was used to generate a matrix of scatter plots which helps to understand the relations hip between distinct variables.

```
> film1=filmx[,-9]
> pairs(filmx)
```



(Pairs plot in R)

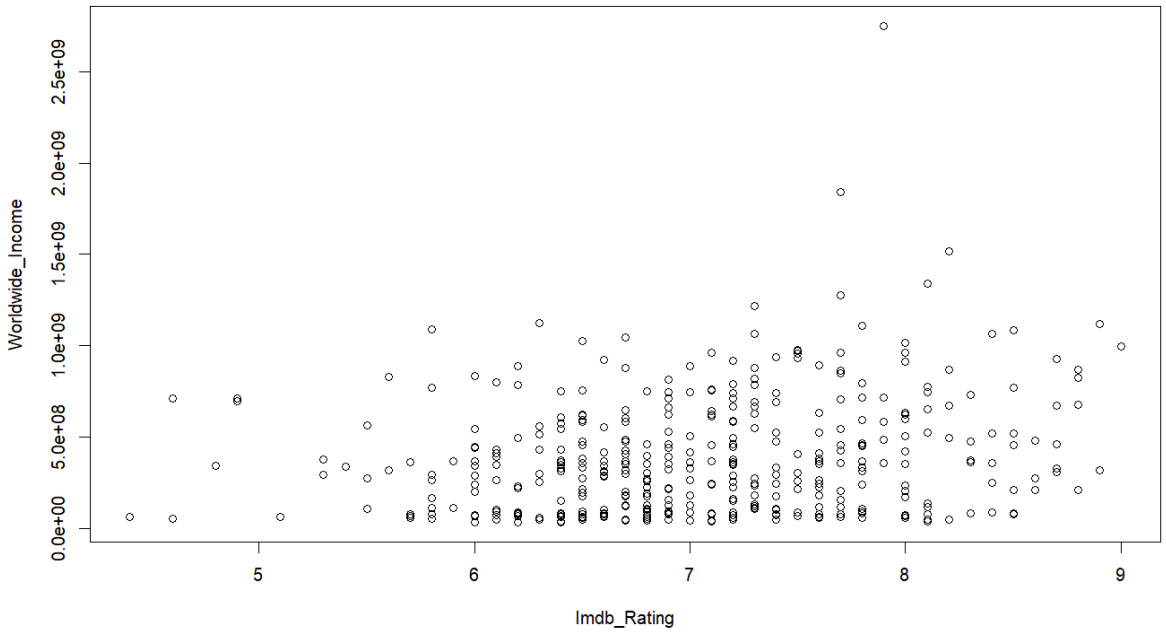


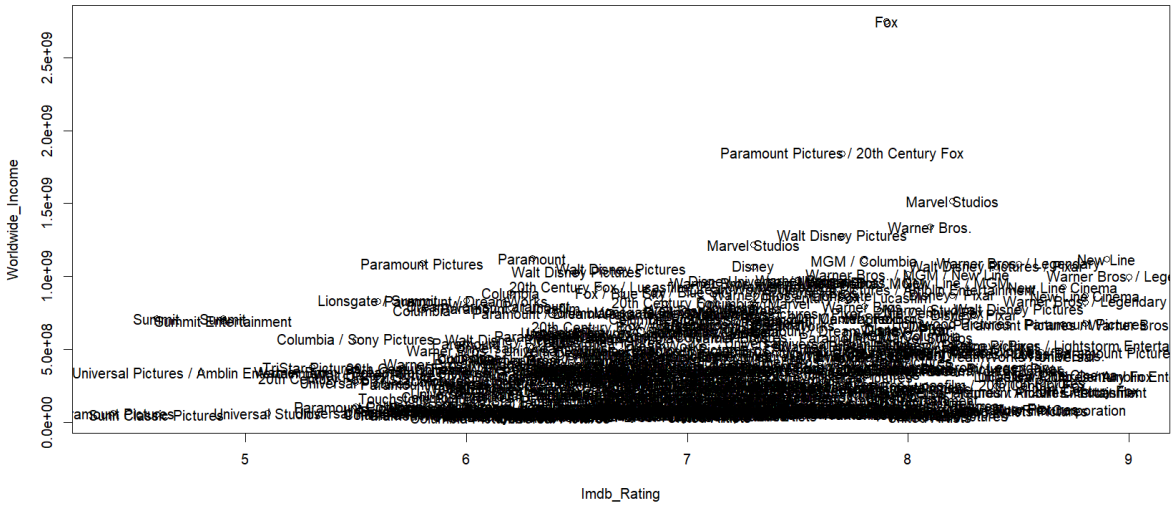
(Pairs plot in Python)

- Create the Dot Plot

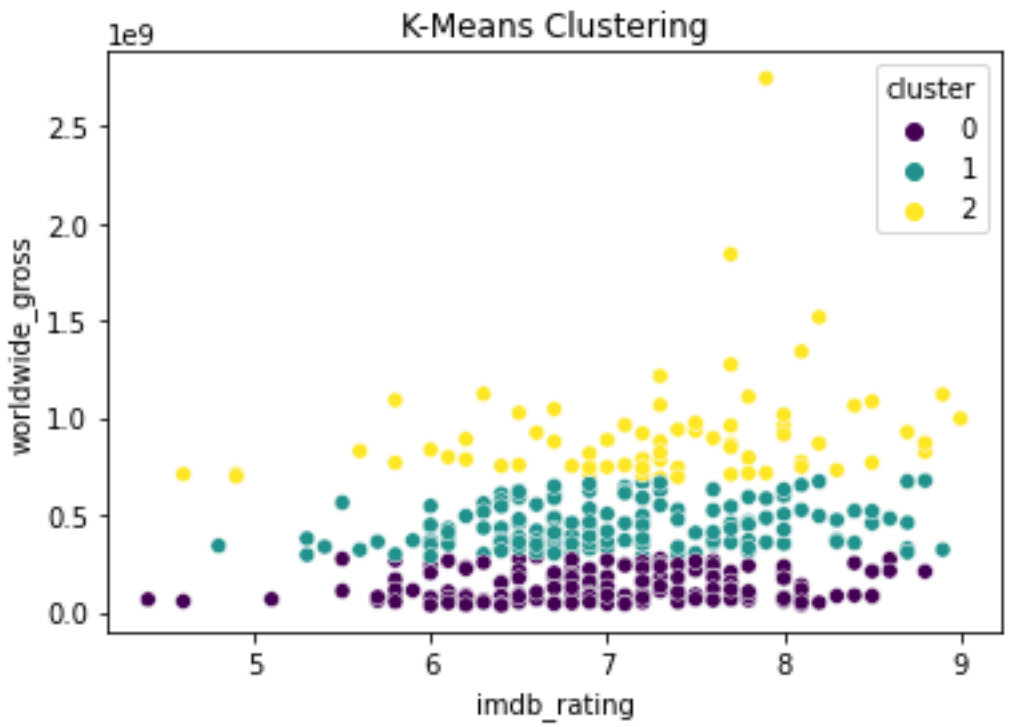
Following R codes were used to generate the relationship between the “world_wide_gross” an d “Imdb_rating”

```
> #plots###
> plot(film1$worldwide_gross~film1$imdb_rating,xlab="Imdb_Rating",ylab="Worldwide_Income")
> with(film1,text(film1$worldwide_gross~film1$imdb_rating, labels=filmx$studio))
```





(Dot plot in R)



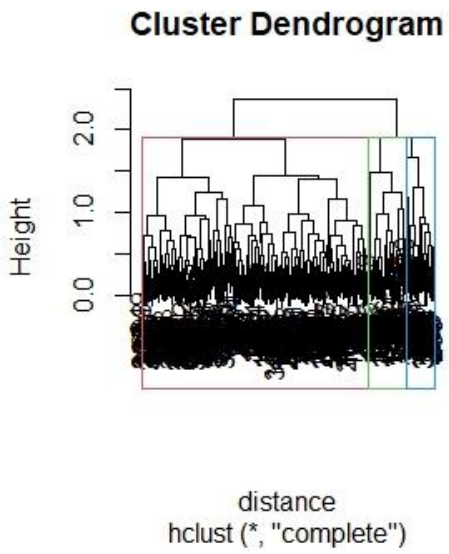
(Dot plot in Python)

- **hierarchical clustering**

```
63  ##Hierarchical clstering##
64
65  hc_filmx_n=hclust(distance,method = "complete")
66  hc_filmx_n
67  plot(hc_filmx_n)
68  rect.hclust(hc_filmx_n,k=3,border=2:5)
69
```

By grouping data points according to how similar they are, hierarchical clustering creates a hier archy of clusters. This hierarchy is represented graphically by the dendrogram.

The hierarchical clustering in the code is handled by hclust, and the dendrogram is produced usi ng plot. As clusters combine throughout the clustering process, the distance between them is displayed on the plot's rightmost side.



normalization

The process of rescaling the values of the variables in data collection. similar scale is referred to as normalization. Then we normalized the dataset using 'normalise' function.

```
> #normalize function
> normalize<-function(df)
+ {
+   return(((df- min(df)) /(max(df)-min(df))*(1-0))+0)
+ }
> head(filmx)
# A tibble: 6 x 11
  audience_freshness rt_audience_score rt_freshness rt_score `2015_inflation`
               <dbl>             <dbl>        <dbl>    <dbl>            <dbl>
1                 92               4.3           89      7.5          -0.0026
2                 89               4.2           90      7.9          -0.0026
3                 93               4.4           91      7.7          -0.0026
4                 86               4.2           72      7            -0.0026
5                 71               3.8           49      5.7          -0.0026
6                 66               3.7           53      5.9          -0.0026
# i 6 more variables: imdb_rating <dbl>, length <dbl>, rank_in_year <dbl>,
#   studio <chr>, worldwide_gross <dbl>, year <dbl>
```

```
> filmx_n=filmx[,c(1,2,3,4,5,6,7,8,10,11)]
> filmx_n=as.data.frame(lapply(filmx_n,normalize))
> filmx_n$studio=studio
>
```

Scale the data scaling the data is an important pre-processing step in data analysis and machine learning and can help to ensure that the results of the analysis are accurate and unbiased. Fro m the following code, we can scale the data set.

```
###rearrange dataset##

filmx_n<-filmx_n[,c(9,1,2,3,4,5,6,7,8,10,11)]
View(filmx_n)
```

```
> filmx_n<-filmx_n[,c(9,1,2,3,4,5,6,7,8,10,11)]
> View(filmx_n)
```

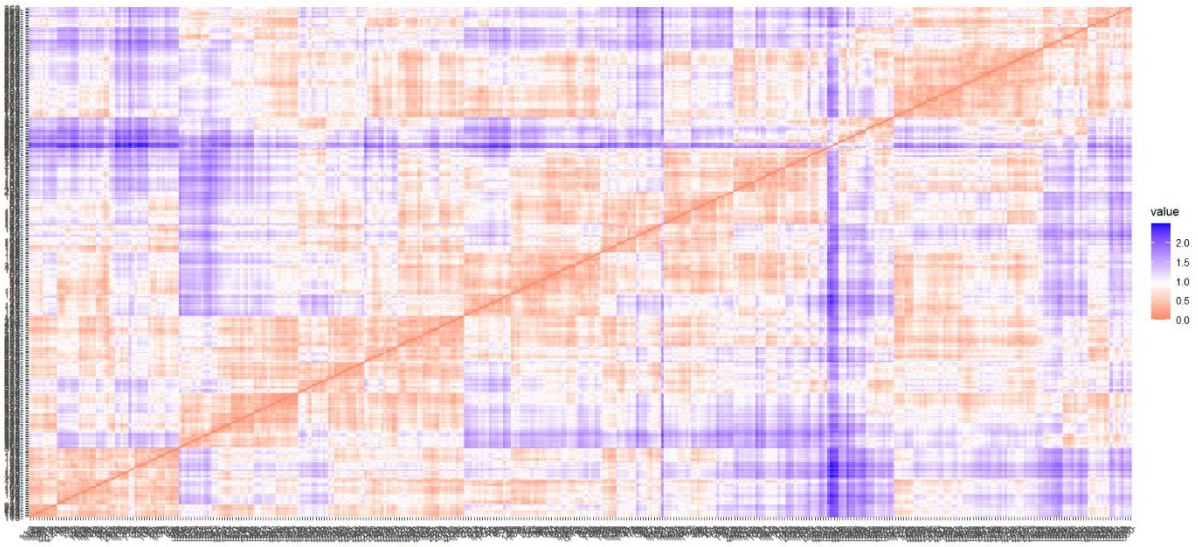
Calculate the distance matrix using “Euclidean” method After scaling the data we can run the fo llowing code to get the distance matrix. Below distance matrix shows the numbers rounded to 4 decimal.

```
> distance=dist(filmx_n,method = "euclidean")
Warning message:
In dist(filmx_n, method = "euclidean") : NAs introduced by coercion
> print(distance,digits = 4)
        1       2       3       4       5       6       7       8       9      10
2   0.24800
3   0.48294 0.71420
       11      12      13      14      15      16      17      18      19      20
2
3
       21      22      23      24      25      26      27      28      29      30
2
3
       31      32      33      34      35      36      37      38      39      40
2
3
       41      42      43      44      45      46      47      48      49      50
2
3
       51      52      53      54      55      56      57      58      59      60
2
3
       61      62      63      64      65      66      67      68      69      70
2
3
       71      72      73      74      75      76      77      78      79      80
2
3
       81      82      83      84      85      86      87      88      89      90
```

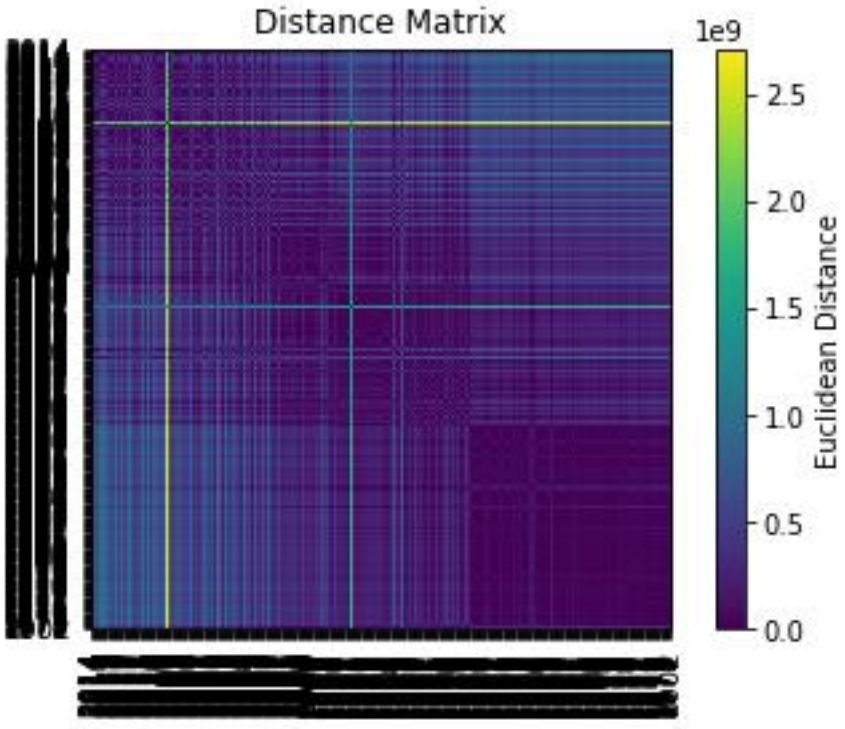
The Euclidean distance is the most used distance measure in clustering. A scatterplot matrix can reveal the relationships between numerous variables. After charting two-way combinations of the studios to highlight which associations are likely to be important, the matrix can reveal linka ges between variables.

The following code was used to visualize the distance matrix.

```
> distance=dist(filmx_n,method = "euclidean")
> fviz_dist(distance)
>
```



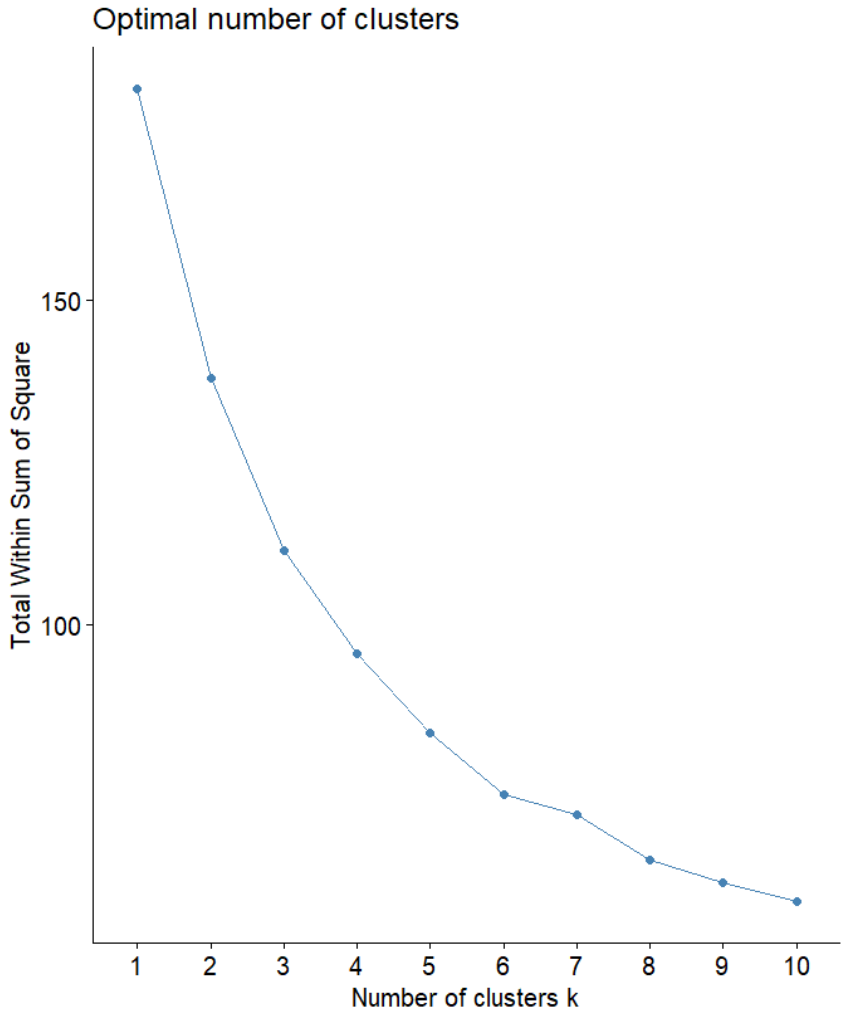
(Heatmap in R)



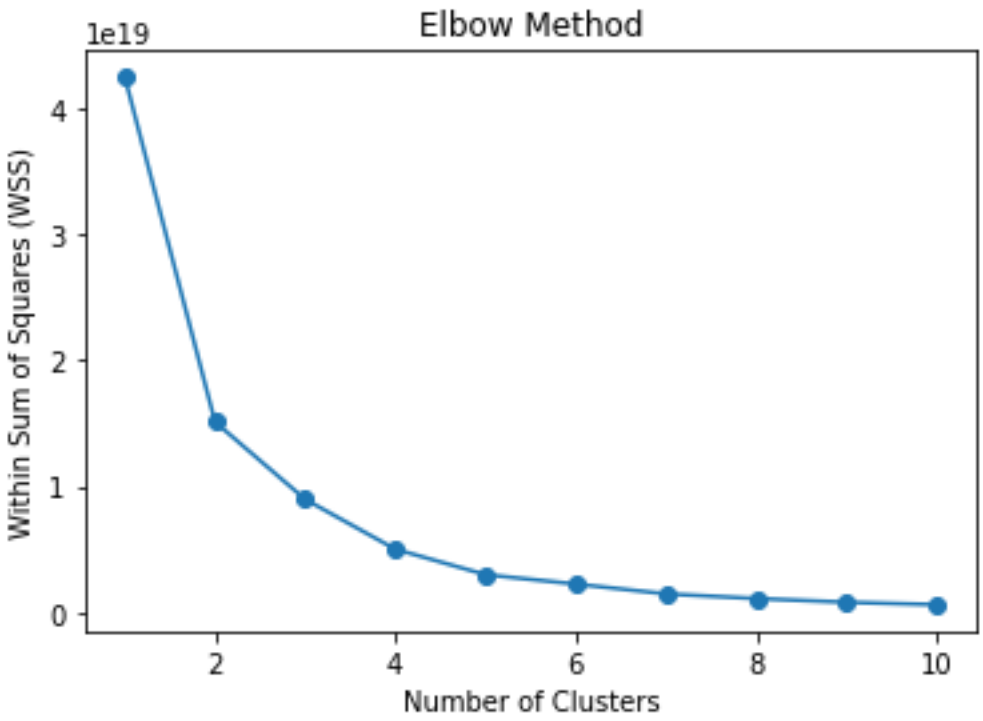
(Heatmap in Python)

K-means clustering determined how many clusters the data should be divided into. The ELBOW METHOD is one of the most popular methods for calculating the optimal value of k.

```
> fviz_nbclust(filmx_n, kmeans, method = "wss")
> set.seed(123)
>
```



(Elbow plot in R)



(Elbow plot in Python)

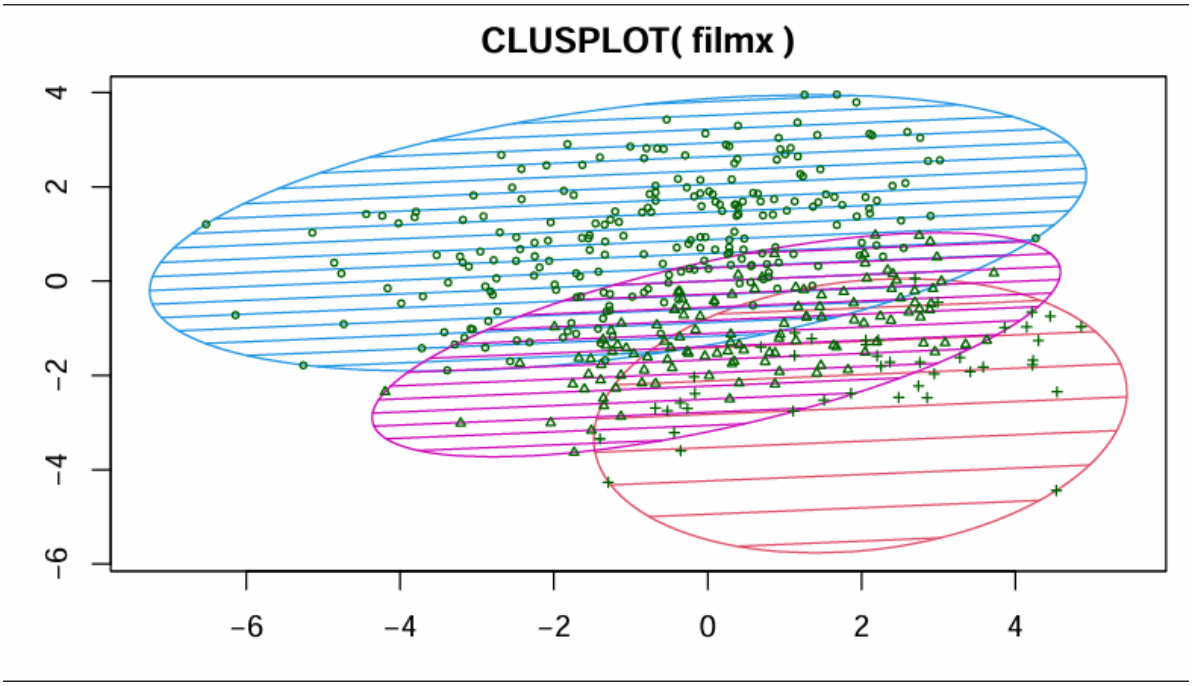
Perform the k-mean clustering From the following code we can perform the k mean clustering a lgorithm for the data set.

Cluster= clusters The output km.fit$cluster is the clustering result, which shows which cluster ea ch observation in the data set belongs to. The values in the output represent the cluster memb ership of each observation, ranging from 1 to the number of clusters in the model.

```
> kc = kmeans(filmx_n[,-(2:4)],3) #k=3
> kc
K-means clustering with 3 clusters of sizes 135, 112, 151

Cluster means:
  rank_in_year rt_freshness  rt_score X2015_inflation imdb_rating    length      year
1    0.1901235    0.7835556 0.7859992       0.1488895   0.6787440 0.5873563 0.6045584
2    0.5615079    0.6927679 0.6920133       0.5854678   0.5421196 0.4896346 0.1504121
3    0.7402502    0.6143709 0.6674914       0.1080419   0.5113735 0.5091726 0.6770250

Clustering vector:
  [1] 3 3 1 3 3 3 1 3 1 1 1 3 1 3 1 3 3 1 1 3 3 3 3 1 3 1 1 1 3 3 3 3 1 3 1 1 3 3 3 1 1 3
 [43] 1 3 1 3 3 3 3 1 3 3 1 1 1 3 3 3 3 1 3 1 3 1 3 3 3 3 1 3 3 1 3 3 3 1 1 3 1 3 3 3 1 3 1
 [85] 3 3 1 3 1 3 3 3 1 3 1 3 3 1 1 1 1 3 3 3 1 1 3 1 3 3 3 3 1 1 3 3 1 1 3 3 3 1 3 3 3 3 3
[127] 3 1 1 1 3 1 3 1 1 3 3 1 1 3 1 3 1 3 3 1 3 3 3 3 3 1 1 1 3 1 3 1 1 1 3 3 3 3 3 1 3 1
[169] 3 1 1 1 3 1 3 3 3 1 1 1 3 3 1 3 1 3 1 3 1 1 1 3 3 1 1 3 3 3 1 3 3 3 1 3 3 1 3 1 1 1
[211] 3 3 1 1 3 1 3 1 1 3 3 1 1 3 3 1 3 3 1 3 1 3 3 3 3 1 1 1 3 1 3 1 3 1 1 3 3 1 3 1 1 1
[253] 3 1 3 3 1 2 2 3 3 3 3 1 3 1 2 1 2 2 1 2 1 1 2 2 2 2 2 3 1 1 2 2 2 1 2 2 3 2 1 1 2 2 2
[295] 2 3 1 2 2 2 1 2 1 2 1 2 2 2 2 2 2 2 2 2 1 2 2 1 2 2 2 2 2 2 1 2 2 2 2 2 1 2 2 2 2 2 1 2
[337] 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
[379] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3

Within cluster sum of squares by cluster:
[1] 25.38852 33.30905 28.39353
 (between_SS / total_SS =  42.7 %)
```

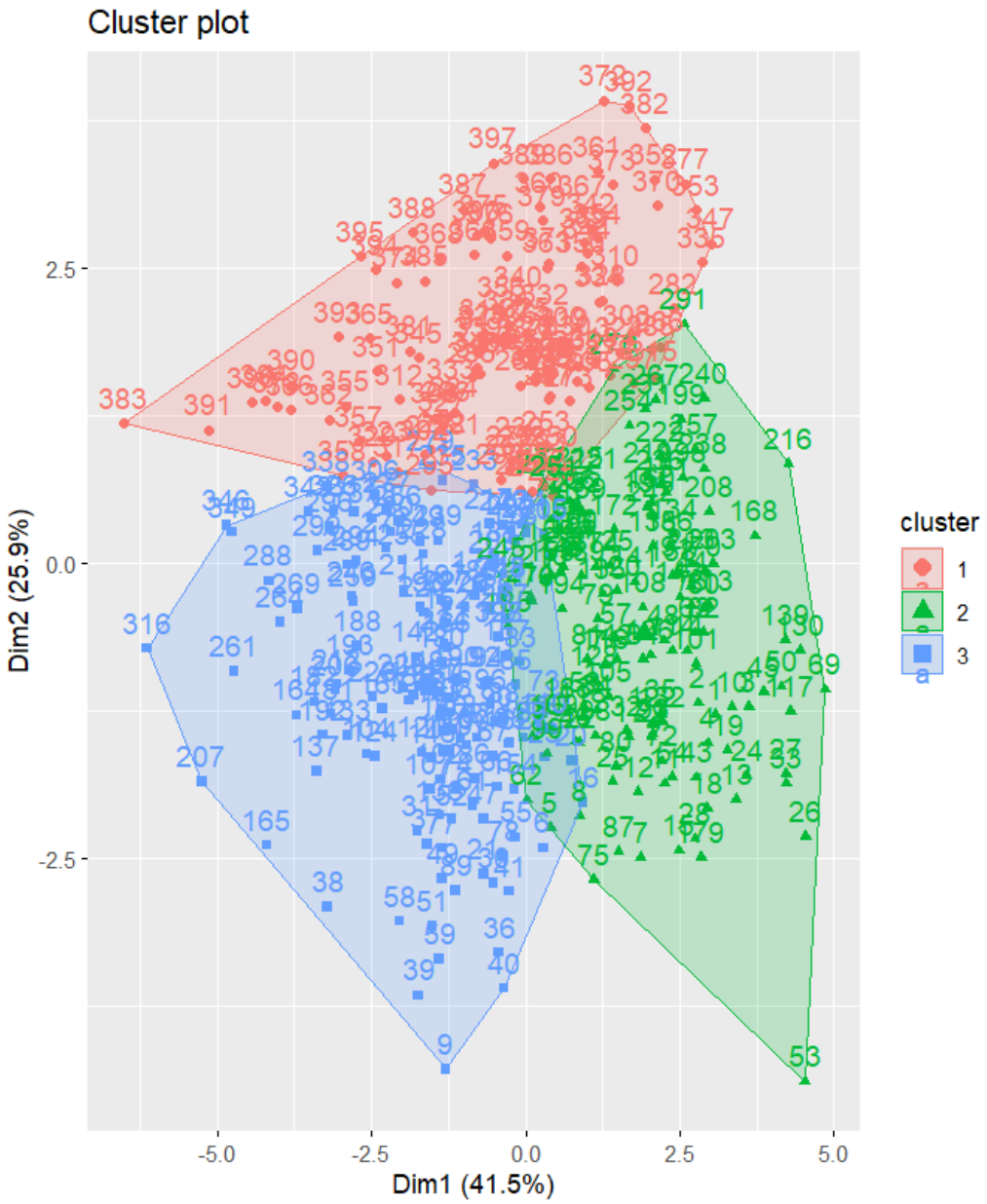
```
Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
[6] "betweenss"    "size"         "iter"         "ifault"
> clusplot(filmx_n, kc$cluster, color = TRUE, shade = TRUE, lines = 0)
> tendency <- get_clust_tendency(filmx_n, 40, graph = TRUE)
> tendency$hopkins_stat
[1] 0.8096303
```

In our analysis, we observed a clear tendency or trend within the dataset, indicating a prominen t pattern or direction of movement. By identifying and highlighting this trend, we can provide v aluable insights into the factors driving the observed patterns and their potential implications. Overall, recognizing and reporting on the best tendency within the dataset enriches our analysis and strengthens the validity and relevance of our findings.



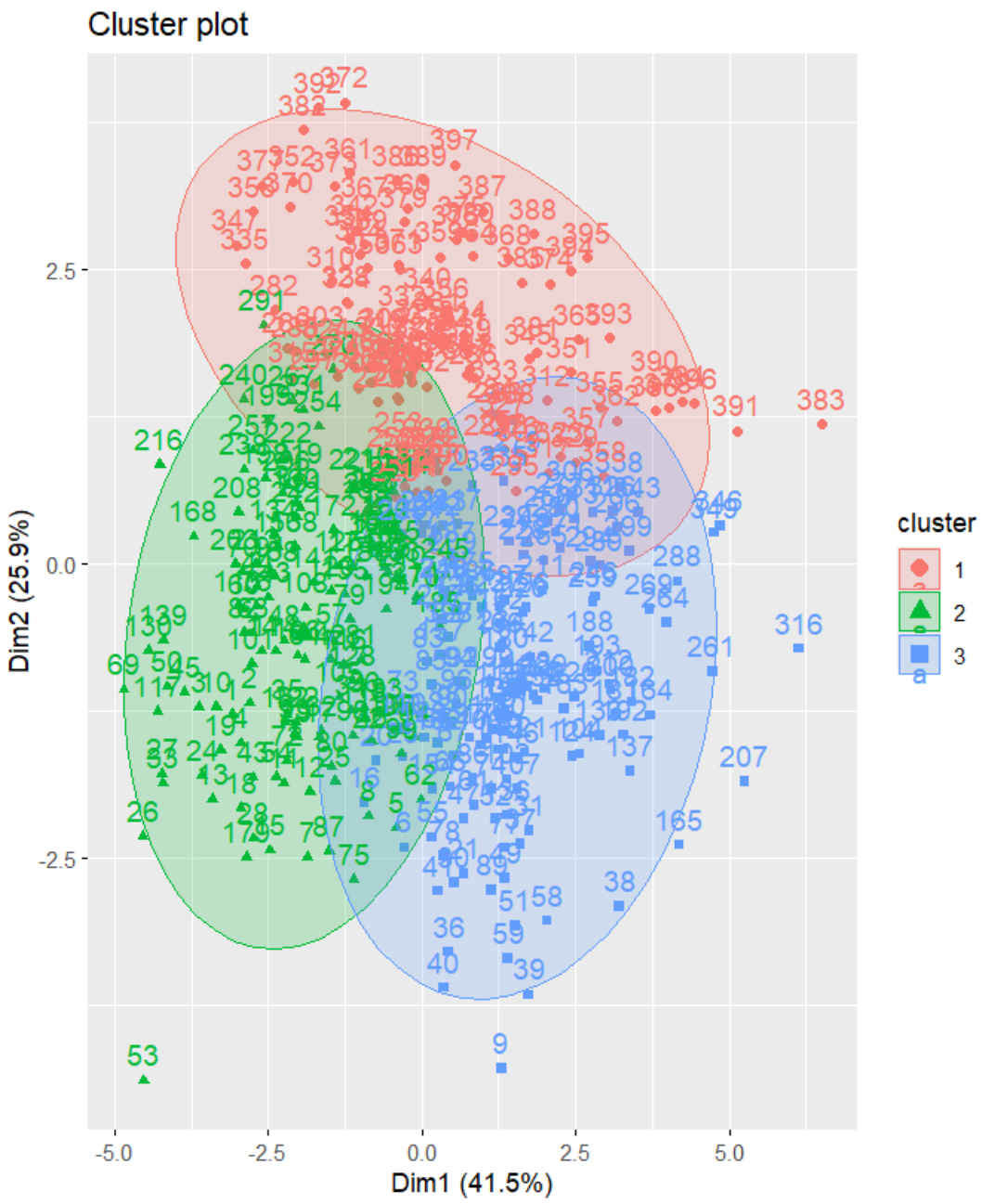
Following R codes were used to visualize clusters by using fviz_cluster function.

```
> set.seed(123)
> km.fit <- kmeans(filmx_n, 3, nstart = 30)
> km.fit$cluster
  [1] 2 2 2 2 2 3 2 2 3 2 2 2 2 2 2 3 2 2 2 3 3 2 3 2 2 2 2 2 2 3 3 2 2 2 2 3 3 3 3 3
 [43] 2 2 2 2 3 2 3 2 3 3 2 2 3 2 2 3 3 2 3 2 2 2 3 3 3 2 2 2 2 3 3 2 2 3 3 2 2 2 2
 [85] 3 3 2 3 3 3 2 3 2 3 2 3 3 2 2 3 2 3 3 3 2 2 3 2 2 3 3 3 2 2 3 2 2 2 3 2 3 2 2 3
[127] 3 2 3 2 3 2 3 2 2 3 3 2 2 3 2 3 2 3 2 3 3 3 3 2 2 3 2 3 2 2 3 2 3 2 2 3 3 3 3 3
[169] 2 2 3 2 3 2 3 2 2 3 2 3 3 3 2 2 2 3 3 3 2 3 2 3 3 2 2 3 3 2 2 3 3 2 2 3 2 3 2 2
[211] 3 3 2 2 2 2 3 3 2 3 2 2 3 3 1 3 3 3 3 1 2 1 3 3 3 2 3 2 3 2 1 2 3 2 2 3 1 3 3 1
[253] 1 2 3 3 2 1 3 1 3 1 1 3 1 1 2 1 3 2 3 1 1 1 1 1 3 1 3 3 1 1 1 1 1 1 3 3 1 2 1 1
[295] 1 3 1 1 3 1 1 1 1 1 1 3 1 1 1 1 3 1 3 1 1 3 1 1 1 1 1 1 1 1 3 1 1 1 1 1 1 1 1 1
[337] 1 3 1 1 1 1 3 1 1 3 1 1 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
[379] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 3
> fviz_cluster(km.fit,filmx_n)
>
```




```
> fviz_cluster(km.fit,filmx_n,ellipse.type = "norm")
>
```

6. Result Analysis and Discussion



We'll discuss the outcomes of our clustering attempts in this section. To aid in the understanding of this data, a final picture plot called the "cluster plot" has been included.

In order for the k-means algorithm to function, data points are iteratively assigned to clusters, the centroid (or mean) of each cluster is determined, and data points are then reassigned to the nearest centroid. Until the centroids stop changing or until a predetermined number of iterations have been made, this process is repeated.

For instance, the data points in the upper right cluster typically have higher values for imdb_rating and rt_score, indicating that viewers and critics may generally enjoy these films. These variables tend to have lower values for the data points in the lower left cluster, which implies that these films might not be as well-liked.

7. Conclusion

As is evident, in order to get the best model, we may play about with the parameters and try out various clustering techniques. The idea was to become knowledgeable about various clustering techniques and then apply them to the blockbluster movie data to extract some useful information.
Using k-means clustering to analyse the top 10 blockbuster films of the last 40 years, groupings of films were identified according to box office performance, audience rating, and critical reaction. But even while these clusters provide some beginning information, it is important not to draw firm conclusions from this approach alone. K-means clustering has limits, and more research is necessary to have a deeper understanding. To get a more complete view of the elements that go into making a blockbuster successful, it is essential to examine individual qualities, their relationships, and contextual considerations like genre and historical context.

TASK 03

Build a Dynamic Dashboard in Power BI.

Power BI dashboard for Blockblaster Top Ten Movie Per Year.

1.Introduction

Microsoft offers Power BI, a business analytics solution that lets users visualise and share insights from several data sources. It provides an easy-to-use interface for connecting, modelling, and transforming data to produce dynamic dashboards and reports. With its powerful analytical features and support for a wide range of data types, Power BI enables organisations to make data-driven choices and boost productivity.

In this dashboard is based on the top 10 blockbusters of the previous 40 years (1984–2024) to gain more insight into the realm of blockbuster films. This dataset will offer insightful information about the kinds of films that have held audiences' attention throughout time, what makes a blockbuster successful, and how the film industry is always changing.

2.Exploration of Data Set

The dataset contains statistics on blockbuster films that have been released in the last 40 years. It includes parameters like audience satisfaction, movie duration, rank within release years, studio information, and worldwide gross income. These facts include information about how the public reacted to the film, how well it did financially, and what production trends were seen in various years and studios.

Attributes of dataset are,

• audience_freshness
• poster_url
• rt_audience_score
• rt_freshness
• rt_score
• 2015_inflation
• Adjusted
• Genres
• genre_1
• genre_2
• genre_3

• imdb_rating
• Length
• rank_in_year
• rating
• release_date
• Studio
• Title
• worldwide_gross
• Year

3.Dashboard Design & Implementation

A dashboard could improve communication, increase efficiency, and properly direct the strategic purpose. The dynamic and interactive aspect of the dashboard allows users to examine the top 10 blockbuster films over the past 40 years by applying filters. Dashboards provide a thorough overview of the situation by aggregating data from multiple systems and sources into a single interface. We started the operation in a different way before importing the data set, using the

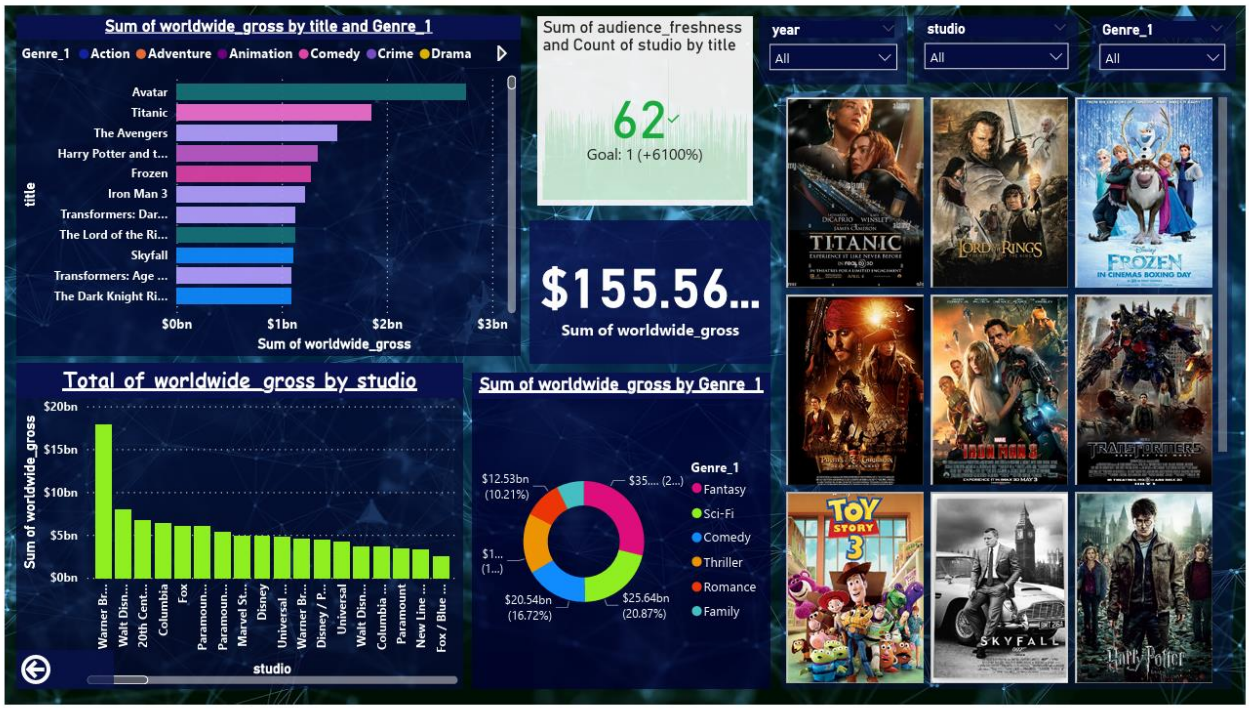
Start POWER BI ⟶ GET DATA ⟶ MORE ⟶ R SCRIPT
Within the field section, Power Bi automatically generated two data sets, one normal and one normalised, after we entered the R codes that we used for clustering with the working directory.

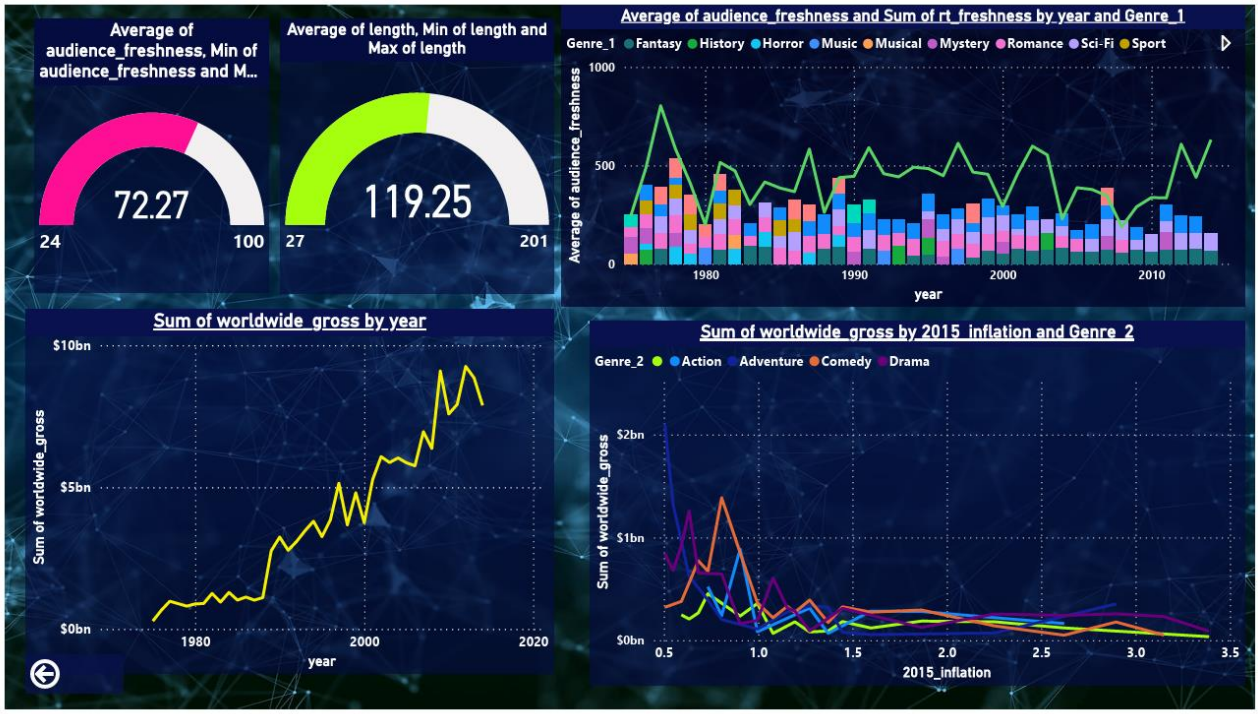
The top 10 blockbuster films of the previous 40 years (1975 - 2015) were analysed using k-means clustering, which allowed for the identification of film groups based on audience rating, _wide gross,critical reception, rank in the year and box office performance.

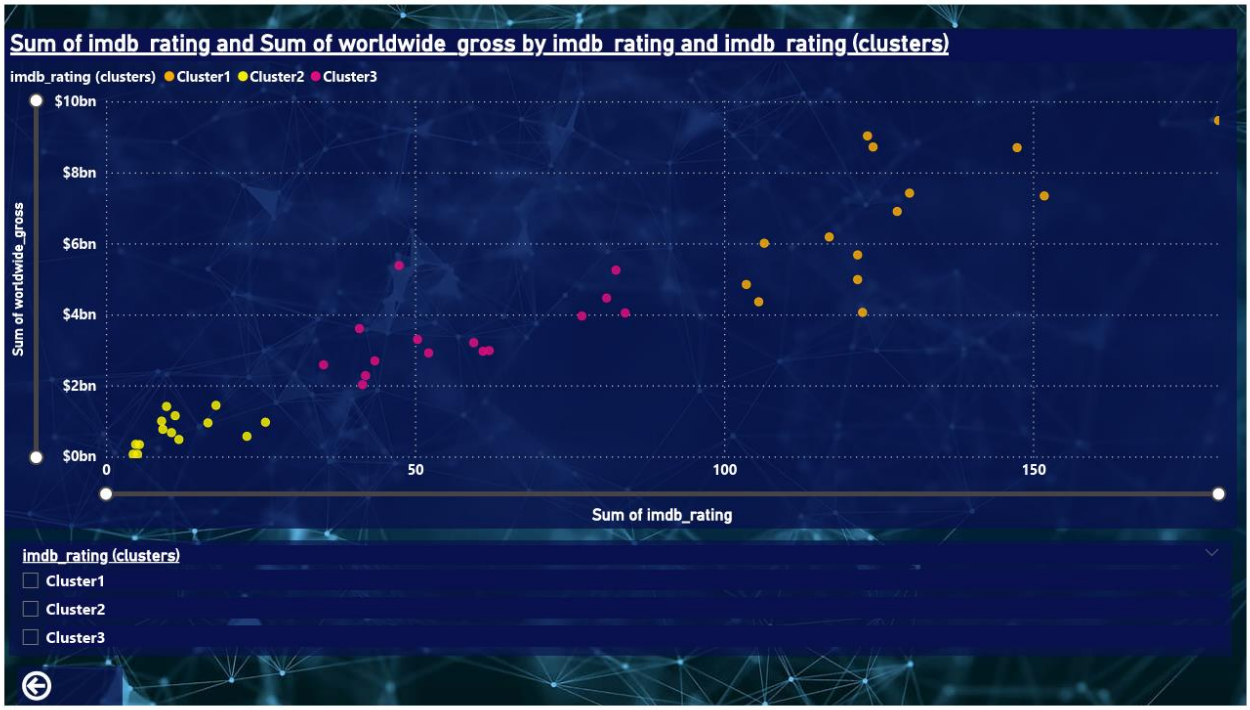
Title Page

Content page

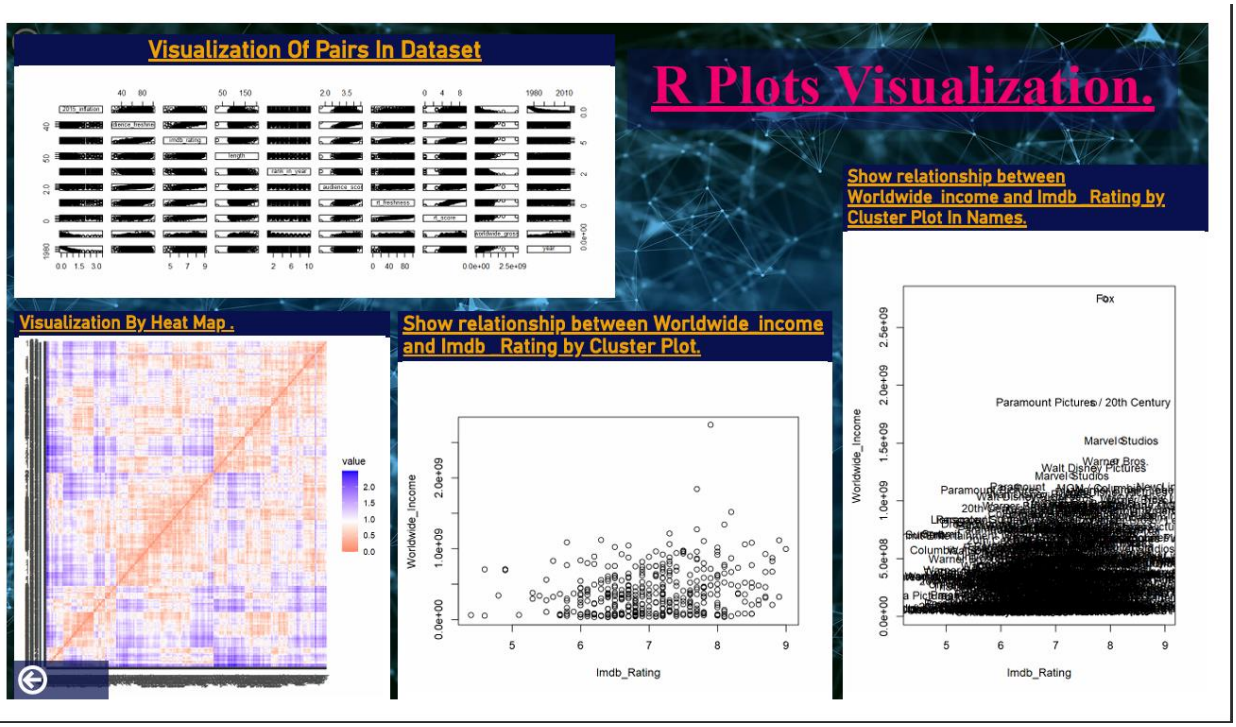
Dashboard


Sum of worldwide_gross by title and Genre_1
Genre_1 Action Adventure Animation Comedy Crime Drama
Avatar
Titanic
The Avengers
Harry Potter and t...
Frozen
Iron Man 3
Transformers: Dar...
The Lord of the Ri...
Skyfall
Transformers: Age ...
The Dark Knight Ri...
title
$0bn
$1bn
$2bn
$3bn
Sum of worldwide_gross
Sum of audience_freshness and Count of studio by title
62
Goal: 1 (+6100%)
$155.56...
Sum of worldwide_gross
year
All
studio
All
Genre_1
All
Total of worldwide_gross by studio
$20bn
$15bn
$10bn
$5bn
$0bn
Sum of worldwide_gross
Warner Br...
Walt Disn...
20th Cent...
Columbia
Fox
Paramoun...
Paramoun...
Marvel St...
Disney
Universal ...
Warner Br...
Disney / P...
Universal
Walt Disn...
Columbia ...
Paramount
New Line ...
Fox / Blue ...
studio
Sum of worldwide_gross by Genre_1
$12.53bn (10.21%)
$35.... (2...)
$1... (1...)
$20.54bn (16.72%)
$25.64bn (20.87%)
Genre_1
Fantasy
Sci-Fi
Comedy
Thriller
Romance
Family

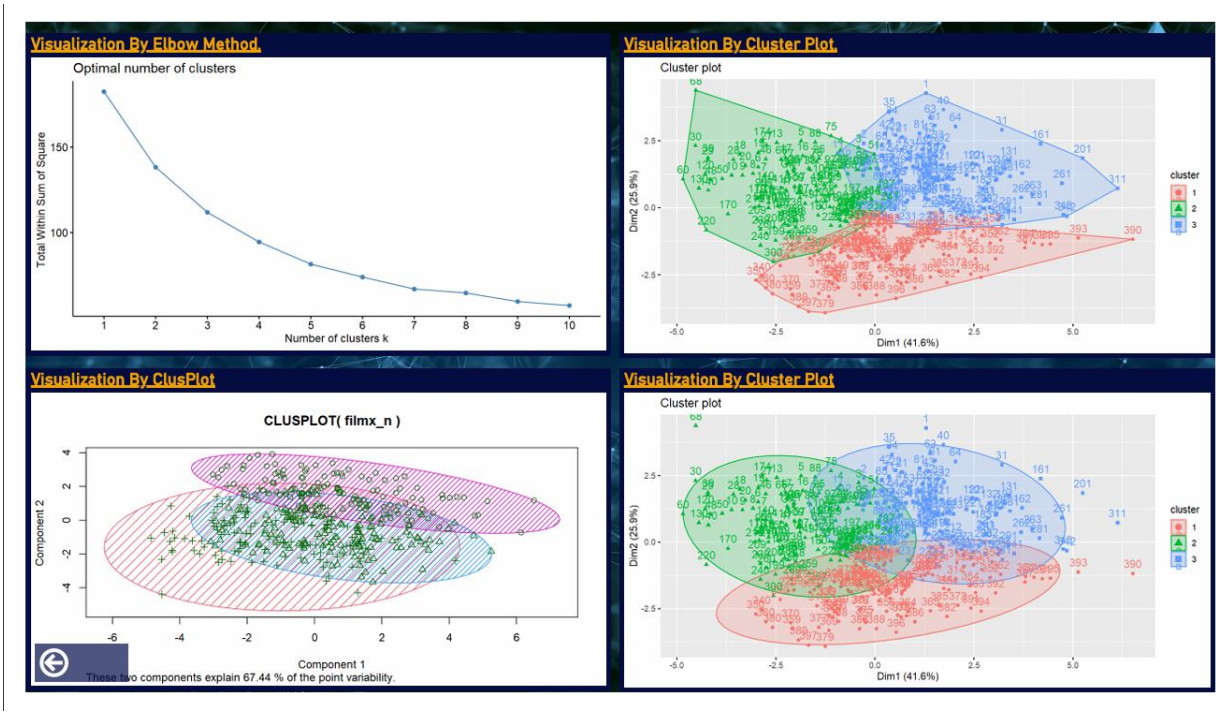

Average of audience_freshness, Min of audience_freshness and M...
72.27
24
100
Average of length, Min of length and Max of length
119.25
27
201
Average of audience_freshness and Sum of rt_freshness by year and Genre_1
Genre_1 Fantasy History Horror Music Musical Mystery Romance Sci-Fi Sport
1000
500
0
Average of audience_freshness
1980
1990
2000
2010
year
Sum of worldwide_gross by year
$10bn
$5bn
$0bn
Sum of worldwide_gross
1980
2000
2020
year
Sum of worldwide_gross by 2015_inflation and Genre_2
Genre_2 Action Adventure Comedy Drama
$2bn
$1bn
$0bn
Sum of worldwide_gross
0.5
1.0
1.5
2.0
2.5
3.0
3.5
2015_inflation



Clustering 1



Clustering 2



Thank You Page.

Data pairs visualization

For the "data pairs visualisation" chart, we used a scatter plot matrix. Numerous variables' relationships can be shown in a scatter plot matrix. After charting all possible two-way combinations of the variables to highlight which associations, the matrix can indicate relationships between the variables are probably going to be crucial. Using the matrix, outliers in many scatter plots can also be identified.

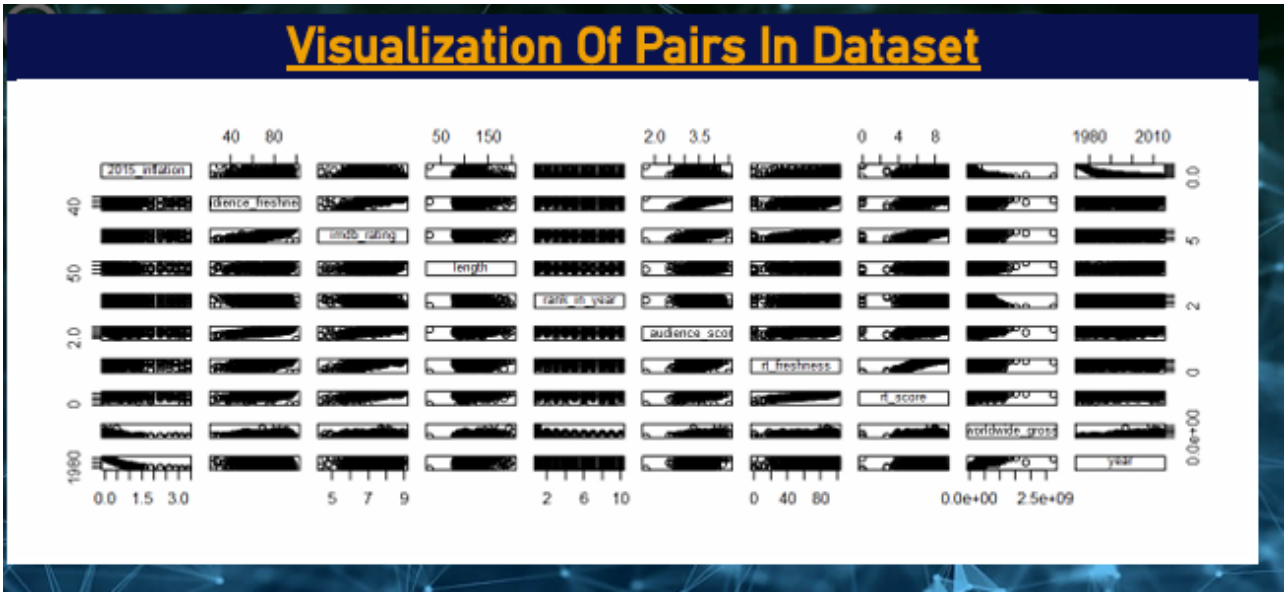
In this Blockbluster Data set scatter plot matrix is used to visualize and compare the variables of the 2015_inflation, audience_freshness, imdb_rating, length, rank_in_year, rt_audience_score, rt_freshness, rt_score, worldwide_gross, year).

The following R code was used in the R script of Power BI to create the above visual.

R script editor

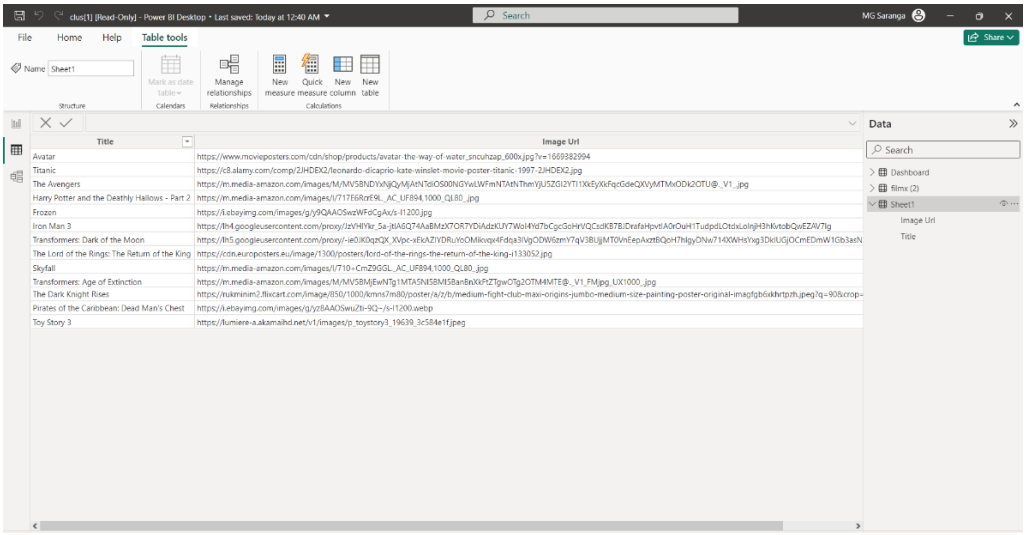
⚠ Duplicate rows will be removed from the data.

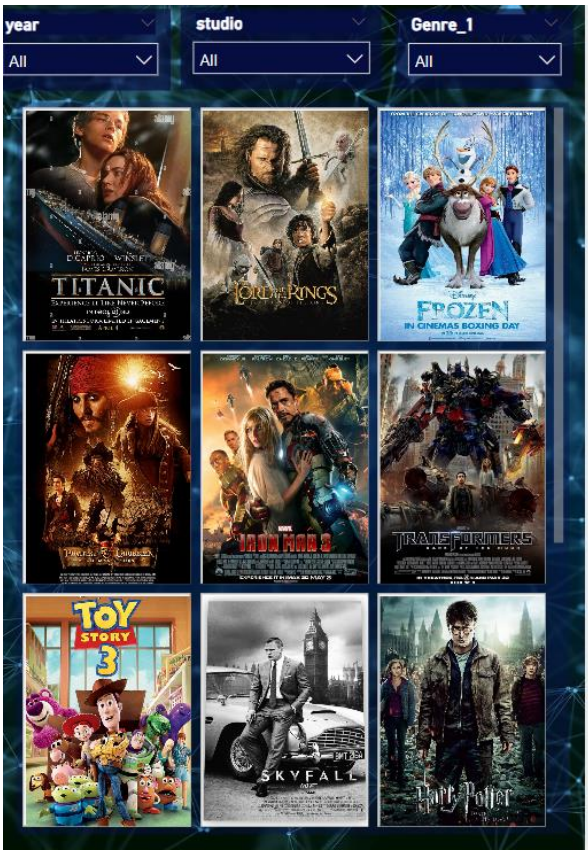
```
# The following code to create a dataframe and remove duplicated rows is always executed and acts as a preamble for your script:

# dataset <- data.frame(2015_inflation, audience_freshness, imdb_rating, length, rank_in_year, rt_audience_score, rt_freshness, rt_score,
worldwide_gross, year)
# dataset <- unique(dataset)

# Paste or type your script code here:
library(cluster)

## Get pairs ##

film1=dataset
pairs(film1)
```

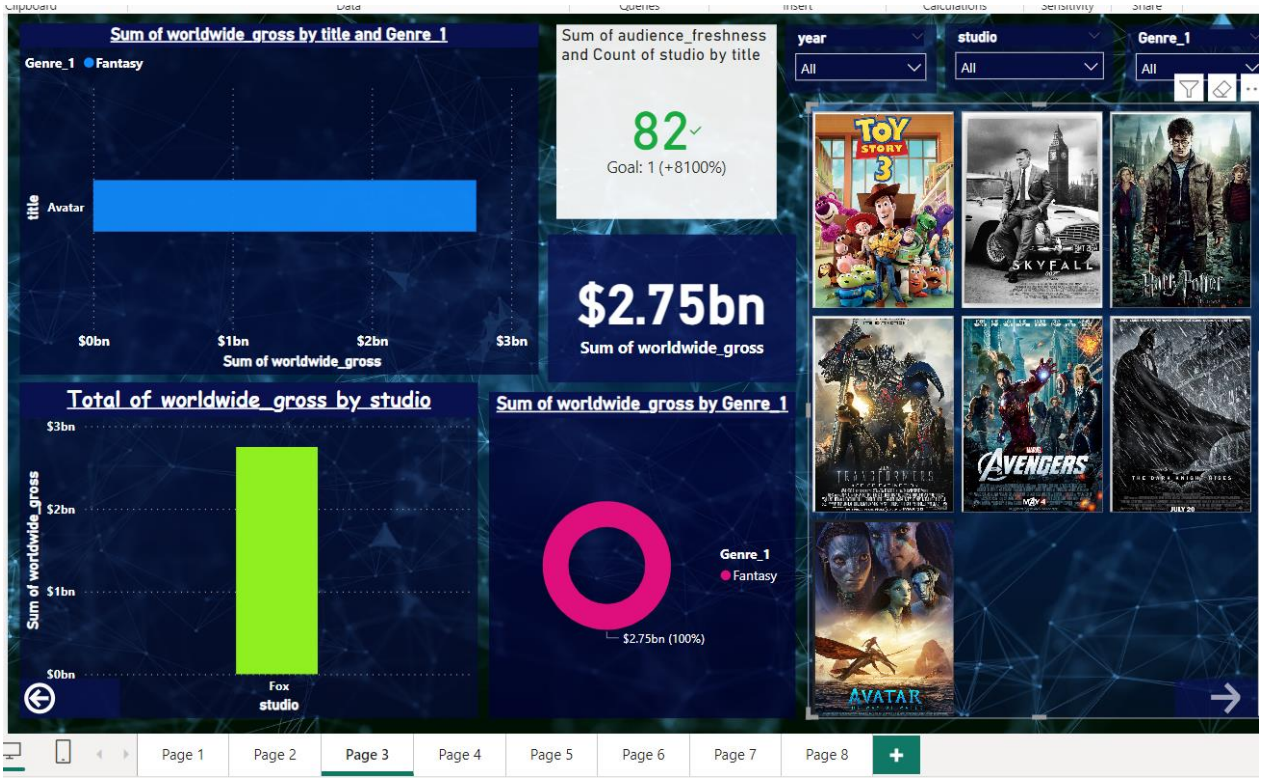


Top BlockBluster Movies

To display Top Blockbluster movies we installed a new visualization tool called “Image by CloudScope Under this section users can find the film of each year and the studio, genre and income of the film well with in by clicking the image .
By using the image URLs of the films as well as the we prepared a new csv file and then imported it into Power BI. The new csv file is as follows,



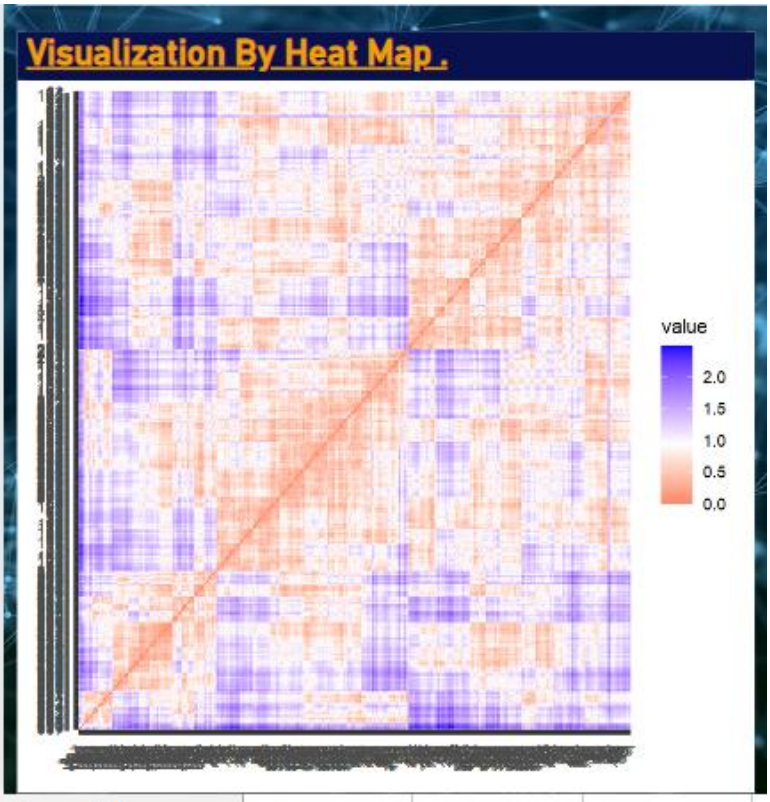




In here can we see when we click on avatar image its shows only regarding to avtar film .

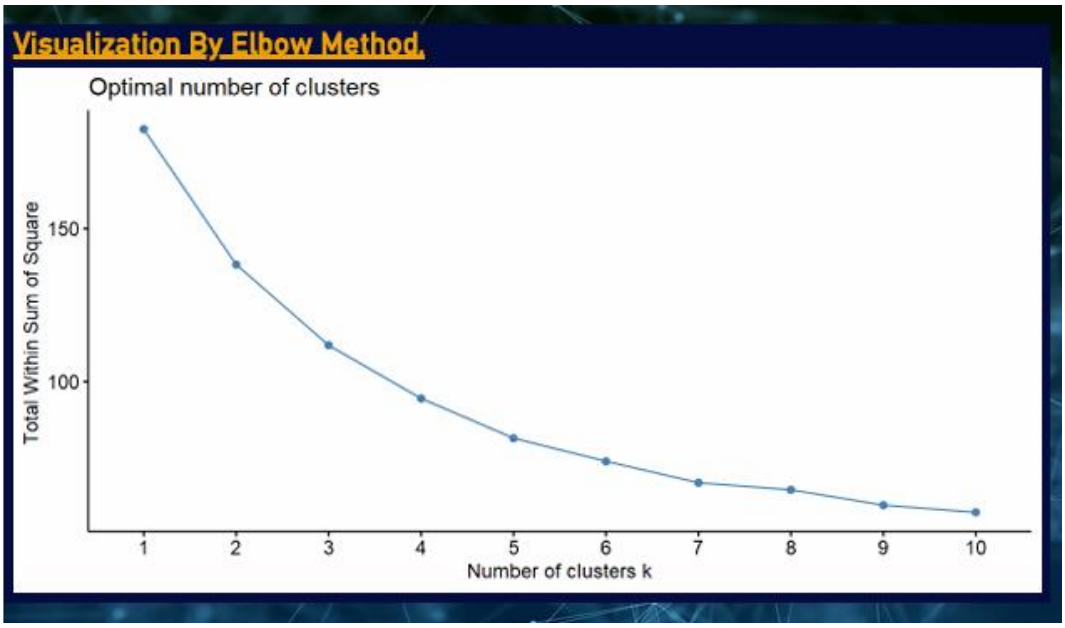
Dissimilarities between studio.

The below plot displays the distance between rows of the dataset. Factoextra R Package is used to create the above plot. Distance matrix is used to visualize dissimilarities of indexes. According to the chart red color shows high similarity (low dissimilarity) and blue color shows low similarity. The following R code was used in the R script of Power BI to create the above plot.


Visualization By Heat Map .
value
2.0
1.5
1.0
0.5
0.0

Elbow Method Plot

The below graph represents the predicted number of clusters which is known as k value. The following R code was used in the R script of Power BI to create the above plot.

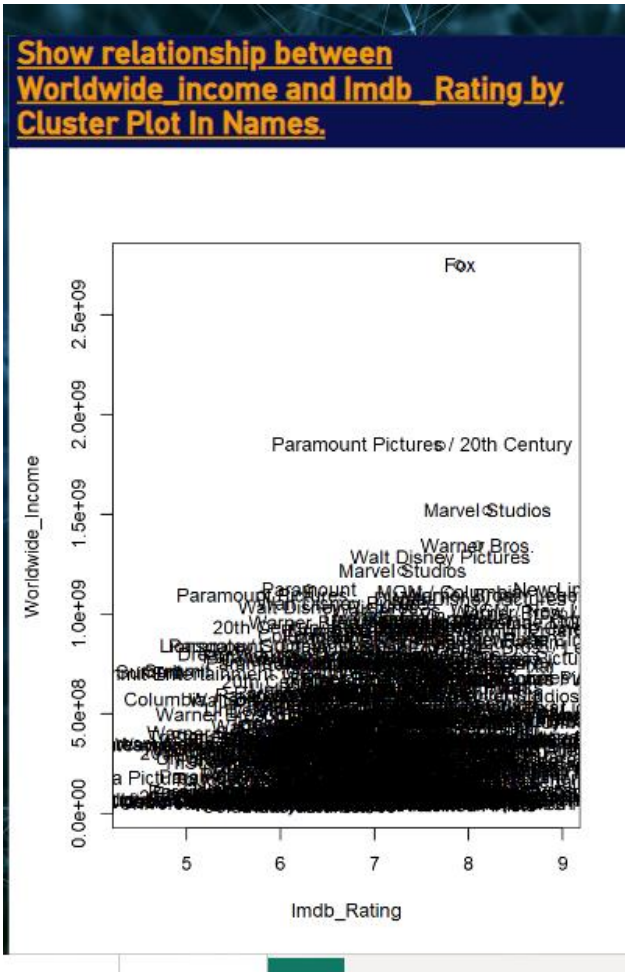


⚠ Duplicate rows will be removed from the data.

```
25 library(factoextra)
26
27 kc = kmeans(filmx_n[,-(2:4)],3) #k=3
28
29 library(factoextra)
30
31 distance=dist(filmx_n,method = "euclidean")
32 ## Calculate the tendency##
33
34 tendency <- get_clust_tendency(filmx_n, 40, graph = TRUE)
35
36 fviz_nbclust(filmx_n, kmeans, method = "wss")
```

Page 1 | Page 2 | Page 3 | Page 4 | Page 5 | Page 6 | Page 7 | Page 8 | +

The below overlapping plot represents the relationship between world wide income and Imdb rating . The following R code was used in the R script of Power BI to create the above plot.



R script editor

⚠ Duplicate rows will be removed from the data.

```r
# The following code to create a dataframe and remove duplicated rows is always executed and acts as a preamble for your script:

# dataset <- data.frame(2015_inflation, audience_freshness, imdb_rating, length, rank_in_year, rt_audience_score, rt_freshness, rt_score, worldwide_gross, studio, year)
# dataset <- unique(dataset)

# Paste or type your script code here:
filmx=dataset
film1=filmx[,-10]
plot(film1$worldwide_gross~film1$imdb_rating,xlab="Imdb_Rating",ylab="Worldwide_Income")
with(film1,text(film1$worldwide_gross~film1$imdb_rating, labels=filmx$studio))

```

Cluster Plot

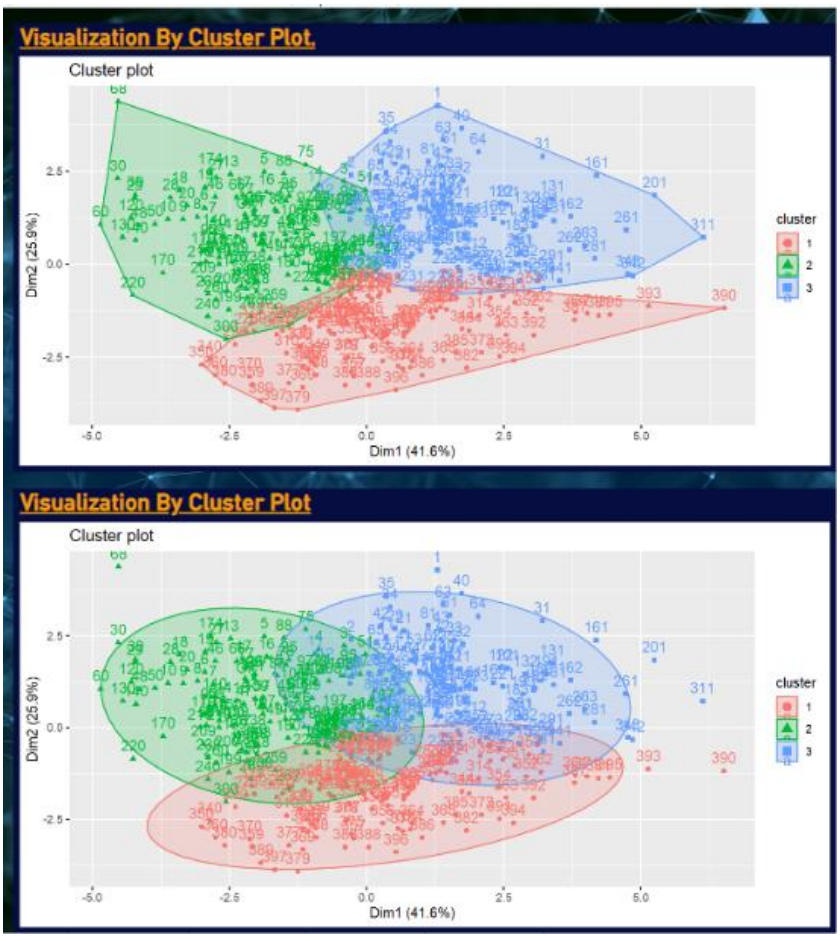
According to the task we used the following R code in the Power BI R script editor to create this visualization.

R script editor

⚠ Duplicate rows will be removed from the data.

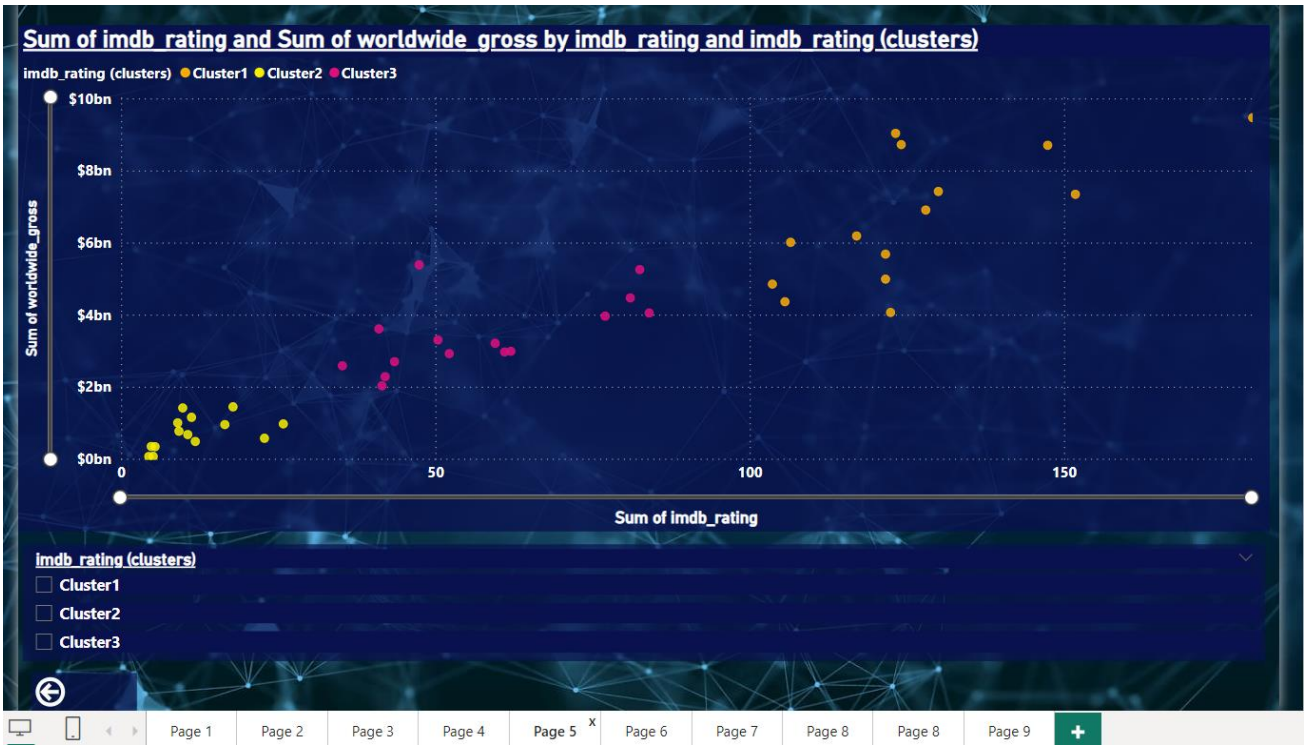
```
library(factoextra)

distance=dist(filmx_n,method = "euclidean")
## Calculate the tendency##

tendency <- get_clust_tendency(filmx_n, 40, graph = TRUE)
set.seed(123)

##clustering the data##

km.fit <- kmeans(filmx_n, 3, nstart = 40)
fviz_cluster(km.fit,filmx_n)
```

the data points in the upper right cluster generally have higher values for rt_score and imdb_rating, suggesting that critics and audiences may find these films to be generally enjoyable. The data points in the lower left cluster tend to have lower values for these variables, suggesting that these films may not be as popular.

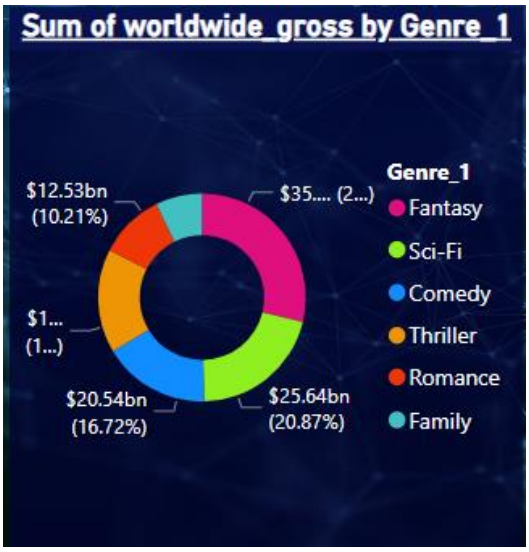


Three main cluster

We used a plot to display list of the clustered imdb rates with the cluster centers mean values, world wide income, and mean value of centers. Users can use to find the imbdn rate by using the plot and image visuals very easily.

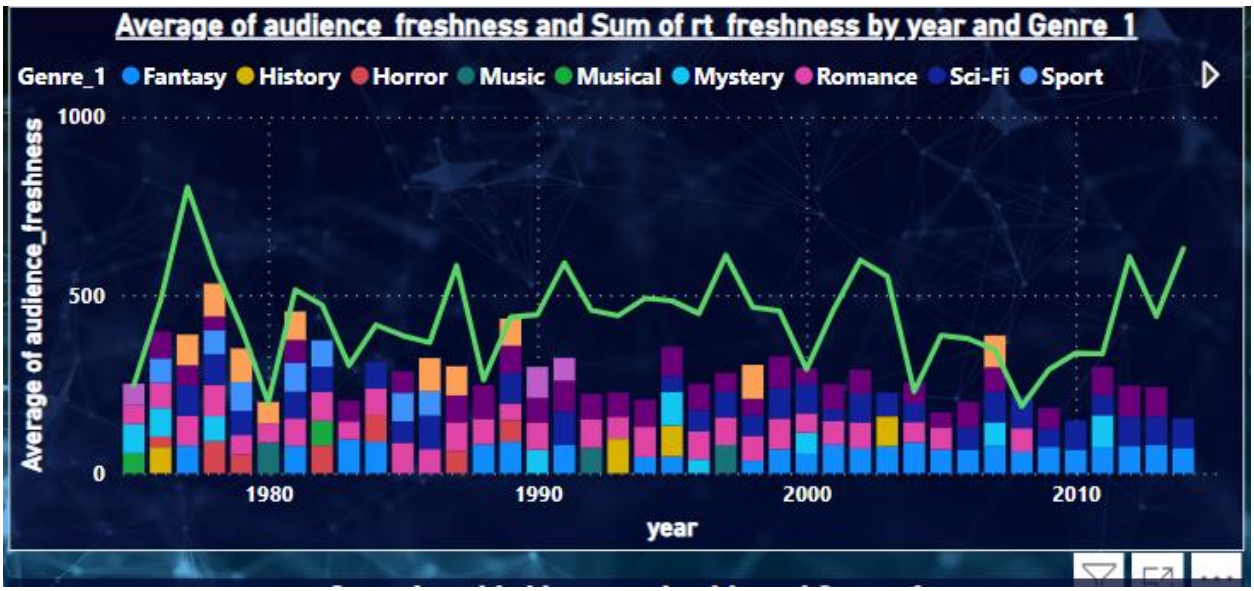


Visualization by Dounut chart.



The total global gross by genre is displayed in this doughnut chart. The genre that brought in the most money overall worldwide throughout the given period is represented by the largest slice, "Fantasy" (35.86%).

Visualization by Line and Staked Bar Chart.



Each dot in the plot represents a movie from the dataset. The centroids of each cluster are shown as black circles in the plot. The centroid, which represents the central tendency of the data points in a cluster, is the average of all the data points within that cluster.

All things considered, the line and staked column chart offers a graphic depiction of how the most popular blockbuster films can be arranged according to the selected characteristics. Understanding the particular traits that set the movie clusters apart will require more investigation that takes into account the original attributes and the context of the data.

Visualization by Bar Chart.

A horizontal bar chart uses bars arranged horizontally to compare different data types. The top studios are represented in this particular chart according to the total global box office revenue of their motion pictures between 1984 and 2024.
All things considered, this horizontal bar chart offers a rapid and simple means of comparing the leading film studios in terms of their overall global box office receipts over a forty-year span. It makes it possible for spectators to quickly determine which studios have made the most money from their motion pictures.

Visualization by Horizontal Bar chart

A horizontal bar chart is shown in the image you submitted. This kind of graphic uses horizontally oriented bars to compare different data types. Based on their global box office receipts, the top 10 most lucrative super films are the categories in this particular ranking.

4.Conclusion.

A dashboard report is a graphic representation of the status of an organisation or anything that has a lot of data. However, a good dashboard offers more than simply a synopsis of important information; it also offers analysis and insight, which makes it a useful tool for quickly reviewing and acting upon important information. Since those who compile and evaluate Decisions are rarely made just by the facts; instead, the dashboard's insights and suggestions can enhance the report's value. This may also help with communication problems for analysts and executives who need specialised data to make strategic decisions. A dashboard could facilitate goal achievement, enhance communication, and save time.
Dashboards give users a complete picture of the situation by combining data from many systems and sources into a single interface. Lastly, a well-designed dashboard makes it possible for users to quickly access all important data.
Through the analysis of this dashboard ,Themes and genres that have dominated the world of blockbusters include that we can decided people still want to see films in genres other than superhero films, such as comedy, sci-fi, or animation?
The production organisations and studios responsible for these hit films are can known What are the successful techniques employed by the studios that regularly deliver hits at the box office?

Technological developments' effects on filmmaking What impact have technology advancements, special effects, and animation approaches had on the blockbuster experience?
The changing tastes of the audience Over the past forty years, how have the expectations and tastes of the audience for blockbusters changed?
Examining this dataset will provide insightful knowledge about the inner workings of the film industry and be an exciting voyage into the realm of blockbuster films.