

SAVEETHA SCHOOL OF ENGINEERING
SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES
ITA 0451 - STATISTICS WITH R PROGRAMMING
DAY 4 – LAB ASSESSMENT Part 3

Reg No:192125001

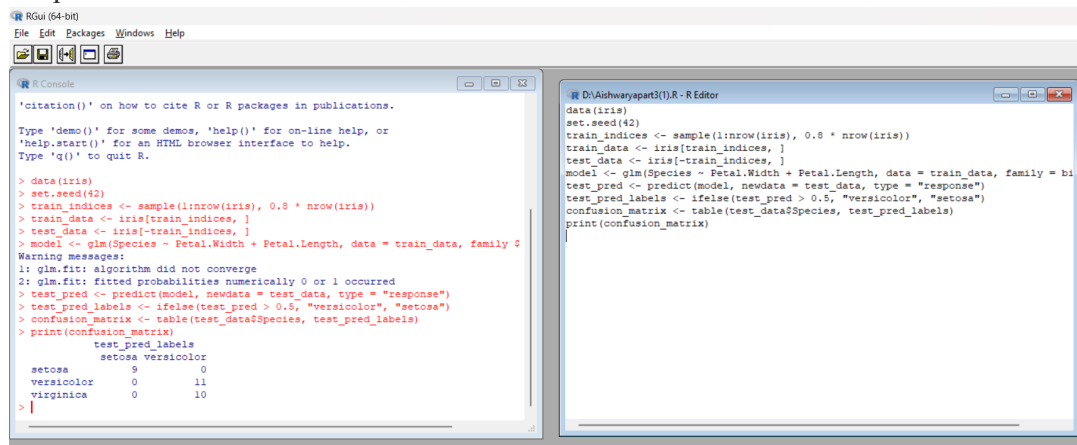
Name:R.Aishwarya

1. Randomly Sample the iris dataset such as 80% data for training and 20% for test and create Logistics regression with train data, use species as target and petals width and length as feature variables, Predict the probability of the model using test data, Create Confusion matrix for above test model

Code:

```
data(iris)
set.seed(42)
train_indices <- sample(1:nrow(iris), 0.8 * nrow(iris))
train_data <- iris[train_indices, ]
test_data <- iris[-train_indices, ]
model <- glm(Species ~ Petal.Width + Petal.Length, data = train_data, family = binomial)
test_pred <- predict(model, newdata = test_data, type = "response")
test_pred_labels <- ifelse(test_pred > 0.5, "versicolor", "setosa")
confusion_matrix <- table(test_data$Species, test_pred_labels)
print(confusion_matrix)
```

Output:



2. (i) Write suitable R code to compute the mean, median, mode of the following values
c(90, 50, 70, 80, 70, 60, 20, 30, 80, 90, 20)

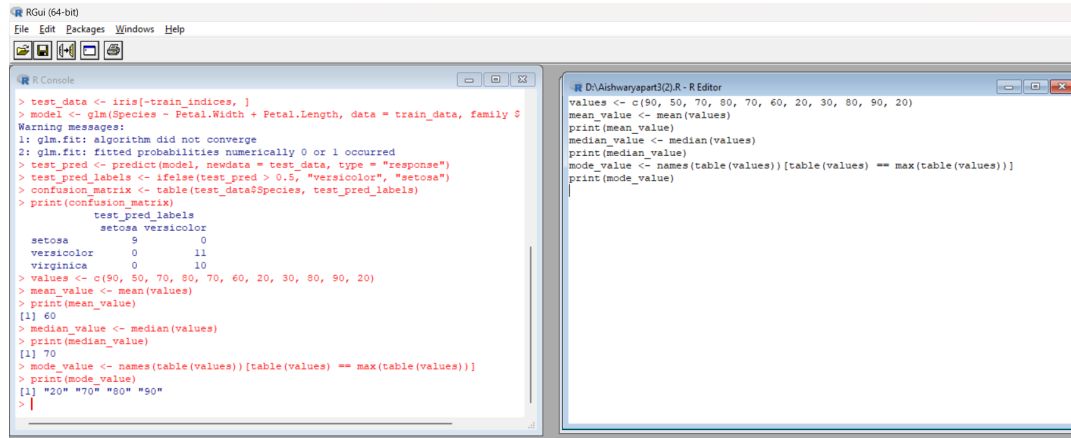
(ii) Write R code to find 2nd highest and 3rd Lowest value of above problem.

Code:

```
values <- c(90, 50, 70, 80, 70, 60, 20, 30, 80, 90, 20)
mean_value <- mean(values)
print(mean_value)
median_value <- median(values)
print(median_value)
```

```
mode_value <- names(table(values))[table(values) == max(table(values))]
print(mode_value)
```

Output:



The screenshot shows two windows from the R environment. The left window, titled 'RGui (64-bit)', contains the R Console with the following code and output:

```
> test_data <- iris[train_indices, ]
> model <- glm(Species ~ Petal.Width + Petal.Length, data = train_data, family = "binomial")
Warning messages:
1: glm.fit: algorithm did not converge
2: glm.fit: fitted probabilities numerically 0 or 1 occurred
> test_pred <- predict(model, newdata = test_data, type = "response")
> test_pred_labels <- ifelse(test_pred > 0.5, "versicolor", "setosa")
> confusion_matrix <- table(test_data$Species, test_pred_labels)
> print(confusion_matrix)
      test_pred_labels
      setosa versicolor
setosa      9         0
versicolor  0        11
virginica   0        10
> values <- c(90, 50, 70, 80, 70, 60, 20, 30, 80, 90, 20)
> mean_value <- mean(values)
> print(mean_value)
[1] 60
> median_value <- median(values)
> print(median_value)
[1] 70
> mode_value <- names(table(values))[table(values) == max(table(values))]
> print(mode_value)
[1] "20" "70" "80" "90"
```

The right window, titled 'D:\Aishwaryapart3\2\R - R Editor', contains the following code:

```
values <- c(90, 50, 70, 80, 70, 60, 20, 30, 80, 90, 20)
mean_value <- mean(values)
print(mean_value)
median_value <- median(values)
print(median_value)
mode_value <- names(table(values))[table(values) == max(table(values))]
print(mode_value)
```

3. Explore the airquality dataset. It contains daily air quality measurements from New York during a period of five months:

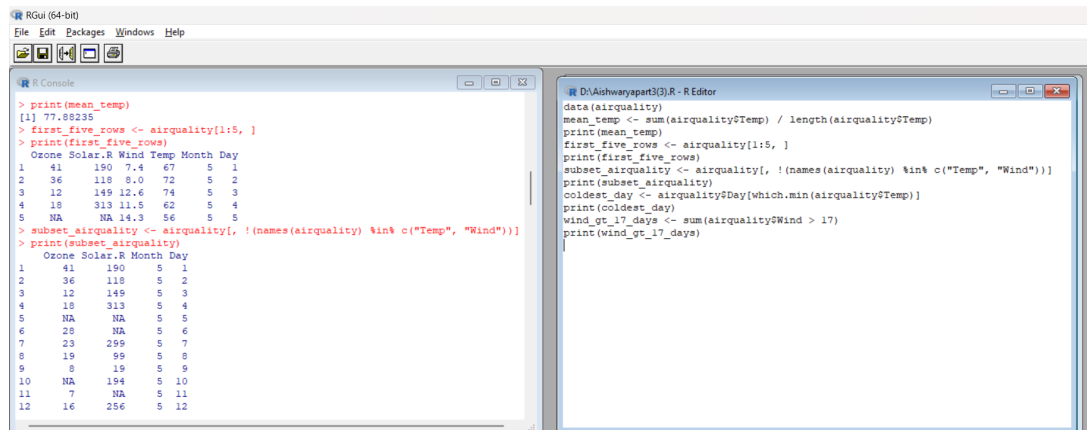
- Ozone: mean ozone concentration (ppb),
- Solar.R: solar radiation (Langley),
- Wind: average wind speed (mph),
- Temp: maximum daily temperature in degrees Fahrenheit,
- Month: numeric month (May=5, June=6, and so on),
- Day: numeric day of the month (1-4).

- i. Compute the mean temperature(don't use build in function)
- ii.Extract the first five rows from airquality.
- iii.Extract all columns from airquality except Temp and Wind
- iv.Which was the coldest day during the period?
- v.How many days was the wind speed greater than 17 mph?

Code:

```
data(airquality)
mean_temp <- sum(airquality$Temp) / length(airquality$Temp)
print(mean_temp)
first_five_rows <- airquality[1:5, ]
print(first_five_rows)
subset_airquality <- airquality[, !(names(airquality) %in% c("Temp", "Wind"))]
print(subset_airquality)
coldest_day <- airquality$Day[which.min(airquality$Temp)]
print(coldest_day)
wind_gt_17_days <- sum(airquality$Wind > 17)
print(wind_gt_17_days)
```

Output:



4. (i) Get the Summary Statistics of air quality dataset
 - (ii) Melt airquality data set and display as a long – format data?
 - (iii) Melt airquality data and specify month and day to be “ID variables”?
 - (iv) Cast the molten airquality data set with respect to month and date features
 - (v) Use cast function appropriately and compute the average of Ozone, Solar.R, Wind and temperature per month?

Code:

```
data(airquality)
library(reshape2)
summary_stats <- summary(airquality)
print(summary_stats)
melted_data <- melt(airquality)
print(melted_data)
melted_data_id <- melt(airquality, id.vars = c("Month", "Day"))
print(melted_data_id)
casted_data <- dcast(melted_data_id, Month + Day ~ variable)
print(casted_data)
average_data <- dcast(melted_data_id, Month ~ variable, mean)
print(average_data)
```

5. (i) Find any missing values (na) in features and drop the missing values if its less than 10% else replace that with mean of that feature.
 - (ii) Apply a linear regression algorithm using Least Squares Method on “Ozone” and “Solar.R”
 - (iii) Plot Scatter plot between Ozone and Solar and add regression line created by above model.

Code:

```
data(airquality)
missing_values <- sapply(airquality, function(x) sum(is.na(x)))
missing_cols <- names(missing_values[missing_values > 0])
```

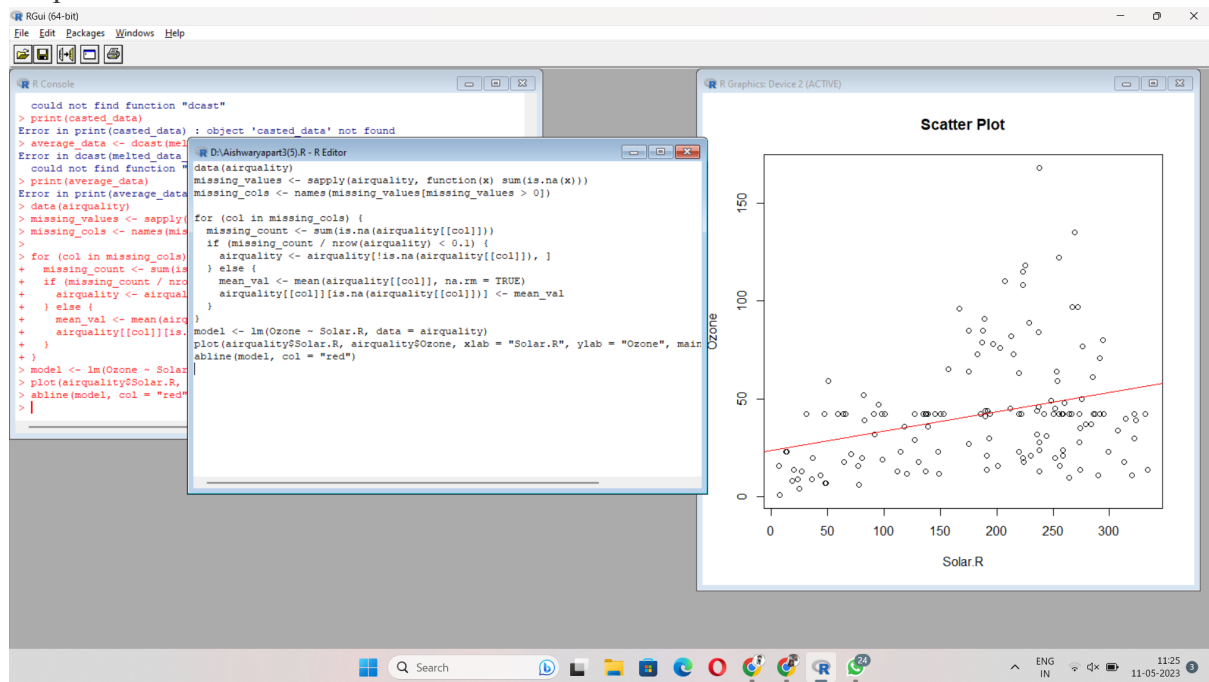
```
for (col in missing_cols) {
  missing_count <- sum(is.na(airquality[[col]]))
  if (missing_count / nrow(airquality) < 0.1) {
    airquality <- airquality[!is.na(airquality[[col]]), ]
  } else {
    mean_val <- mean(airquality[[col]], na.rm = TRUE)
```

```

    airquality[[col]][is.na(airquality[[col]])] <- mean_val
  }
}
model <- lm(Ozone ~ Solar.R, data = airquality)
plot(airquality$Solar.R, airquality$Ozone, xlab = "Solar.R", ylab = "Ozone", main = "Scatter Plot")
abline(model, col = "red")

```

Output:



6. Load dataset named ChickWeight,

- (i). Order the data frame, in ascending order by feature name "weight" grouped by feature "diet" and Extract the last 6 records from order data frame.
- (ii). a. Perform melting function based on "Chick", "Time", "Diet" features as ID variables
 - b. Perform cast function to display the mean value of weight grouped by Diet
 - c. Perform cast function to display the mode of weight grouped by Diet

Code:

```
data(ChickWeight)
```

```
library(reshape2)
```

```
ordered_df <- ChickWeight[order(ChickWeight$weight), ]
```

```
last_six_records <- tail(ordered_df, 6)
```

```
print(last_six_records)
```

```

melted_data <- melt(ChickWeight, id.vars = c("Chick", "Time", "Diet"))
casted_mean_data <- dcast(melted_data, Diet ~ variable, mean)
print(casted_mean_data)
casted_mode_data <- dcast(melted_data, Diet ~ variable, function(x) {
  uniq_x <- unique(x)
  uniq_x[which.max(tabulate(match(x, uniq_x)))]
})
print(casted_mode_data)

```

7. a. Create Box plot for “weight” grouped by “Diet”
- b. Create a Histogram for “weight” features belong to Diet- 1 category
- c. Create Scatter plot for “ weight” vs “Time” grouped by Diet

Code:

```
data(ChickWeight)
```

```
library(ggplot2)
```

```
ggplot(ChickWeight, aes(x = Diet, y = weight)) +
```

```
  geom_boxplot() +
```

```
  xlab("Diet") +
```

```
  ylab("Weight") +
```

```
  ggtitle("Box Plot of Weight Grouped by Diet")
```

```
ggplot(ChickWeight[ChickWeight$Diet == 1, ], aes(x = weight)) +
```

```
  geom_histogram() +
```

```
  xlab("Weight") +
```

```
  ylab("Frequency") +
```

```
  ggtitle("Histogram of Weight in Diet-1")
```

```
ggplot(ChickWeight, aes(x = Time, y = weight, color = factor(Diet))) +
```

```
  geom_point() +
```

```
  xlab("Time") +
```

```
  ylab("Weight") +
```

```
ggtitle("Scatter Plot of Weight vs Time Grouped by Diet")
```