**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: 192124133

Name: Bala Manohar. J

**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.**

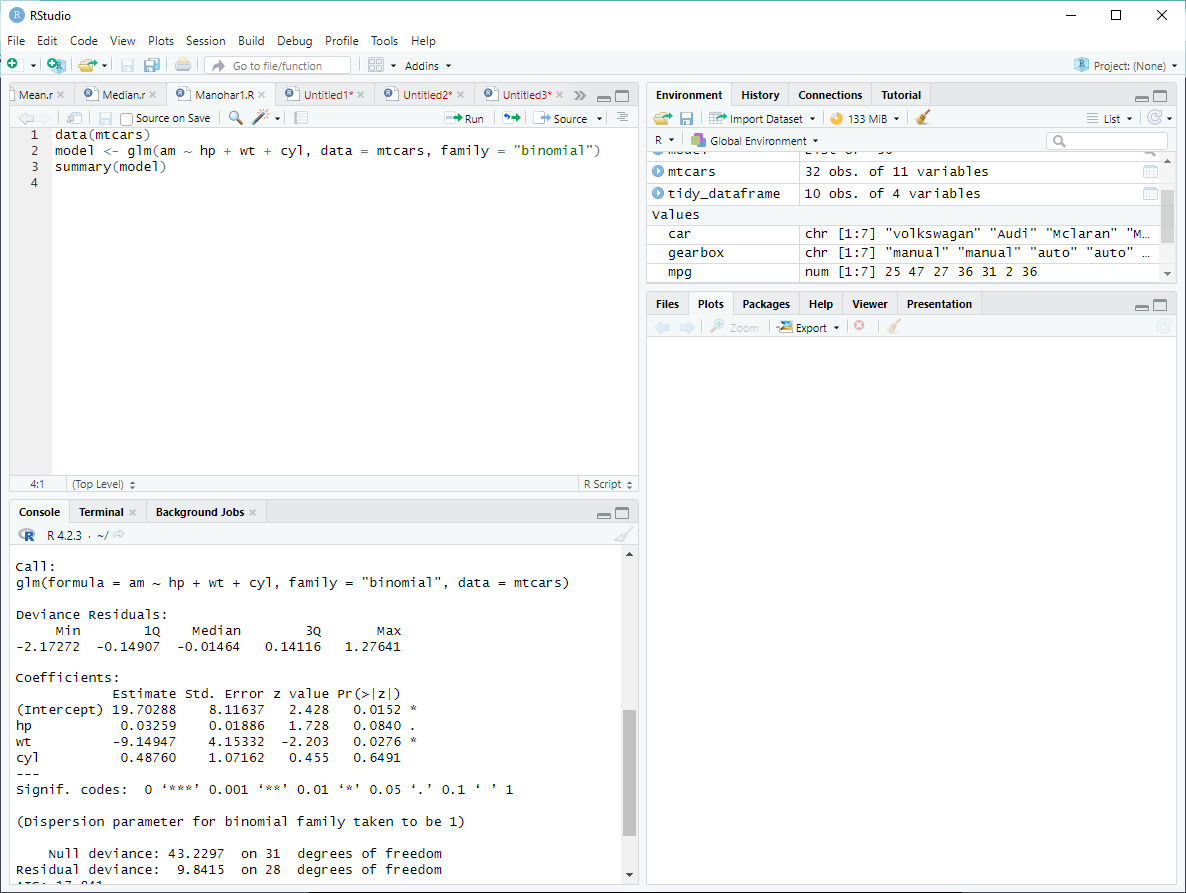
**Source Code:**

data(warpbreaks)

model <- glm(breaks ~ type + tension, data = warpbreaks, family = "poisson")

summary(model)

**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)**

**Source Code:**

# Define the vector

values <- c(90, 50, 70, 80, 70, 60, 20, 30, 80, 90, 20)

# Compute the mean

mean(values)

# Compute the median

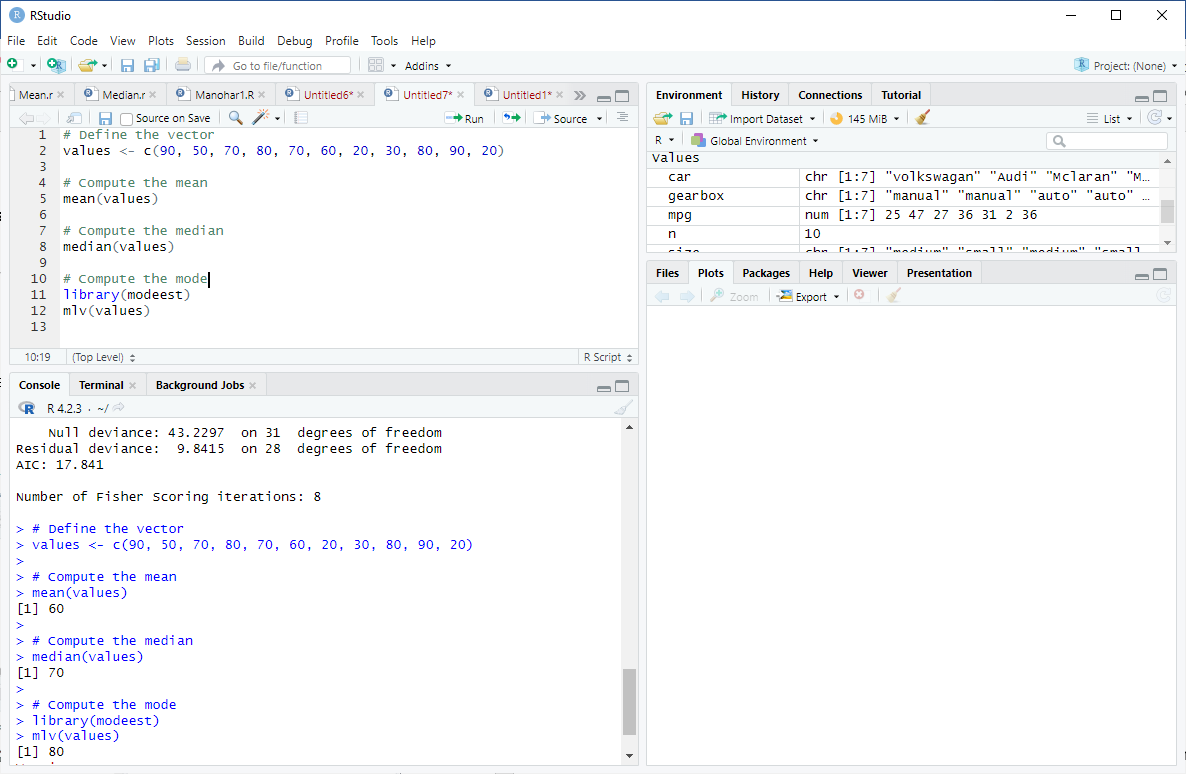
median(values)

# Compute the mode

library(modeest)

mlv(values)

**OUTPUT:**



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

**Source Code:**

# Define the vector

values <- c(90, 50, 70, 80, 70, 60, 20, 30, 80, 90, 20)

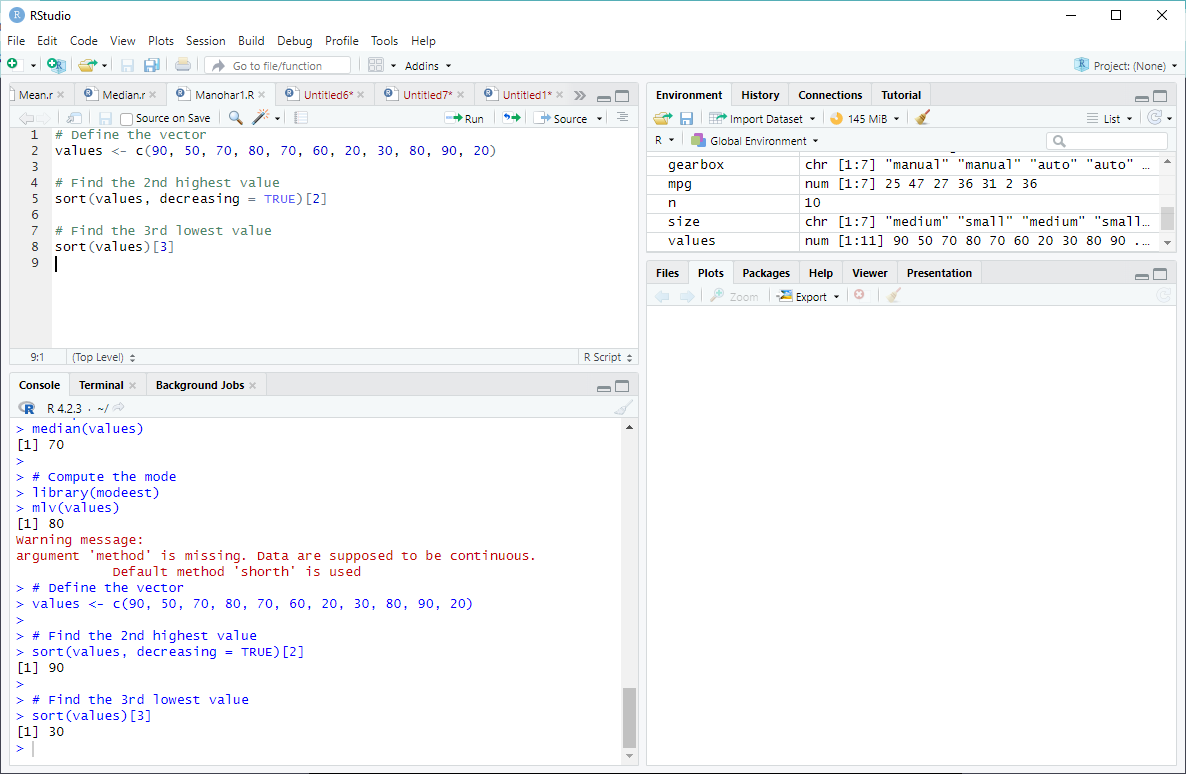
# Find the 2nd highest value

sort(values, decreasing = TRUE)[2]

# Find the 3rd lowest value

sort(values)[3]

**OUTPUT:**



**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)**

**Source Code:**

# Load the dataset

data(airquality)

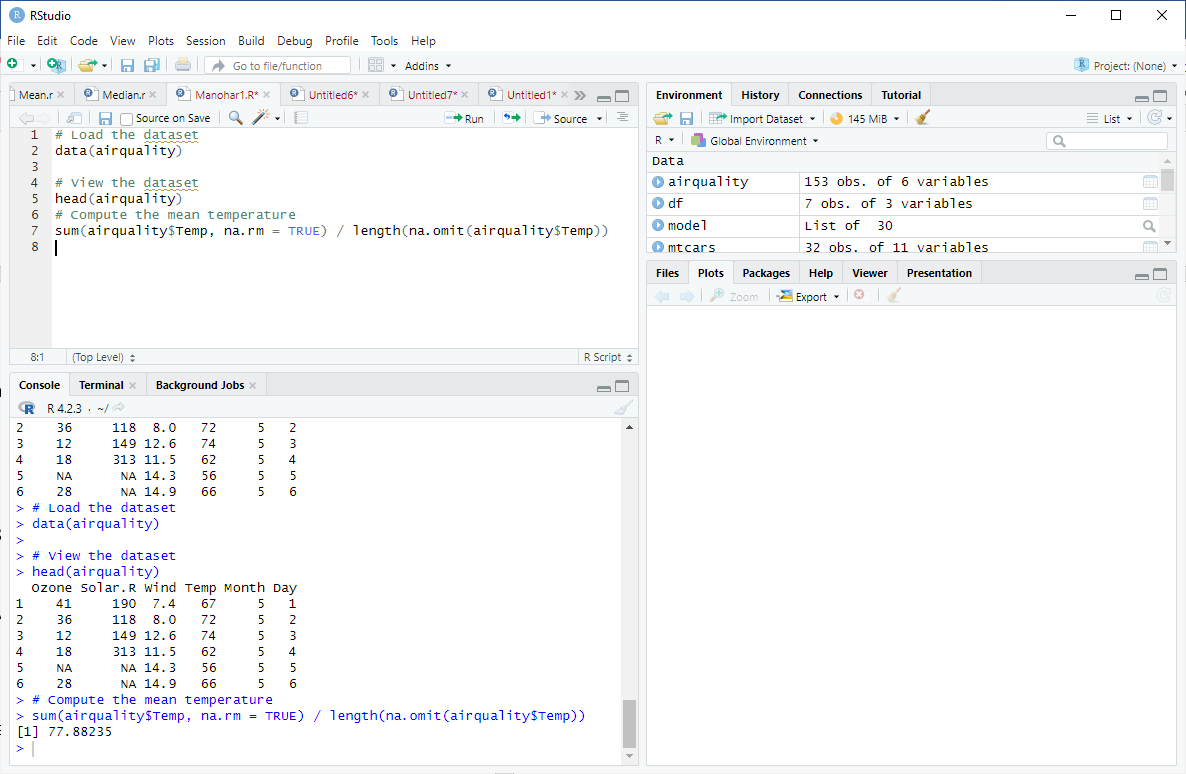
# View the dataset

head(airquality)

# Compute the mean temperature

sum(airquality$Temp, na.rm = TRUE) / length(na.omit(airquality$Temp))

**OUTPUT:**

****

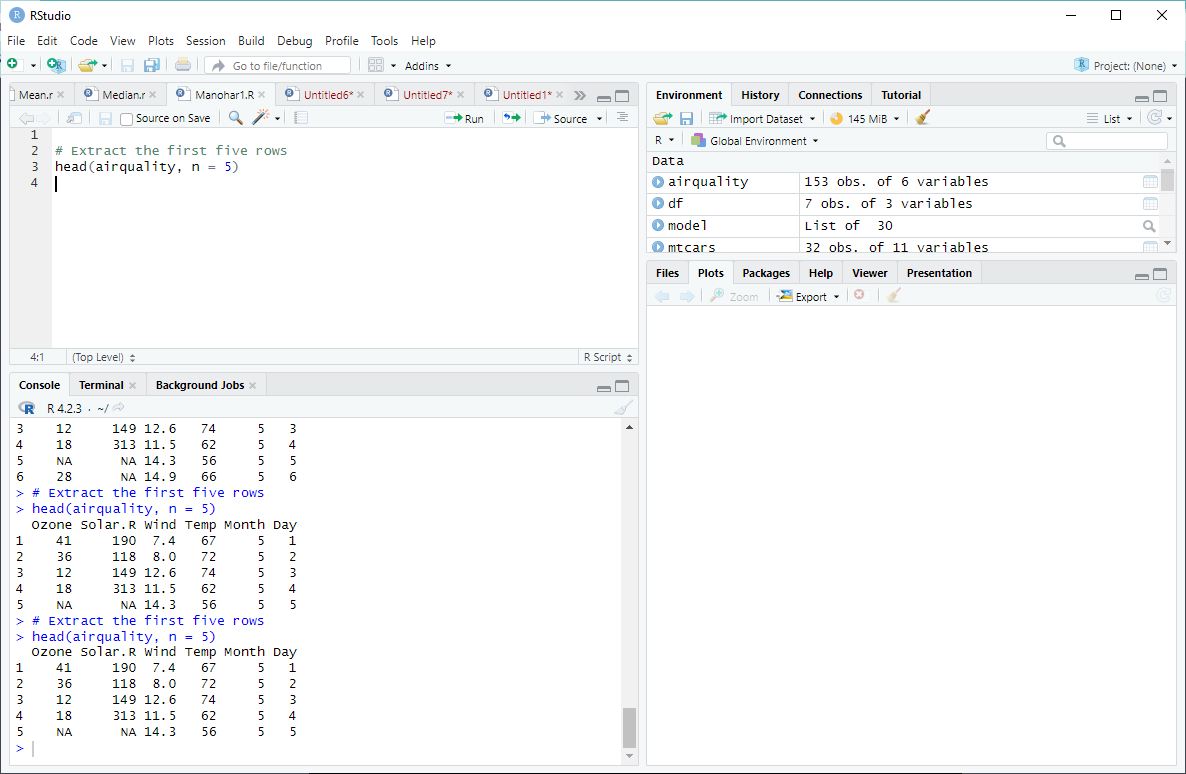
**ii. Extract the first five rows from air quality.**

**Source Code:**

# Extract the first five rows

head(airquality, n = 5)

**OUTPUT:**

****

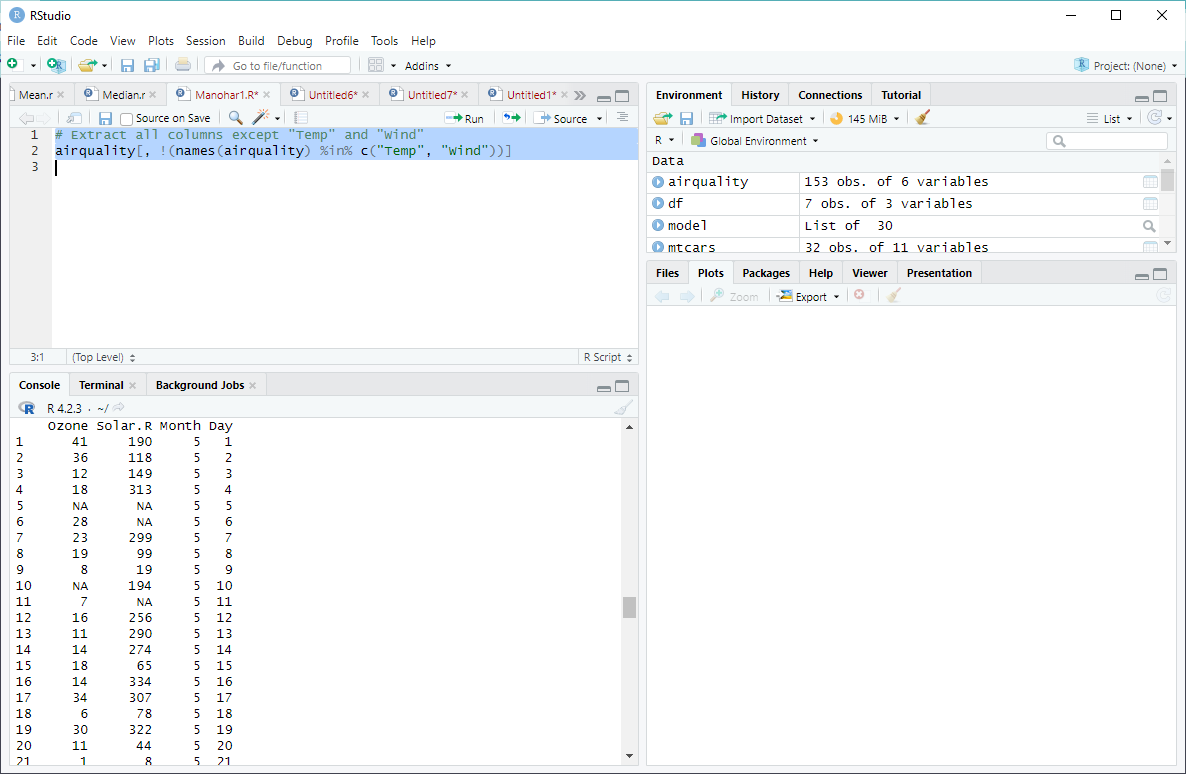
**iii. Extract all columns from air quality except Temp and Wind**

**Source Code:**

# Extract all columns except "Temp" and "Wind"

airquality[, !(names(airquality) %in% c("Temp", "Wind"))]

**OUTPUT:**

****

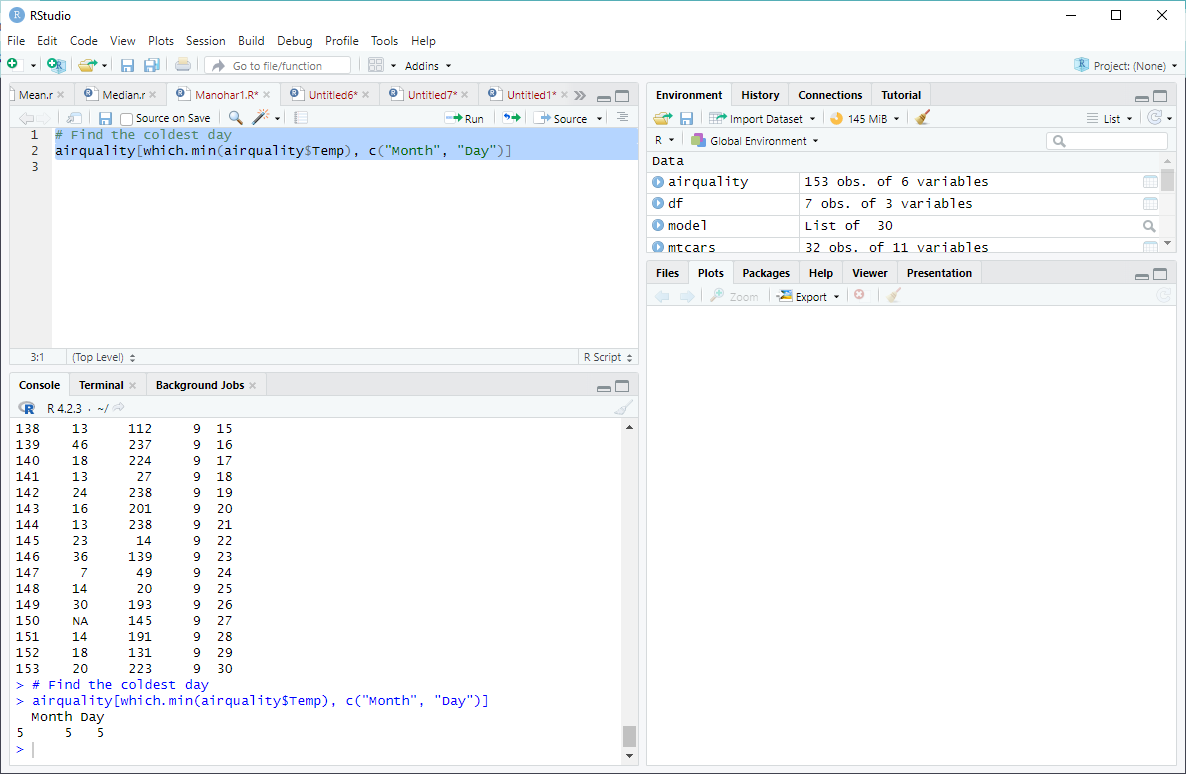
**iv. Which was the coldest day during the period?**

**Source Code:**

# Find the coldest day

airquality[which.min(airquality$Temp), c("Month", "Day")]

**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”?**

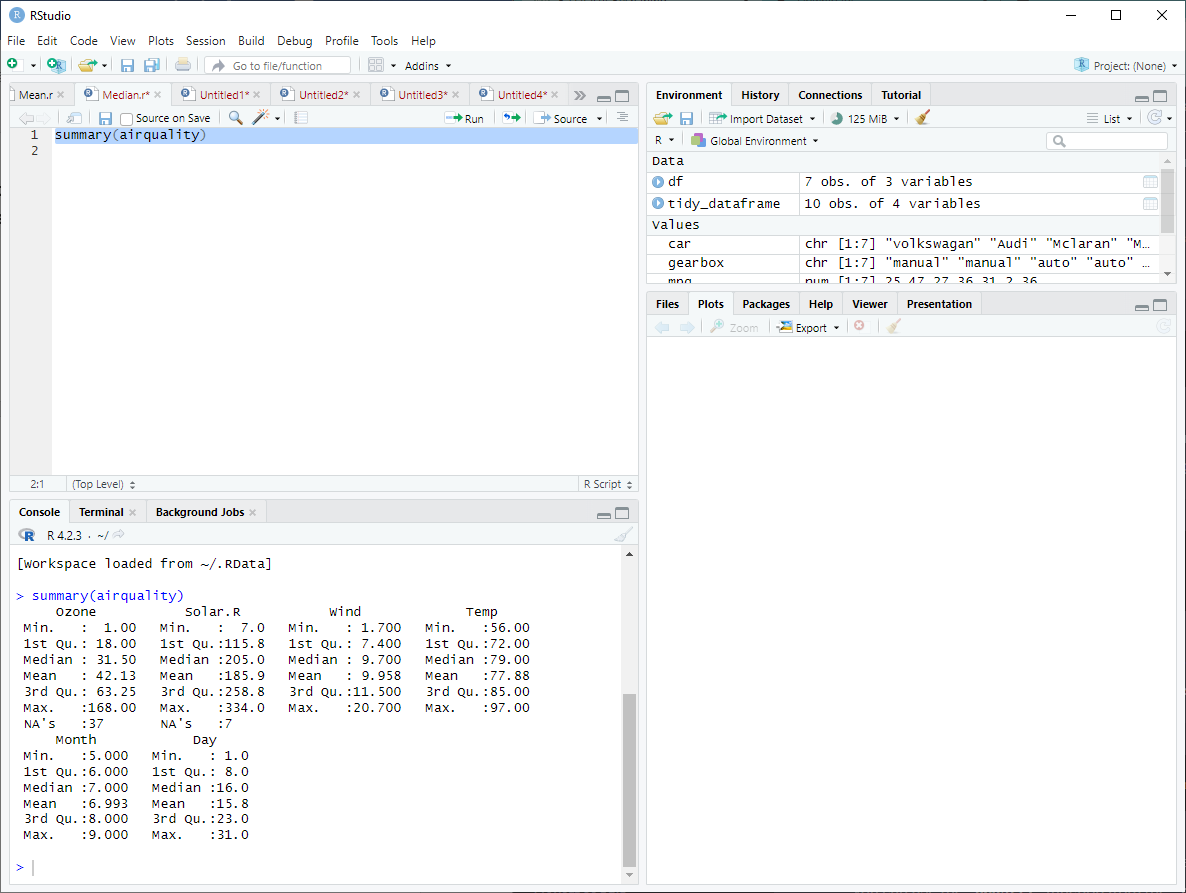
**(iv)Cast the molten airquality data set with respect to month and date features**

**(v) Use cast function appropriately and compute the average of Ozone, Solar.R, Windand temperature per month?**

1. **Source Code:**

summary(airquality)

**OUTPUT:**

****

**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.**

**Source Code:**

library(DescTools)

missing\_percent <- na.percent(airquality)

for (i in 1:length(missing\_percent)) {

if (missing\_percent[i] < 10) {

airquality <- airquality[complete.cases(airquality[, i]), ]

} else {

airquality[, i] <- replace(airquality[, i], is.na(airquality[, i]), mean(airquality[, i], na.rm = TRUE))

}

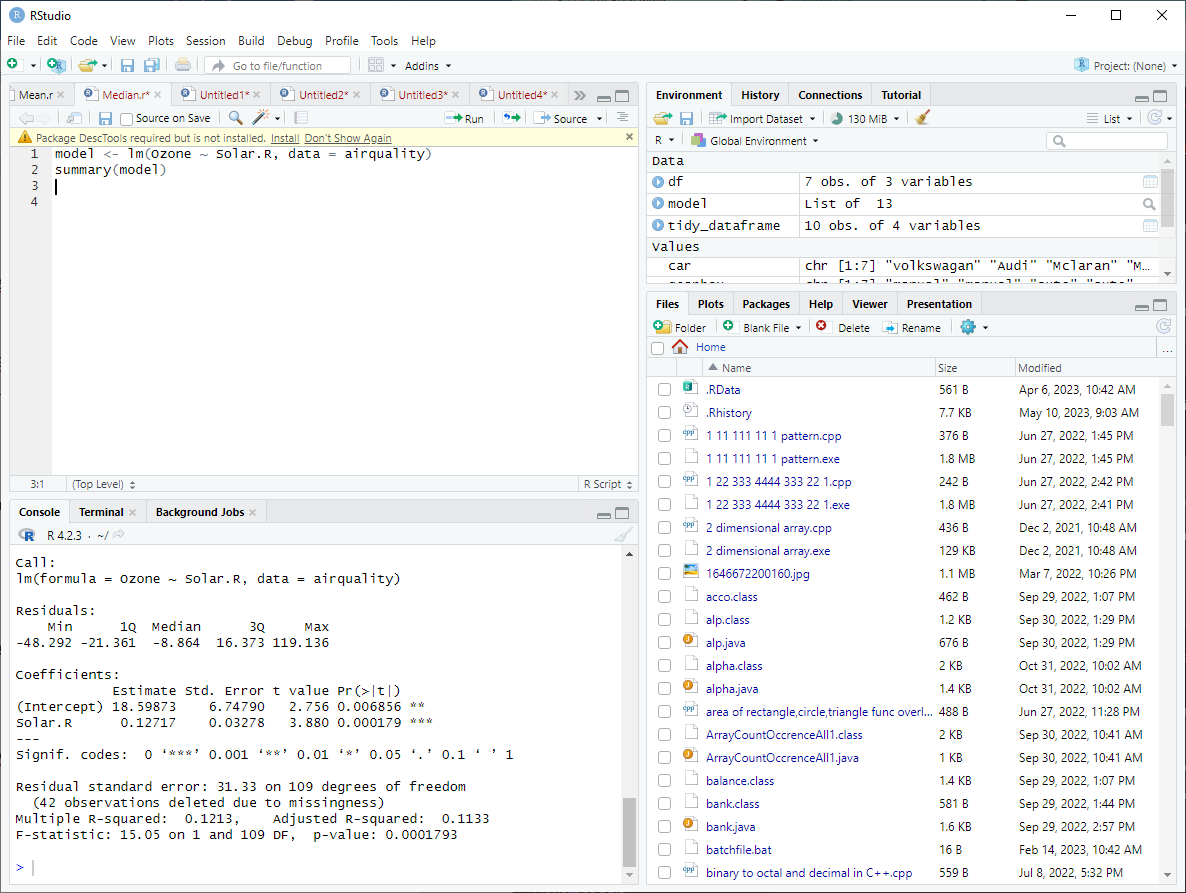
}

1. **Apply a linear regression algorithm using Least Squares Method on “Ozone” and “Solar.R”**

**Source Code:**

model <- lm(Ozone ~ Solar.R, data = airquality)

summary(model)

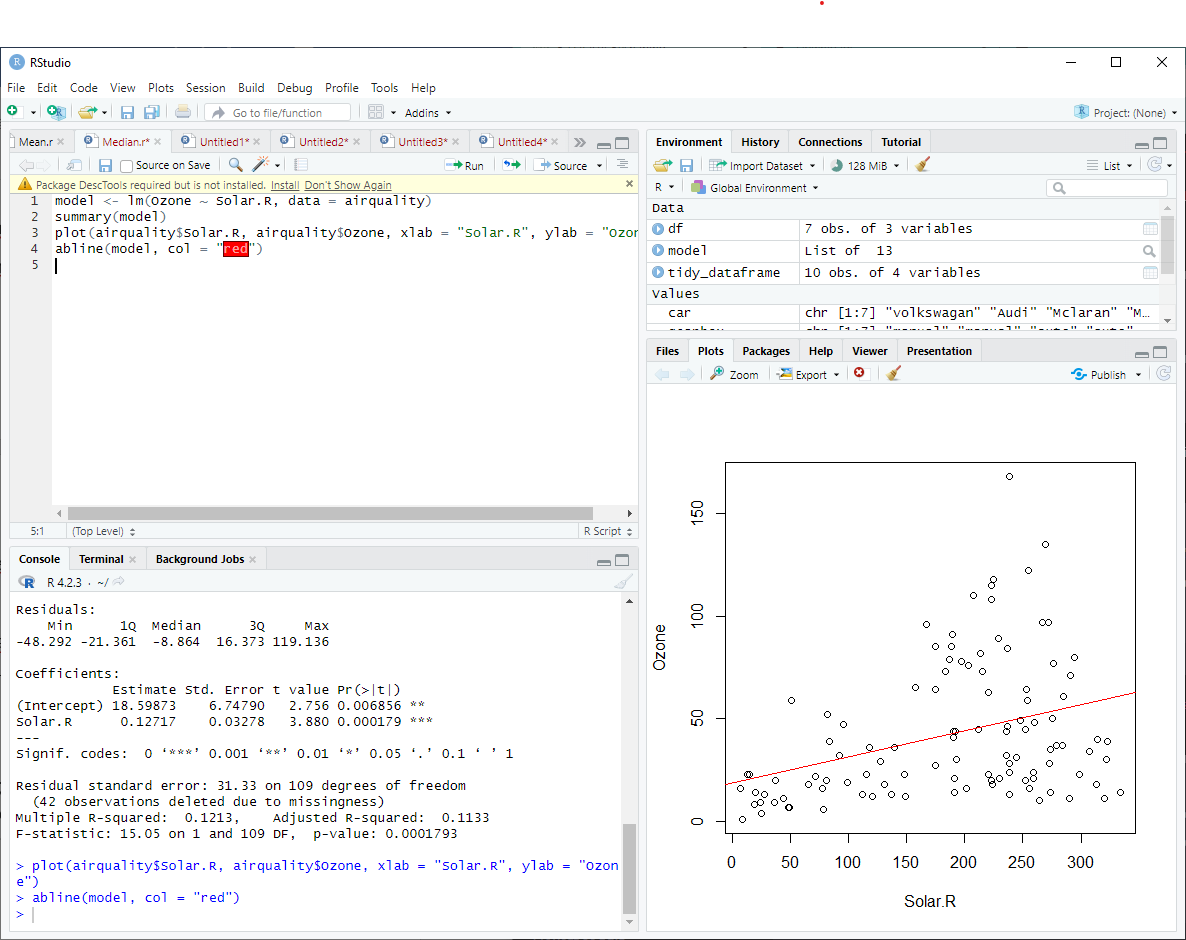
****

1. **Plot Scatter plot between Ozone and Solar and add regression line created by above mode**

**Source Code:**

plot(airquality$Solar.R, airquality$Ozone, xlab = "Solar.R", ylab = "Ozone")

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.**

**Source Code:**

data(ChickWeight)

**(ii).a Perform melting function based on “Chick", "Time", "Diet" features as ID variables**

**Source Code:**

library(dplyr)

ordered\_data <- ChickWeight %>% arrange(diet, weight)

last\_6\_records <- tail(ordered\_data, 6)

**b. Perform cast function to display the mean value of weight grouped by Diet**

**Source Code:**

casted\_data\_mean <- dcast(melted\_data, Diet ~ variable, mean)

**c. Perform cast function to display the mode of weight grouped by Die**

**Source Code:**

mode\_func <- function(x) {

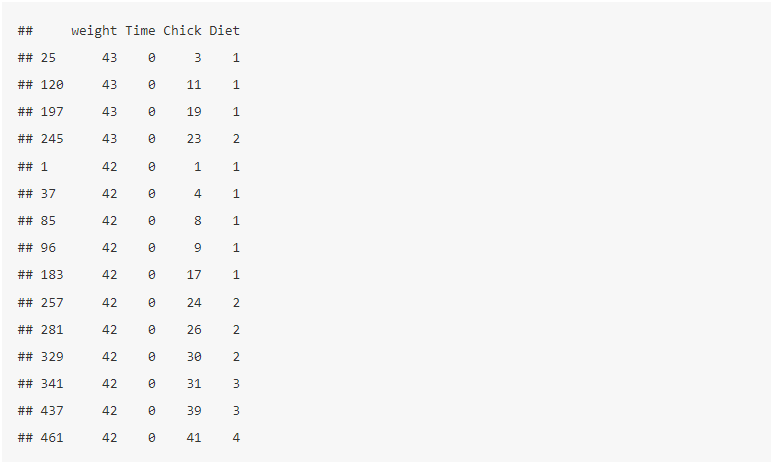
ux <- unique(x)

ux[which.max(tabulate(match(x, ux)))]

}

casted\_data\_mode <- aggregate(ChickWeight$weight, by = list(Diet = ChickWeight$Diet), mode\_func)

**OUTPUT:**

****

**7. a. Create Box plot for “weight” grouped by “Diet”**

**Source Code:**

library(ggplot2)

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

geom\_boxplot() +

labs(x = "Diet", y = "Weight") +

ggtitle("Box plot of Weight by Diet")

**b. Create a Histogram for “weight” features belong to Diet- 1 category**

**Source Code:**

ggplot(subset(ChickWeight, Diet == 1), aes(x = weight)) +

geom\_histogram() +

labs(x = "Weight", y = "Count") +

ggtitle("Histogram of Weight for Diet-1")

**c. Create Scatter plot for “ weight” vs “Time” grouped by Die**

**Source Code:**

ggplot(subset(ChickWeight, Diet == 1), aes(x = weight)) +

geom\_histogram() +

labs(x = "Weight", y = "Count") +

ggtitle("Histogram of Weight for Diet-1")