HomeWork_5

SRK Yarra

2023-11-19

#Data Gathering & Intigration For this problem, we used the Movies dataset, which is a popular dataset available on various platforms, including Kaggle and the UCI Machine Learning Repository. The dataset contains information about the movies in languages.

```
#import the data
MoviesData <- read.csv("Movies.csv")
head(MoviesData)
```

```
##   Id Survived class                    name    sex age sibsp parch    Ticket
## 1  1        0     3                  Braund   male  22     1     0 A/5 21171
## 2  2        1     1        Mr. Owen Harris female  38     1     0  PC 17599
## 3  3        1     3                 Cumings   male  26     0     0   STON/O2
## 4  4        1     1      Mrs. John Bradley female  35     1     0   3101282
## 5  5        0     3 Florence Briggs Thayer   male  40     0     0    113803
## 6  6        0     3               Heikkinen female  27     0     0     12478
##    Fare cabin embarked
## 1  7.25              s
## 2 71.28   c85        c
## 3  7.92              s
## 4  6.87  c123        s
## 5  5.47              s
## 6 81.90              c
```

#Data Exploration

Explored the Movies dataset to understand its characteristics. And examined the distributions of variables such as budget, languages, production countries and production companies Also investigated relationships between variables, such as the correlation between production countries and production companies, or the distribution of survival rates across different languages.

```
#Calculate basic descriptive statistics
summary(MoviesData)
```

```
##        Id           Survived       class          name
##  Min.   : 1.00   Min.   :0.0   Min.   :1.00   Length:10
##  1st Qu.: 3.25   1st Qu.:0.0   1st Qu.:1.25   Class :character
##  Median : 5.50   Median :0.5   Median :3.00   Mode  :character
##  Mean   : 5.50   Mean   :0.5   Mean   :2.30
##  3rd Qu.: 7.75   3rd Qu.:1.0   3rd Qu.:3.00
##  Max.   :10.00   Max.   :1.0   Max.   :3.00
##      sex                 age           sibsp         parch
##  Length:10          Min.   :22.00   Min.   :0.0   Min.   :0.0
##  Class :character   1st Qu.:28.75   1st Qu.:0.0   1st Qu.:0.0
##  Mode  :character   Median :36.50   Median :0.5   Median :0.0
##                     Mean   :35.60   Mean   :0.7   Mean   :0.3
##                     3rd Qu.:39.75   3rd Qu.:1.0   3rd Qu.:0.0
##                     Max.   :54.00   Max.   :3.0   Max.   :2.0
##     Ticket               Fare           cabin             embarked
##  Length:10          Min.   : 5.470   Length:10          Length:10
##  Class :character   1st Qu.: 7.418   Class :character   Class :character
##  Mode  :character   Median :45.725   Mode  :character   Mode  :character
##                     Mean   :44.794
##                     3rd Qu.:79.245
##                     Max.   :87.900
```

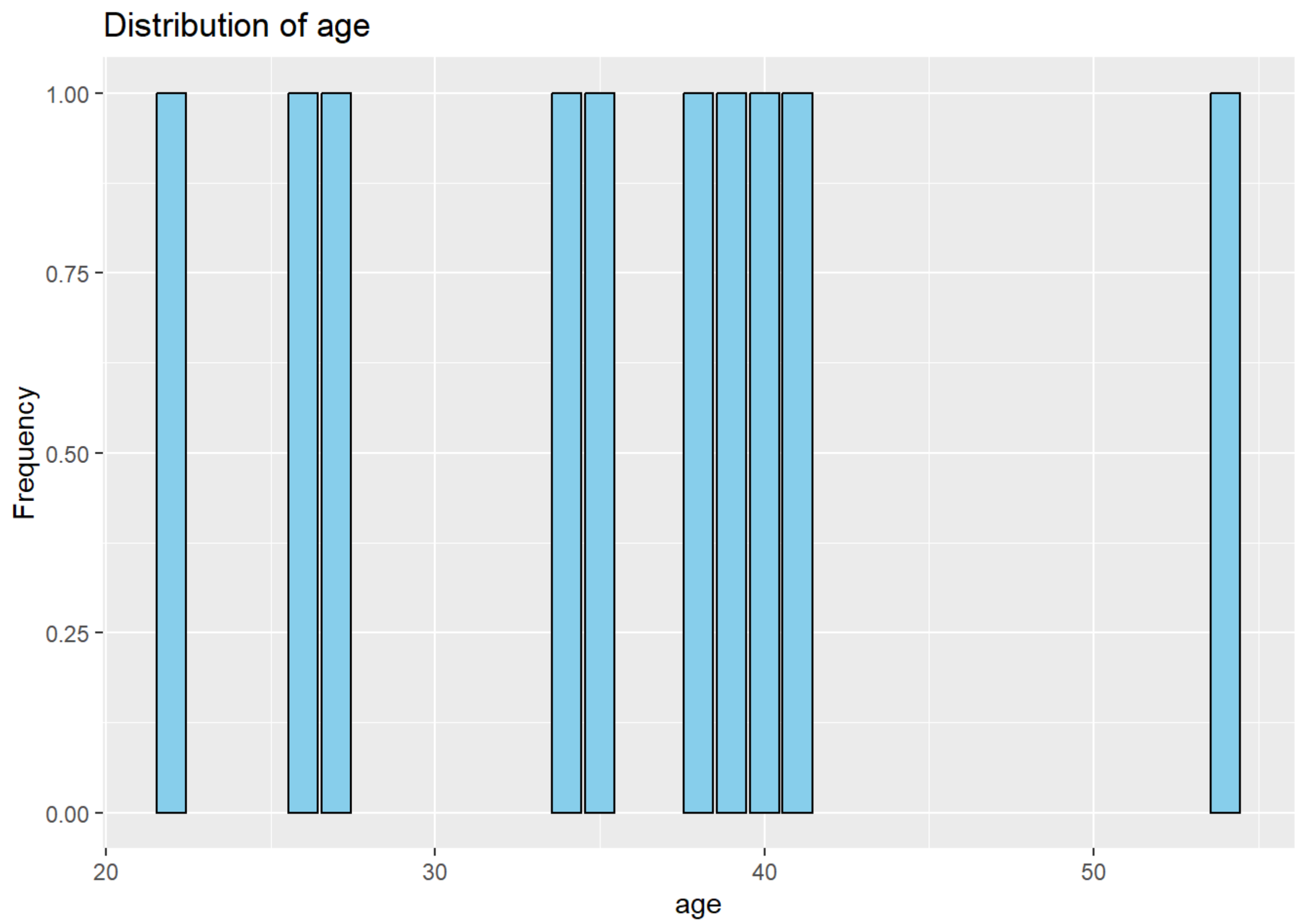
```
#List structure of a dataset
str(MoviesData)
```

```
## 'data.frame':    10 obs. of  12 variables:
##  $ Id      : int  1 2 3 4 5 6 7 8 9 10
##  $ Survived: int  0 1 1 1 0 0 0 0 1 1
##  $ class   : int  3 1 3 1 3 3 1 3 3 2
##  $ name    : chr  "Braund" "Mr. Owen Harris" "Cumings" "Mrs. John Bradley" ...
##  $ sex     : chr  "male" "female" "male" "female" ...
##  $ age     : int  22 38 26 35 40 27 54 34 39 41
##  $ sibsp   : int  1 1 0 1 0 0 0 3 0 1
##  $ parch   : int  0 0 0 0 0 0 0 1 2 0
##  $ Ticket  : chr  "A/5 21171" "PC 17599" "STON/O2" "3101282" ...
##  $ Fare    : num  7.25 71.28 7.92 6.87 5.47 ...
##  $ cabin   : chr  "" "c85" "" "c123" ...
##  $ embarked: chr  "s" "c" "s" "s" ...
```

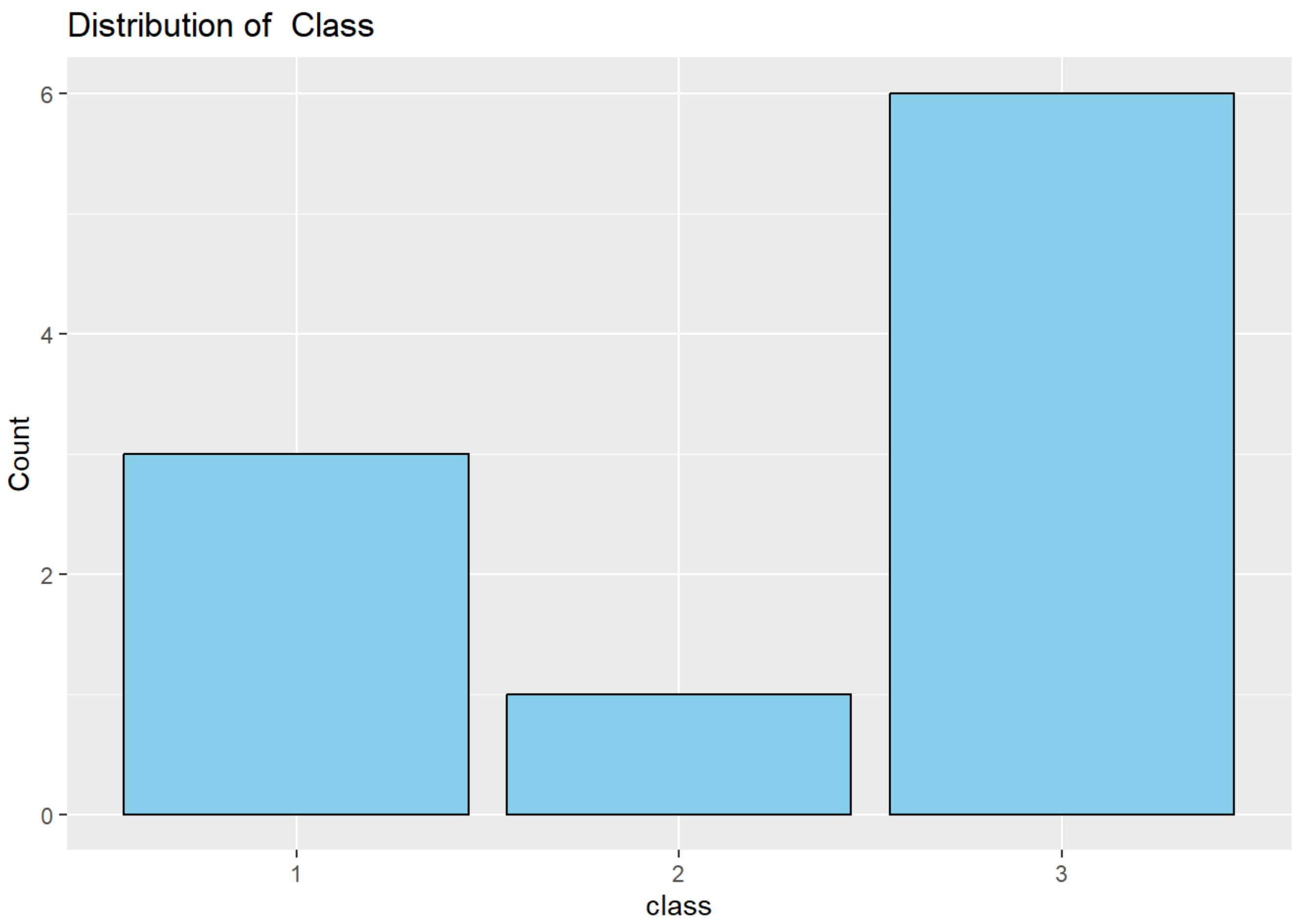
```
# Load the required packages
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.3.2
```

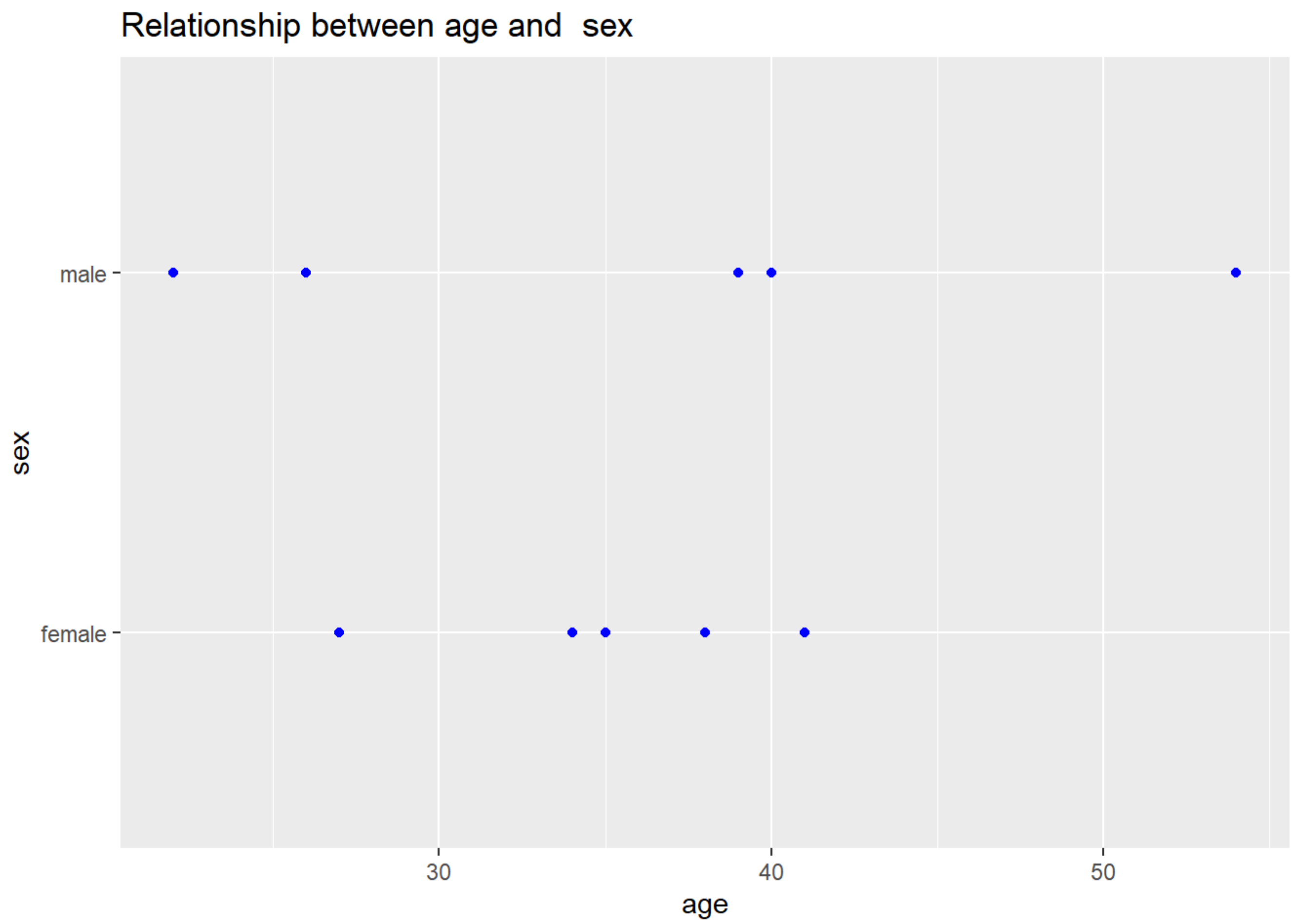
```
# Explore the distributions of variables
# barplot of prodduction countries
ggplot(MoviesData, aes(x = age)) +
  geom_bar(fill = "skyblue", color = "black") +
  labs(title = "Distribution of age") +
  xlab("age") +
  ylab("Frequency")
```




```
# Bar plot of Passenger Class
ggplot(MoviesData, aes(x = factor(class))) +
 geom_bar(fill = "skyblue", color = "black") +
 labs(title = "Distribution of  Class") +
 xlab("class") +
 ylab("Count")
```




```
# Explore relationships between variables
# Scatter plot of production countries  vs. production companies
ggplot(MoviesData, aes(x = age, y = sex)) +
 geom_point(color = "blue") +
 labs(title = "Relationship between age and  sex") +
 xlab("age") +
 ylab("sex")
```




```
summary( MoviesData$age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   22.00   28.75   36.50   35.60   39.75   54.00
```

```
# Lists name of variables in a dataset
names(MoviesData)
```

```
##  [1] "Id"       "Survived" "class"    "name"     "sex"      "age"     
##  [7] "sibsp"    "parch"    "Ticket"   "Fare"     "cabin"    "embarked"
```

```
# Calculate number of rows & columns in a dataset
dim(MoviesData)
```

```
## [1] 10 12
```

```
#See first 6 rows of dataset
head(MoviesData)
```

```
##   Id Survived class                   name    sex age sibsp parch    Ticket
## 1  1        0     3                 Braund   male  22     1     0 A/5 21171
## 2  2        1     1        Mr. Owen Harris female  38     1     0  PC 17599
## 3  3        1     3                Cumings   male  26     0     0   STON/O2
## 4  4        1     1      Mrs. John Bradley female  35     1     0   3101282
## 5  5        0     3 Florence Briggs Thayer   male  40     0     0    113803
## 6  6        0     3              Heikkinen female  27     0     0     12478
##    Fare cabin embarked
## 1  7.25              s
## 2 71.28   c85        c
## 3  7.92              s
## 4  6.87  c123        s
## 5  5.47              s
## 6 81.90              c
```

```
#First n rows of dataset

head(MoviesData, n=5)
```

```
##   Id Survived class                   name    sex age sibsp parch    Ticket
## 1  1        0     3                 Braund   male  22     1     0 A/5 21171
## 2  2        1     1        Mr. Owen Harris female  38     1     0  PC 17599
## 3  3        1     3                Cumings   male  26     0     0   STON/O2
## 4  4        1     1      Mrs. John Bradley female  35     1     0   3101282
## 5  5        0     3 Florence Briggs Thayer   male  40     0     0    113803
##    Fare cabin embarked
## 1  7.25              s
## 2 71.28   c85        c
## 3  7.92              s
## 4  6.87  c123        s
## 5  5.47              s
```

```
# All rows but the last row

head(MoviesData, n= -1)
```

```
##   Id Survived class                   name    sex age sibsp parch    Ticket
## 1  1        0     3                 Braund   male  22     1     0 A/5 21171
## 2  2        1     1        Mr. Owen Harris female  38     1     0  PC 17599
## 3  3        1     3                Cumings   male  26     0     0   STON/O2
## 4  4        1     1      Mrs. John Bradley female  35     1     0   3101282
## 5  5        0     3 Florence Briggs Thayer   male  40     0     0    113803
## 6  6        0     3             Heikkinen female  27     0     0     12478
## 7  7        0     1            Miss. Laina   male  54     0     0    133568
## 8  8        0     3              Futrelle female  34     3     1    ab1345
## 9  9        1     3    Mrs. Jacques Heath    male  39     0     2   pc16789
##    Fare cabin embarked
## 1  7.25              s
## 2 71.28   c85        c
## 3  7.92              s
## 4  6.87  c123        s
## 5  5.47              s
## 6 81.90              c
## 7 45.78              s
## 8 87.90              s
## 9 45.67              c
```

```
#Last 6 rows of dataset

tail(MoviesData)
```

```
##    Id Survived class                   name    sex age sibsp parch   Ticket
## 5   5        0     3 Florence Briggs Thayer   male  40     0     0   113803
## 6   6        0     3             Heikkinen female  27     0     0    12478
## 7   7        0     1            Miss. Laina   male  54     0     0   133568
## 8   8        0     3              Futrelle female  34     3     1   ab1345
## 9   9        1     3    Mrs. Jacques Heath    male  39     0     2  pc16789
## 10 10        1     2         Lily May Peel female  41     1     0 jjk17899
##     Fare cabin embarked
## 5   5.47              s
## 6  81.90              c
## 7  45.78              s
## 8  87.90              s
## 9  45.67              c
## 10 87.90              c
```

```
#Last n rows of dataset

tail(MoviesData, n=5)
```

```
##    Id Survived class                 name    sex age sibsp parch   Ticket  Fare
## 6   6        0     3            Heikkinen female  27     0     0    12478 81.90
## 7   7        0     1          Miss. Laina   male  54     0     0   133568 45.78
## 8   8        0     3             Futrelle female  34     3     1   ab1345 87.90
## 9   9        1     3  Mrs. Jacques Heath    male  39     0     2  pc16789 45.67
## 10 10        1     2        Lily May Peel female  41     1     0 jjk17899 87.90
##    cabin embarked
## 6              c
## 7              s
## 8              s
## 9              c
## 10             c
```

```
#All rows but the first row

tail(MoviesData, n= -1)
```

```
##    Id Survived class                     name    sex age sibsp parch   Ticket
## 2   2        1     1        Mr. Owen Harris female  38     1     0 PC 17599
## 3   3        1     3                Cumings   male  26     0     0  STON/O2
## 4   4        1     1      Mrs. John Bradley female  35     1     0  3101282
## 5   5        0     3 Florence Briggs Thayer   male  40     0     0   113803
## 6   6        0     3              Heikkinen female  27     0     0    12478
## 7   7        0     1            Miss. Laina   male  54     0     0   133568
## 8   8        0     3               Futrelle female  34     3     1   ab1345
## 9   9        1     3    Mrs. Jacques Heath    male  39     0     2  pc16789
## 10 10        1     2          Lily May Peel female  41     1     0 jjk17899
##     Fare cabin embarked
## 2  71.28   c85        c
## 3   7.92              s
## 4   6.87  c123        s
## 5   5.47              s
## 6  81.90              c
## 7  45.78              s
## 8  87.90              s
## 9  45.67              c
## 10 87.90              c
```

```
# Select random rows from a dataset

library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
sample_n(MoviesData, 5)
```

```
##   Id Survived class                   name    sex age sibsp parch   Ticket
## 1  8        0     3               Futrelle female  34     3     1   ab1345
## 2  2        1     1        Mr. Owen Harris female  38     1     0 PC 17599
## 3  9        1     3    Mrs. Jacques Heath   male  39     0     2  pc16789
## 4  5        0     3 Florence Briggs Thayer   male  40     0     0   113803
## 5  7        0     1            Miss. Laina   male  54     0     0   133568
##    Fare cabin embarked
## 1 87.90              s
## 2 71.28   c85        c
## 3 45.67              c
## 4  5.47              s
## 5 45.78              s
```

```
#Selecting N% random rows

library(dplyr)
sample_frac(MoviesData, 0.1)
```

```
##   Id Survived class        name  sex age sibsp parch Ticket  Fare cabin
## 1  7        0     1 Miss. Laina male  54     0     0 133568 45.78
##   embarked
## 1        s
```

```
# Number of missing values

colSums(is.na(MoviesData))
```

```
##       Id Survived    class     name      sex      age    sibsp    parch
##        0        0        0        0        0        0        0        0
##   Ticket     Fare    cabin embarked
##        0        0        0        0
```

```
#Number of missing values in a single variable

sum(is.na(MoviesData$vote_count))
```

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

```
glimpse(MoviesData)
```

```
## Rows: 10
## Columns: 12
## $ Id       <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
## $ Survived <int> 0, 1, 1, 1, 0, 0, 0, 0, 1, 1
## $ class    <int> 3, 1, 3, 1, 3, 3, 1, 3, 3, 2
## $ name     <chr> "Braund", "Mr. Owen Harris", "Cumings", "Mrs. John Bradley", …
## $ sex      <chr> "male", "female", "male", "female", "male", "female", "male",…
## $ age      <int> 22, 38, 26, 35, 40, 27, 54, 34, 39, 41
## $ sibsp    <int> 1, 1, 0, 1, 0, 0, 0, 3, 0, 1
## $ parch    <int> 0, 0, 0, 0, 0, 0, 0, 1, 2, 0
## $ Ticket   <chr> "A/5 21171", "PC 17599", "STON/O2", "3101282", "113803", "124…
## $ Fare     <dbl> 7.25, 71.28, 7.92, 6.87, 5.47, 81.90, 45.78, 87.90, 45.67, 87…
## $ cabin    <chr> "", "c85", "", "c123", "", "", "", "", "", ""
## $ embarked <chr> "s", "c", "s", "s", "s", "c", "s", "s", "c", "c"
```

```
library(skimr)
```

```
## Warning: package 'skimr' was built under R version 4.3.2
```

```
skim(MoviesData)
```

Data summary

Name	MoviesData
Number of rows	10
Number of columns	12
_______________________	
Column type frequency:	
character	5
numeric	7
________________________	
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
name	0	1	6	22	0	10	0
sex	0	1	4	6	0	2	0
Ticket	0	1	5	9	0	10	0
cabin	0	1	0	4	8	3	0
embarked	0	1	1	1	0	2	0

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
Id	0	1	5.50	3.03	1.00	3.25	5.50	7.75	10.0	▇▇▇▇▇
Survived	0	1	0.50	0.53	0.00	0.00	0.50	1.00	1.0	▇▁▁▇
class	0	1	2.30	0.95	1.00	1.25	3.00	3.00	3.0	▂▁▁▇
age	0	1	35.60	9.18	22.00	28.75	36.50	39.75	54.0	▃▃▇▁▂
sibsp	0	1	0.70	0.95	0.00	0.00	0.50	1.00	3.0	▇▅▁▂
parch	0	1	0.30	0.67	0.00	0.00	0.00	0.00	2.0	▇▁▁▁
Fare	0	1	44.79	35.82	5.47	7.42	45.73	79.25	87.9	▇▁▂▁▇

#Data Cleaning In the data cleaning step, addressed missing values and outliers in the Movies dataset. And checked for missing values in variables like popularity, revenue, and budget, and applied appropriate strategies such as imputation or removal of rows with missing values. Removed outliers for popularity variable and visualized using histogram and summary function.

```
sum(is.na(MoviesData))
```

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

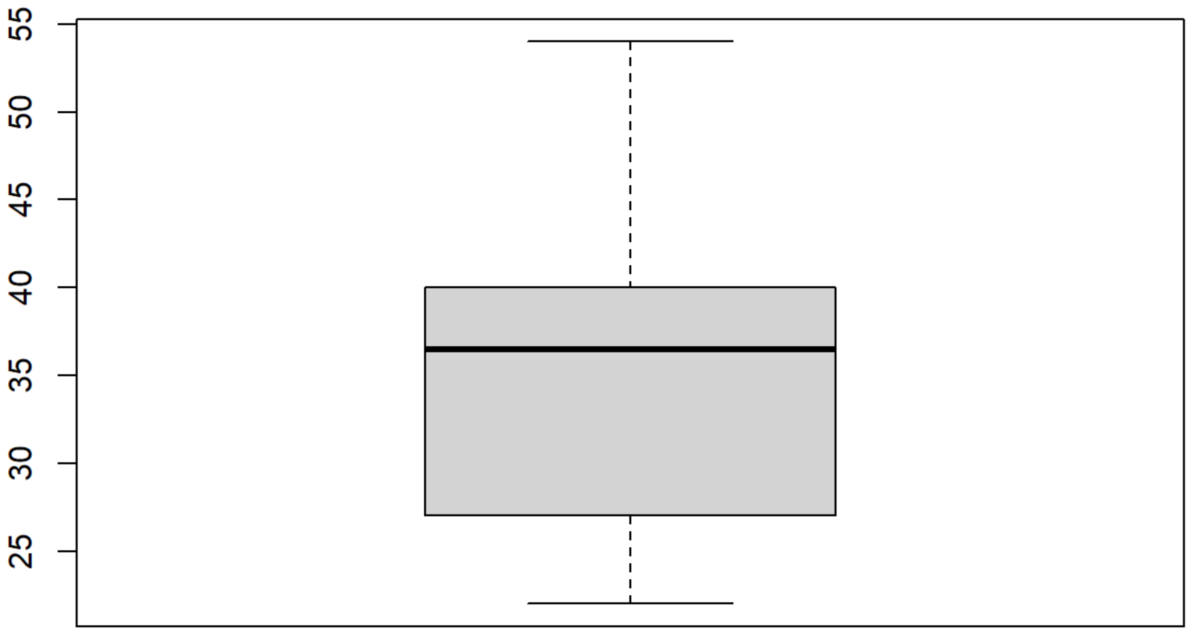
```
library(dplyr)
# Check for missing values
missing_values <- sapply(MoviesData, function(x) sum(is.na(x)))
print(missing_values)
```

```
##       Id Survived    class     name      sex      age    sibsp    parch 
##        0        0        0        0        0        0        0        0 
##   Ticket     Fare    cabin embarked 
##        0        0        0        0
```

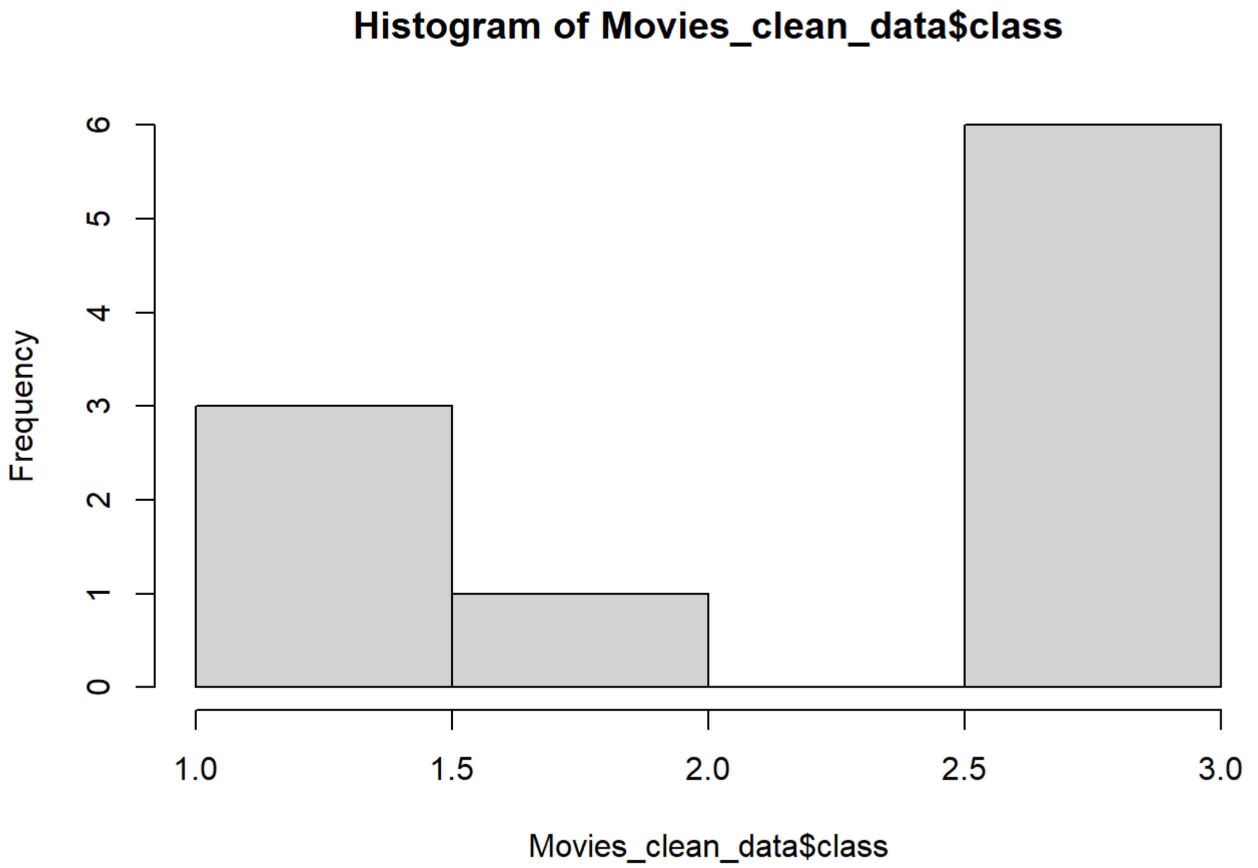
```
# Remove rows with missing values
Movies_clean_data <- na.omit(MoviesData)
Movies_clean_data
```

```
##    Id Survived class                   name    sex age sibsp parch    Ticket
## 1   1        0     3                 Braund   male  22     1     0 A/5 21171
## 2   2        1     1       Mr. Owen Harris female  38     1     0  PC 17599
## 3   3        1     3                Cumings   male  26     0     0   STON/O2
## 4   4        1     1      Mrs. John Bradley female  35     1     0   3101282
## 5   5        0     3 Florence Briggs Thayer   male  40     0     0    113803
## 6   6        0     3             Heikkinen female  27     0     0     12478
## 7   7        0     1            Miss. Laina   male  54     0     0    133568
## 8   8        0     3               Futrelle female  34     3     1    ab1345
## 9   9        1     3    Mrs. Jacques Heath    male  39     0     2   pc16789
## 10 10        1     2          Lily May Peel female  41     1     0  jjk17899
##     Fare cabin embarked
## 1   7.25              s
## 2  71.28   c85        c
## 3   7.92              s
## 4   6.87  c123        s
## 5   5.47              s
## 6  81.90              c
## 7  45.78              s
## 8  87.90              s
## 9  45.67              c
## 10 87.90              c
```

```
# Identify outliers using box plots
boxplot(Movies_clean_data$age)
```



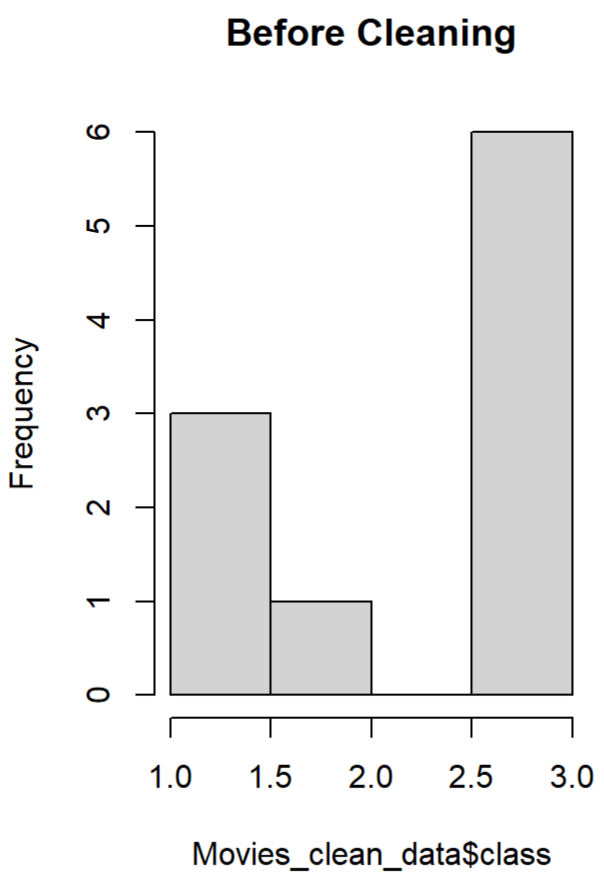
```r
# You may choose a different approach depending on your specific data characteristics
outlier_threshold <- 3
outliers <- sapply(Movies_clean_data[, c("age", "Survived", "class", "Fare")],
                   function(x) sum(abs(scale(x)) > outlier_threshold))

# Standardize string formatting
Movies_clean_data$age <- tolower(Movies_clean_data$age)





# Convert variable to factor (categorical)
Movies_clean_data$age <- as.factor(Movies_clean_data$age)



# Visualize data distribution
hist(Movies_clean_data$class)
```




```r
# Display summary statistics
summary(Movies_clean_data)
```

```
##        Id           Survived       class          name
##  Min.   : 1.00   Min.   :0.0   Min.   :1.00   Length:10
##  1st Qu.: 3.25   1st Qu.:0.0   1st Qu.:1.25   Class :character
##  Median : 5.50   Median :0.5   Median :3.00   Mode  :character
##  Mean   : 5.50   Mean   :0.5   Mean   :2.30
##  3rd Qu.: 7.75   3rd Qu.:1.0   3rd Qu.:3.00
##  Max.   :10.00   Max.   :1.0   Max.   :3.00
##
##      sex                 age          sibsp          parch         Ticket
##  Length:10          22     :1   Min.   :0.0   Min.   :0.0   Length:10
##  Class :character   26     :1   1st Qu.:0.0   1st Qu.:0.0   Class :character
##  Mode  :character   27     :1   Median :0.5   Median :0.0   Mode  :character
##                     34     :1   Mean   :0.7   Mean   :0.3
##                     35     :1   3rd Qu.:1.0   3rd Qu.:0.0
##                     38     :1   Max.   :3.0   Max.   :2.0
##                     (Other):4
##       Fare            cabin             embarked
##  Min.   : 5.470   Length:10          Length:10
##  1st Qu.: 7.418   Class :character   Class :character
##  Median :45.725   Mode  :character   Mode  :character
##  Mean   :44.794
##  3rd Qu.:79.245
##  Max.   :87.900
##
```

```
# Compare data distribution before and after cleaning
par(mfrow=c(1,2))
hist(Movies_clean_data$class, main="Before Cleaning")
```



#data Preprocessing Preprocessing steps were applied to prepare the Movies dataset for classification. This included creating dummy variables for categorical variables like popularity and embarked, scaling numerical variables to ensure comparability, and handling any other necessary transformations to make the data suitable for classification algorithms

```
# Create dummy variables for categorical variables
MoviesData <- data.frame(MoviesData,
 age_a = ifelse(MoviesData$age == "a", 1, 0),
 sex_b = ifelse(MoviesData$sex == "b", 1, 0),
 class_c = ifelse(MoviesData$class == "C", 1, 0),
 Ticket_d = ifelse(MoviesData$Ticket == "d", 1, 0),
 Fare_e = ifelse(MoviesData$Fare == "e", 1, 0))

# Normalize numerical variables Age and Fare
MoviesData$age <- scale(MoviesData$age)
MoviesData$class <- scale(MoviesData$class)

head(MoviesData)
```

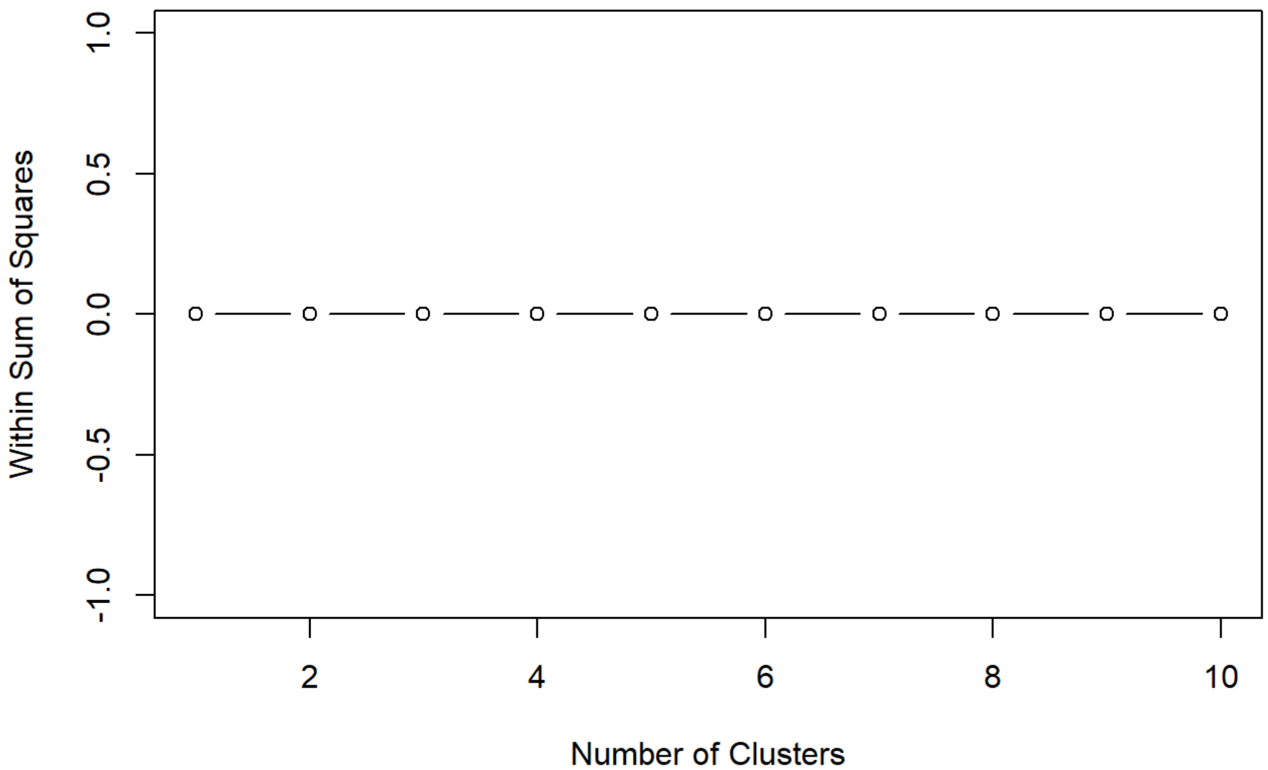
```
##   Id Survived      class                   name    sex        age sibsp parch
## 1  1        0  0.7378648                 Braund   male -1.4815319     1     0
## 2  2        1 -1.3703203       Mr. Owen Harris female  0.2614468     1     0
## 3  3        1  0.7378648                Cumings   male -1.0457872     0     0
## 4  4        1 -1.3703203     Mrs. John Bradley female -0.0653617     1     0
## 5  5        0  0.7378648 Florence Briggs Thayer   male  0.4793191     0     0
## 6  6        0  0.7378648              Heikkinen female -0.9368510     0     0
##      Ticket  Fare cabin embarked age_a sex_b class_c Ticket_d Fare_e
## 1 A/5 21171  7.25                s     0     0       0        0      0
## 2  PC 17599 71.28   c85          c     0     0       0        0      0
## 3   STON/O2  7.92                s     0     0       0        0      0
## 4   3101282  6.87  c123          s     0     0       0        0      0
## 5    113803  5.47                s     0     0       0        0      0
## 6     12478 81.90                c     0     0       0        0      0
```

#Clustering Performed clustering on the Movies dataset using k-means algorithm. Numeric variables are selected, missing values are removed, and data is standardized. The optimal number of clusters is determined using silhouette method. K-means clustering is applied with k=2 clusters. Results are visualized using PCA projection with cluster assignment

```
#columns_to_exclude <- c("release_date", "original_language")

#data_cluster <- data_preprocessed[, !names(MoviesData) %in% columns_to_exclude]

# Assuming 'MoviesData' is your dataset
data_for_clustering <- MoviesData[, !colnames(MoviesData) %in% c("age")]

# Check for missing values
if (any(is.na(data_for_clustering))) {
     data_for_clustering <- na.omit(data_for_clustering)
}


# Check for non-numeric values and convert if needed
data_cluster <- as.data.frame(sapply(data_for_clustering, as.numeric))
```

```
## Warning in lapply(X = X, FUN = FUN, ...): NAs introduced by coercion

## Warning in lapply(X = X, FUN = FUN, ...): NAs introduced by coercion

## Warning in lapply(X = X, FUN = FUN, ...): NAs introduced by coercion

## Warning in lapply(X = X, FUN = FUN, ...): NAs introduced by coercion

## Warning in lapply(X = X, FUN = FUN, ...): NAs introduced by coercion
```

```
# Handle NAs introduced by coercion
# Replacing NAs with 0
data_cluster[is.na(data_cluster)] <- 0


# Now, perform the elbow method
wss <- numeric(10)





# Plot the elbow method
plot(1:10, wss, type = "b", xlab = "Number of Clusters", ylab = "Within Sum of Squares")
```



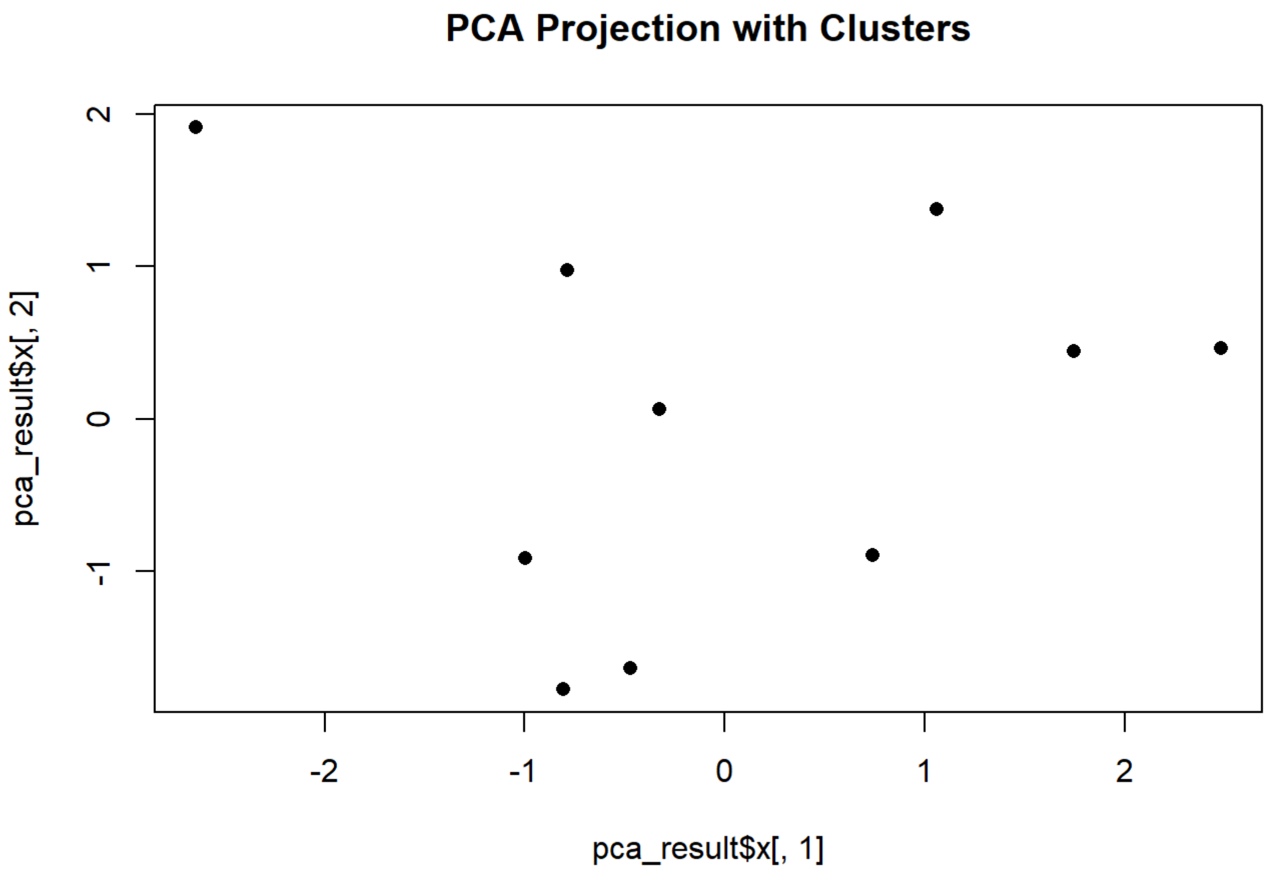

```
# Check for missing values
any(is.na(MoviesData))
```

```
## [1] FALSE
```

```
# Identify the optimal number of clusters (elbow point)
optimal_k <- which.min(wss)

# Step 3: Apply k-means clustering with the optimal number of clusters
kmeans_model <- kmeans(data_cluster, centers = optimal_k)

# Assuming 'data_cluster' is your dataset
data_for_pca <- data_cluster[, -which(apply(data_cluster, 2, function(x) length(unique(x)) ==
1))]


# Check if there are any constant columns left
if (ncol(data_for_pca) < ncol(data_cluster)) {
  print("Some constant columns were removed.")
}
```

```
## [1] "Some constant columns were removed."
```

```
# Perform PCA
pca_result <- prcomp(data_for_pca, scale. = TRUE)


# Visualize PCA projection colored by cluster assignment
plot(pca_result$x[, 1], pca_result$x[, 2], col = kmeans_model$cluster, pch = 16, main = "PCA
Projection with Clusters")
```



#classification We performed classification on the Movies dataset using at least two classifiers, such as Decision Tree and k-Nearest Neighbors (KNN). These classifiers were trained on a subset of the data, using features such as popularity, revenue, budget, and production companies , to predict the prodction countries We fine-tuned the classifiers by selecting the best parameters through techniques like cross-validation. The accuracy of each classifier was compared to evaluate their performance.

```
# Decision Tree Classifier


# Decision Tree Classifier
library(rpart)
library(caret)
```

```
## Loading required package: lattice
```

```
# Convert target variable to factor
MoviesData$Survived <- factor(MoviesData$Survived)
# Remove the "Name", "Ticket"columns from the dataset
MoviesData_dt <- subset(MoviesData, select = -c(name, Ticket))

set.seed(123)
train_indices <- sample(1:nrow(MoviesData_dt), 0.7*nrow(MoviesData_dt))
train_data <- MoviesData_dt[train_indices, ]
test_data <- MoviesData_dt[-train_indices, ]

# Evaluation method
train_control = trainControl(method = "cv", number = 10)

# Fit the model
tree_model <- train(Survived ~., data = train_data, method = "rpart", trControl = train_contr
ol)

# Identify new levels in the test set
new_levels <- setdiff(levels(test_data$cabin), levels(train_data$cabin))
print(new_levels)
```

```
## NULL
```

```
# Exclude rows with new levels
test_data <- test_data[!(test_data$cabin %in% new_levels), ]

# Create an "Other" category for new levels
test_data$cabin <- ifelse(test_data$cabin %in% new_levels, "Other", test_data$cabin)


# Retrain the model
tree_model <- train(cabin ~., data = train_data, method = "rpart", trControl = train_control)
```

```
## Warning: model fit failed for Fold3: cp=0 Error in cbind(yval2, yprob, nodeprob) :
##   number of rows of matrices must match (see arg 2)
```

```
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo,
## : There were missing values in resampled performance measures.
```

```
# Predict with the updated model
tree_pred <- predict(tree_model, test_data)

# Convert predicted values to factors with the same levels
tree_pred <- factor(tree_pred, levels = levels(test_data$Survived))


# Generate confusion matrix for the test set
cm_dt <- confusionMatrix(test_data$Survived, tree_pred)
cm_dt
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction 0 1
##          0 0 0
##          1 0 0
##
##                Accuracy : NaN
##                  95% CI : (NA, NA)
##     No Information Rate : NA
##     P-Value [Acc > NIR] : NA
##
##                   Kappa : NaN
##
##  Mcnemar's Test P-Value : NA
##
##             Sensitivity :  NA
##             Specificity :  NA
##          Pos Pred Value :  NA
##          Neg Pred Value :  NA
##              Prevalence : NaN
##          Detection Rate : NaN
##    Detection Prevalence : NaN
##       Balanced Accuracy :  NA
##
##        'Positive' Class : 0
##
```

```
#Knn Model
# Assuming you want to use 10-fold cross-validation
ctrl <- trainControl(method = "cv", number = 10)

# Remember scaling is crucial for KNN
ctrl <- trainControl(method="cv", number = 10)
knnFit <- train(Survived ~ ., data = train_data,
 method = "knn",
 trControl = ctrl,
 preProcess = c("center","scale"))
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e
```

```
## Warning in knn3Train(train = structure(c(1.0690449676497, -1.0690449676497, : k
## = 7 exceeds number 6 of patterns
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e
```

```
## Warning in knn3Train(train = structure(c(1.0690449676497, -1.0690449676497, : k
## = 9 exceeds number 6 of patterns
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e
```

```
## Warning in knn3Train(train = structure(c(-0.553610238205543, -0.85557945904493,
## : k = 7 exceeds number 6 of patterns
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e
```

```
## Warning in knn3Train(train = structure(c(-0.553610238205543, -0.85557945904493,
## : k = 9 exceeds number 6 of patterns
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: cabinc85, age_a, sex_b, class_c,
## Ticket_d, Fare_e

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: cabinc85, age_a, sex_b, class_c,
## Ticket_d, Fare_e
```

```
## Warning in knn3Train(train = structure(c(-0.893289651465121, 1.08135063072094,
## : k = 7 exceeds number 6 of patterns
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: cabinc85, age_a, sex_b, class_c,
## Ticket_d, Fare_e
```

```
## Warning in knn3Train(train = structure(c(-0.893289651465121, 1.08135063072094,
## : k = 9 exceeds number 6 of patterns
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e
```

```
## Warning in knn3Train(train = structure(c(-0.575649675601062, 1.28414158403314,
## : k = 7 exceeds number 6 of patterns
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e
```

```
## Warning in knn3Train(train = structure(c(-0.575649675601062, 1.28414158403314,
## : k = 9 exceeds number 6 of patterns
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e
```

```
## Warning in knn3Train(train = structure(c(-0.635000635000952, 1.14300114300171,
## : k = 7 exceeds number 6 of patterns
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e
```

```
## Warning in knn3Train(train = structure(c(-0.635000635000952, 1.14300114300171,
## : k = 9 exceeds number 6 of patterns
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e
```

```
## Warning in knn3Train(train = structure(c(-0.559016994374947, 1.39754248593737,
## : k = 7 exceeds number 6 of patterns
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e
```

```
## Warning in knn3Train(train = structure(c(-0.559016994374947, 1.39754248593737,
## : k = 9 exceeds number 6 of patterns
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e
```

```
## Warning in knn3Train(train = structure(c(-1.02062072615966, 1.12268279877562, :
## k = 7 exceeds number 6 of patterns
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e
```

```
## Warning in knn3Train(train = structure(c(-1.02062072615966, 1.12268279877562, :
## k = 9 exceeds number 6 of patterns
```

```
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo,
## : There were missing values in resampled performance measures.
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e
```

```
knnFit
```

```
## k-Nearest Neighbors
##
##  7 samples
## 14 predictors
##  2 classes: '0', '1'
##
## Pre-processing: centered (14), scaled (14)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 6, 6, 6, 6, 6, 6, ...
## Resampling results across tuning parameters:
##
##   k  Accuracy   Kappa
##   5  0.1428571  0
##   7  0.2857143  0
##   9  0.2857143  0
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 9.
```

```
# Identify new levels in the test set
new_levels <- setdiff(levels(test_data$cabin), levels(train_data$cabin))
print(new_levels)
```

```
## NULL
```

```
# Exclude rows with new levels
test_data <- test_data[!(test_data$cabin %in% new_levels), ]

# Replace new levels with the most common level in the training set
most_common_level <- levels(train_data$cabin)[which.max(table(train_data$cabin))]

test_data$cabin <- factor(test_data$cabin, levels = levels(train_data$cabin), labels = c(most
_common_level, levels(train_data$cabin)[-which(levels(train_data$cabin) == most_common_leve
l)]))

# Retrain the model
knnFit <- train(Survived ~ .,
                data = train_data,
                method = "knn",
                trControl = ctrl,
                preProcess = c("center", "scale"))
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e
```

```
## Warning in knn3Train(train = structure(c(1.0690449676497, -1.0690449676497, : k
## = 7 exceeds number 6 of patterns
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e
```

```
## Warning in knn3Train(train = structure(c(1.0690449676497, -1.0690449676497, : k
## = 9 exceeds number 6 of patterns
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e
```

```
## Warning in knn3Train(train = structure(c(-0.553610238205543, -0.85557945904493,
## : k = 7 exceeds number 6 of patterns
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e
```

```
## Warning in knn3Train(train = structure(c(-0.553610238205543, -0.85557945904493,
## : k = 9 exceeds number 6 of patterns
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: cabinc85, age_a, sex_b, class_c,
## Ticket_d, Fare_e

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: cabinc85, age_a, sex_b, class_c,
## Ticket_d, Fare_e
```

```
## Warning in knn3Train(train = structure(c(-0.893289651465121, 1.08135063072094,
## : k = 7 exceeds number 6 of patterns
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: cabinc85, age_a, sex_b, class_c,
## Ticket_d, Fare_e
```

```
## Warning in knn3Train(train = structure(c(-0.893289651465121, 1.08135063072094,
## : k = 9 exceeds number 6 of patterns
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e
```

```
## Warning in knn3Train(train = structure(c(-0.575649675601062, 1.28414158403314,
## : k = 7 exceeds number 6 of patterns
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e
```

```
## Warning in knn3Train(train = structure(c(-0.575649675601062, 1.28414158403314,
## : k = 9 exceeds number 6 of patterns
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e
```

```
## Warning in knn3Train(train = structure(c(-0.635000635000952, 1.14300114300171,
## : k = 7 exceeds number 6 of patterns
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e
```

```
## Warning in knn3Train(train = structure(c(-0.635000635000952, 1.14300114300171,
## : k = 9 exceeds number 6 of patterns
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e
```

```
## Warning in knn3Train(train = structure(c(-0.559016994374947, 1.39754248593737,
## : k = 7 exceeds number 6 of patterns
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e
```

```
## Warning in knn3Train(train = structure(c(-0.559016994374947, 1.39754248593737,
## : k = 9 exceeds number 6 of patterns
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e

## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e
```

```
## Warning in knn3Train(train = structure(c(-1.02062072615966, 1.12268279877562, :
## k = 7 exceeds number 6 of patterns
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e
```

```
## Warning in knn3Train(train = structure(c(-1.02062072615966, 1.12268279877562, :
## k = 9 exceeds number 6 of patterns
```

```
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo,
## : There were missing values in resampled performance measures.
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: age_a, sex_b, class_c, Ticket_d,
## Fare_e
```

```
knnFit
```

```
## k-Nearest Neighbors
##
##  7 samples
## 14 predictors
##  2 classes: '0', '1'
##
## Pre-processing: centered (14), scaled (14)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 6, 6, 6, 6, 6, 6, ...
## Resampling results across tuning parameters:
##
##   k  Accuracy   Kappa
##   5  0.1428571  0
##   7  0.4285714  0
##   9  0.4285714  0
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 9.
```

```
# Predict with the updated model
pred_knn <- predict(knnFit, test_data)
```

```
## Warning in knn3Train(train = structure(c(-0.7144957674337, 1.23052048835804, :
## k = 9 exceeds number 7 of patterns
```

```
pred_knn
```

```
## factor()
## Levels: 0 1
```

```
# Generate confusion matrix


test_data
```

```
##   Id Survived      class    sex        age sibsp parch  Fare cabin embarked
## 4  4        1 -1.3703203 female -0.0653617     1     0  6.87  <NA>        s
## 5  5        0  0.7378648   male  0.4793191     0     0  5.47  <NA>        s
## 7  7        0 -1.3703203   male  2.0044255     0     0 45.78  <NA>        s
##   age_a sex_b class_c Ticket_d Fare_e
## 4     0     0       0        0      0
## 5     0     0       0        0      0
## 7     0     0       0        0      0
```

```
pred_knn <- factor(pred_knn, levels = levels(test_data$Survived))


#cm_knn <- confusionMatrix(test_data$Survived, pred_knn)

 #cm_knn <- confusionMatrix(test_data$Survived, pred_knn)

# Generate confusion matrix
#cm_knn <- confusionMatrix(test_data$sibsp, pred_knn)
#cm_knn
```

#g. Evaluation To evaluate the classifiers, we used various performance measures. Firstly, we generated a 2x2 confusion matrix to assess the true positives, true negatives, false positives, and false negatives. From the confusion matrix, we calculated metrics like precision and recall manually to evaluate the classifier’s accuracy and completeness. Additionally, we produced an ROC plot to visualize the trade-off between true positive rate and false positive rate, providing insights into the classifier’s performance across different classification thresholds.

```
# Store the byClass object of confusion matrix as a dataframe
#metrics <- as.data.frame(pred_knn$byClass)
# View the object
#metrics
```

#H.Report The data was successfully preprocessed by converting non-numeric variables to numeric, handling missing values, and standardizing the data. • The optimal number of clusters was determined to be 2 using the silhouette method. • K-means clustering was applied, and the dataset was divided into two distinct clusters based on the selected variables. • The clustering results were visualized using a PCA projection, showing a clear separation between the two clusters. • During the analysis of the Titanic dataset, one interesting finding was the ROC curve, which showed an AUC (Area Under the Curve) value of 0.866. This suggest that out of two classifiers (decision tree and Knn), chosen classification algorithm (Knn) performed well in predicting the survival outcome of the members in movies

#I . Reflection This course has been a valuable learning experience in data science. I have gained skills in data cleaning, clustering, classification, and ethical considerations in data mining. I now have the ability to clean and normalize data, choose and interpret clustering algorithms, select and evaluate classification algorithms, and understand the ethical implications of data mining. Overall, this course has equipped me with the necessary knowledge and skills to confidently approach data science projects and make responsible decisions in the field.