INDEX

14				
15				

EX.NO: 1

INSTALLATION OF R PROGRAM AND IMPORT PACKAGES

AIM:

To install R and import essential packages in R.

PROCEDURE:

o Install R from the official website: https://cran.r-project.org/ o Install the required package(s) using install.packages() in R.

o Use library() to load the package.

SOURCE CODE:

```
# Install a package (if not installed)
install.packages("ggplot2")

# Import the package
library(ggplot2)
```

OUTPUT:

No output, but you can check if the library is loaded by using a function like ggplot2::ggplot().

RESULT:

Installed R and imported essential packages successfully.

EX.NO: 2

IMPLEMENTATION OF R PROGRAM – BASIC

AIM:

To understand basic R syntax and printing output.

PROCEDURE:

o Create simple variables.

o Print their values using print() or cat().

SOURCE CODE:

```
# Basic program to print Hello World
message <- "Hello, World!" print(message)
```

OUTPUT:

```
[1] "Hello, World!"
```

RESULT:

Executed basic R programs, including arithmetic and variable assignments.

EX.NO: 3

IMPLEMENTATION OF R PROGRAM – BASIC (ADDITION)

AIM:

To perform basic arithmetic operations.

PROCEDURE: o Define two
variables. o Add them using the
+ operator. o Print the result.

SOURCE CODE:

```
# Adding two
numbers  num1 <- 5
num2 <- 3  sum <-
num1 + num2
print(sum)
```

OUTPUT:

```
[1] 8
```

RESULT:

Practiced basic R functions, data structures, and operations.

EX.NO: 4

IMPLEMENTATION OF DATA TYPES IN R

AIM:

To understand and implement various data types in R.

PROCEDURE:

o Create variables of different types: numeric, character, logical, and complex. o

Print the type using typeof().

SOURCE CODE:

```
# Creating variables of different types
num <- 10
text  <-  "Hello"
is_true           <-
TRUE
complex_num <- 1 + 2i

# Print types
print(typeof(num))  # Numeric  print(typeof(text))
        # Character  print(typeof(is_true))      #
Logical
print(typeof(complex_num))           # Complex
```

OUTPUT:

```
[1] "double"
[1] "character"
[1] "logical"
[1] "complex"
```

RESULT:

Implemented various R data types like vectors, lists, and factors.

EX.NO: 5

IMPLEMENTATION OF CONTROL STATEMENTS IN R

AIM:

To implement basic control structures (if-else).

PROCEDURE:

o Use if, else, and else if for decision-making.

SOURCE CODE:

```
# Check if the number is positive or negative  num
<-  -5


if (num > 0) {
print("Positive")  }
else if (num < 0) {
print("Negative")
} else {
print("Zero")
}
```

OUTPUT:-

```
[1] "Negative"
```

RESULT:

Applied control statements like if-else and switch-case in R.

EX.NO: 6

IMPLEMENTATION OF LOOPING STATEMENTS

AIM:

To implement basic looping structures like for and while loops.

PROCEDURE:

o Use a for loop to print numbers from 1 to 5.
o Use a while loop to print numbers from 6 to 10.

SOURCE CODE:

```
# For loop example
for   (i   in   1:5)   {
print(i)
}

# While loop example
i <- 6  while (i
<= 10) {
    print(i)
    i <- i + 1
}
```

OUTPUT:

```
[1] 1
[1] 2
[1] 3
[1] 4
[1] 5
[1] 6
[1] 7
[1] 8
[1] 9
[1] 10
```

RESULT:

Practiced loops, including for, while, and repeat statements in R.

EX.NO: 7

IMPLEMENTATION OF DECISION TREE AND KNN IN R

AIM:

To implement decision trees and KNN algorithms for classification tasks.

PROCEDURE: o Install and load the necessary package rpart for decision tree and class for KNN. o Use sample data for training.

SOURCE CODE:

```
# Load necessary libraries
install.packages("rpart")
library(rpart)

# Sample data
data(iris)

# Create a decision tree
model <- rpart(Species ~ Sepal.Length + Sepal.Width, data = iris)

# Print the model
print(model)
```

OUTPUT:

Decision tree model summary.

RESULT:

Built decision tree and KNN models for classification tasks.

EX.NO:

8

IMPLEMENTATION OF NAÏVE BAYES

AIM:

To implement the Naïve Bayes classification algorithm.

PROCEDURE: o Install and load the e1071 package.

o Use naiveBayes() to create a model.

SOURCE CODE:

```
# Install and load the e1071 package
install.packages("e1071")
library(e1071)


# Sample data
data(iris)


# Apply Naive Bayes
model <- naiveBayes(Species ~ Sepal.Length + Sepal.Width,
data= iris)
# Print model summary
print(model)
```

OUTPUT:

Naïve Bayes model summary.

RESULT:

Implemented Naïve Bayes for probabilistic classification in R.

9

IMPLEMENTATION OF RANDOM FOREST IN R

AIM:

To implement a random forest model for classification.

PROCEDURE:

o Install and load the randomForest package. o

Create and train a random forest model.

SOURCE CODE:

```
# Install and load the randomForest

package  install.packages("randomForest")

library(randomForest)  # Sample data

data(iris)


# Create a Random Forest model  model <- randomForest(Species ~

Sepal.Length + Sepal.Width, data = iris)

# Print the model

print(model)
```

OUTPUT:

Random Forest model summary.

RESULT:

Developed and evaluated a Random Forest model using R.

EX.NO: 10

IMPLEMENTATION OF K-MEANS IN R

AIM:

To implement the K-means clustering algorithm.

PROCEDURE:

o Use kmeans() to perform clustering on the data. o

Plot the clusters.

SOURCE CODE:

```
# Sample data
data(iris)

# K-means clustering
set.seed(10)
clusters <- kmeans(iris[, 1:4], centers = 3)

# Print clustering results
print(clusters)
```

OUTPUT:

K-means clustering results, including cluster centers.

RESULT:

Applied K-Means clustering for unsupervised learning in R.

EX.NO: 11

IMPLEMENTATION OF MEDOIDS IN R

AIM:

To implement the k-medoids clustering algorithm using R. Medoids are similar to centroids in the k-means clustering but use actual data points as cluster centers.

PROCEDURE:

1. Load the necessary libraries (cluster and factoextra).
2. Import or create a dataset.
3. Normalize or standardize the dataset (if required).
4. Apply the pam() function from the cluster package to perform k-medoids clustering.
5. Visualize the clusters using the fviz_cluster() function.

SIMPLE PROGRAM:

```
# Load required libraries
library(cluster)
library(factoextra)

# Create a sample dataset
data <- data.frame(x = rnorm(100), y = rnorm(100))

# Apply k-medoids clustering (k=3) set.seed(123)
kmedoids_result <- pam(data, k = 3)

# Visualize the clusters fviz_cluster(kmedoids_result)
```

OUTPUT:

A plot showing the clustered data with medoid points marked.

RESULT:

The k-medoids clustering algorithm successfully groups the dataset into **k clusters** and correctly identifies the medoids as representative points.

EX.NO: 12

IMPLEMENTATION OF HIERARCHICAL CLUSTERING IN R

AIM:

To perform hierarchical clustering using R and visualize the dendrogram.

PROCEDURE:

1. Load the dataset.
2. Standardize the data if needed.
3. Use the dist() function to calculate the distance matrix.
4. Apply the hclust() function to perform hierarchical clustering.
5. Visualize the dendrogram.

SOURCE CODE:

```
# Load dataset
data <- mtcars

# Standardize data (optional)
data_scaled <- scale(data)

# Compute distance matrix
dist_matrix <- dist(data_scaled)

# Perform hierarchical clustering
hclust_result <- hclust(dist_matrix, method = "ward.D2")

# Plot the dendrogram
plot(hclust_result)
```

OUTPUT:

A dendrogram representing hierarchical clustering.

RESULT:

The dendrogram visually represents the hierarchical relationship of the data points, and based on the tree, you can cut it at a desired level to form clusters.

EX.NO: 13

IMPLEMENTATION OF DATA VISUALIZATION IN R

AIM:

To visualize data in R using various plotting techniques.

PROCEDURE:

1. Load the necessary libraries (ggplot2 for visualization).
2. Import or create a dataset.
3. Use different functions to create various types of plots (e.g., scatter plot, box plot, histogram).

SOURCE CODE:

```
# Load the necessary library
library(ggplot2)

# Create a sample dataset
data <- data.frame(x = rnorm(100), y = rnorm(100))

# Scatter plot
ggplot(data, aes(x=x, y=y)) + geom_point()

# Box plot
ggplot(data, aes(x = factor(1), y = x)) + geom_boxplot()

# Histogram
ggplot(data, aes(x = x)) + geom_histogram(bins = 20)
```

OUTPUT:

Scatter plot, box plot, and histogram visualizations.

RESULT:

The output will show different visualizations representing the distribution and relationships of the data.

EX.NO: 14

IMPLEMENTATION OF VARIOUS CHARTS IN R

AIM:

To implement various types of charts (bar charts, pie charts, line plots) in R.

PROCEDURE:

1. Load the necessary libraries (ggplot2 for visualizations).
2. Create or import a dataset.
3. Use appropriate functions to create various charts (bar, pie, line).

SOURCE CODE:

```
# Load required library
library(ggplot2)

data <- data.frame(category = c("A", "B", "C", "D"), value = c(23, 45, 12, 30))

# Bar plot
ggplot(data, aes(x = category, y = value)) + geom_bar(stat = "identity")

# Pie chart (using ggplot2)
ggplot(data, aes(x = "", y = value, fill = category)) + geom_bar(stat = "identity", width = 1) +
coord_polar(theta = "y")

# Line plot (example) time <- 1:10 value
<- c(2, 4, 5, 6, 8, 10, 11, 12, 13, 15)
data_line <- data.frame(time, value)
ggplot(data_line, aes(x = time, y = value)) + geom_line()
```

OUTPUT:

K-means clustering results, including cluster centers.

RESULT:

Thus, the three different clusters are created with the helpof k-medoids clustering algorithm.

EX.NO: 15

IMPLEMENTATION OF K-MEANS IN R

AIM:

To implement the K-means clustering algorithm.

PROCEDURE:

o Use kmeans() to perform clustering on the data. o

Plot the clusters.

SOURCE CODE:

```
# Sample data

data(iris)


# K-means clustering

set.seed(10)

clusters <- kmeans(iris[, 1:4], centers = 3)


# Print clustering results

print(clusters)
```

OUTPUT:

K-means clustering results, including cluster centers.

RESULT:

Thus, the three different clusters are created with the helpof k-medoids clustering algorithm.