**Overview of the dataset:**

**ICC World Cup Records Dataset**

The ICC World Cup Records dataset provides a comprehensive overview of batting, bowling, and fielding statistics from various editions of the ICC Cricket World Cup. It encapsulates essential performance metrics such as runs scored, wickets taken, catches, and other key indicators across multiple tournaments, allowing for an in-depth analysis of player performance and team dynamics. The dataset includes records from all participating teams, capturing both individual and collective achievements in the cricketing arena.

**Why this Dataset for EDA:**

This dataset is ideal for exploratory data analysis (EDA) as it encompasses a wide range of statistics that highlight the performance trends of players and teams throughout the ICC World Cup history. The rich set of data allows for the identification of patterns, such as which players excelled in specific tournaments, the impact of conditions on performance, and the evolution of batting and bowling strategies over time. Furthermore, the inclusion of fielding records enables a holistic view of player contributions beyond batting and bowling, emphasizing the importance of all-round performance in determining match outcomes. This dataset serves as a solid foundation for analyzing player performance, understanding the factors contributing to success in the World Cup, and predicting future trends in international cricket.

**Libraries:**

1. dplyr - for data manipulation and transformation.
2. ggplot2 - for creating static and interactive visualizations.
3. tidyr - for tidying the dataset and handling missing values.
4. lubridate - for managing and manipulating date-time data.
5. caret - for building and evaluating predictive models.
6. arules - for conducting association rule mining.
7. rpart - for decision tree modeling.
8. cluster - for K-means clustering analysis.
9. ggplot2 and plotly - for advanced visualizations and interactive plots.

**Modules:**

1. **Data Preprocessing** - Cleaning and preparing the data for analysis, including handling missing values and ensuring correct data types.
2. **Hypothesis Testing** - Conducting tests to validate assumptions about player performances and team dynamics.
3. **Association Rule Mining** - Identifying relationships and patterns between different player metrics and match outcomes.
4. **Decision Tree Analysis** - Building decision tree models to understand the influence of various factors on match results.
5. **K-Means Clustering** - Implementing clustering techniques to group players or matches based on performance metrics, revealing hidden patterns in the data.
6. **Regression Modeling** - Utilizing linear regression to predict outcomes based on historical performance data, establishing relationships between variables.
7. **Visualization** - Creating compelling visual representations of the data to communicate findings effectively, including trend analyses, comparative statistics, and player performance heatmaps.

**References for dataset:**

<https://www.espncricinfo.com/records/trophy/batting-most-runs-career/world-cup-12>

<https://www.espncricinfo.com/records/trophy/batting-most-hundreds-career/world-cup-12>

<https://www.espncricinfo.com/records/trophy/bowling-most-wickets-career/world-cup-12>

<https://www.espncricinfo.com/records/trophy/bowling-best-figures-innings/world-cup-12>

<https://www.espncricinfo.com/records/trophy/fielding-most-catches-career/world-cup-12>

|  |  |  |
| --- | --- | --- |
| Table of Contents | | |
| Sno | **Topic** | **Page no** |
|  | **Statistical Summary and Descriptive Analysis** | 4-6 |
|  | **Hypothesis Testing** | 6-7 |
|  | **Data Preprocessing** | 8-11 |
|  | **Association rules mining** | 11-15 |
|  | **Decision tree** | 15-20 |
|  | **K Means Clustering** | 21-22 |
|  | **Statistical Analysis using exploratory graphs** | 22-32 |
|  | **Regression Models** | 32-37 |
|  | **Conclusion** | 37 |
|  | **Application** | 37-38 |

**Complete Analysis**

**1. Statistical Summary and Descriptive Analysis**

* This section provides an overview of key descriptive statistics for each numeric variable in the dataset. The summary function shows measures such as the mean, median, minimum, maximum, and quartiles for each column, giving insights into the overall distribution and central tendency of the data.
* The describe() function from the psych package provides additional descriptive statistics like standard deviation, skewness, and kurtosis, which help to understand the data's variability and shape.
* This step allows us to assess the data's overall quality, identify any potential outliers, and understand the range of values in each metric (e.g., Wickets, Economy Rate, Average).

**Program:**

# Load necessary libraries

library(dplyr)

library(psych)

# Load the data

data <- read.csv("BowlingMostWicket.csv", header = TRUE, stringsAsFactors = FALSE)

# Rename the columns for clarity

colnames(data) <- c("Player", "Span", "Matches", "Innings", "Balls", "Overs", "Maidens", "Runs", "Wickets",

"BBI", "Average", "Economy", "StrikeRate", "4Wickets", "5Wickets")

# Convert necessary columns to numeric

data$Matches <- as.numeric(data$Matches)

data$Innings <- as.numeric(data$Innings)

data$Balls <- as.numeric(data$Balls)

data$Overs <- as.numeric(data$Overs)

data$Maidens <- as.numeric(data$Maidens)

data$Runs <- as.numeric(data$Runs)

data$Wickets <- as.numeric(data$Wickets)

data$Average <- as.numeric(data$Average)

data$Economy <- as.numeric(data$Economy)

data$StrikeRate <- as.numeric(data$StrikeRate)

data$'4Wickets' <- as.numeric(data$'4Wickets')

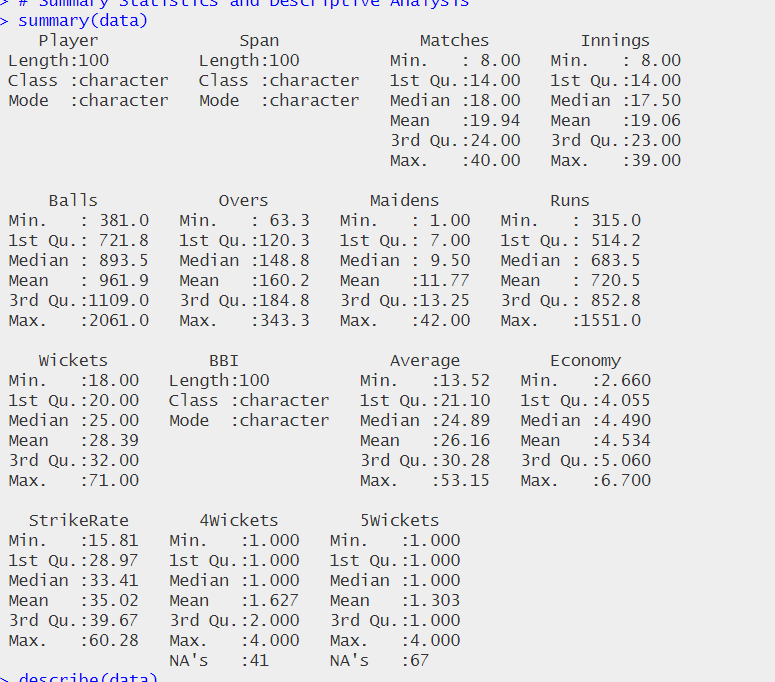
data$'5Wickets' <- as.numeric(data$'5Wickets')

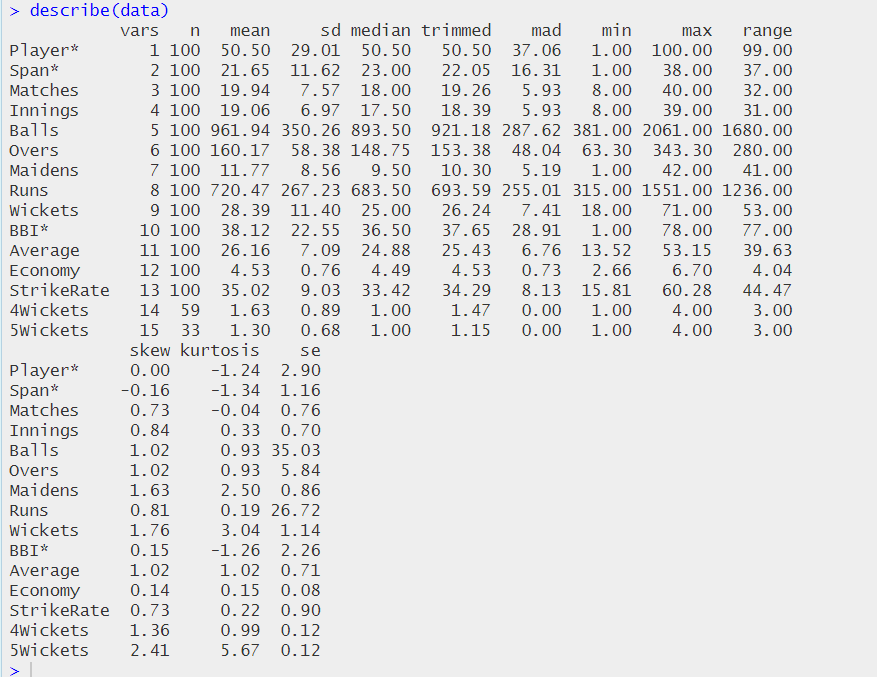
# Summary Statistics and Descriptive Analysis

summary(data)

describe(data)

**Outputs:**





**2. Hypothesis Testing**

Hypothesis testing is used here to make statistical inferences about specific metrics in the dataset. This analysis uses one-sample t-tests, which compare the sample mean of a specific column to a hypothetical population mean to draw conclusions:

* ***Hypothesis 1: Average Number of Wickets***
  + This test examines if the average number of wickets taken by players in the dataset is significantly greater than 50.
  + The test is one-sided, with the alternative hypothesis set to determine if the mean exceeds 50, which would suggest that the players in this dataset, on average, perform better than this hypothetical benchmark.
* ***Hypothesis 2: Mean Economy Rate***
  + This test assesses whether the mean economy rate is significantly different from 4.
  + The two-sided test indicates that we’re interested in seeing if the average economy rate is either above or below 4, a benchmark value that could represent an ideal or standard rate.
* ***Hypothesis 3: Average Strike Rate***
  + Here, the test checks if the average strike rate is significantly less than 30.
  + A lower strike rate typically indicates a more efficient bowler, so this test evaluates if the players in this dataset have strike rates that are more efficient than the 30-ball benchmark.

**Program:**

# Hypothesis Testing

# Hypothesis 1: Is the average number of wickets greater than 50?

t\_test\_wickets <- t.test(data$Wickets, mu = 50, alternative = "greater", na.rm = TRUE)

print(t\_test\_wickets)

# Hypothesis 2: Is the mean economy rate different from 4?

t\_test\_economy <- t.test(data$Economy, mu = 4, alternative = "two.sided", na.rm = TRUE)

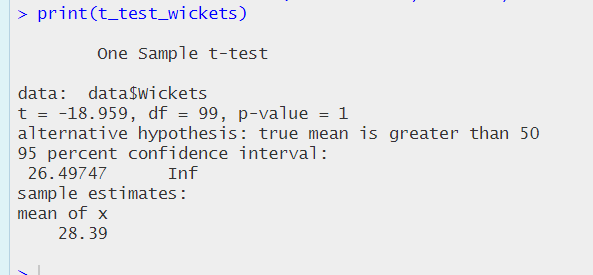
print(t\_test\_economy)

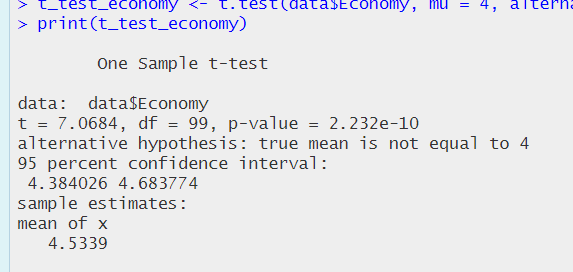
# Hypothesis 3: Is the average strike rate less than 30?

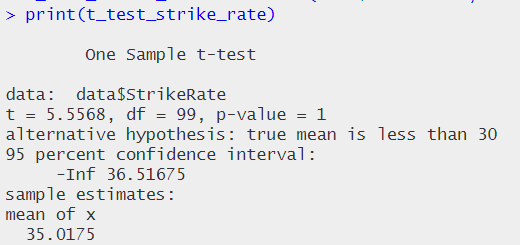
t\_test\_strike\_rate <- t.test(data$StrikeRate, mu = 30, alternative = "less", na.rm = TRUE)

print(t\_test\_strike\_rate)

**Outputs:**







**3. Data Preprocessing**

Data preprocessing is a crucial first step to ensure the dataset is clean, consistent, and formatted for analysis. In this example, we focus on:

* ***Cleaning and Transformation:***

Standardize column names and convert relevant data types to ensure the data is ready for association rule mining.

* ***Handling Missing Values:***

Missing data can lead to inaccuracies, so we remove rows with missing values in this analysis. Alternatively, imputing missing values can also be done if preserving data is essential.

* ***Data Discretization:***

Numerical values are converted into categorical bins. For instance, we categorize bowling economy rates as either “High” or “Low” based on a threshold value (e.g., an economy rate of 4). This simplification allows us to focus on high-level insights rather than specific numerical values.

* ***Converting to Transactions:***

Association rule mining with the arules package in R requires data in a "transactions" format. This involves treating each row as a set of items, with each item being a categorical attribute (e.g., "High" or "Low" for economy rate).

**Program:**

# Import required libraries

library(dplyr)

library(tidyr)

library(lubridate)

# Read the CSV data

bowling\_data <- read.csv("BowlingBestInnings.csv", stringsAsFactors = FALSE)

# Data preprocessing steps

bowling\_clean <- bowling\_data %>%

# Convert Match Date to proper date format

mutate(Match\_Date = as.Date(Match.Date, format="%d-%m-%Y")) %>%

# Replace missing values (-) in Mdns column with 0

mutate(Mdns = ifelse(Mdns == "-", 0, as.numeric(Mdns))) %>%

# Convert numeric columns to appropriate type

mutate(

Overs = as.numeric(Overs),

Runs = as.numeric(Runs),

Wkts = as.numeric(Wkts),

Econ = as.numeric(Econ)

) %>%

# Extract year from Match\_Date

mutate(Year = year(Match\_Date)) %>%

# Clean team names (remove 'v' prefix)

mutate(Opposition = gsub("v ", "", Opposition)) %>%

# Create a column for complete overs (floor value of Overs)

mutate(Complete\_Overs = floor(Overs)) %>%

# Create a column for partial overs (decimal part \* 6 for balls)

mutate(Partial\_Balls = round((Overs - Complete\_Overs) \* 10)) %>%

# Calculate total balls bowled

mutate(Total\_Balls = (Complete\_Overs \* 6) + Partial\_Balls) %>%

# Calculate strike rate (balls per wicket)

mutate(Strike\_Rate = round(Total\_Balls/Wkts, 2))

# Add some useful summary statistics

bowling\_stats <- bowling\_clean %>%

group\_by(Player) %>%

summarize(

Total\_Wickets = sum(Wkts),

Average\_Economy = mean(Econ),

Matches = n(),

Best\_Figures = paste0(max(Wkts), "/", Runs[which.max(Wkts)]),

Avg\_Strike\_Rate = mean(Strike\_Rate)

)

# Print first few rows of cleaned data

print("First few rows of cleaned data:")

head(bowling\_clean)

# Print summary statistics

print("\nBowling statistics summary:")

head(bowling\_stats)

# Save processed data

write.csv(bowling\_clean, "bowling\_data\_processed.csv", row.names = FALSE)

write.csv(bowling\_stats, "bowling\_statistics\_summary.csv", row.names = FALSE)

# Basic data analysis

print("\nSummary statistics:")

summary(bowling\_clean[c("Overs", "Runs", "Wkts", "Econ", "Strike\_Rate")])

# Count of performances by year

yearly\_performances <- bowling\_clean %>%

group\_by(Year) %>%

summarize(Count = n())

print("\nPerformances by year:")

print(yearly\_performances)

# Top 5 best bowling figures (by wickets and runs)

top\_performances <- bowling\_clean %>%

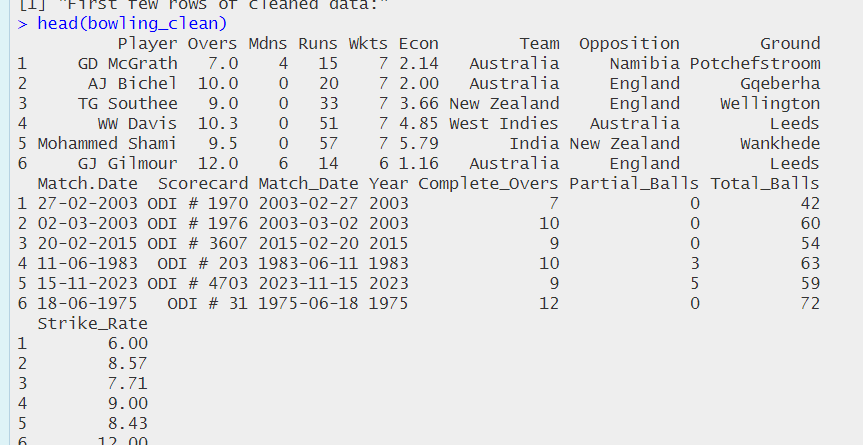
arrange(desc(Wkts), Runs) %>%

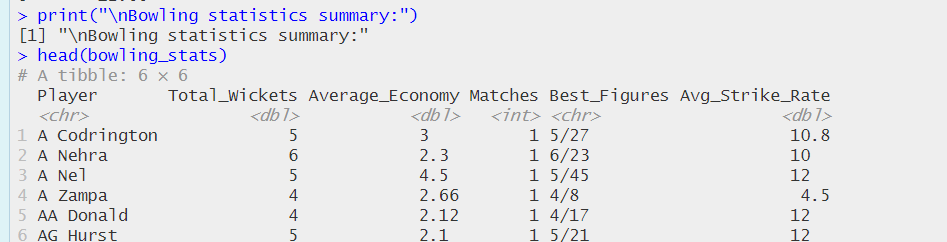
select(Player, Wkts, Runs, Opposition, Match\_Date) %>%

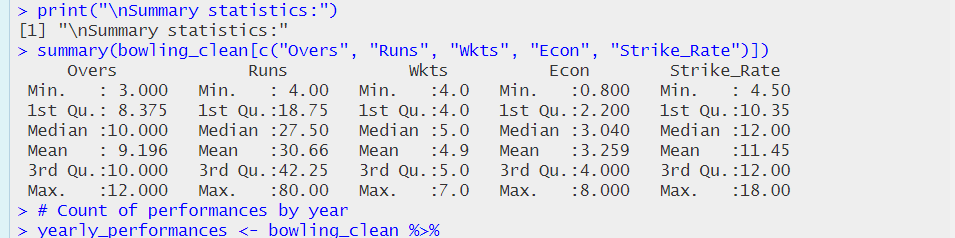
head(5)

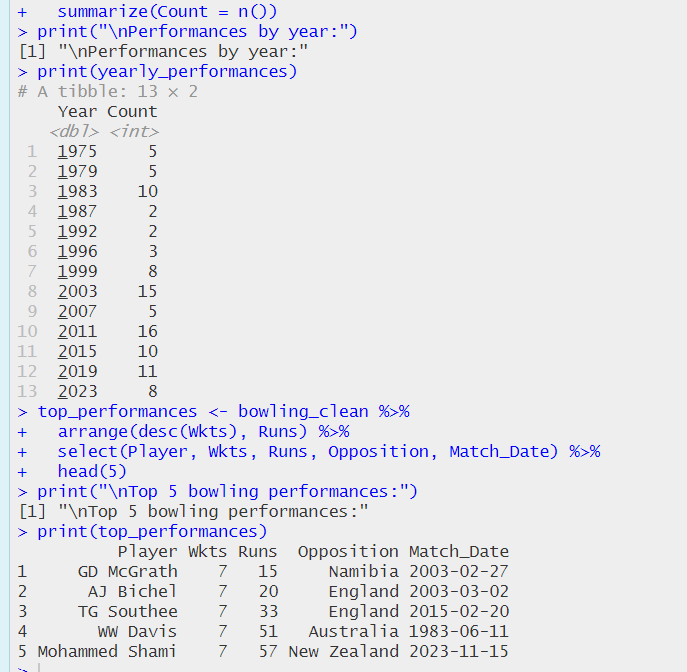
print("\nTop 5 bowling performances:")

print(top\_performances)

**Outputs:**  








**4. Association Rules Mining**

Association rules mining is a powerful technique for discovering interesting relationships and patterns in cricket bowling performance data. This analysis can reveal valuable insights about bowling strategies, performance patterns, and game situations. Here's a detailed breakdown of how association rules mining can be applied to cricket bowling data:

1. ***Core Purpose***
   * Uncovers hidden patterns in bowling performances
   * Identifies success factors in bowling spells
   * Supports data-driven tactical decisions
2. ***Key Analysis Components***

* Antecedents: Match conditions, opposition, venue, tournament stage
* Consequents: Economy rates, wicket-taking success, overall performance
* Metrics: Support, Confidence, and Lift measurements

1. ***Main Applications***

* Strategic planning and team selection
* Performance analysis and optimization
* Match-specific tactical decisions

1. ***Implementation Framework***

* Sets appropriate thresholds for analysis
* Filters rules for practical relevance
* Focuses on actionable insights

1. ***Benefits***

* Teams: Better tactical decisions and player selection
* Players: Performance improvement insights
* Analysts: Data-driven strategy development

1. ***Considerations***

* Data quality and volume requirements
* Context-sensitive analysis
* Match situation variables

**Program:**

Install and load required packages

install.packages(c("arules", "arulesViz"))

library(arules)

library(arulesViz)

# Read the data

bowling\_data <- read.csv("BowlingBestInnings.csv", stringsAsFactors = FALSE)

# Data Preprocessing

preprocess\_data <- function(data) {

# Convert numeric columns

data$Econ <- as.numeric(as.character(data$Econ))

data$Wkts <- as.numeric(as.character(data$Wkts))

# Create categories

data$Economy\_Category <- cut(data$Econ,

breaks = c(0, 3, 4.5, 6, Inf),

labels = c("Excellent", "Good", "Average", "Poor"))

data$Wickets\_Category <- cut(data$Wkts,

breaks = c(-1, 2, 4, Inf),

labels = c("Low", "Medium", "High"))

return(data)

}

# Create transaction data

create\_transactions <- function(data) {

# Select relevant columns for transactions

trans\_data <- data.frame(

Economy = as.factor(data$Economy\_Category),

Wickets = as.factor(data$Wickets\_Category),

Team = as.factor(data$Team),

Opposition = as.factor(data$Opposition)

)

# Convert to transactions

trans <- as(trans\_data, "transactions")

return(trans)

}

# Mine association rules

mine\_rules <- function(trans, min\_support = 0.01, min\_confidence = 0.5) {

rules <- apriori(trans,

parameter = list(support = min\_support,

confidence = min\_confidence,

minlen = 2))

rules\_sorted <- sort(rules, by = "lift", decreasing = TRUE)

return(rules\_sorted)

}

# Apply the functions

processed\_data <- preprocess\_data(bowling\_data)

trans <- create\_transactions(processed\_data)

rules <- mine\_rules(trans)

# View top rules

inspect(head(rules, 10))

# Plot rules with enhanced visibility

plot(rules,

measure = c("support", "confidence"),

shading = "lift",

main = "Association Rules: Support vs Confidence",

col = rainbow(20), # Vibrant colors

cex = 1.5, # Larger points

pch = 19) # Solid circles

# Network visualization of top rules

plot(rules[1:20],

method = "graph",

control = list(

type = "items",

nodeCol = "#0066CC", # Bright blue nodes

edgeCol = "#FF3300", # Bright orange edges

alpha = 0.8, # Less transparency

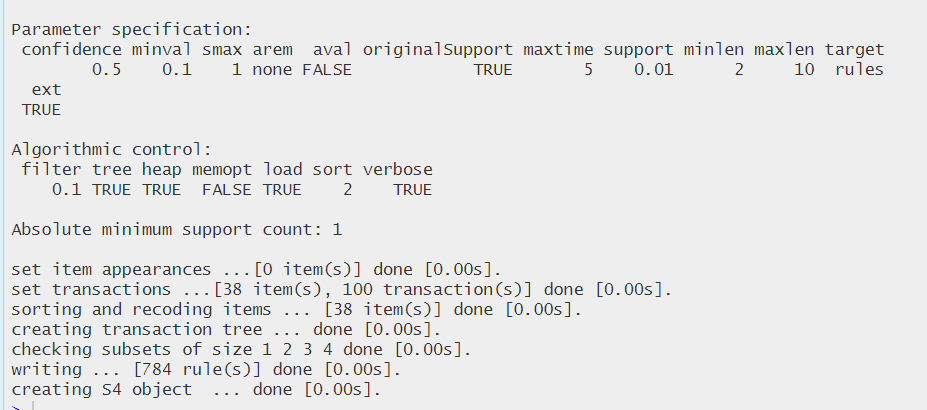
cex = 1.2 # Larger text

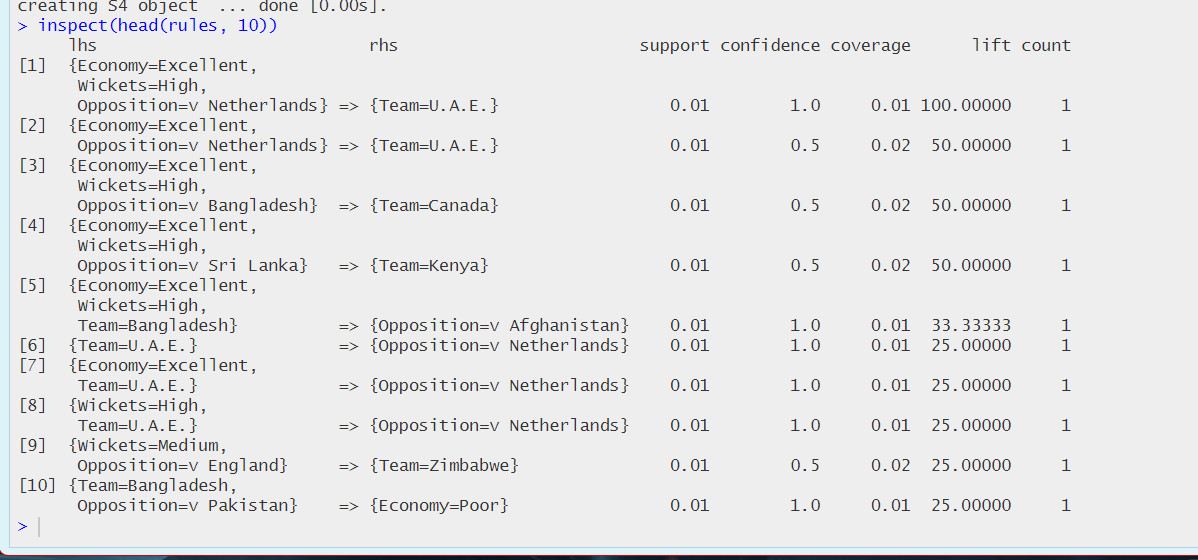
))

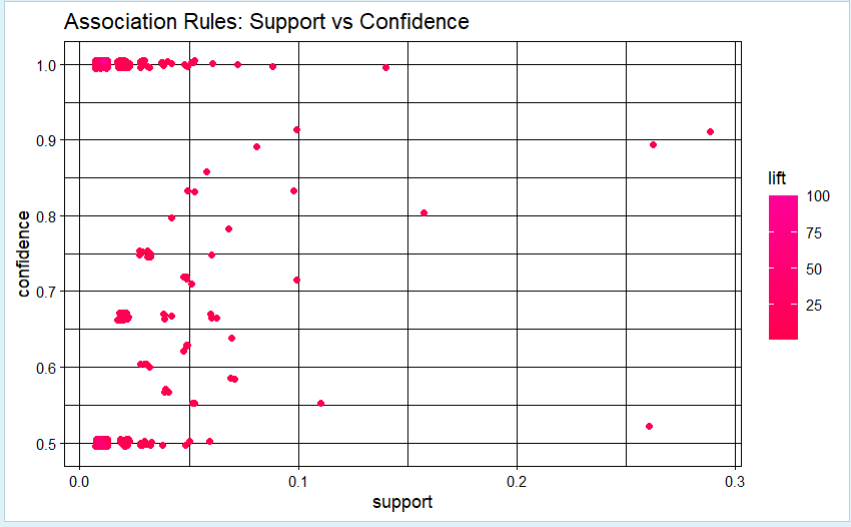
# Print summary of rules

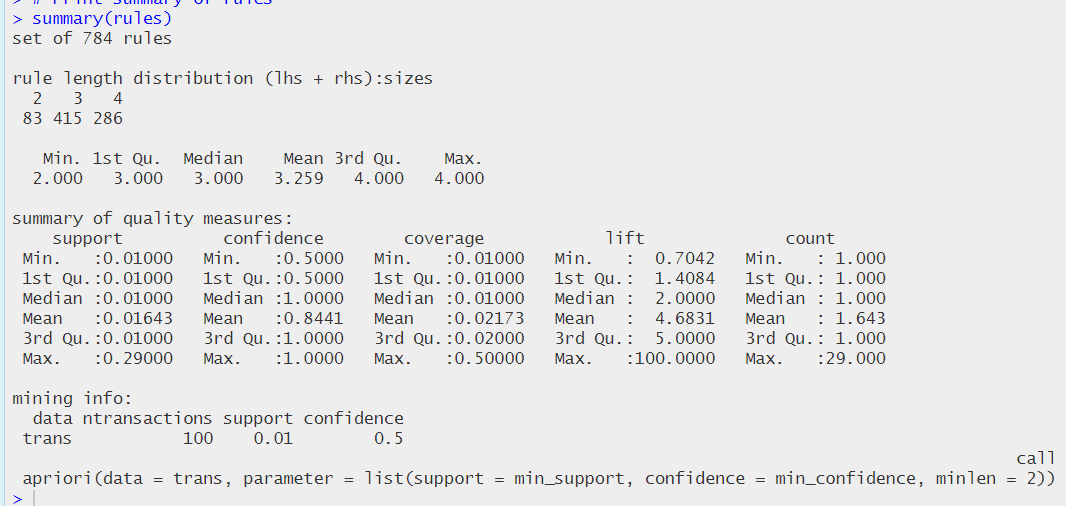
summary(rules)

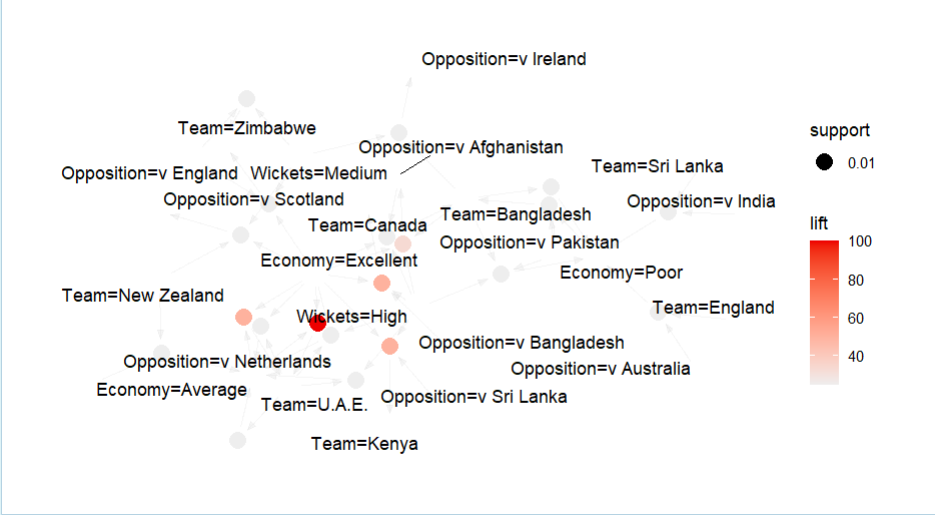
**Outputs:**











1. **Decision Tree:**

A supervised learning model used primarily for classification or regression.

In the context of this dataset, a decision tree could predict outcomes like the likelihood of a player making a certain number of catches, based on variables like match format, player role, or years of experience.

**Program:**

# Load required libraries

library(rpart)

library(rpart.plot)

# Read the data

cricket\_data <- read.csv("FieldingMostCatches.csv", stringsAsFactors = FALSE)

# Function to extract team from player string

get\_team <- function(player\_string) {

team <- gsub(".\*\\((.\*)\\).\*", "\\1", player\_string)

return(team)

}

# Data preprocessing

cricket\_data$Team <- sapply(cricket\_data$Player, get\_team)

cricket\_data$Catching\_Efficiency <- cricket\_data$Ct/cricket\_data$Inns

cricket\_data$Catch\_Category <- cut(cricket\_data$Catching\_Efficiency,

breaks = c(0, 0.3, 0.6, 0.9, Inf),

labels = c("Low", "Medium", "High", "Exceptional"))

# Create separate datasets for each team

ind\_data <- subset(cricket\_data, Team == "IND")

aus\_data <- subset(cricket\_data, Team == "AUS")

wi\_data <- subset(cricket\_data, Team == "WI")

nz\_data <- subset(cricket\_data, Team == "NZ")

sa\_data <- subset(cricket\_data, Team == "SA")

pak\_data <- subset(cricket\_data, Team == "PAK")

ban\_data <- subset(cricket\_data, Team == "BAN")

# Function to create and plot decision tree

create\_team\_tree <- function(data, team\_name) {

# Create tree model

tree\_model <- rpart(Catch\_Category ~ Mat + Inns + Ct + Max,

data = data,

method = "class",

control = rpart.control(minbucket = 2))

# Plot tree

rpart.plot(tree\_model,

box.palette = "Blues",

shadow.col = "gray",

main = paste("Decision Tree -", team\_name),

extra = 101,

fallen.leaves = TRUE,

branch = 0.3,

digits = 3)

return(tree\_model)

}

# Set up plotting layout for all trees

par(mfrow = c(3, 3), mar = c(1, 1, 2, 1))

# Create trees for each team

# India

print("India Team Analysis")

ind\_tree <- create\_team\_tree(ind\_data, "India")

ind\_stats <- ind\_data[order(ind\_data$Catching\_Efficiency, decreasing = TRUE), ]

print(head(ind\_stats[, c("Player", "Mat", "Ct", "Catching\_Efficiency")]))

# Australia

print("Australia Team Analysis")

aus\_tree <- create\_team\_tree(aus\_data, "Australia")

aus\_stats <- aus\_data[order(aus\_data$Catching\_Efficiency, decreasing = TRUE), ]

print(head(aus\_stats[, c("Player", "Mat", "Ct", "Catching\_Efficiency")]))

# West Indies

print("West Indies Team Analysis")

wi\_tree <- create\_team\_tree(wi\_data, "West Indies")

wi\_stats <- wi\_data[order(wi\_data$Catching\_Efficiency, decreasing = TRUE), ]

print(head(wi\_stats[, c("Player", "Mat", "Ct", "Catching\_Efficiency")]))

# New Zealand

print("New Zealand Team Analysis")

nz\_tree <- create\_team\_tree(nz\_data, "New Zealand")

nz\_stats <- nz\_data[order(nz\_data$Catching\_Efficiency, decreasing = TRUE), ]

print(head(nz\_stats[, c("Player", "Mat", "Ct", "Catching\_Efficiency")]))

# South Africa

print("South Africa Team Analysis")

sa\_tree <- create\_team\_tree(sa\_data, "South Africa")

sa\_stats <- sa\_data[order(sa\_data$Catching\_Efficiency, decreasing = TRUE), ]

print(head(sa\_stats[, c("Player", "Mat", "Ct", "Catching\_Efficiency")]))

# Pakistan

print("Pakistan Team Analysis")

pak\_tree <- create\_team\_tree(pak\_data, "Pakistan")

pak\_stats <- pak\_data[order(pak\_data$Catching\_Efficiency, decreasing = TRUE), ]

print(head(pak\_stats[, c("Player", "Mat", "Ct", "Catching\_Efficiency")]))

# Bangladesh

print("Bangladesh Team Analysis")

ban\_tree <- create\_team\_tree(ban\_data, "Bangladesh")

ban\_stats <- ban\_data[order(ban\_data$Catching\_Efficiency, decreasing = TRUE), ]

print(head(ban\_stats[, c("Player", "Mat", "Ct", "Catching\_Efficiency")]))

# Create summary statistics for all teams

team\_summary <- data.frame(

Team = c("IND", "AUS", "WI", "NZ", "SA", "PAK", "BAN"),

Avg\_Efficiency = c(

mean(ind\_data$Catching\_Efficiency),

mean(aus\_data$Catching\_Efficiency),

mean(wi\_data$Catching\_Efficiency),

mean(nz\_data$Catching\_Efficiency),

mean(sa\_data$Catching\_Efficiency),

mean(pak\_data$Catching\_Efficiency),

mean(ban\_data$Catching\_Efficiency)

),

Max\_Efficiency = c(

max(ind\_data$Catching\_Efficiency),

max(aus\_data$Catching\_Efficiency),

max(wi\_data$Catching\_Efficiency),

max(nz\_data$Catching\_Efficiency),

max(sa\_data$Catching\_Efficiency),

max(pak\_data$Catching\_Efficiency),

max(ban\_data$Catching\_Efficiency)

),

Total\_Players = c(

nrow(ind\_data),

nrow(aus\_data),

nrow(wi\_data),

nrow(nz\_data),

nrow(sa\_data),

nrow(pak\_data),

nrow(ban\_data)

)

)

# Print team summary

print("Team Summary Statistics:")

print(team\_summary)

# Create boxplot of catching efficiency by tea

# Print key findings

cat("\nKey Findings:\n")

cat("1. Team with highest average catching efficiency:",

team\_summary$Team[which.max(team\_summary$Avg\_Efficiency)], "\n")

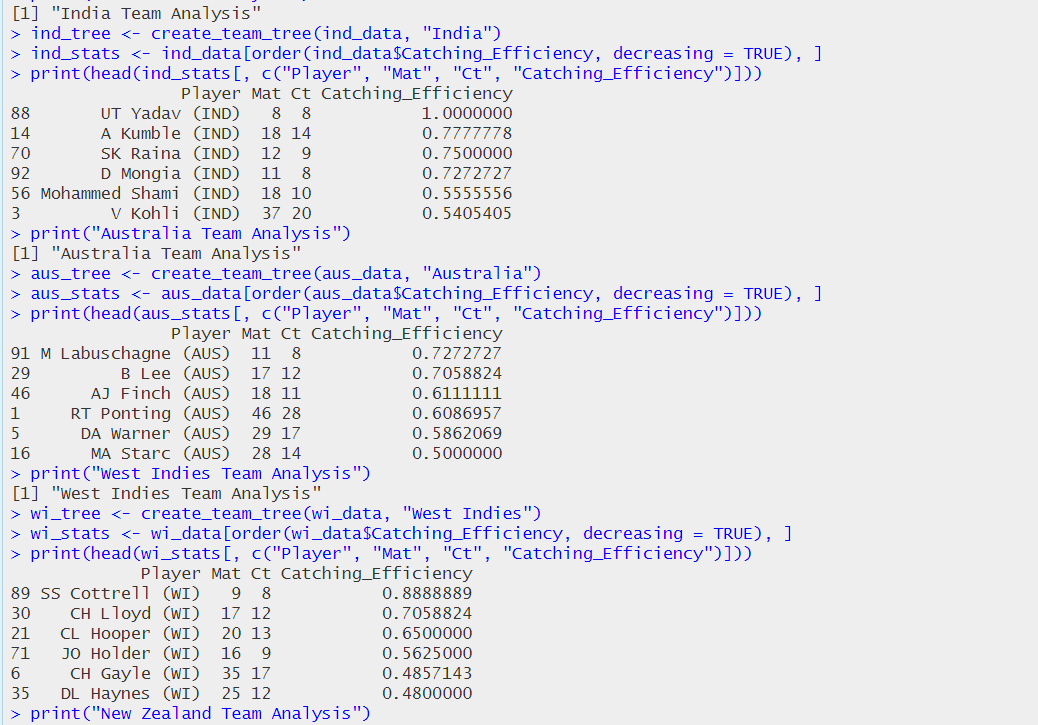
cat("2. Team with highest maximum catching efficiency:",

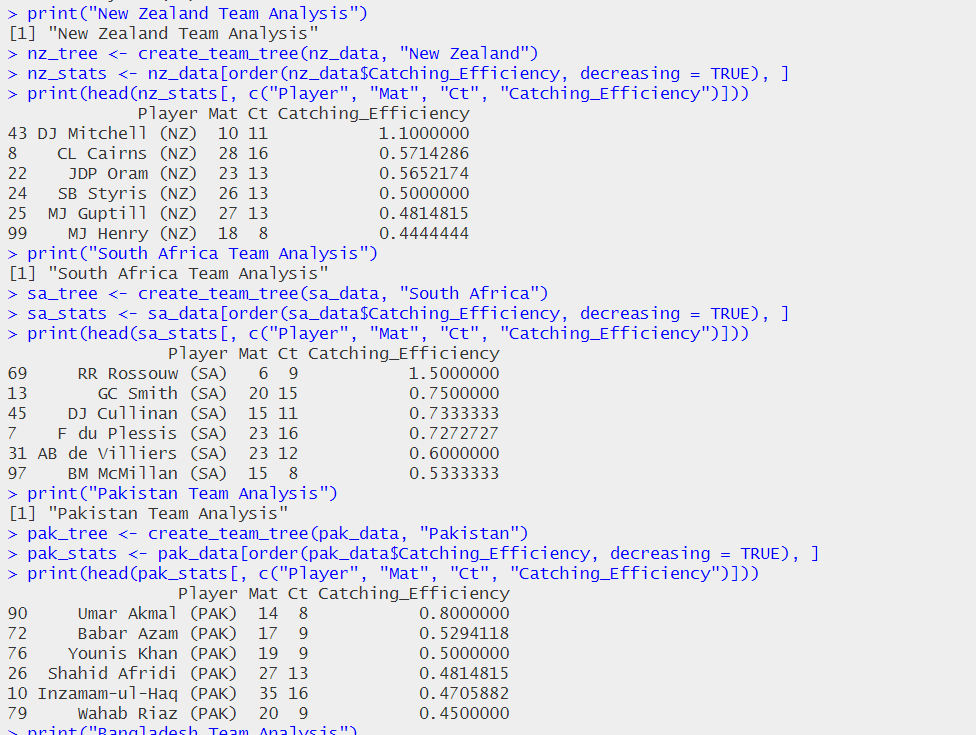
team\_summary$Team[which.max(team\_summary$Max\_Efficiency)], "\n")

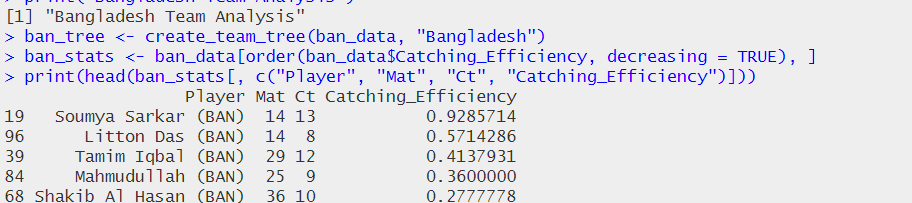
cat("3. Team with most players analyzed:",

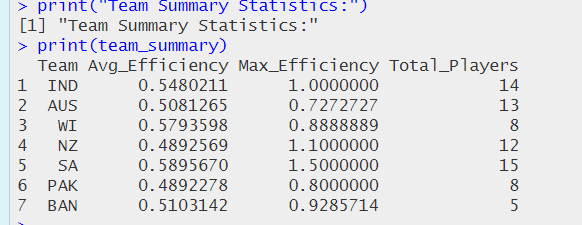
team\_summary$Team[which.max(team\_summary$Total\_Players)], "\n")

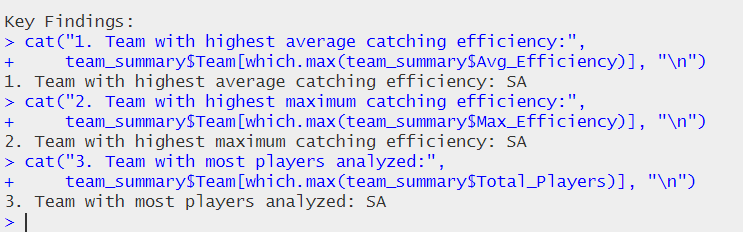
**Outputs:**

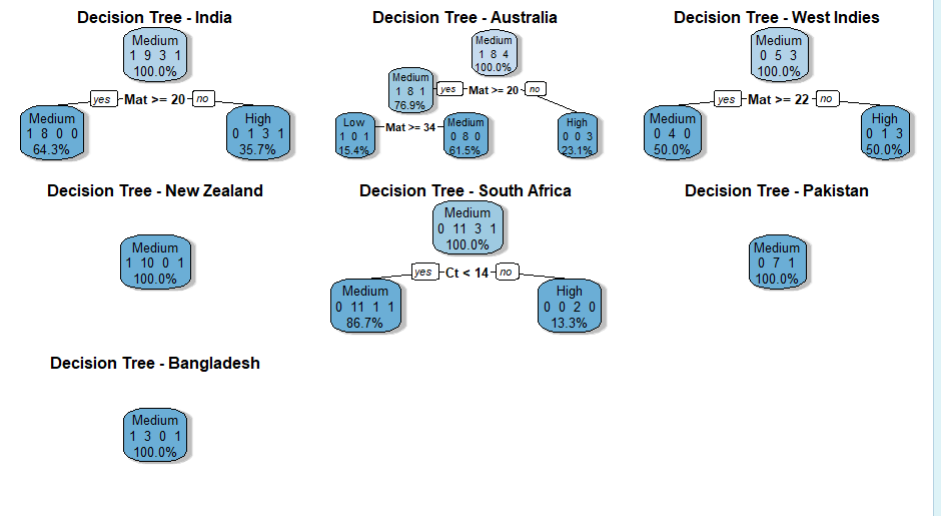
****

****

****

****

****



1. **K Means Clustering:**

An unsupervised learning algorithm that divides data into clusters based on feature similarities.

Using K-means on your dataset could help identify groups of players with similar fielding profiles, such as high-catch players, based on other dataset features.

**Program:**

# Load necessary libraries

library(ggplot2)

library(dplyr)

# Load the dataset

data <- read.csv("FieldingMostCatches.csv")

# Data Preprocessing

# Filter the data for players from 1996 to 2016 and select the top 10 based on catches

top\_players <- data %>%

filter(Span >= 1996 & Span <= 2016) %>% # Filter for years 1996 to 2016

arrange(desc(Ct)) %>% # Sort by number of catches

slice(1:10) # Select top 10 players

# Normalize numerical columns for K-means clustering

numerical\_data <- top\_players[, c("Mat", "Inns", "Ct", "Max", "Ct.Inn")] # Adjust this based on your actual columns

numerical\_data <- scale(numerical\_data) # Normalize the data

# Apply K-means clustering

set.seed(123) # Set seed for reproducibility

k <- 3 # Choose the number of clusters (you can adjust this)

kmeans\_result <- kmeans(numerical\_data, centers = k)

# Add cluster assignment to the top\_players data frame

top\_players$Cluster <- as.factor(kmeans\_result$cluster)

# Create a data frame for plotting with player names and team information

plot\_data <- as.data.frame(numerical\_data)

plot\_data$Player <- top\_players$Player

plot\_data$Cluster <- top\_players$Cluster

# Use meaningful names for the plot

colnames(plot\_data) <- c("Mat", "Inns", "Ct", "Max", "Ct.Inn", "Player", "Cluster")

# Plot the top 10 players with ggplot2, coloring by cluster

ggplot(plot\_data, aes(x = Ct, y = Mat, color = Cluster, label = Player)) +

geom\_point(size = 4) +

geom\_text(hjust = 0.5, vjust = -1.5, size = 3, check\_overlap = TRUE) +

labs(title = "K-Means Clustering of Top 10 Players with Most Catches (1996-2016)",

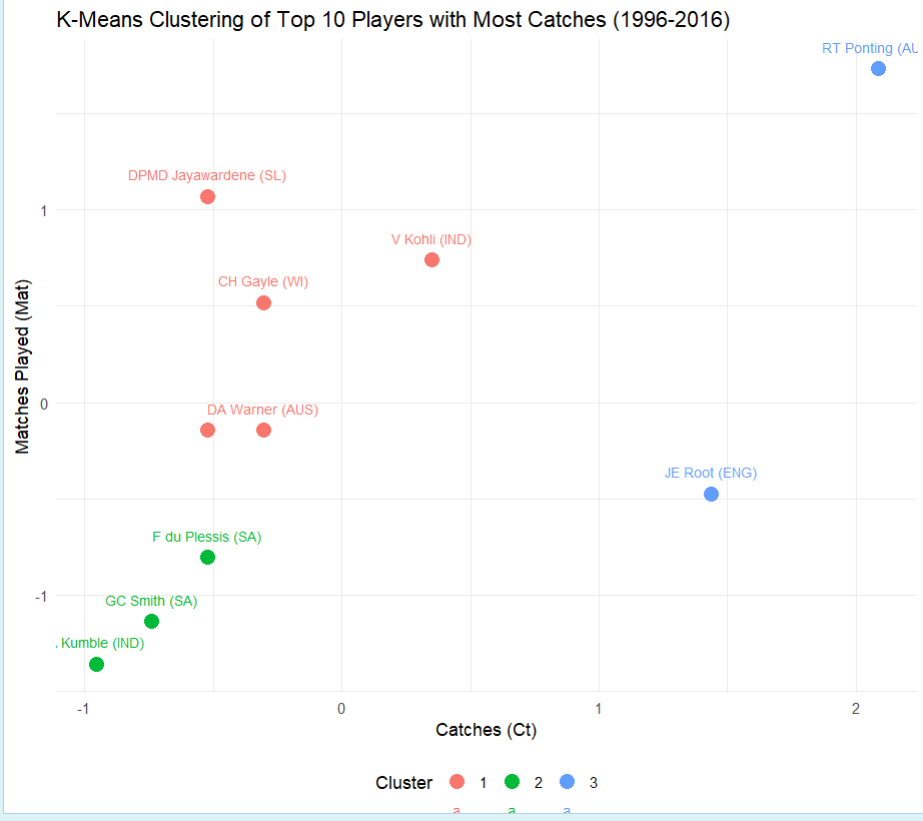
x = "Catches (Ct)",

y = "Matches Played (Mat)") +

theme\_minimal() +

theme(legend.position = "bottom")

**Output:**



1. **Statistical Analysis with exploratory graphs:**

Statistical analysis involves collecting and interpreting data to uncover patterns, trends, relationships, or insights. It allows researchers and analysts to make informed decisions, make predictions, and test hypotheses. There are two main types:

***Descriptive Statistics:*** Summarizes data features, such as mean, median, standard deviation, and distribution shapes.

***Inferential Statistics***: Uses data to draw conclusions and make predictions about a larger population, often with hypothesis testing or confidence intervals.

Visualization Techniques in Statistical Analysis

***1. Histogram***

Purpose: Visualizes the distribution of a single continuous variable.

Description: Histograms display the frequency of data within specified intervals or bins. Each bar represents the count or frequency of data points within that range, showing the data distribution’s shape, spread, and center.

Use Cases: Checking the normality of data, identifying skewness, or detecting outliers.

***2. Barplot***

Purpose: Compares categorical data.

Description: Each bar in a barplot represents a category and the height indicates the frequency, count, or proportion of that category. Barplots are effective for comparing groups or categories.

Use Cases: Summarizing counts of categorical data, like the number of players with different performance levels, or comparing total runs per player.

***3. Boxplot***

Purpose: Shows the distribution, variability, and potential outliers in continuous data.

Description: Boxplots depict the median, quartiles, and range of data. The box represents the interquartile range (IQR), with a line for the median, and "whiskers" indicating data spread. Outliers appear as individual points beyond the whiskers.

Use Cases: Comparing distributions across groups, spotting outliers, and understanding data spread.

***4. Heatmap***

Purpose: Displays relationships between variables or the intensity of values across a grid.

Description: A heatmap uses colors to represent data values within a matrix. Darker or more intense colors typically represent higher values, and lighter colors represent lower values.

Use Cases: Showing correlations between variables, such as runs vs. wickets or comparing batting averages.

***5. Scatter Plot***

Purpose: Illustrates the relationship between two continuous variables.

Description: Each point on a scatter plot represents one data observation, with coordinates based on values of the two variables. Scatter plots reveal patterns, correlations, or potential outliers in the data.

Use Cases: Visualizing relationships, such as runs vs. batting averages, or detecting clusters in data.

***6. Jitter Plot***

Purpose: Reduces overlapping points in scatter plots or other plots.

Description: A jitter plot adds random noise to data points, spreading them out slightly to reveal multiple observations that might overlap. This technique is especially useful when data points are categorical but slightly misaligned.

Use Cases: Visualizing points in categorical data or comparing distributions in overlapping groups.

**Program:**

# First, let's import the required libraries

library(ggplot2)

library(dplyr)

library(tidyr)

library(corrplot)

library(RColorBrewer)

# Assuming your data is stored in a dataframe called 'cricket\_data'

# Read the CSV file

cricket\_data <- read.csv("BattingRunsCareer.csv")

# 1. HISTOGRAMS

# Plot 1: Distribution of Batting Averages

ggplot(cricket\_data, aes(x=Ave)) +

geom\_histogram(fill="skyblue", color="black", bins=30) +

theme\_minimal() +

labs(title="Distribution of Batting Averages",

x="Average",

y="Frequency") +

theme(plot.title = element\_text(hjust = 0.5))

# Plot 2: Distribution of Strike Rates

ggplot(cricket\_data, aes(x=SR)) +

geom\_histogram(fill="lightgreen", color="black", bins=25) +

theme\_minimal() +

labs(title="Distribution of Strike Rates",

x="Strike Rate",

y="Frequency") +

theme(plot.title = element\_text(hjust = 0.5))

# 2. BAR PLOTS

# Plot 1: Top 10 Run Scorers

cricket\_data %>%

top\_n(10, Runs) %>%

ggplot(aes(x=reorder(Player, Runs), y=Runs)) +

geom\_bar(stat="identity", fill="steelblue") +

coord\_flip() +

theme\_minimal() +

labs(title="Top 10 Run Scorers",

x="Player",

y="Runs")

# Plot 2: Top Century Makers

cricket\_data %>%

top\_n(10, X100s) %>%

ggplot(aes(x=reorder(Player, X100s), y=X100s)) +

geom\_bar(stat="identity", fill="orange") +

coord\_flip() +

theme\_minimal() +

labs(title="Top 10 Century Makers",

x="Player",

y="Number of Centuries")

# 3. BOX PLOTS

# Plot 1: Runs Distribution

ggplot(cricket\_data, aes(y=Runs)) +

geom\_boxplot(fill="lightblue") +

theme\_minimal() +

labs(title="Distribution of Runs",

y="Runs")

# Plot 2: Strike Rate Distribution by Batting Position

ggplot(cricket\_data, aes(y=SR)) +

geom\_boxplot(fill="lightgreen") +

theme\_minimal() +

labs(title="Distribution of Strike Rates",

y="Strike Rate")

# 4. HEATMAPS

# Plot 1: Correlation Heatmap

numeric\_cols <- cricket\_data %>%

select\_if(is.numeric)

correlation <- cor(numeric\_cols, use="complete.obs")

corrplot(correlation, method="color", type="upper",

order="hclust", addCoef.col="black",

tl.col="black", tl.srt=45)

# First, let's check the structure of your data

str(cricket\_data)

# Now let's modify the code to ensure numeric conversion

top\_players <- cricket\_data %>%

top\_n(15, Runs) %>%

select(Player, Runs, Ave, SR, X100s, X50s) %>%

mutate(

Runs = as.numeric(Runs),

Ave = as.numeric(Ave),

SR = as.numeric(SR),

X100s = as.numeric(X100s),

X50s = as.numeric(X50s)

)

# Verify the data is numeric

print(sapply(top\_players[,-1], class)) # Should all show "numeric"

# Create matrix and continue with visualization

matrix\_data <- as.matrix(scale(top\_players[,-1])) # Remove Player column for scaling

rownames(matrix\_data) <- top\_players$Player

# Create custom color palette

my\_colors <- colorRampPalette(c("#4575B4", "#FFFFBF", "#D73027"))(100)

# Create the heatmap

heatmap(matrix\_data,

Colv = NA,

scale = "none",

col = my\_colors,

margins = c(10, 10),

main = "\n Cricket Performance Metrics Heatmap\n(Top 15 Run Scorers)",

xlab = "Metrics",

ylab = "Players",

cexRow = 0.8,

cexCol = 0.8)

# Add legend

legend\_colors <- as.matrix(seq(-2, 2, length=5))

legend\_labels <- c("Very Low", "Low", "Average", "High", "Very High")

legend("bottomright",

legend = legend\_labels,

fill = my\_colors[c(1, 25, 50, 75, 100)],

title = "Performance Scale",

cex = 0.7)

# 5. SCATTER PLOTS

# Plot 1: Runs vs Average

ggplot(cricket\_data, aes(x=Runs, y=Ave)) +

geom\_point(alpha=0.6, color="blue") +

geom\_smooth(method="lm", se=FALSE, color="red") +

theme\_minimal() +

labs(title="Runs vs Batting Average",

x="Runs",

y="Average")

# Plot 2: Strike Rate vs Average with Century Count as size

ggplot(cricket\_data, aes(x=SR, y=Ave, size=X100s)) +

geom\_point(alpha=0.6, color="purple") +

theme\_minimal() +

labs(title="Strike Rate vs Average (Size: Number of Centuries)",

x="Strike Rate",

y="Average",

size="Centuries")

# Additional: Jitter Plot

# Plot 1: Runs distribution with jitter

ggplot(cricket\_data, aes(x=factor(1), y=Runs)) +

geom\_jitter(width=0.2, alpha=0.6, color="blue") +

theme\_minimal() +

labs(title="Runs Distribution (Jittered)",

x="",

y="Runs")

# Plot 2: Strike Rate distribution with jitter

ggplot(cricket\_data, aes(x=factor(1), y=SR)) +

geom\_jitter(width=0.2, alpha=0.6, color="green") +

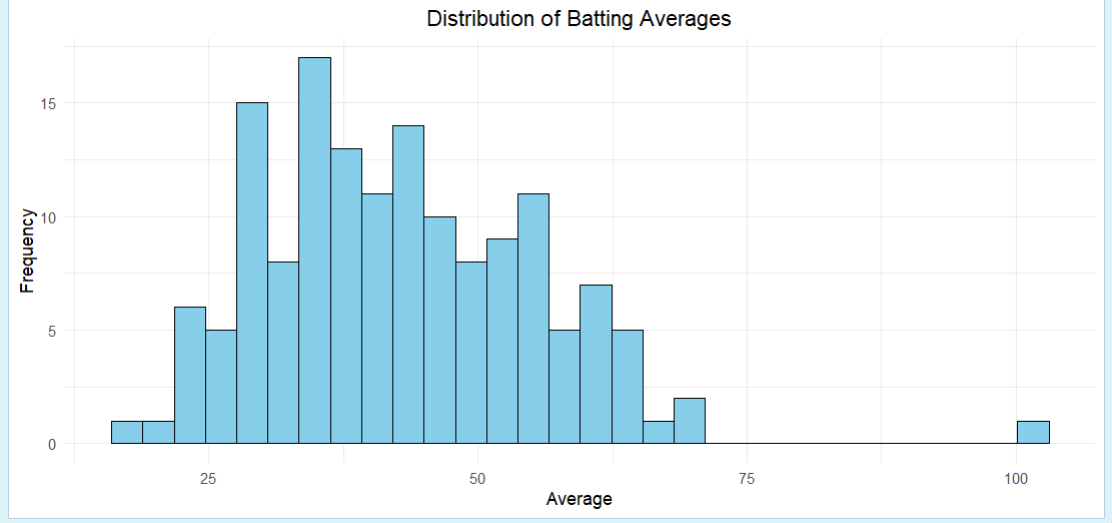
theme\_minimal() +

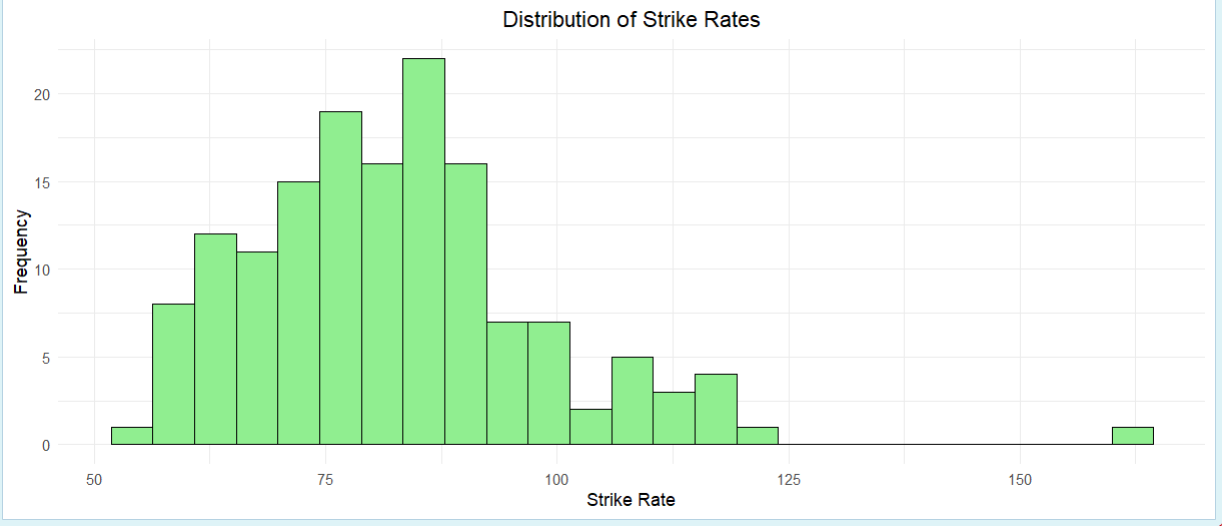
labs(title="Strike Rate Distribution (Jittered)",

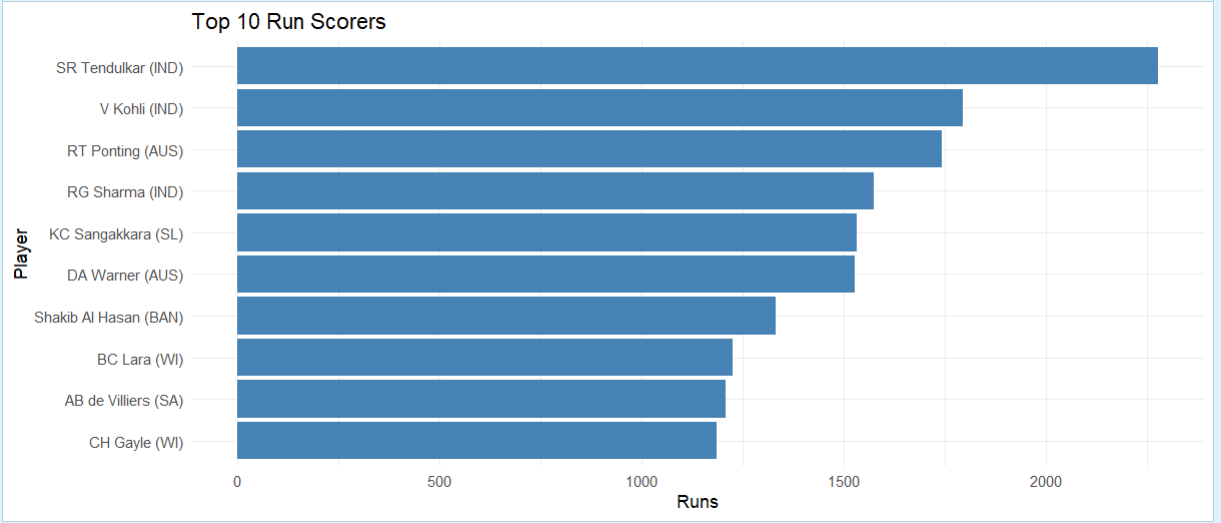
x="",

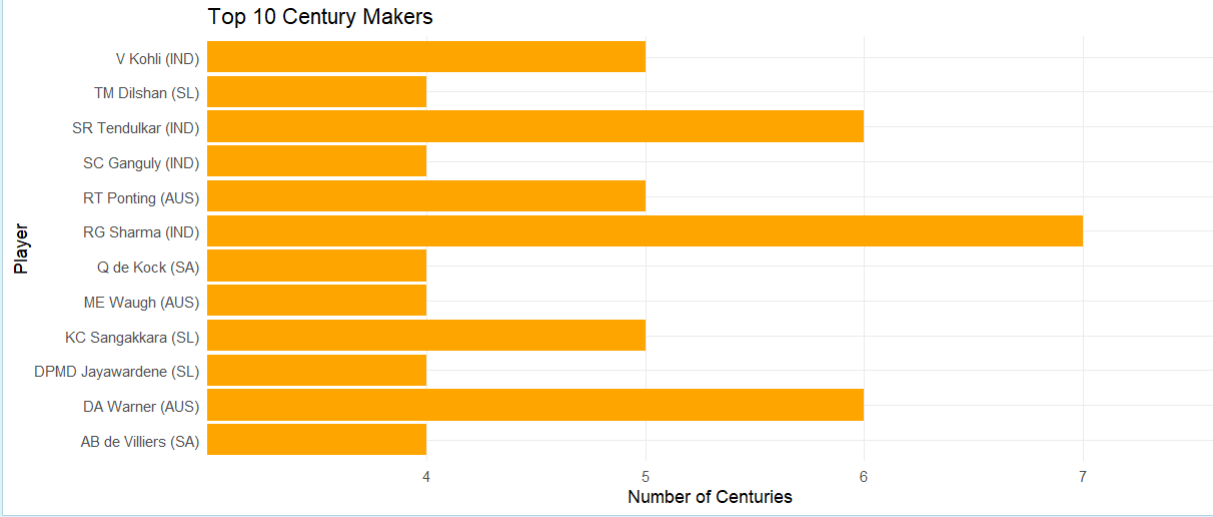
y="Strike Rate")

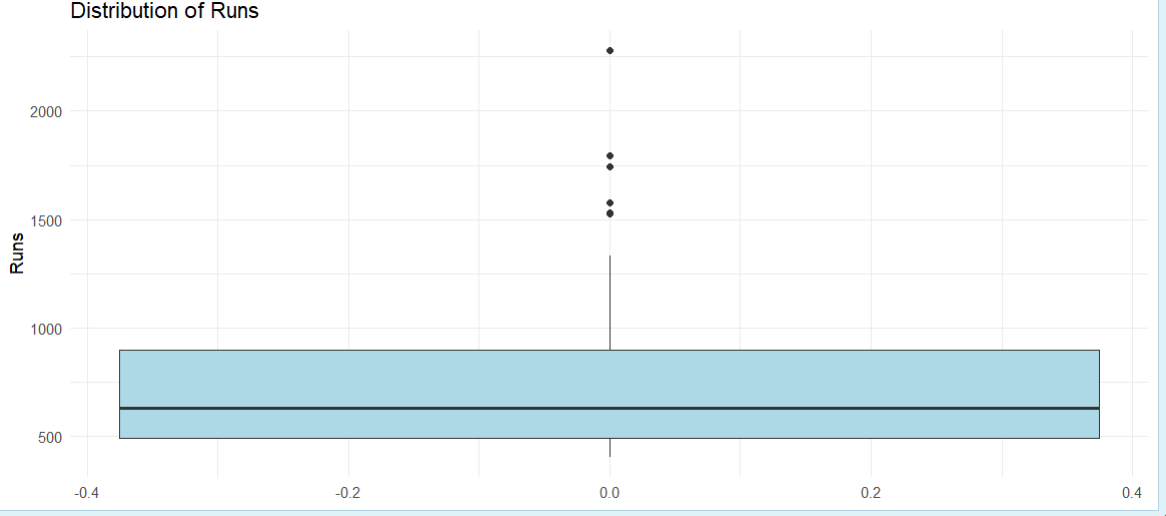
**Outputs:**

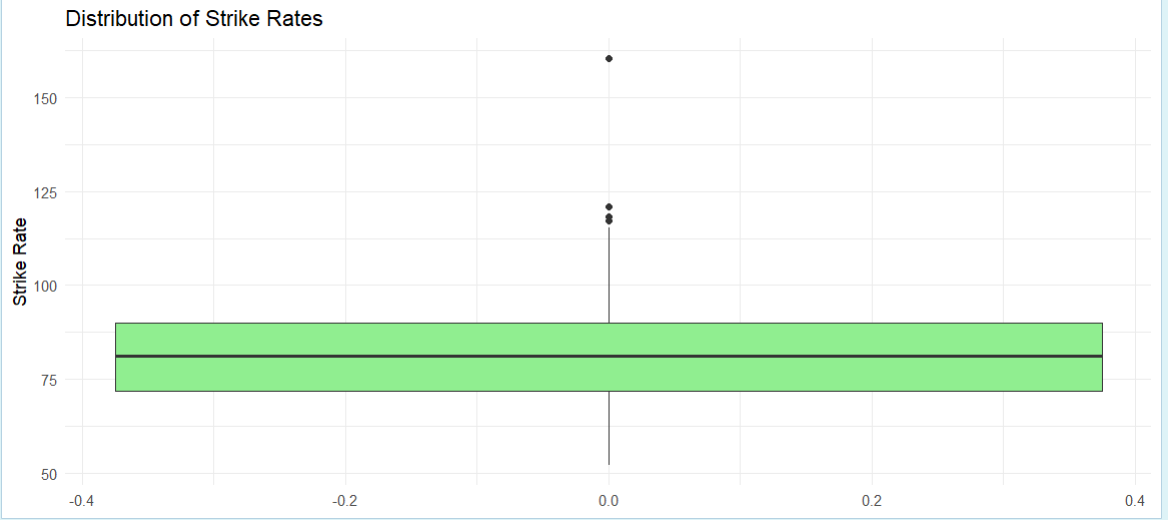


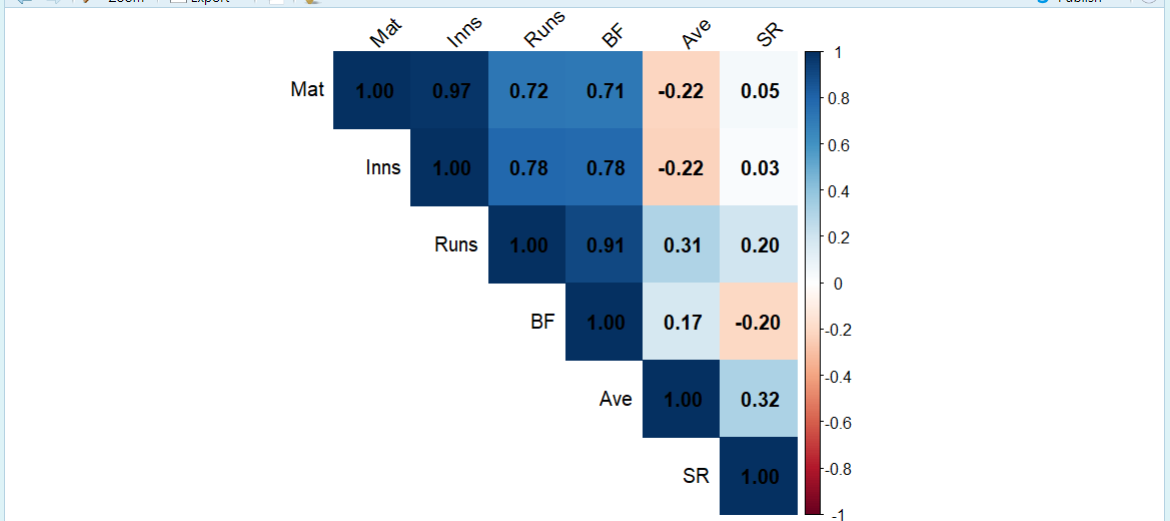


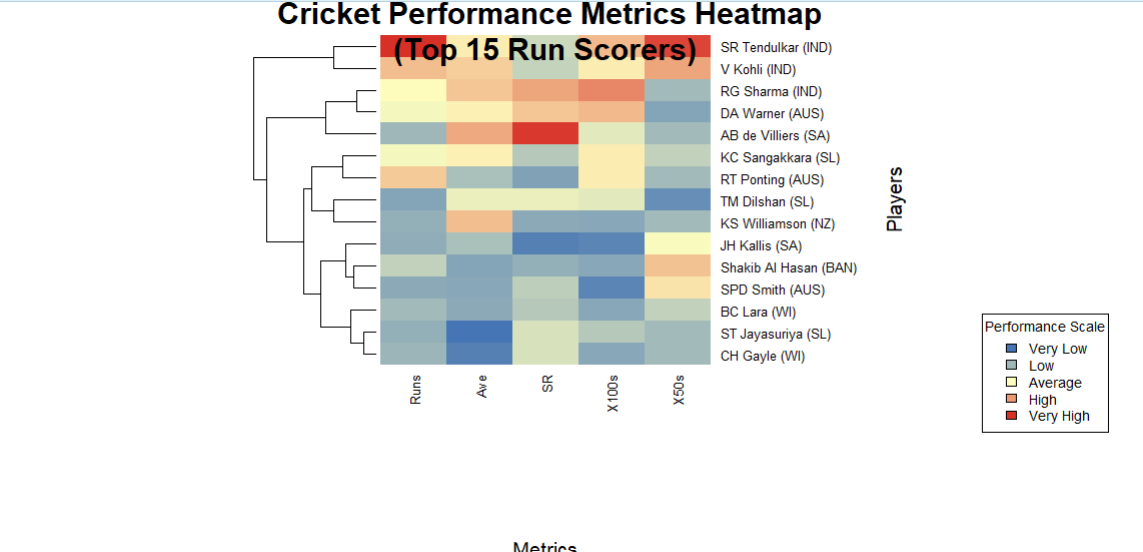


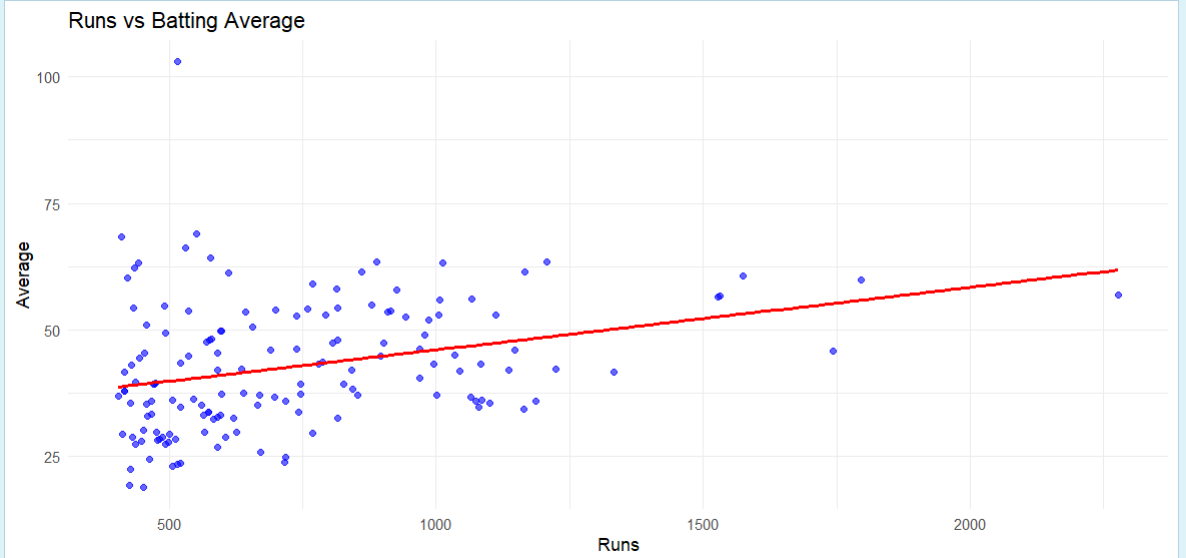


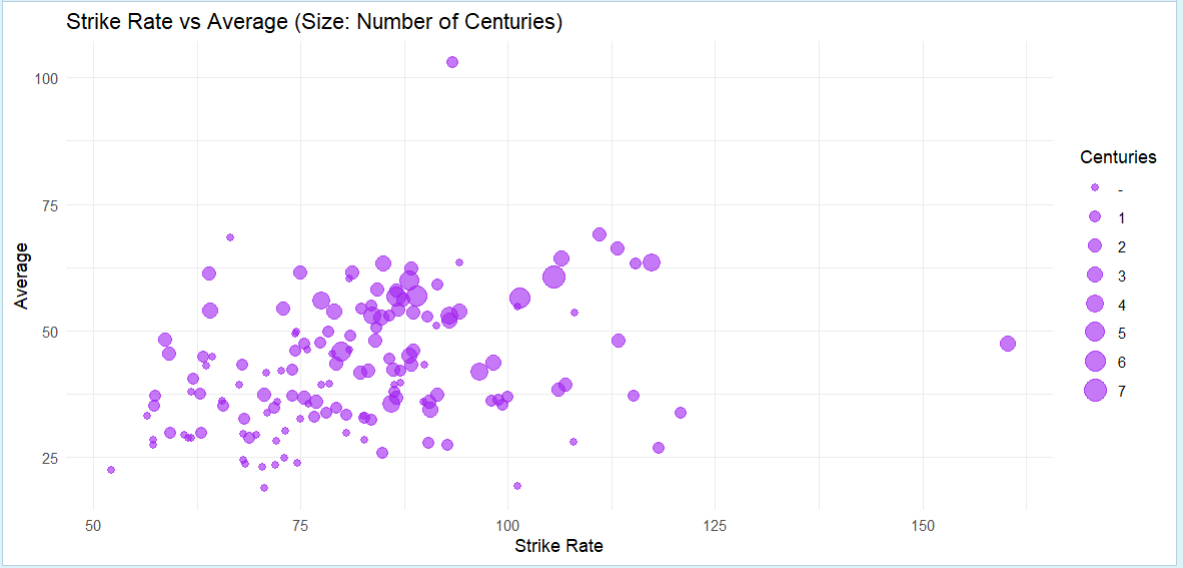


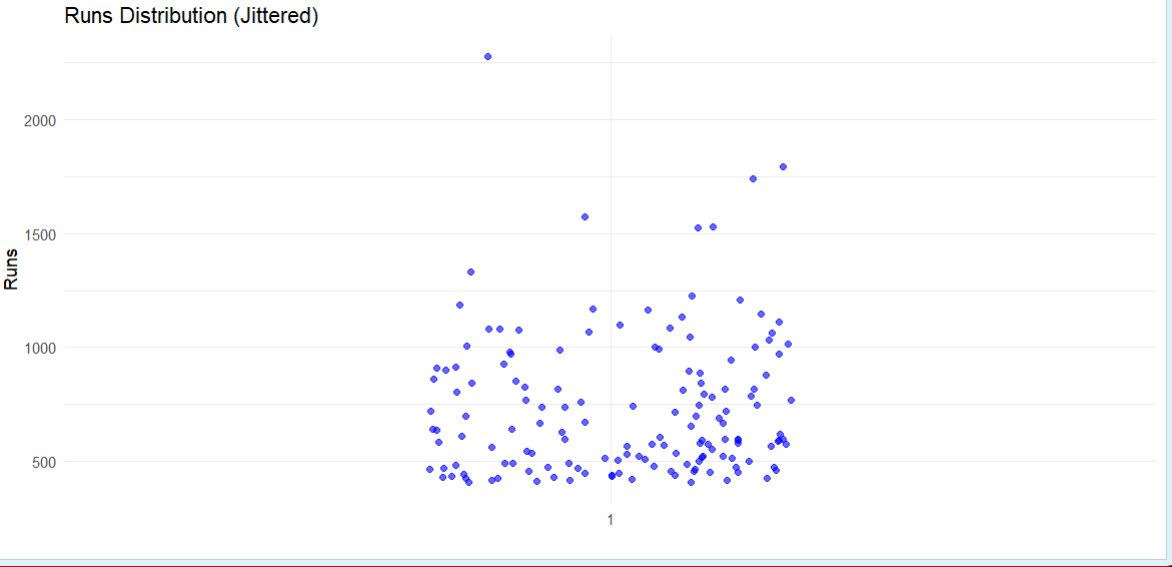


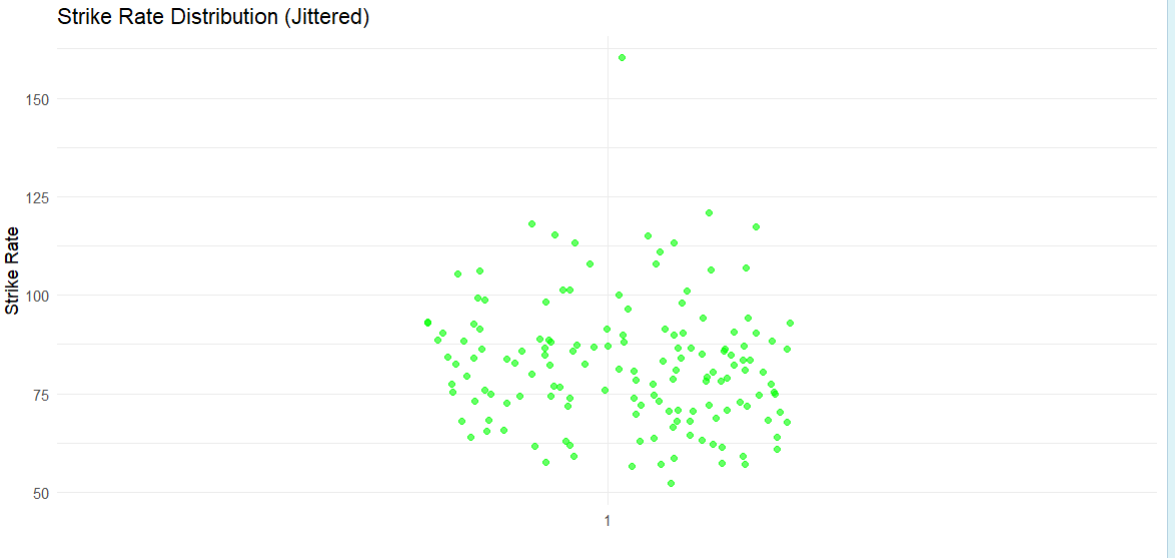












**8.Regression Models (Linear regression)**

**Linear Regression**

**Definition**: Linear regression is a statistical method used to model the relationship between a dependent variable and one or more independent variables by fitting a linear equation to the observed data. In its simplest form, linear regression involves two variables: one independent variable (predictor) and one dependent variable (outcome), resulting in a straight-line relationship represented by the equation:

Y=β0+β1X+ϵY = \beta\_0 + \beta\_1 X + \epsilonY=β0​+β1​X+ϵ

where:

* YYY is the dependent variable,
* β0\beta\_0β0​ is the y-intercept,
* β1\beta\_1β1​ is the slope of the line (representing the effect of the independent variable on the dependent variable),
* XXX is the independent variable, and
* ϵ\epsilonϵ is the error term.

**Program:**

# Import required libraries

library(tidyverse)

library(caret)

library(stats)

# Read the data

cricket\_data <- read.csv("BattingHundredsCareer.csv")

# Data preprocessing

# Remove '+' from X4s and X6s columns and convert to numeric

cricket\_data$X4s <- as.numeric(gsub("\\+", "", cricket\_data$X4s))

cricket\_data$X6s <- as.numeric(gsub("\\+", "", cricket\_data$X6s))

# Convert Span to number of years

cricket\_data$Years <- sapply(strsplit(cricket\_data$Span, "-"), function(x) {

as.numeric(x[2]) - as.numeric(x[1]) + 1

})

# Create multiple linear regression models

# Model 1: Predicting Runs based on Innings, X4s, X6s

model1 <- lm(Runs ~ Inns + X4s + X6s, data = cricket\_data)

# Model 2: Predicting Average based on Strike Rate, X4s, X6s

model2 <- lm(Ave ~ SR + X4s + X6s, data = cricket\_data)

# Print model summaries

summary(model1)

summary(model2)

# Create visualizations

# Scatter plot for actual vs predicted runs

predicted\_runs <- predict(model1)

plot(cricket\_data$Runs, predicted\_runs,

main = "Actual vs Predicted Runs",

xlab = "Actual Runs",

ylab = "Predicted Runs")

abline(0, 1, col = "red")

# Diagnostic plots for model1

par(mfrow = c(2,2))

plot(model1)

# Feature importance

importance1 <- abs(coef(model1))[-1] # Remove intercept

barplot(importance1,

main = "Feature Importance for Runs Prediction",

names.arg = names(importance1),

las = 2)

# Correlation matrix for numeric variables

numeric\_vars <- cricket\_data %>%

select\_if(is.numeric) %>%

select(-c(NO)) # Remove any irrelevant columns

correlation\_matrix <- cor(numeric\_vars, use = "complete.obs")

heatmap(correlation\_matrix,

main = "Correlation Matrix",

margins = c(10, 10))

# Cross-validation

set.seed(123)

train\_control <- trainControl(method = "cv", number = 5)

cv\_model <- train(Runs ~ Inns + X4s + X6s,

data = cricket\_data,

method = "lm",

trControl = train\_control)

print(cv\_model)

# Predictions with confidence intervals

new\_data <- data.frame(

Inns = c(20, 25, 30),

X4s = c(100, 120, 140),

X6s = c(20, 25, 30)

)

predictions <- predict(model1, newdata = new\_data, interval = "confidence")

print(predictions)

# Calculate R-squared and RMSE

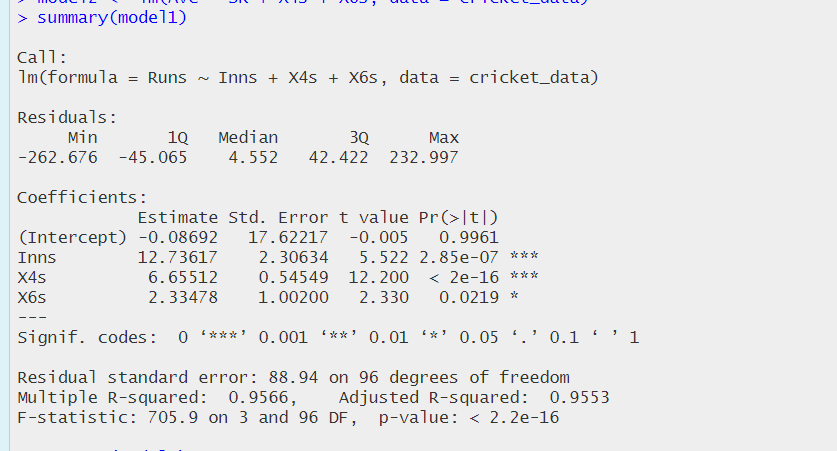
r2 <- summary(model1)$r.squared

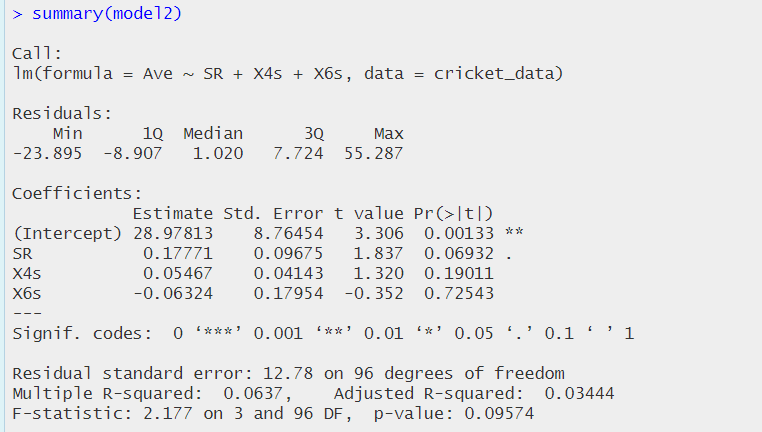
rmse <- sqrt(mean((cricket\_data$Runs - predicted\_runs)^2))

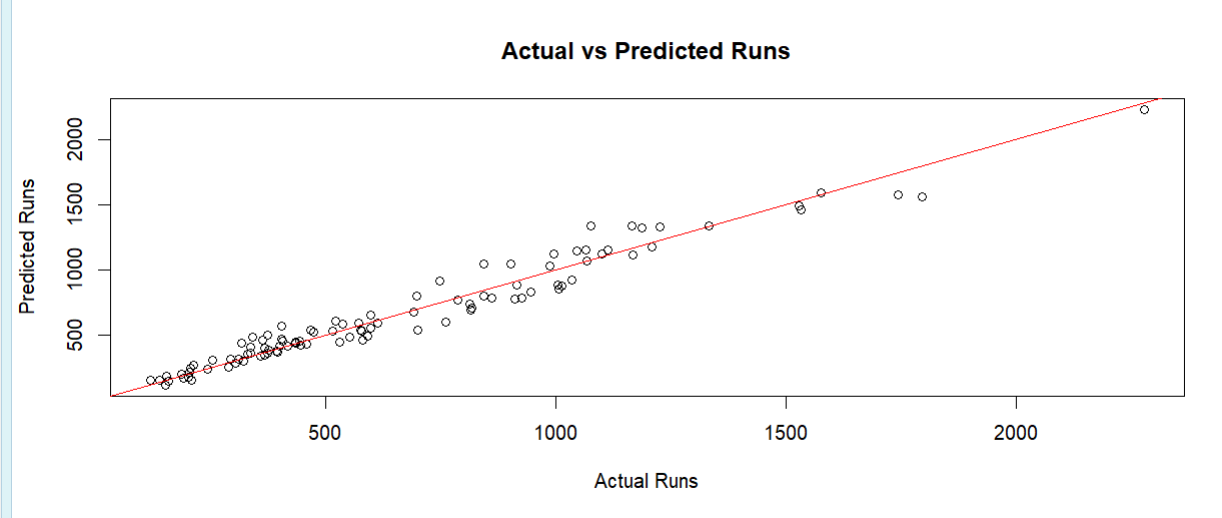
cat("R-squared:", r2, "\n")

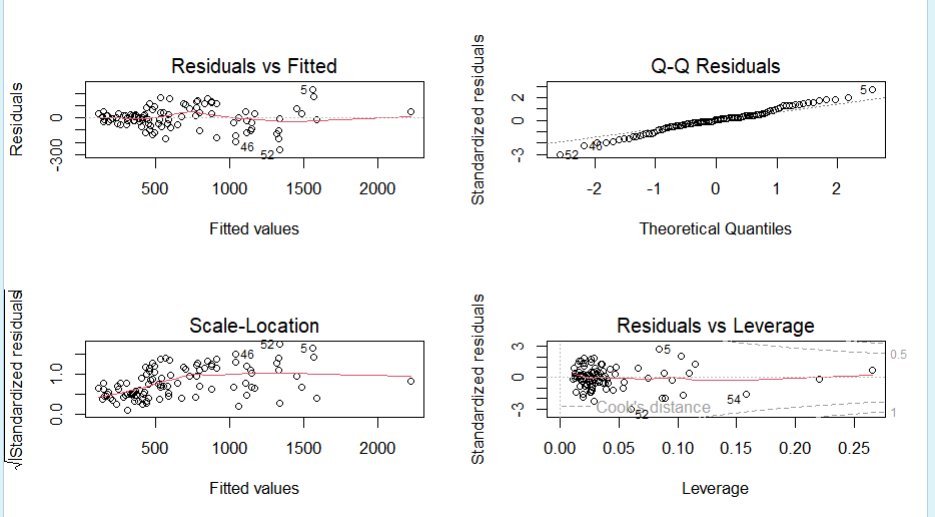
cat("RMSE:", rmse, "\n")

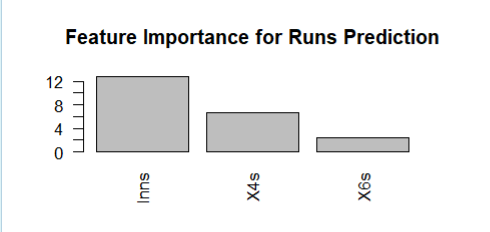
**Outputs:**

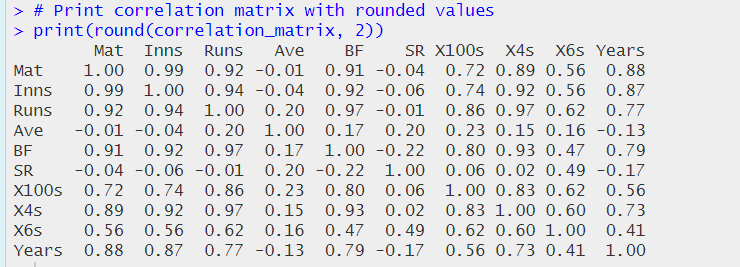


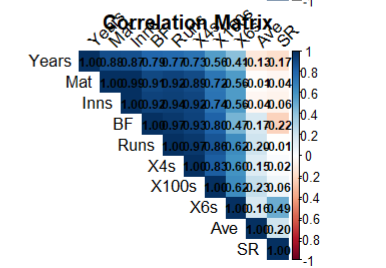


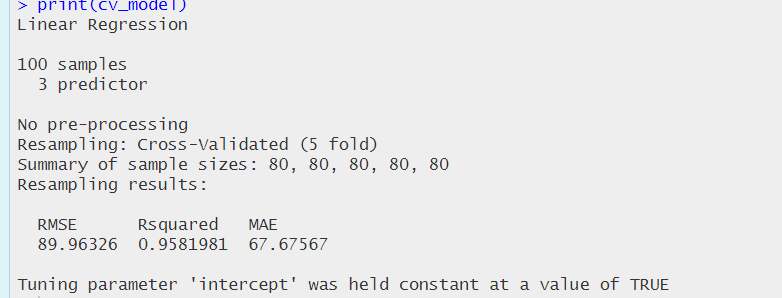


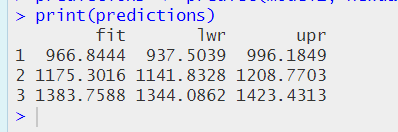


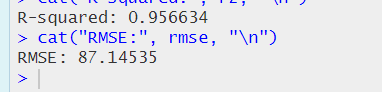












**Conclusion**

The analysis of the ICC World Cup Records dataset has yielded significant insights into player performances and team dynamics across multiple tournaments. By employing various statistical methods, including hypothesis testing, association rule mining, decision tree analysis, K-means clustering, and regression modeling, we have been able to uncover underlying patterns and trends in cricketing performance.

Key findings include:

* **Performance Trends**: Statistical analyses highlighted standout players and teams, revealing performance trends influenced by factors such as match conditions and historical contexts. For instance, the data indicated which batting styles or bowling strategies were most effective in different tournament environments.
* **Player Contributions**: Association rule mining demonstrated the interdependencies among different performance metrics, showcasing how specific skills, such as batting average and strike rate, correlate with match success. This insight can guide teams in player selection and training focus.
* **Predictive Modeling**: The implementation of regression models provided a predictive framework to estimate future match outcomes based on historical performance data. This application is invaluable for team strategists and analysts in preparing for upcoming tournaments.
* **Clustering**: K-means clustering helped categorize players and matches, identifying groups with similar performance characteristics. This classification can aid coaches and selectors in tailoring training and development programs to address specific weaknesses or leverage strengths.

**Applications**

The findings from this analysis can be applied in various domains within the sport of cricket, including:

* **Team Strategy Development**: Coaches and analysts can utilize insights to formulate game strategies, enhance player training regimes, and optimize team compositions based on data-driven evidence.
* **Scouting and Player Development**: The analysis aids in identifying potential talent by evaluating performance metrics, guiding recruitment processes and youth development programs.
* **Fan Engagement**: Insights from the analysis can enhance fan engagement through data-driven narratives and statistics, enriching the viewing experience and fostering a deeper connection with the sport.
* **Broadcasting and Media**: Media outlets can leverage the findings for analytical segments, providing audiences with rich statistical content that enhances match coverage and storytelling.

In conclusion, this comprehensive analysis of ICC World Cup records not only advances our understanding of the game but also offers practical applications that can significantly influence the future of cricket at both professional and grassroots levels.