Experiment No 10

Title: Create bar charts, line, scatter plots, histogram using R.

Objective:

The objective of this practical is to learn how to create various types of data visualizations in R, including bar charts, line charts, scatter plots, and histograms. By using R's built-in plotting functions, we will gain hands-on experience in visualizing data and interpreting graphical outputs.

Theory :

R language is mostly used for statistics and data analytics purposes to represent the data graphically in the software. To represent those data graphically, charts and graphs are used in R.

R – graphs

There are hundreds of charts and graphs present in R. For example, bar plot, box plot, mosaic plot, dot chart, coplot, histogram, pie chart, scatter graph, etc.

Types of R – Charts

- Line Chart
- Bar Plot or Bar Chart
- Pie Diagram or Pie Chart
- Histogram
- Scatter Plot
- Box Plot

Bar Plot or Bar Chart

Bar plot or Bar Chart in R is used to represent the values in data vector as height of the bars. The data vector passed to the function is represented over y-axis of the graph. Bar chart can behave like histogram by using table() function instead of data vector.

Syntax: barplot(data, xlab, ylab)

where:

- *data is the data vector to be represented on y-axis*
- *xlab is the label given to x-axis*
- *ylab is the label given to y-axis*

Histogram

Histogram is a graphical representation used to create a graph with bars representing the frequency of grouped data in vector. Histogram is same as bar chart but only difference between them is histogram represents frequency of grouped data rather than data itself.

Syntax: hist(x, col, border, main, xlab, ylab)

where:

- *x is data vector*
- *col specifies the color of the bars to be filled*
- *border specifies the color of border of bars*
- *main specifies the title name of histogram*
- *xlab specifies the x-axis label*
- *ylab specifies the y-axis label*

Scatter Plot

A Scatter plot is another type of graphical representation used to plot the points to show relationship between two data vectors. One of the data vectors is represented on x-axis and another on y-axis.

Syntax: plot(x, y, type, xlab, ylab, main)

Where,

- x is the data vector represented on x-axis
- y is the data vector represented on y-axis
- type specifies the type of plot to be drawn. For example, “l” for lines, “p” for points, “s” for stair steps, etc.
- xlab specifies the label for x-axis
- ylab specifies the label for y-axis
- main specifies the title name of the graph

Line Graph

A line graph is a chart that is used to display information in the form of a series of data points. It utilizes points and lines to represent change over time. Line graphs are drawn by plotting different points on their X coordinates and Y coordinates, then by joining them together through a line from beginning to end. The graph represents different values as it can move up and down based on the suitable variable.

R – Line Graphs

The plot() function in R is used to create the line graph.

Syntax: plot(v, type, col, xlab, ylab)

Parameters:

- v: This parameter is a contains only the numeric values
- type: This parameter has the following value:

    1. “p” : This value is used to draw only the points.
    2. “l” : This value is used to draw only the lines.
    3. “o”: This value is used to draw both points and lines
- xlab: This parameter is the label for x axis in the chart.
- ylab: This parameter is the label for y axis in the chart.
- main: This parameter main is the title of the chart.
- col: This parameter is used to give colors to both the points and lines.

1. **Practical Approach:**

1. Bar Plot:

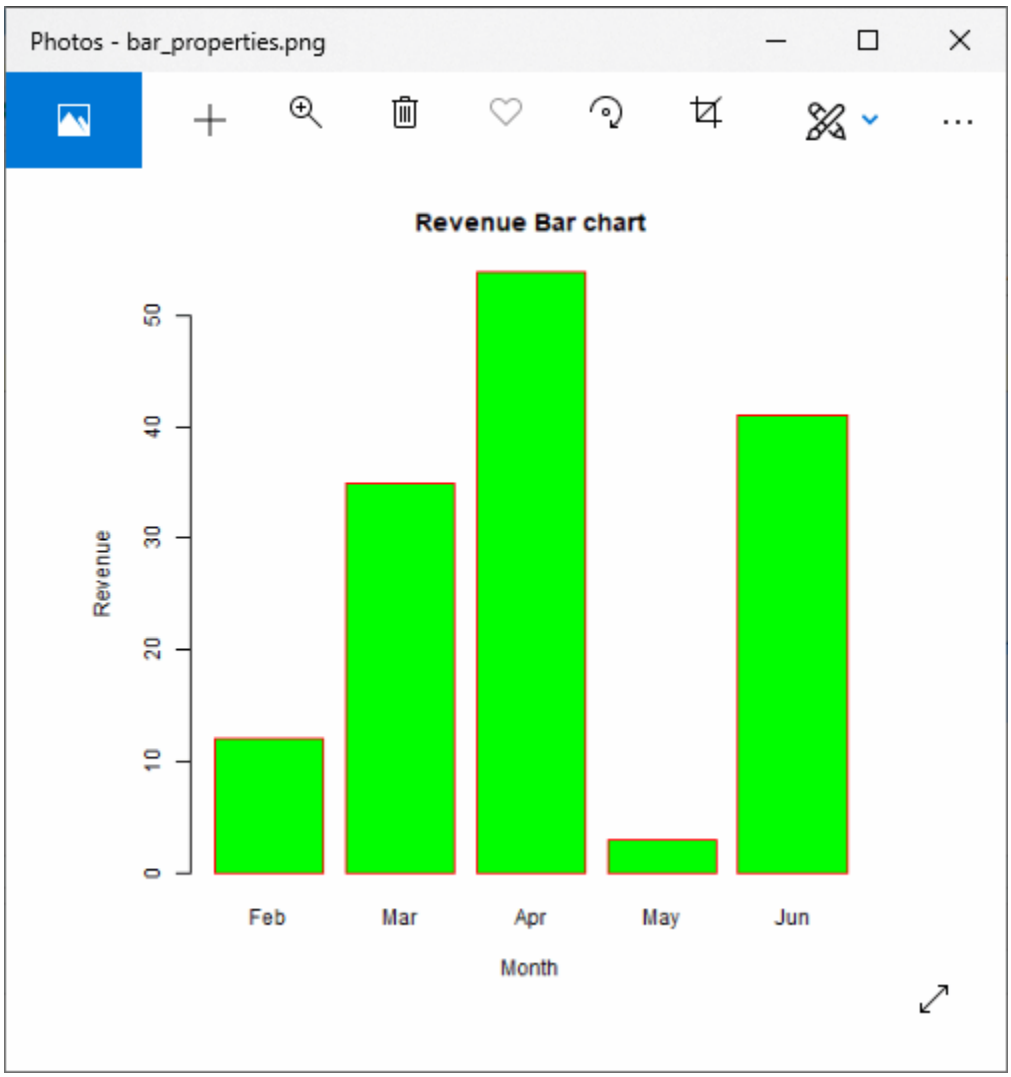
```
# Creating the data for Bar chart
H <- c(12,35,54,3,41)
M<- c("Feb","Mar","Apr","May","Jun")

# Giving the chart file a name
png(file = "bar_properties.png")

# Plotting the bar chart
barplot(H,names.arg=M,xlab="Month",ylab="Revenue",col="Green",
        main="Revenue Bar chart",border="red")
# Saving the file
```

12. dev.off()

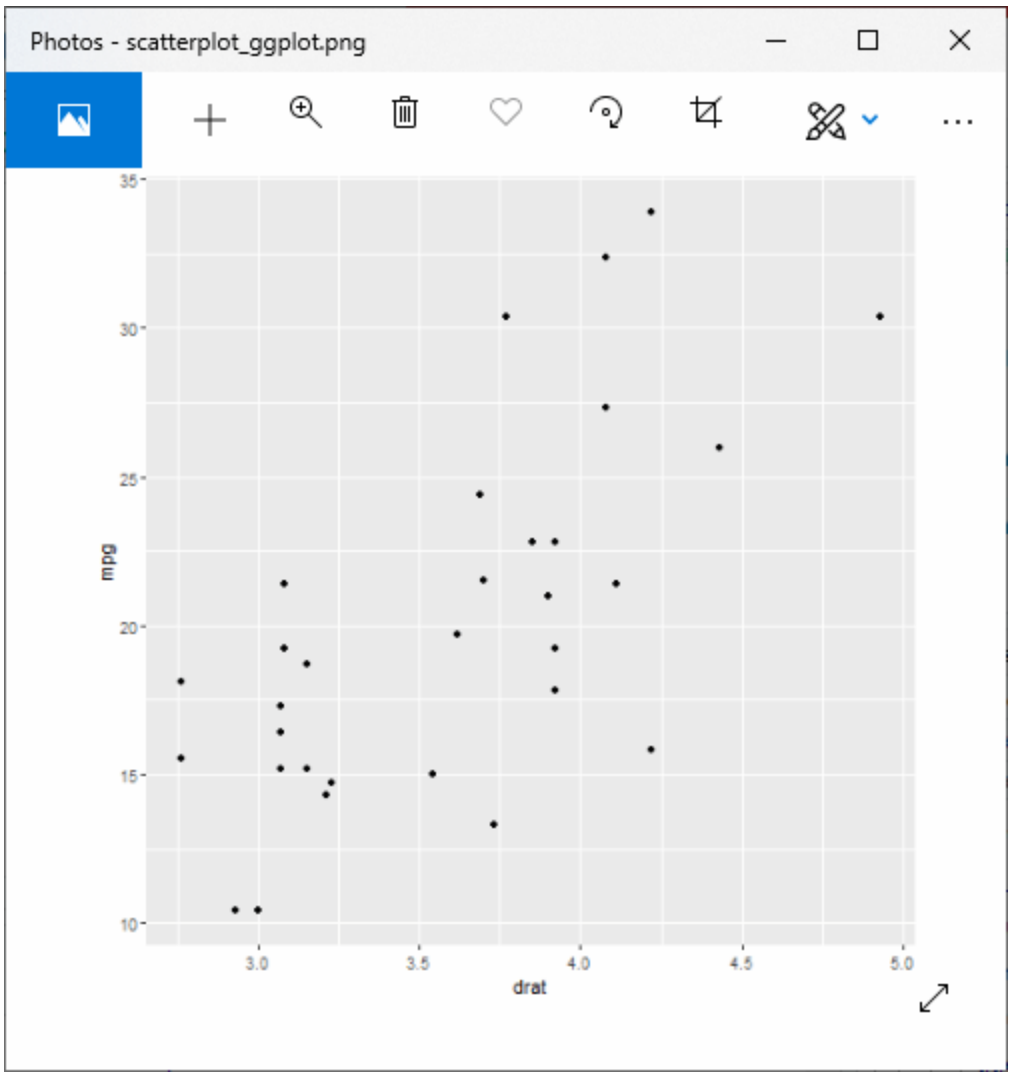
Output:



2.Scatter Plot

1. #Loading ggplot2 package
2. library(ggplot2)
3. # Giving a name to the chart file.
4. png(file = "scatterplot_ggplot.png")
5. # Plotting the chart using ggplot() and geom_point() functions.
6. ggplot(mtcars, aes(x = drat, y = mpg)) +geom_point()
7. # Saving the file.

8. dev.off()



3. Histogram:

1. # Creating data for the graph.
2. v <- c(12,24,16,38,21,13,55,17,39,10,60)

3. # Giving a name to the chart file.
4. png(file = "histogram_chart.png")
5.
6. # Creating the histogram.
7. hist(v,xlab = "Weight",ylab="Frequency",col = "green",border = "red")
8.

9. # Saving the file.
10. dev.off()

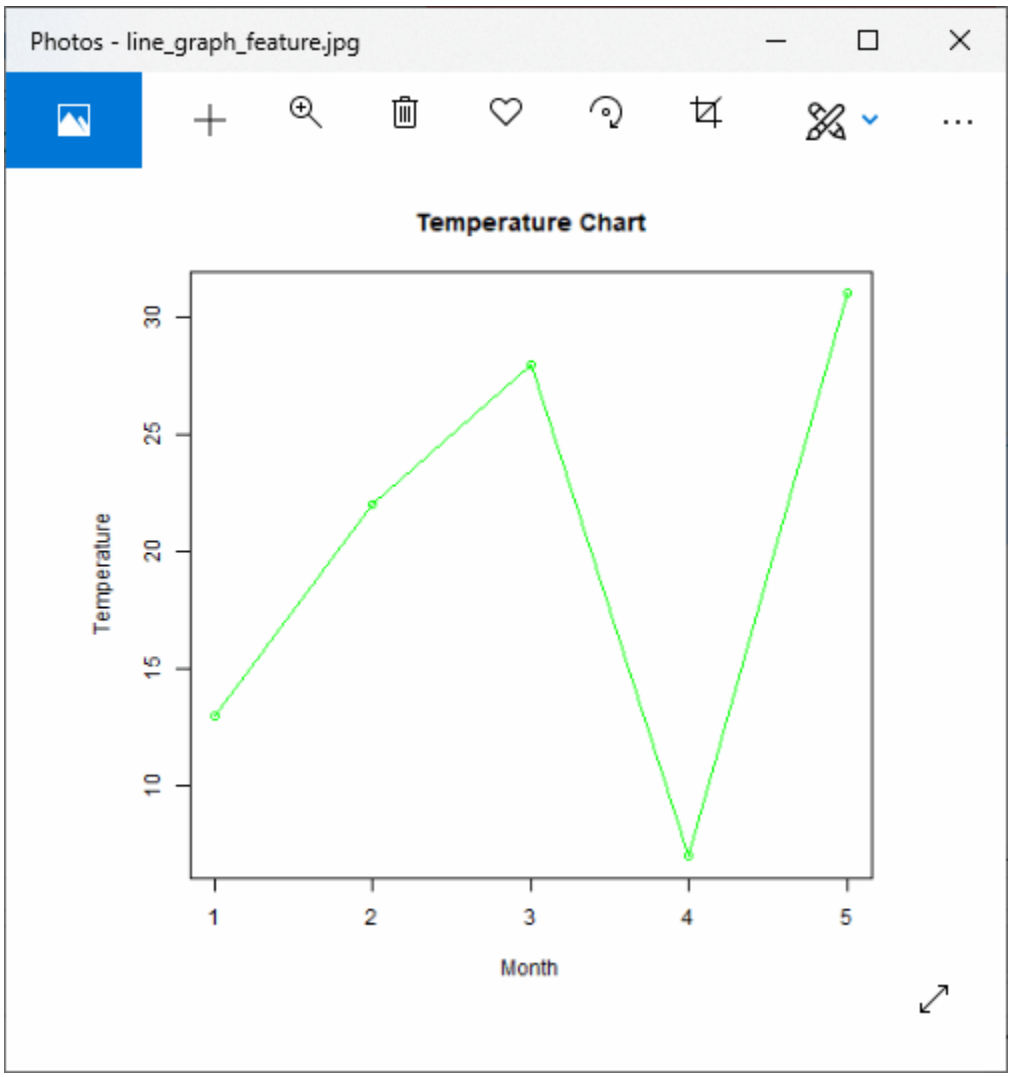
Output:

Line Chart:

1. # Creating the data for the chart.
2. v <- c(13,22,28,7,31)
3. # Giving a name to the chart file.
4. png(file = "line_graph_feature.jpg")
5. # Plotting the bar chart.

6. plot(v,type = "o",col="green",xlab="Month",ylab="Temperature")
7. # Saving the file.
8. dev.off()

Output:



Outcome:

The outcome of this practical is to learn how to make different types of charts in R to show and understand data better.

Online Reference Websites:

- https://bioinformatics-core-shared-training.github.io/graphics/graphics-exercises.pdf
- https://www.bioinformatics.babraham.ac.uk/training/Core_R_Plotting/Core%20R%20Plotting%20Exercises.pdf

Viva Questions:

1.	**What is the purpose of a bar chart, and when would you use it?**
2.	**How does a line chart differ from a bar chart?**
3.	**What kind of data is best visualized with a scatter plot?**
4.	**Why are histograms useful for analyzing data distributions?**
5.	**How do you add a title and labels to a plot in R?**
6.	**What function do you use to create a bar chart in R?**
7.	**How can you customize the colors of your chart in R?**
8.	**What is the significance of adding a trendline to a scatter plot?**
9.	**How can you adjust the number of bins in a histogram in R?**
10.	**How do you save a plot as an image file in R?**
11.	**What function in R allows you to add labels to data points on a chart?**
12.	**How does a histogram differ from a bar chart?**

Conclusion:

Students learn how to create different types of charts in R to display and understand data. Students now know which chart type works best for different kinds of data, how to make the charts look clearer, and how to highlight important patterns.

Experiment No 11

Title: Implementation of the linear and multiple linear regression using R.

Objective: The objective of the practical is to implement and understand linear and multiple linear regression models in R. It involves fitting the models, interpreting outputs, and visualizing relationships between variables.

Theory :

Linear Regression is a commonly used type of predictive analysis. Linear Regression is a statistical approach for modelling the relationship between a dependent variable and a given set of independent variables. It is predicted that a straight line can be used to approximate the relationship. The goal of linear regression is to identify the line that minimizes the discrepancies between the observed data points and the line's anticipated values.

There are two types of linear regression.

- Simple Linear Regression
- Multiple Linear Regression

Simple Linear Regression in R

In Machine Learning Linear regression is one of the easiest and most popular Machine Learning algorithms.

- It is a statistical method that is used for predictive analysis.
- Linear regression makes predictions for continuous/real or numeric variables such as sales, salary, age, product price, etc.
- Linear regression algorithm shows a linear relationship between a dependent (y) and one or more independent (y) variables, hence called as linear regression. Since linear regression shows the linear relationship, which means it finds how the value of the dependent variable changes according to the value of the independent variable.

Linear Regression Line

A linear line showing the relationship between the dependent and independent variables is called a regression line. A regression line can show two types of relationship:

- Positive Linear Relationship: If the dependent variable increases on the Y-axis and the independent variable increases on the X-axis, then such a relationship is termed as a Positive linear relationship.

- Negative Linear Relationship: If the dependent variable decreases on the Y-axis and independent variable increases on the X-axis, then such a relationship is called a negative linear relationship.

Assumptions of Linear Regression

Below are some important assumptions of Linear Regression. These are some formal checks while building a Linear Regression model, which ensures to get the best possible result from the given dataset.

1. Linear relationship between the features and target: Linear regression assumes the linear relationship between the dependent and independent variables.

1. Small or no multicollinearity between the features: Multicollinearity means high-correlation between the independent variables. Due to multicollinearity, it may difficult to find the true relationship between the predictors and target variables. Or we can say, it is difficult to determine which predictor variable is affecting the target variable and which is not. So, the model assumes either little or no multicollinearity between the features or independent variables.

1. Homoscedasticity: Homoscedasticity is a situation when the error term is the same for all the values of independent variables. With homoscedasticity, there should be no clear pattern distribution of data in the scatter plot. We want to have homoscedasticity. To check for homoscedasticity we use a scatterplot of the residuals against the independent variable. Once this plot is created we are looking for a rectangular shape. This rectangle is indicative of homoscedasticity. If we see a cone shape, that is indicative of heteroscedasticity.

1. Normal distribution of error terms: Linear regression assumes that the error term should follow the normal distribution pattern. If error terms are not normally distributed, then confidence intervals will become either too wide or too narrow, which may cause difficulties in finding coefficients. It can be checked using the q-q plot. If the plot shows a straight line without any deviation, which means the error is normally distributed.

1. No autocorrelations: The linear regression model assumes no autocorrelation in error terms. If there will be any correlation in the error term, then it will drastically reduce the accuracy of the model. Autocorrelation usually occurs if there is a dependency between residual errors.

It is a statistical method that allows us to summarize and study relationships between two continuous (quantitative) variables. One variable denoted x is regarded as an independent variable and the other one denoted y is regarded as a dependent variable. It is assumed that the two variables are linearly related. Hence, we try to find a linear function that predicts the response value(y) as accurately as possible as a function of the feature or independent variable(x).

$Y = \beta_o + \beta_1 X + \varepsilon$

- The dependent variable, also known as the response or outcome variable, is represented by the letter Y.

- The independent variable, often known as the predictor or explanatory variable, is denoted by the letter X.

- The intercept, or value of Y when X is zero, is represented by the β_o.
- The slope or change in Y resulting from a one-unit change in X is represented by the β_1.
- The error term or the unexplained variation in Y is represented by the ε.

Multiple Linear Regression :

It is the most common form of Linear Regression. Multiple Linear Regression basically describes how a single response variable Y depends linearly on a number of predictor variables.

The basic examples where Multiple Regression can be used are as follows:

1. The selling price of a house can depend on the desirability of the location, the number of bedrooms, the number of bathrooms, the year the house was built, the square footage of the lot, and a number of other factors.
2. The height of a child can depend on the height of the mother, the height of the father, nutrition, and environmental factors.

Estimation of the Model Parameters
Consider a multiple linear Regression model with k independent predictor variable x1, x2……, xk, and one response variable y.

$$y = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k + \epsilon$$

Suppose we have n observation on the k+1 variables and the variable of n should be greater than k.

$$y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_k x_{ik} + \epsilon_i, \quad i = 1, \ldots, n$$

The basic goal in least-squares regression is to fit a hyper-plane into (k + 1)-dimensional space that minimizes the sum of squared residuals.

$$\sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{k} \beta_j x_{ij} \right)^2$$

Before taking the derivative with respect to the model parameters set them equal to zero and derive the least-squares normal equations that the parameters would have to fulfill.
These equations are formulated with the help of vectors and matrices.
Let

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \qquad \mathbf{X} = \begin{bmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1k} \\ 1 & x_{21} & x_{22} & \cdots & x_{2k} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_{n1} & x_{n2} & \cdots & x_{nk} \end{bmatrix}$$

$$\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{bmatrix}, \qquad \epsilon = \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{bmatrix}$$

The linear Regression model is written in the form as follows:

$$\mathbf{y} = \mathbf{X}\beta + \epsilon$$

In linear regression the least square parameters estimate b

$$\sum_{i=1}^{n} \epsilon_i^2 = \epsilon'\epsilon = (\mathbf{y} - \mathbf{X}\beta)'(\mathbf{y} - \mathbf{X}\beta)$$

Imagine the columns of X to be fixed, they are the data for a specific problem and say b to be variable. We want to find the “best” b in the sense that the sum of squared residuals is minimized. The smallest that the sum of squares could be is zero.

$$\hat{\mathbf{y}} = \mathbf{X}\hat{\beta}$$

Here y is the estimated response vector.

1. **Practical Approach:**

- **Implementation of linear regression**

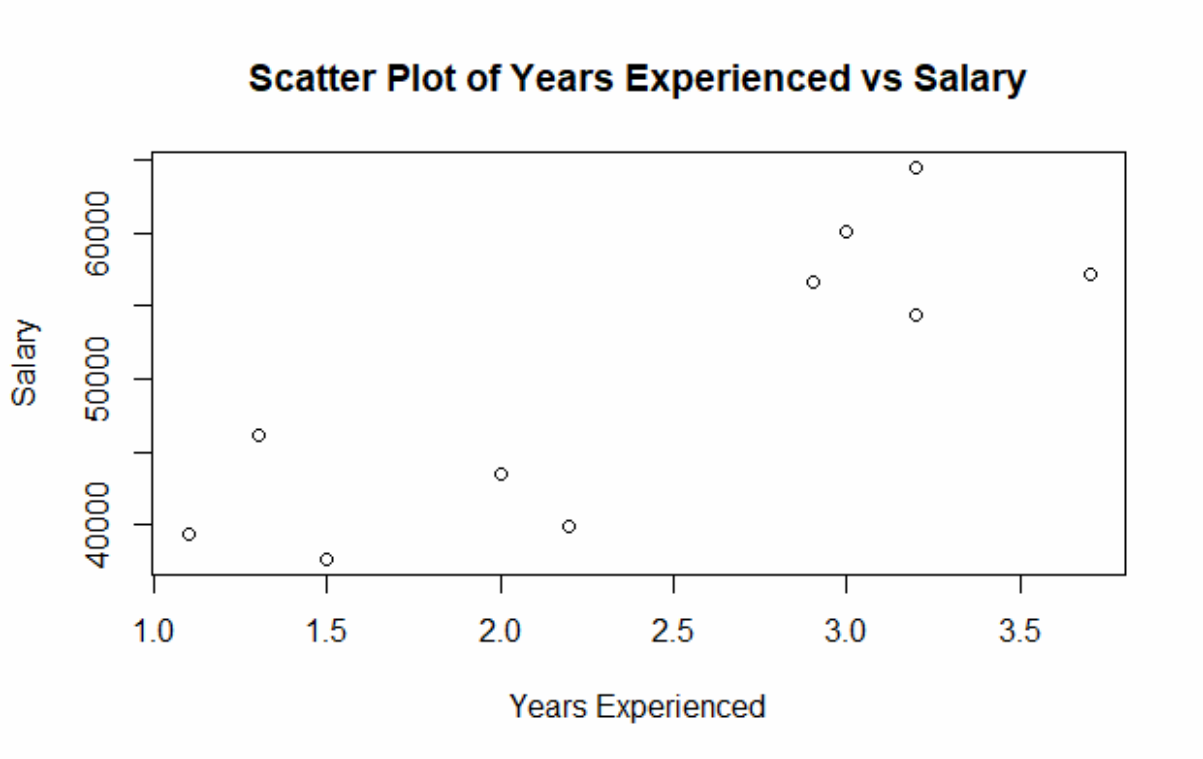
1.Create the Data Frame

```
# Create the data frame
data <- data.frame(
Years_Exp = c(1.1, 1.3, 1.5, 2.0, 2.2, 2.9, 3.0, 3.2, 3.2, 3.7),
Salary = c(39343.00, 46205.00, 37731.00, 43525.00,
                                39891.00, 56642.00, 60150.00, 54445.00, 64445.00, 57189.00)
)
```

2. **Scatter plot of the given dataset**

```
# Create the scatter plot
plot(data$Years_Exp, data$Salary,
        xlab = "Years Experienced",
        ylab = "Salary",
        main = "Scatter Plot of Years Experienced vs Salary")
```

Output:



3.Implement Simple Linear Regression

```
install.packages('caTools')
library(caTools)
split = sample.split(data$Salary, SplitRatio = 0.7)
trainingset = subset(data, split == TRUE)
testset = subset(data, split == FALSE)

# Fitting Simple Linear Regression to the Training set
lm.r= lm(formula = Salary ~ Years_Exp,
                    data = trainingset)
#Summary of the model
summary(lm.r)
```

Output:

```
Call:
lm(formula = Salary ~ Years_Exp, data = trainingset)
Residuals:
    1     2     3     5     6     8    10
```

```
  463.1  5879.1 -4041.0 -6942.0  4748.0   381.9  -489.1
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)    30927       4877   6.341  0.00144 **
Years_Exp       7230       1983   3.645  0.01482 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
Residual standard error: 4944 on 5 degrees of freedom
Multiple R-squared:  0.7266,    Adjusted R-squared:  0.6719
F-statistic: 13.29 on 1 and 5 DF,  p-value: 0.01482
```

4. **Predict values using predict function**

```
# Create a data frame with new input values
new_data <- data.frame(Years_Exp = c(4.0, 4.5, 5.0))

# Predict using the linear regression model
predicted_salaries <- predict(lm.r, newdata = new_data)

# Display the predicted salaries
print(predicted_salaries)
```

Output:

```
     1       2       3
65673.14 70227.40 74781.66
```

5. **Visualizing the Training set results:**

```
# Visualising the Training set results
ggplot() + geom_point(aes(x = trainingset$Years_Ex,
                                                y = trainingset$Salary), colour = 'red') +
geom_line(aes(x = trainingset$Years_Ex,
```

```
                                    y = predict(lm.r, newdata = trainingset)), colour = 'blue') +

ggtitle('Salary vs Experience (Training set)') +
xlab('Years of experience') +
ylab('Salary')
```

Output:

6.**Visualizing the Testing set results:**

```
# Visualising the Test set results
ggplot() +
geom_point(aes(x = testset$Years_Exp, y = testset$Salary),
                                    colour = 'red') +
geom_line(aes(x = trainingset$Years_Exp,
                                    y = predict(lm.r, newdata = trainingset)),
                                    colour = 'blue') +
ggtitle('Salary vs Experience (Test set)') +
xlab('Years of experience') +
ylab('Salary')
```

Output:

- **Implementation of Multi Linear Regression**

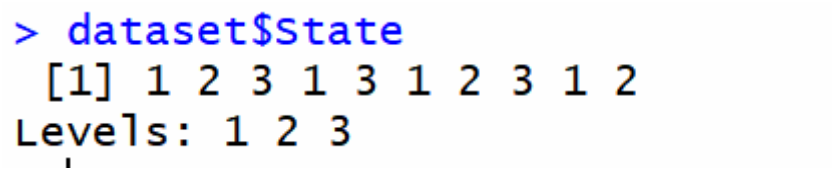
1.Encoding Categorical Data

```
# Multiple Linear Regression

# Importing the dataset
dataset = read.csv('data2.csv')

# Encoding categorical data
dataset$State = factor(dataset$State,
                       levels = c('New York', 'California', 'Florida'),
                       labels = c(1, 2, 3))
dataset$State
```

Output:



2. Predicting Results

```
# Splitting the dataset into the Training set and Test set
# install.packages('caTools')
library(caTools)
set.seed(123)
split = sample.split(dataset$Profit, SplitRatio = 0.8)
training_set = subset(dataset, split == TRUE)
test_set = subset(dataset, split == FALSE)

# Feature Scaling
# training_set = scale(training_set)
# test_set = scale(test_set)

# Fitting Multiple Linear Regression to the Training set
regressor = lm(formula = Profit ~ .,
               data = training_set)

# Predicting the Test set results
y_pred = predict(regressor, newdata = test_set)
```

Output:

```
> regressor

Call:
lm(formula = Profit ~ ., data = training_set)

Coefficients:
    (Intercept)        R.D.Spend   Administration  Marketing.Spend
      2.816e+04        8.884e-01        5.670e-02        2.859e-02
         State2           State3
     -2.861e+03        9.172e+03
```

```
> y_pred
       5        8
179233.6 170602.2
```

Outcome:

The outcome of this practical is to learn how to create and analyze linear regression models in R. You'll understand how to find relationships between variables and assess how well the model predicts outcomes.

Viva Questions:

1.	What is linear regression?
2.	What is the difference between simple linear regression and multiple linear regression?
3.	Explain the role of the dependent and independent variables in linear regression.
4.	What are the main assumptions of linear regression?
5.	What is the purpose of the intercept and slope in a linear regression model?
6.	How do you perform linear regression in R?
7.	What does the **summary()** function do in the context of a regression model in R?
8.	What is the purpose of the **lm()** function in R?
9.	How do you interpret the coefficients of a multiple linear regression model in R?
10.	What is R-squared, and how do you interpret its value?

Conclusion:

This practical helps students to learn how to build and analyze linear regression models in R. Students gain an understanding of how to find relationships between variables, make predictions, and evaluate the model's performance using statistical measures.