R Practical

Q1. Data Pre-processing Write a R program to find all null

#values in a given data set and remove them. (Download dataset from github.com)

Start ->

 # install library(tidyverse)

```
url <- "https://raw.githubusercontent.com/suneet10/DataPreprocessing/main/Data.csv"
dataset <- read.csv(url)
cat("Original Dataset:\n")
print(dataset)
cat("\nCount of null values in each column:\n")
null_values <- sapply(dataset, function(x) sum(is.na(x)))
print(null_values)
cleaned_dataset <- na.omit(dataset)
cat("\nDataset after removing null values:\n")
print(cleaned_dataset)
```

Q2. Write a python program to implement complete data pre-

#processing in a given data set.(missing value, encoding categorical value, Splitting the dataset into the training and test sets and feature scaling.(Download dataset from github.com).

start ->

```
# library(caTools)
# library(dplyr)
# library(caret)

    url <- "https://raw.githubusercontent.com/suneet10/DataPreprocessing/main/Data.csv"
    dataset <- read.csv(url)
```

```
cat("Display Dataset:\n")
print(head(dataset))

cat("Handling missing values and replace with mean of column:\n")
        dataset$Age <- ifelse(is.na(dataset$Age),
                ave(dataset$Age, FUN = function(x) mean(x, na.rm = TRUE)),
                dataset$Age)
        dataset$Salary <- ifelse(is.na(dataset$Salary),
                ave(dataset$Salary, FUN = function(x) mean(x, na.rm = TRUE)),
                dataset$Salary)

cat("\nEncoding categorical values into numerical values:\n")
dataset$Country <- as.factor(dataset$Country)
dataset$Purchased <- factor(dataset$Purchased, levels = c('No', 'Yes'))

cat("\nDisplay Column names in dataset:\n")
print(colnames(dataset))

cat("\nSplit dataset into Training and Test dataset:\n")
set.seed(123)
if ("Purchased" %in% colnames(dataset)) {
split <- sample.split(dataset$Purchased, SplitRatio = 0.8)
training_set <- subset(dataset, split == TRUE)
test_set <- subset(dataset, split == FALSE)

cat("\nTraining Set:\n")
print(head(training_set))

cat("\nTest Set:\n")
print(head(test_set))
```

```
cat("\nApplying feature scaling:\n")
training_set[, c('Age', 'Salary')] <- scale(training_set[, c('Age', 'Salary')])
test_set[, c('Age', 'Salary')] <- scale(test_set[, c('Age', 'Salary')])
cat("\nTraining Set after scaling:\n")
print(head(training_set))

cat("\nTest Set after scaling:\n")
print(head(test_set))

cat("\nSave Processed Data as CSV Files:\n")
write.csv(training_set, "training_set.csv", row.names = FALSE)
write.csv(test_set, "test_set.csv", row.names = FALSE)
} else {
cat("\nError: 'Purchased' column not found in the dataset!\n")
}
```

Q4. Consider following dataset weather=['Sunny','Sunny','Overcast','Rainy','Rainy','Rainy','Over cast','Sunny','Sunny','Rainy','Sunny','Overcast','Overcast','Rainy']temp=['Hot','Hot','Hot','Mild','Cool',' Cool','Cool','Mild','Cool','Mild','Mild','Mild','Hot','Mild']
play=['No','No','Yes','Yes','Yes','No','Yes','No','Yes','Yes','Yes','Yes','Yes','No']. Use Naïve Bayes algorithm to predict[0:Overcast, 2:Mild] tuple belongs to which class whether to play the sports or not. (Using R Studio)

Start ->

```
# library(e1071)

weather <- c('Sunny', 'Sunny', 'Overcast', 'Rainy', 'Rainy', 'Rainy', 'Overcast',
             'Sunny', 'Sunny', 'Rainy', 'Sunny', 'Overcast', 'Overcast', 'Rainy')
temp <- c('Hot', 'Hot', 'Hot', 'Mild', 'Cool', 'Cool', 'Cool', 'Mild', 'Cool',
          'Mild', 'Mild', 'Mild', 'Hot', 'Mild')
play <- c('No', 'No', 'Yes', 'Yes', 'Yes', 'No', 'Yes', 'No', 'Yes', 'Yes',
          'Yes', 'Yes', 'Yes', 'No')
```

```
data <- data.frame(Weather = weather, Temperature = temp, Play = play)

model <- naiveBayes(Play ~ Weather + Temperature, data = data)

new_data <- data.frame(Weather = 'Overcast', Temperature = 'Mild')

prediction <- predict(model, new_data)

cat("Prediction for Weather = Overcast and Temperature = Mild:\n")
cat("Play =", prediction, "\n")
cat("Play =", as.character(prediction), "\n")

View(data)
```

Q5. Association Rules Write a R Programme to read the dataset (“Iris.csv”). dataset download from (https://archive.ics.uci.edu/ml/datasets/iris) and apply Apriori algorithm.

Start ->

```
# library(arules)
# library(arulesViz)
url <- "https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data"
iris_data <- read.csv(url, header = FALSE)
colnames(iris_data) <- c("Sepal.Length", "Sepal.Width", "Petal.Length", "Petal.Width", "Species")
iris_data$Sepal.Length <- cut(iris_data$Sepal.Length, breaks = 3, labels = c("Short", "Medium", "Long"))
iris_data$Sepal.Width <- cut(iris_data$Sepal.Width, breaks = 3, labels = c("Narrow", "Medium", "Wide"))
iris_data$Petal.Length <- cut(iris_data$Petal.Length, breaks = 3, labels = c("Short", "Medium", "Long"))
iris_data$Petal.Width <- cut(iris_data$Petal.Width, breaks = 3, labels = c("Narrow", "Medium", "Wide"))
iris_transactions <- as(iris_data, "transactions")
```

```
rules <- apriori(iris_transactions, parameter = list(supp = 0.2, conf = 0.8))

inspect(rules)

plot(rules, method = "graph", control = list(type = "items"))

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

inspect(top_rules[1:5])
```

Q6. Write a R program to read “StudentsPerformance.csv” file.Solve following:- To display the shape of dataset.To display the top rows of the dataset with their columns.To display the number of rows randomly.

```
#To display the number of columns and names of the columns.Note: Download dataset from following link :(https://www.kaggle.com/spscientist/students-performance-in-

#exams?select=StudentsPerformance.csv) (Add External student perfom.file.csv)
```

Start ->

```
# library(dplyr)

# library(readr)


#url <- "https://www.kaggle.com/spscientist/students-performance-in-“

#url <- “https://www.kaggle.com/datasets/spscientist/students-performance-in-exams”

#url <- "https://www.kaggle.com/datasets/spscientist/students-performance-in-exams"


dataset <- read_csv("StudentsPerformance.csv")

cat("Shape of the dataset:\n")

cat("Number of rows: ", nrow(dataset), "\n")

cat("Number of columns: ", ncol(dataset), "\n")

cat("\nTop rows of the dataset:\n")

print(head(dataset))

set.seed(123)

cat("\nRandom sample of rows:\n")

random_rows <- dataset %>% sample_n(5) # Display 5 random rows

print(random_rows)

cat("\nNumber of columns: ", ncol(dataset), "\n")

cat("Names of the columns:\n")
```

```
print(colnames(dataset))
```

Q7. Regression Analysis and Outlier Detection Consider following observations/data. And apply simple linear regression and find out estimated coefficients b1 and b1 Also analyse the performance of the model

#(Use sklearn package) x = np.array([1,2,3,4,5,6,7,8]) y = np.array([7,14,15,18,19,21,26,23])

Start ->

```
# install library(ggplot2)


x <- c(1, 2, 3, 4, 5, 6, 7, 8)

y <- c(7, 14, 15, 18, 19, 21, 26, 23)

data <- data.frame(x = x, y = y)

model <- lm(y ~ x, data = data)

summary(model)

coefficients <- coef(model)

b0 <- coefficients["(Intercept)"]

b1 <- coefficients["x"]

cat("Estimated coefficients:\n")

cat("Intercept (b0): ", b0, "\n")

cat("Slope (b1): ", b1, "\n")

data$predicted <- predict(model, data)

data$residuals <- data$y - data$predicted

cat("\nResiduals:\n")

print(data$residuals)

ggplot(data, aes(x = x, y = y)) +

  geom_point(color = "blue") +
```

```
geom_smooth(method = "lm", color = "red") +
labs(title = "Simple Linear Regression",
    x = "X",
    y = "Y") +
theme_minimal()
```