CSV

Write a R program to read the content of the CSV file and show as data frame.

# Read CSV file into a data frame

df <- read.csv("C:\\Users\\repan\\OneDrive\\Desktop\\R\\sample\_data.csv")

# Display the data frame

print(df)

Write a R Program to [export CSV File without Row Names](https://www.geeksforgeeks.org/export-csv-file-without-row-names-in-r/)

# Sample Dataset

data <- data.frame(

Name = c("Alice", "Bob", "Charlie"),

Age = c(25, 30, 22),

Salary = c(50000, 60000, 45000)

)

# Display the sample dataset

print("Sample Dataset:")

print(data)

# Export the dataset to a CSV file without row names

write.csv(data, "output\_without\_row\_names.csv", row.names = FALSE)

# Display a message indicating the export is successful

cat("\nCSV file exported successfully without row names.\n")

[Write a R Program to write to CSV without index ?](https://www.geeksforgeeks.org/how-to-write-to-csv-in-r-without-index/)

# Sample Dataset

data <- data.frame(

Name = c("Alice", "Bob", "Charlie"),

Age = c(25, 30, 22),

Salary = c(50000, 60000, 45000)

)

# Display the sample dataset

print("Sample Dataset:")

print(data)

# Export the dataset to a CSV file without row names and indices

write.csv(data, "output\_without\_index.csv", row.names = FALSE)

# Display a message indicating the export is successful

cat("\nCSV file exported successfully without row names and indices.\n")

Write a R Program to [Append row to CSV](https://www.geeksforgeeks.org/append-row-to-csv-using-r/) file.

# CSV file path

csv\_file <- "C:\\Users\\repan\\OneDrive\\Documents\\existing\_data.csv"

# Append a new row to the CSV file

new\_row <- data.frame(Name = "David", Age = 28, Salary = 55000)

# Append the new row to the CSV file

write.csv(new\_row, csv\_file, row.names = FALSE, append = TRUE)

# Display a message indicating the append is successful

cat("Row appended to CSV file successfully.\n")

[Write a R Program to calculate mean, median and variance of a CSV file ?](https://www.geeksforgeeks.org/how-to-calculate-mean-of-a-csv-file-in-r/)

data <- read.csv(text = "Name, Age, Salary

Alice, 25, 50000

Bob, 30, 60000

Charlie, 22, 45000

David, 35, 70000

Eva, 28, 55000

Frank, 40, 80000

Grace, 32, 65000

Hank, 27, 48000

Ivy, 38, 72000

Jack, 29, 59000")

cat("Loaded Dataset:\n")

print(data)

# Specify the column for calculations

column\_name <- "Salary"

# Calculate mean, median, and variance

mean\_value <- mean(data[[column\_name]], na.rm = TRUE)

median\_value <- median(data[[column\_name]], na.rm = TRUE)

variance\_value <- var(data[[column\_name]], na.rm = TRUE)

# Display the results

cat("\nResults:\n")

cat("Mean:", mean\_value, "\n")

cat("Median:", median\_value, "\n")

cat("Variance:", variance\_value, "\n")

Write a R program to display the structure of the CSV file.

data <- read.csv(text = "Name, Age, Salary

Alice, 25, 50000

Bob, 30, 60000

Charlie, 22, 45000

David, 35, 70000

Eva, 28, 55000

Frank, 40, 80000

Grace, 32, 65000

Hank, 27, 48000

Ivy, 38, 72000

Jack, 29, 59000")

# Display the structure of the CSV file

cat("Structure of the CSV file:\n")

str(data)

Write a R program to convert the text file into CSV file.

# Read text file into a data frame

data <- read.table("C:\\Users\\repan\\OneDrive\\Documents\\seven\_data.txt", header = TRUE, sep = ",")

# Display the loaded dataset

cat("Loaded Dataset:\n")

print(data)

# Write the data frame to a CSV file

write.csv(data, "seven\_output.csv", row.names = FALSE)

# Display a message indicating the conversion is successful

cat("\nText file converted to CSV file successfully.\n")

Excel

[Write a R Program to import an Excel File](https://www.geeksforgeeks.org/how-to-import-an-excel-file-into-r/)

install.packages("readxl")

library(readxl)

excel\_file <- "C:\\Users\\repan\\OneDrive\\Desktop\\R\\Iris.xls"

# Read the Excel file into a data frame

data <- read\_excel(excel\_file)

# Display the first few rows of the imported data

head(data)

[Write a R Program to export a DataFrame to Excel File](https://www.geeksforgeeks.org/how-to-export-a-dataframe-to-excel-file-in-r/)

# Install and load the writexl package

install.packages("writexl")

library(writexl)

# Create a sample DataFrame

my\_data <- data.frame(

Name = c("John", "Alice", "Bob", "Eva"),

Age = c(25, 30, 28, 35),

Score = c(85, 92, 78, 96)

)

# Specify the file path for the Excel file

excel\_output\_file <- "C:\\Users\\repan\\OneDrive\\Desktop\\R\\nine\_op\_file.xls"

# Write the DataFrame to an Excel file

write\_xlsx(my\_data, excel\_output\_file)

# Print a message to confirm the export

cat("DataFrame exported to Excel file:", excel\_output\_file, "\n")

Write a R Program to [convert an Excel column into a list of vectors.](https://www.geeksforgeeks.org/convert-an-excel-column-into-a-list-of-vectors-in-r/)

# Install and load the readxl package

install.packages("readxl")

library(readxl)

# Specify the file path to your Excel file

excel\_file <- "C:\\Users\\repan\\OneDrive\\Documents\\ten\_data.xlsx"

# Read the Excel file into a data frame

data <- read\_excel(excel\_file)

# Extract a specific column into a list

column\_name <- "Name" # Replace with the actual column name

column\_list <- as.list(data[[column\_name]])

# Display the list

print(column\_list)

[Write a R Program to convert Excel column to vector in R ?](https://www.geeksforgeeks.org/how-to-convert-excel-column-to-vector-in-r/)

# Install and load the readxl package

install.packages("readxl")

library(readxl)

# Specify the file path to your Excel file

excel\_file <- "C:\\Users\\repan\\OneDrive\\Documents\\ten\_data.xlsx"

# Read the Excel file into a data frame

data <- read\_excel(excel\_file)

# Extract a specific column into a vector

column\_name <- "Name" # Replace with the actual column name

column\_vector <- data[[column\_name]]

# Display the vector

print(column\_vector)

[Write a R Program to convert excel content into DataFrame.](https://www.geeksforgeeks.org/how-to-convert-excel-content-into-dataframe-in-r/)

library(readxl)

# Specify the file path to your Excel file

excel\_file <- "C:\\Users\\repan\\OneDrive\\Documents\\ten\_data.xlsx"

# Read the Excel file into a data frame

data <- read\_excel(excel\_file)

# Display the data frame

print(data)

Write a R Program to [Delete rows with empty cells from Excel.](https://www.geeksforgeeks.org/delete-rows-with-empty-cells-from-excel-using-r/)

# Install and load the tidyxl and readxl packages

install.packages(c("tidyxl"))

library(tidyxl)

library(readxl)

# Specify the file path to your Excel file

excel\_file <- "C:\\Users\\repan\\OneDrive\\Documents\\ten\_data.xlsx"

# Read the Excel file into a data frame

data <- read\_excel(excel\_file)

# Identify rows with empty cells

empty\_rows <- which(rowSums(is.na(data) | data == "") > 0)

# Remove rows with empty cells

data\_cleaned <- data[-empty\_rows, ]

# Display the cleaned data frame

print(data\_cleaned)

Data Visualization

Write a R Program to [adding Colors to Charts.](https://www.geeksforgeeks.org/adding-colors-to-charts-in-r-programming/)

# Load required library

install.packages("ggplot2")

library(ggplot2)

# Load the mtcars dataset

data(mtcars)

# Create a bar chart with colors

ggplot(mtcars, aes(x = factor(cyl), fill = factor(cyl))) +

geom\_bar() +

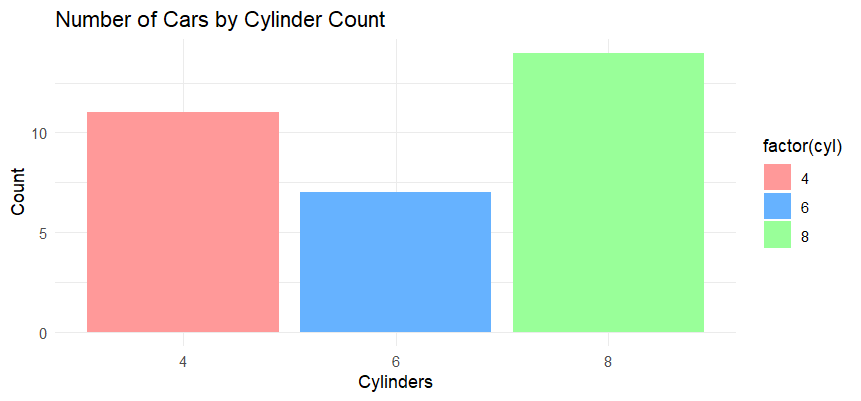
scale\_fill\_manual(values = c("#FF9999", "#66B2FF", "#99FF99")) + # Specify custom colors

labs(title = "Number of Cars by Cylinder Count",

x = "Cylinders",

y = "Count") +

theme\_minimal()



[Write a R Program to show legend in heatmap.](https://www.geeksforgeeks.org/how-to-show-legend-in-heatmap-in-r/)

# Load the mtcars dataset

data(mtcars)

# Select a subset of the mtcars dataset for demonstration

subset\_mtcars <- mtcars[, c("mpg", "hp", "qsec")]

# Create a heatmap with a legend

heatmap(

scale(subset\_mtcars), # Scale the data for better visualization

Colv = NA, # Do not perform hierarchical clustering on columns

Rowv = NA, # Do not perform hierarchical clustering on rows

col = cm.colors(256), # Specify color range

margins = c(5, 10), # Add margins for row and column labels

main = "Heatmap with Legend",

xlab = "Car Features",

ylab = "Cars",

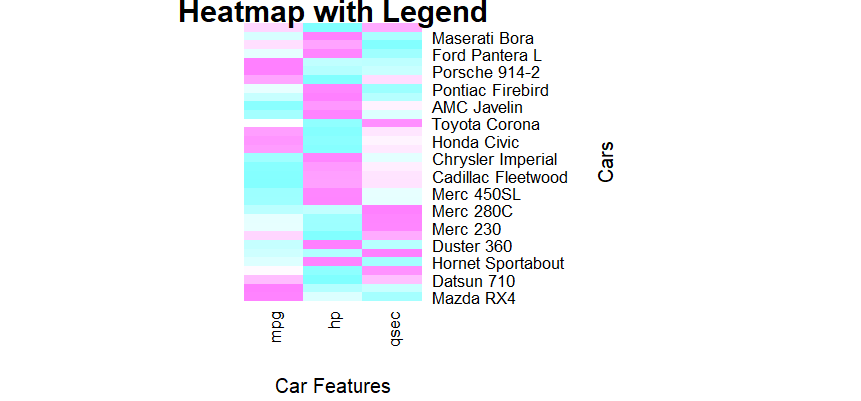
cexRow = 1, # Set the size of row labels

cexCol = 1, # Set the size of column labels

key = TRUE, # Show the legend

symkey = FALSE # Do not add a symmetric key

)



Write a R Program to [display All X-Axis Labels of Barplot.](https://www.geeksforgeeks.org/display-all-x-axis-labels-of-barplot-in-r/)

# Load the mtcars dataset

data(mtcars)

# Create a bar plot with all X-axis labels

bar\_colors <- c("#66B2FF", "#FF9999", "#99FF99", "#FFCC99", "#CC99FF")

barplot(

table(mtcars$cyl),

col = bar\_colors,

main = "Number of Cars by Cylinder Count",

xlab = "Cylinders",

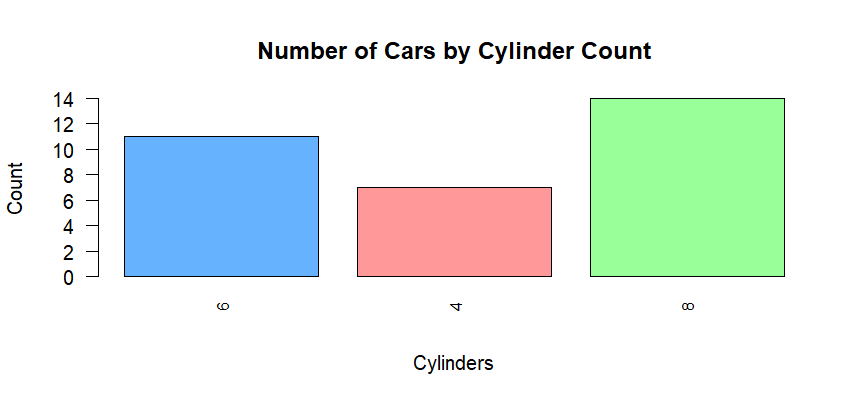
ylab = "Count",

names.arg = unique(mtcars$cyl),

las = 2, # Rotate X-axis labels vertically

cex.names = 0.8 # Adjust the size of X-axis labels

)



[Write a R Program to create a Stacked Dot Plot.](https://www.geeksforgeeks.org/how-to-create-a-stacked-dot-plot-in-r/)

# Load the mtcars dataset

data(mtcars)

# Create a stacked dot plot

dot\_colors <- c("#66B2FF", "#FF9999", "#99FF99")

# Convert the 'cyl' column to a factor for better plotting

mtcars$cyl <- as.factor(mtcars$cyl)

# Use geom\_dotplot from ggplot2 for the stacked dot plot

library(ggplot2)

ggplot(mtcars, aes(x = factor(cyl), y = mpg, fill = factor(cyl))) +

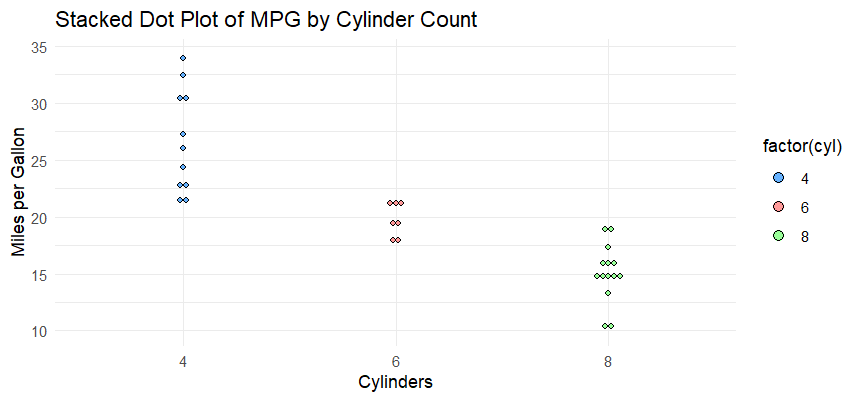
geom\_dotplot(binaxis = "y", stackdir = "center", binwidth = 1, dotsize = 0.5) +

scale\_fill\_manual(values = dot\_colors) +

labs(title = "Stacked Dot Plot of MPG by Cylinder Count",

x = "Cylinders",

y = "Miles per Gallon") +

theme\_minimal()

Write a R Program to [Change Spacing of Axis Tick Marks in Base R Plot](https://www.geeksforgeeks.org/change-spacing-of-axis-tick-marks-in-base-r-plot/)

# Load the mtcars dataset

data(mtcars)

# Create a base R plot with modified axis tick spacing

par(mar = c(5, 4, 4, 4)) # Adjust the margins to make room for labels

plot(mtcars$mpg, mtcars$wt,

xlab = "Miles per Gallon",

ylab = "Weight (1000 lbs)",

main = "Scatter Plot of MPG vs. Weight",

pch = 16, # Set the point character

col = "#66B2FF" # Set the point color

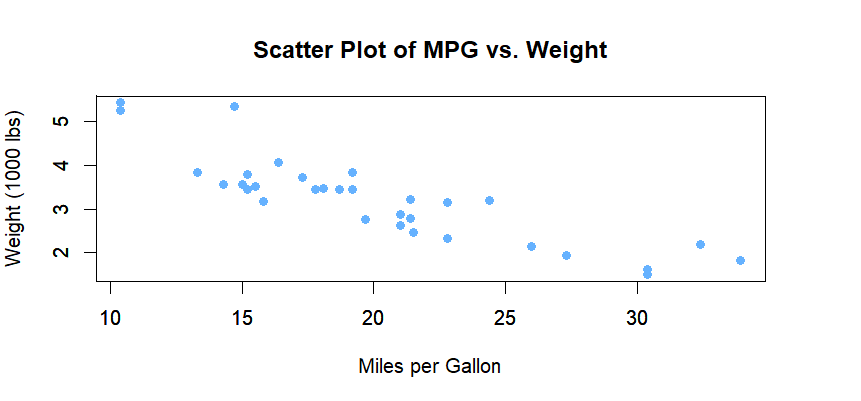
)

# Change the x-axis tick marks and labels

axis(side = 1, at = seq(10, 35, by = 5), labels = seq(10, 35, by = 5))

# Change the y-axis tick marks and labels

axis(side = 2, at = seq(1, 6, by = 1), labels = seq(1, 6, by = 1))



Write a R Program to [Add legends without border and with white background.](https://www.geeksforgeeks.org/add-legends-without-border-and-with-white-background-in-r/)

# Load the mtcars dataset

data(mtcars)

# Create a base R plot

plot(mtcars$mpg, mtcars$wt,

xlab = "Miles per Gallon",

ylab = "Weight (1000 lbs)",

main = "Scatter Plot of MPG vs. Weight",

pch = 16, # Set the point character

col = "#66B2FF" # Set the point color

)

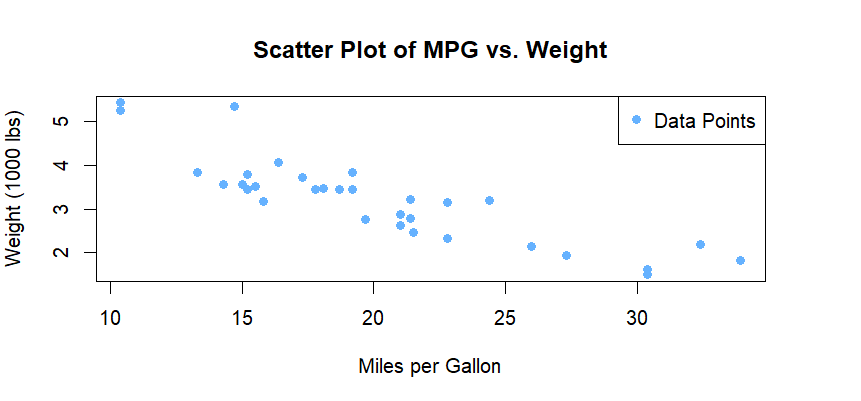
# Add a legend without border and with a white background

legend("topright", legend = "Data Points",

pch = 16, col = "#66B2FF",

bg = "white", # Set background color to white

box.lwd = 0) # Set legend border width to 0



Write a R Program to [Plot Shaded Area between vertical lines.](https://www.geeksforgeeks.org/plot-shaded-area-between-vertical-lines-in-r/)

# Load the mtcars dataset

data(mtcars)

# Define the range of x-values

x\_values <- c(15, 25)

# Create a base R plot

plot(mtcars$mpg, mtcars$wt,

xlab = "Miles per Gallon",

ylab = "Weight (1000 lbs)",

main = "Scatter Plot of MPG vs. Weight",

pch = 16, # Set the point character

col = "#66B2FF" # Set the point color

)

# Add vertical lines

abline(v = x\_values[1], col = "red", lty = 2) # First vertical line

abline(v = x\_values[2], col = "blue", lty = 2) # Second vertical line

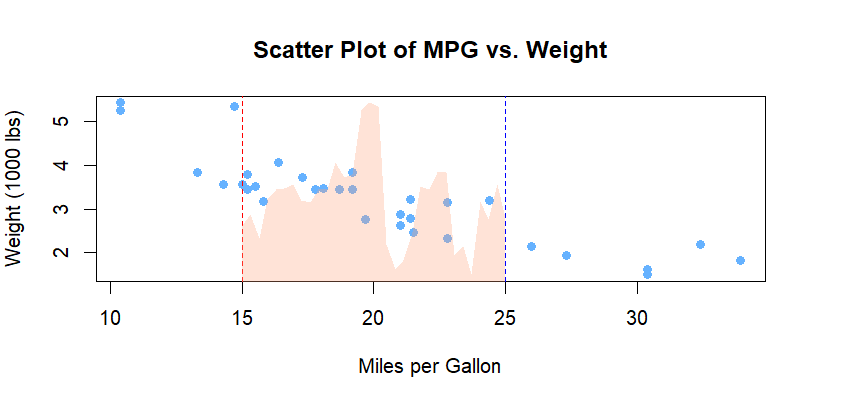
# Plot shaded area between the vertical lines

polygon(c(x\_values[1], seq(x\_values[1], x\_values[2], length.out = nrow(mtcars)), x\_values[2]),

c(0, mtcars$wt, 0),

col = alpha("#FFA07A", 0.3), # Set the fill color with transparency

border = NA) # Remove border



[Write a R Program to add Mean and Median to Histogram](https://www.geeksforgeeks.org/how-to-add-mean-and-median-to-histogram-in-r/).

# Load the mtcars dataset

data(mtcars)

# Create a histogram

hist(mtcars$mpg,

main = "Histogram of Miles per Gallon",

xlab = "Miles per Gallon",

col = "#66B2FF", # Set histogram fill color

border = "white" # Set histogram border color

)

# Add mean and median lines

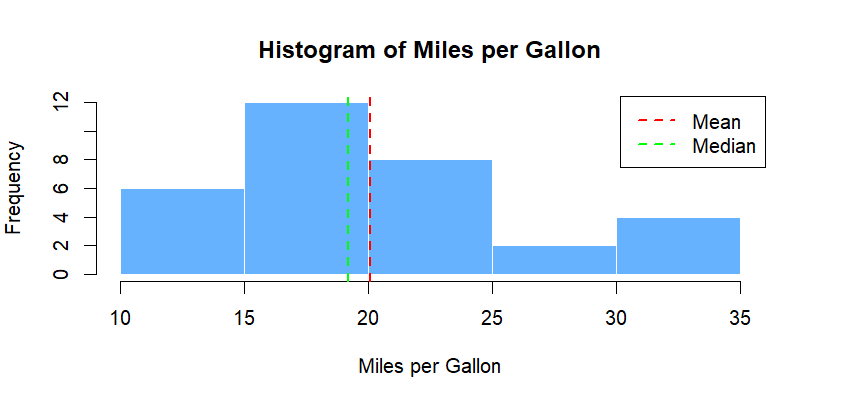
abline(v = mean(mtcars$mpg), col = "red", lty = 2, lwd = 2) # Mean line

abline(v = median(mtcars$mpg), col = "green", lty = 2, lwd = 2) # Median line

# Add a legend

legend("topright", legend = c("Mean", "Median"),

col = c("red", "green"), lty = 2, lwd = 2)



Write a R Program to [create Scatter plot from CSV file.](https://www.geeksforgeeks.org/create-scatter-plot-from-csv-in-r/)

# Load the iris dataset

data(iris)

# Scatter plot using the iris dataset

plot(iris$Sepal.Length, iris$Sepal.Width,

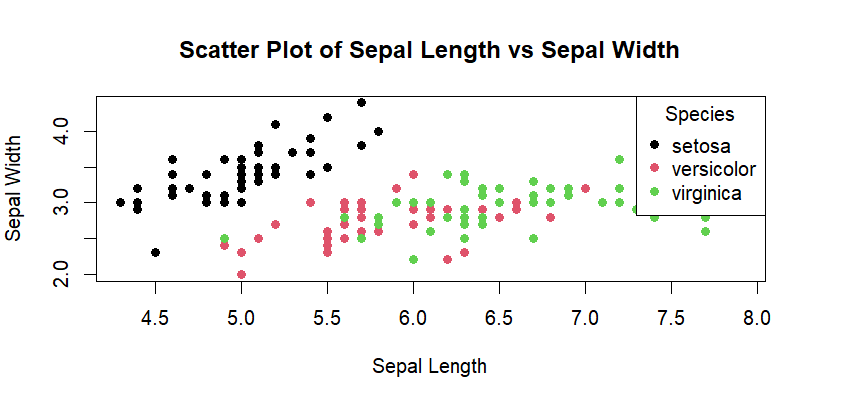
main = "Scatter Plot of Sepal Length vs Sepal Width",

xlab = "Sepal Length", ylab = "Sepal Width",

col = iris$Species, pch = 19)

legend("topright", legend = levels(iris$Species),

col = 1:3, pch = 19, title = "Species")



Others

Write a R program to find the linear regression for a dataset.

# Generate a sample dataset

set.seed(123) # Set seed for reproducibility

x <- 1:20

y <- 3 \* x + rnorm(20, mean = 0, sd = 5) # Simulating a linear relationship with some noise

# Create a data frame

data <- data.frame(x, y)

# Plot the dataset

plot(data$x, data$y, main = "Scatter Plot with Linear Relationship",

xlab = "X", ylab = "Y", pch = 16, col = "blue")

# Perform linear regression

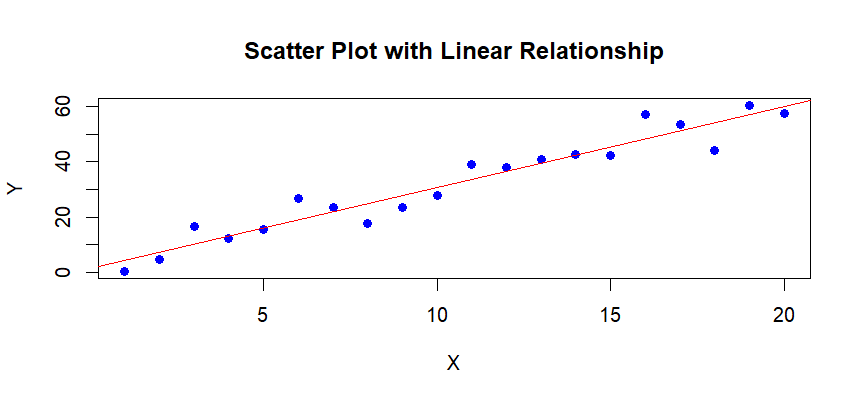
linear\_model <- lm(y ~ x, data = data)

# Add regression line to the plot

abline(linear\_model, col = "red")

# Display the regression summary

summary(linear\_model)



Write a R program to fill the null values in a data set by 3

# Create a sample dataset with missing values

set.seed(123)

data <- data.frame(

A = c(1, 2, NA, 4, 5),

B = c(NA, 8, 9, 10, NA),

C = c(11, 12, 13, NA, 15)

)

# Display the original dataset

print("Original Dataset:")

print(data)

# Replace NA values with 3

data[is.na(data)] <- 3

# Display the dataset after filling NA values with 3

print("\nDataset after replacing NA values with 3:")

print(data)

Write a R program to fine the no of rows and columns in a data set

# Create a sample dataset

data <- data.frame(

Name = c("Alice", "Bob", "Charlie", "David"),

Age = c(25, 30, 22, 28),

Salary = c(50000, 60000, 45000, 55000)

)

# Display the sample dataset

print("Sample Dataset:")

print(data)

# Find the number of rows and columns

num\_rows <- nrow(data)

num\_cols <- ncol(data)

# Display the results

cat("\nNumber of Rows:", num\_rows, "\n")

cat("Number of Columns:", num\_cols, "\n")

Write a R program to display the decision tree for a data set.

# Install and load the rpart.plot package

install.packages("rpart.plot")

library(rpart.plot)

# Load the iris dataset

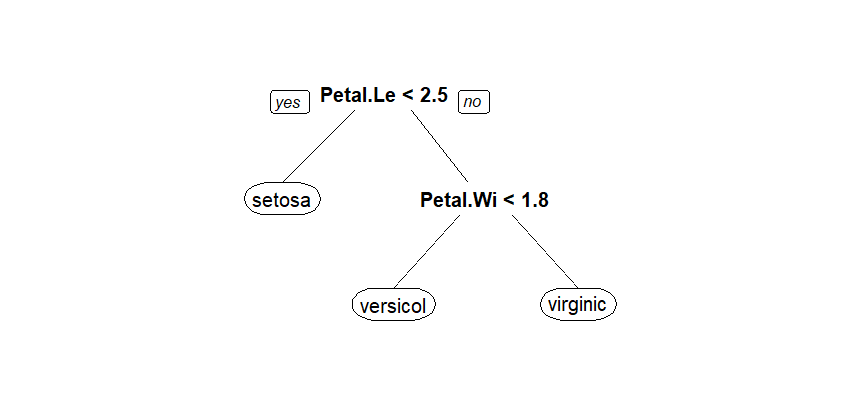
data(iris)

# Build a decision tree model

tree\_model <- rpart(Species ~ ., data = iris, method = "class")

# Display the decision tree with rpart.plot

prp(tree\_model)



Load a dataset for binary classification (e.g., the 'Pima Indians Diabetes' dataset) and perform data preprocessing tasks such as feature scaling and handling imbalanced data.

# Install and load necessary packages

install.packages("mlbench")

library(mlbench)

# Load the Pima Indians Diabetes dataset

data(PimaIndiansDiabetes)

diabetes\_data <- PimaIndiansDiabetes

# Display the structure of the dataset

str(diabetes\_data)

# Check for missing values

missing\_values <- sum(is.na(diabetes\_data))

cat("Number of missing values:", missing\_values, "\n")

# Summary statistics

summary(diabetes\_data)

# Feature scaling (standardization)

scaled\_data <- scale(diabetes\_data[, -9]) # Exclude the outcome variable (column 9)

diabetes\_data\_scaled <- cbind(scaled\_data, Outcome = diabetes\_data$diabetes)

# Display the structure of the scaled dataset

str(diabetes\_data\_scaled)

# Check the summary statistics of the scaled dataset

summary(diabetes\_data\_scaled)

output:-

> # Load the Pima Indians Diabetes dataset

> data(PimaIndiansDiabetes)

> diabetes\_data <- PimaIndiansDiabetes

>

> # Display the structure of the dataset

> str(diabetes\_data)

'data.frame': 768 obs. of 9 variables:

$ pregnant: num 6 1 8 1 0 5 3 10 2 8 ...

$ glucose : num 148 85 183 89 137 116 78 115 197 125 ...

$ pressure: num 72 66 64 66 40 74 50 0 70 96 ...

$ triceps : num 35 29 0 23 35 0 32 0 45 0 ...

$ insulin : num 0 0 0 94 168 0 88 0 543 0 ...

$ mass : num 33.6 26.6 23.3 28.1 43.1 25.6 31 35.3 30.5 0 ...

$ pedigree: num 0.627 0.351 0.672 0.167 2.288 ...

$ age : num 50 31 32 21 33 30 26 29 53 54 ...

$ diabetes: Factor w/ 2 levels "neg","pos": 2 1 2 1 2 1 2 1 2 2 ...

>

> # Check for missing values

> missing\_values <- sum(is.na(diabetes\_data))

> cat("Number of missing values:", missing\_values, "\n")

Number of missing values: 0

>

> # Summary statistics

> summary(diabetes\_data)

pregnant glucose pressure triceps insulin

Min. : 0.000 Min. : 0.0 Min. : 0.00 Min. : 0.00 Min. : 0.0

1st Qu.: 1.000 1st Qu.: 99.0 1st Qu.: 62.00 1st Qu.: 0.00 1st Qu.: 0.0

Median : 3.000 Median :117.0 Median : 72.00 Median :23.00 Median : 30.5

Mean : 3.845 Mean :120.9 Mean : 69.11 Mean :20.54 Mean : 79.8

3rd Qu.: 6.000 3rd Qu.:140.2 3rd Qu.: 80.00 3rd Qu.:32.00 3rd Qu.:127.2

Max. :17.000 Max. :199.0 Max. :122.00 Max. :99.00 Max. :846.0

mass pedigree age diabetes

Min. : 0.00 Min. :0.0780 Min. :21.00 neg:500

1st Qu.:27.30 1st Qu.:0.2437 1st Qu.:24.00 pos:268

Median :32.00 Median :0.3725 Median :29.00

Mean :31.99 Mean :0.4719 Mean :33.24

3rd Qu.:36.60 3rd Qu.:0.6262 3rd Qu.:41.00

Max. :67.10 Max. :2.4200 Max. :81.00

>

> # Feature scaling (standardization)

> scaled\_data <- scale(diabetes\_data[, -9]) # Exclude the outcome variable (column 9)

> diabetes\_data\_scaled <- cbind(scaled\_data, Outcome = diabetes\_data$diabetes)

>

> # Display the structure of the scaled dataset

> str(diabetes\_data\_scaled)

num [1:768, 1:9] 0.64 -0.844 1.233 -0.844 -1.141 ...

- attr(\*, "dimnames")=List of 2

..$ : chr [1:768] "1" "2" "3" "4" ...

..$ : chr [1:9] "pregnant" "glucose" "pressure" "triceps" ...

>

> # Check the summary statistics of the scaled dataset

> summary(diabetes\_data\_scaled)

pregnant glucose pressure triceps insulin

Min. :-1.1411 Min. :-3.7812 Min. :-3.5703 Min. :-1.2874 Min. :-0.6924

1st Qu.:-0.8443 1st Qu.:-0.6848 1st Qu.:-0.3671 1st Qu.:-1.2874 1st Qu.:-0.6924

Median :-0.2508 Median :-0.1218 Median : 0.1495 Median : 0.1544 Median :-0.4278

Mean : 0.0000 Mean : 0.0000 Mean : 0.0000 Mean : 0.0000 Mean : 0.0000

3rd Qu.: 0.6395 3rd Qu.: 0.6054 3rd Qu.: 0.5629 3rd Qu.: 0.7186 3rd Qu.: 0.4117

Max. : 3.9040 Max. : 2.4429 Max. : 2.7327 Max. : 4.9187 Max. : 6.6485

mass pedigree age Outcome

Min. :-4.057829 Min. :-1.1888 Min. :-1.0409 Min. :1.000

1st Qu.:-0.595191 1st Qu.:-0.6885 1st Qu.:-0.7858 1st Qu.:1.000

Median : 0.000941 Median :-0.2999 Median :-0.3606 Median :1.000

Mean : 0.000000 Mean : 0.0000 Mean : 0.0000 Mean :1.349

3rd Qu.: 0.584390 3rd Qu.: 0.4659 3rd Qu.: 0.6598 3rd Qu.:2.000

Max. : 4.452906 Max. : 5.8797 Max. : 4.0611 Max. :2.000