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: 192110399

Name: Gokul S

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

# Load the iris dataset

data(iris)

# Set seed for reproducibility

set.seed(123)

# Split the data into training and test data

train\_idx <- sample(nrow(iris), nrow(iris)\*0.8)

train\_data <- iris[train\_idx,]

test\_data <- iris[-train\_idx,]

# Create a logistic regression model

logreg\_model <- glm(Species ~ Petal.Length + Petal.Width, data=train\_data, family="binomial")

# Predict the probability of the model using the test data

test\_pred\_prob <- predict(logreg\_model, newdata=test\_data, type="response")

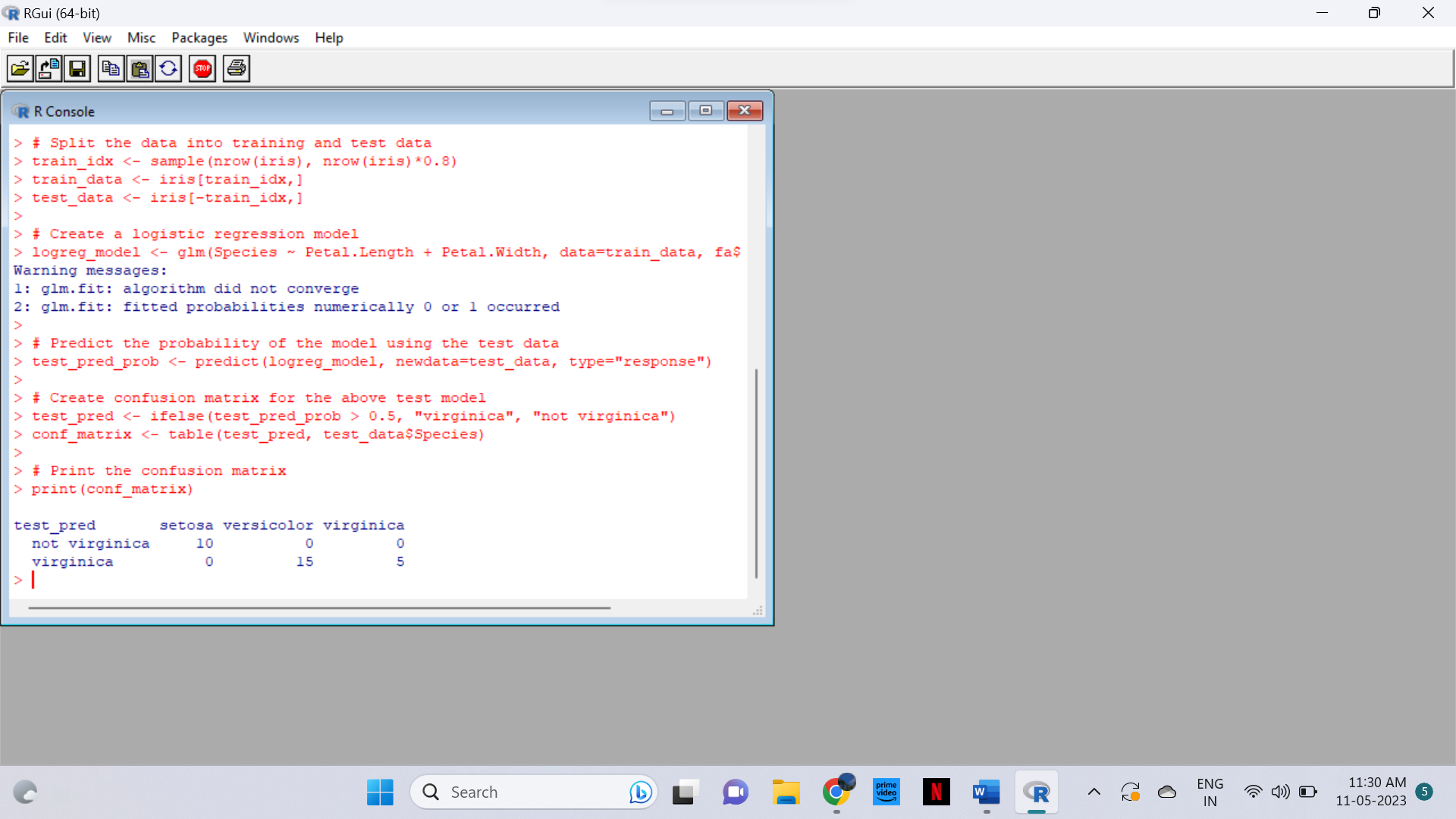
# Create confusion matrix for the above test model

test\_pred <- ifelse(test\_pred\_prob > 0.5, "virginica", "not virginica")

conf\_matrix <- table(test\_pred, test\_data$Species)

# Print the confusion matrix

print(conf\_matrix)



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)

CODE

# Create the vector of values

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

# Compute the mean

mean\_x <- mean(x)

print(paste("Mean:", mean\_x))

# Compute the median

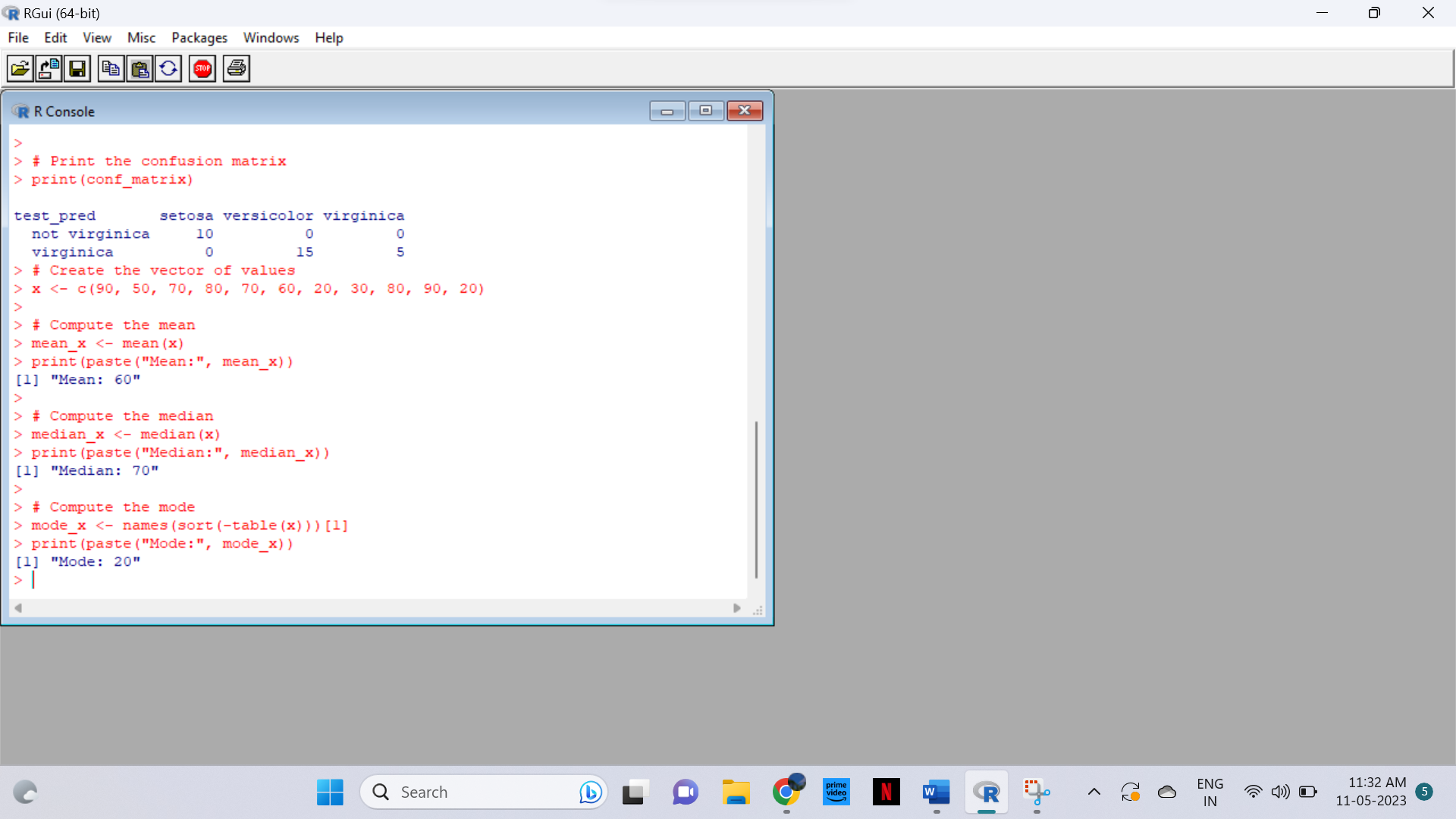
median\_x <- median(x)

print(paste("Median:", median\_x))

# Compute the mode

mode\_x <- names(sort(-table(x)))[1]

print(paste("Mode:", mode\_x))



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

CODE

# Find the 2nd highest value

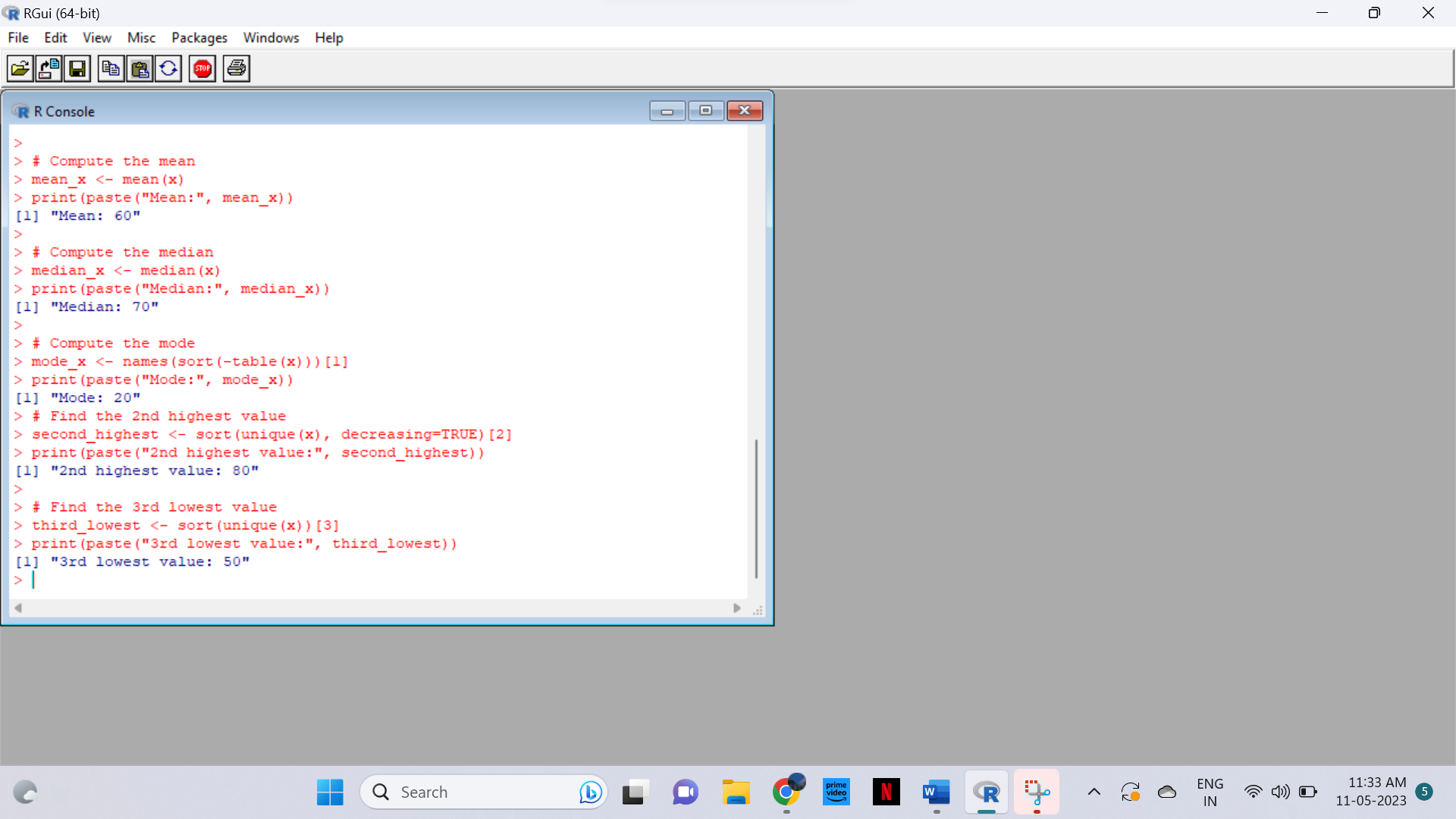
second\_highest <- sort(unique(x), decreasing=TRUE)[2]

print(paste("2nd highest value:", second\_highest))

# Find the 3rd lowest value

third\_lowest <- sort(unique(x))[3]

print(paste("3rd lowest value:", third\_lowest))



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

# Load the airquality dataset

data(airquality)

# i. Compute the mean temperature (without using built-in function)

mean\_temp <- sum(airquality$Temp) / length(airquality$Temp)

print(paste("Mean temperature:", mean\_temp))

# ii. Extract the first five rows

first\_five\_rows <- airquality[1:5, ]

print("First five rows:")

print(first\_five\_rows)

# iii. Extract all columns except Temp and Wind

all\_except\_temp\_wind <- airquality[, !(names(airquality) %in% c("Temp", "Wind"))]

print("All columns except Temp and Wind:")

print(all\_except\_temp\_wind)

# iv. Find the coldest day

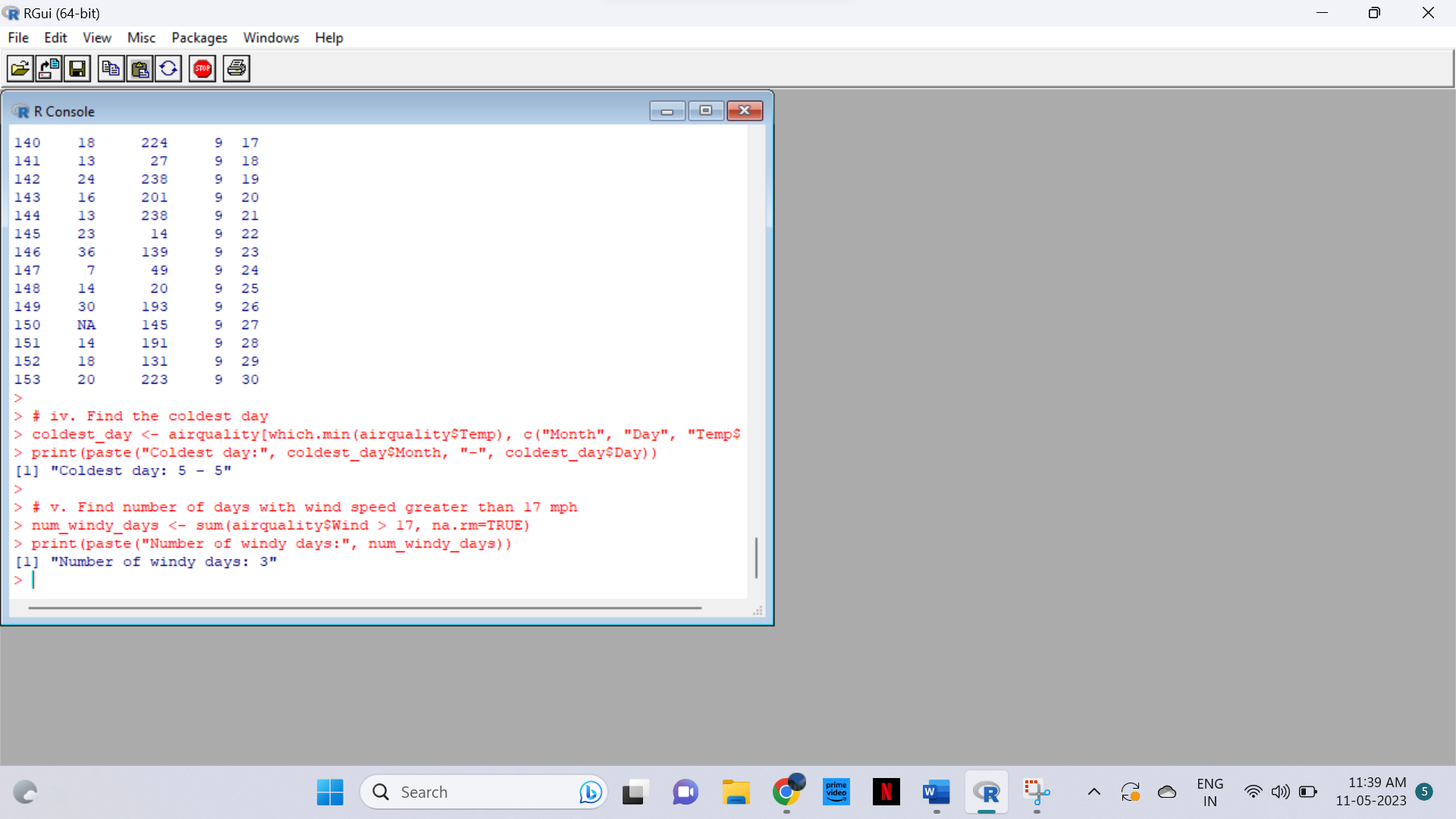
coldest\_day <- airquality[which.min(airquality$Temp), c("Month", "Day", "Temp")]

print(paste("Coldest day:", coldest\_day$Month, "-", coldest\_day$Day))

# v. Find number of days with wind speed greater than 17 mph

num\_windy\_days <- sum(airquality$Wind > 17, na.rm=TRUE)

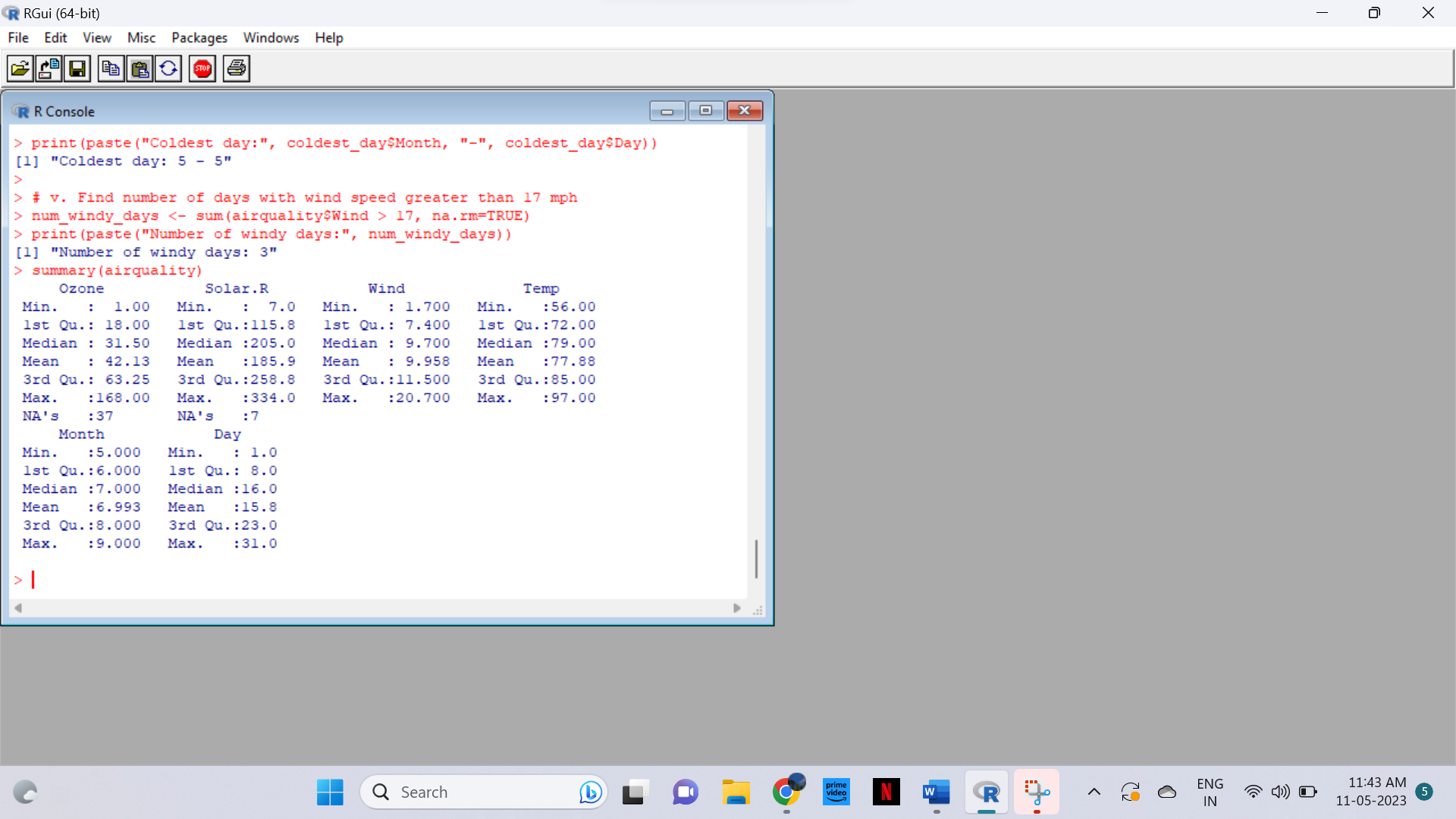
print(paste("Number of windy days:", num\_windy\_days))



4. (i)Get the Summary Statistics of air quality dataset

CODE

summary(airquality)



(ii)Melt airquality data set and display as a long – format data?

CODE

# Load the airquality dataset

data(airquality)

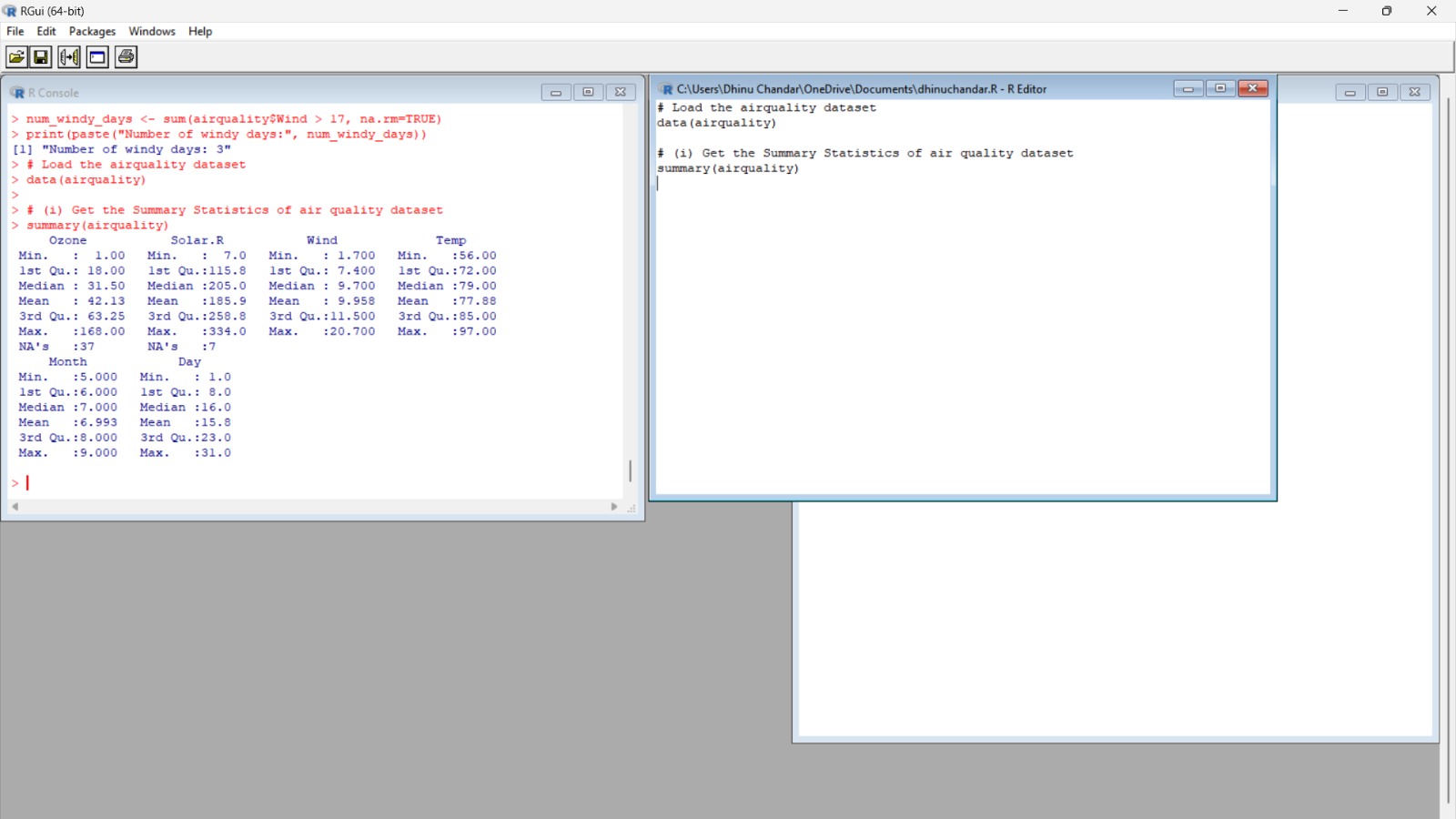
# Get the Summary Statistics of air quality dataset

summary(airquality)

library(reshape2)

melted\_airquality <- melt(airquality)

print(melted\_airquality)



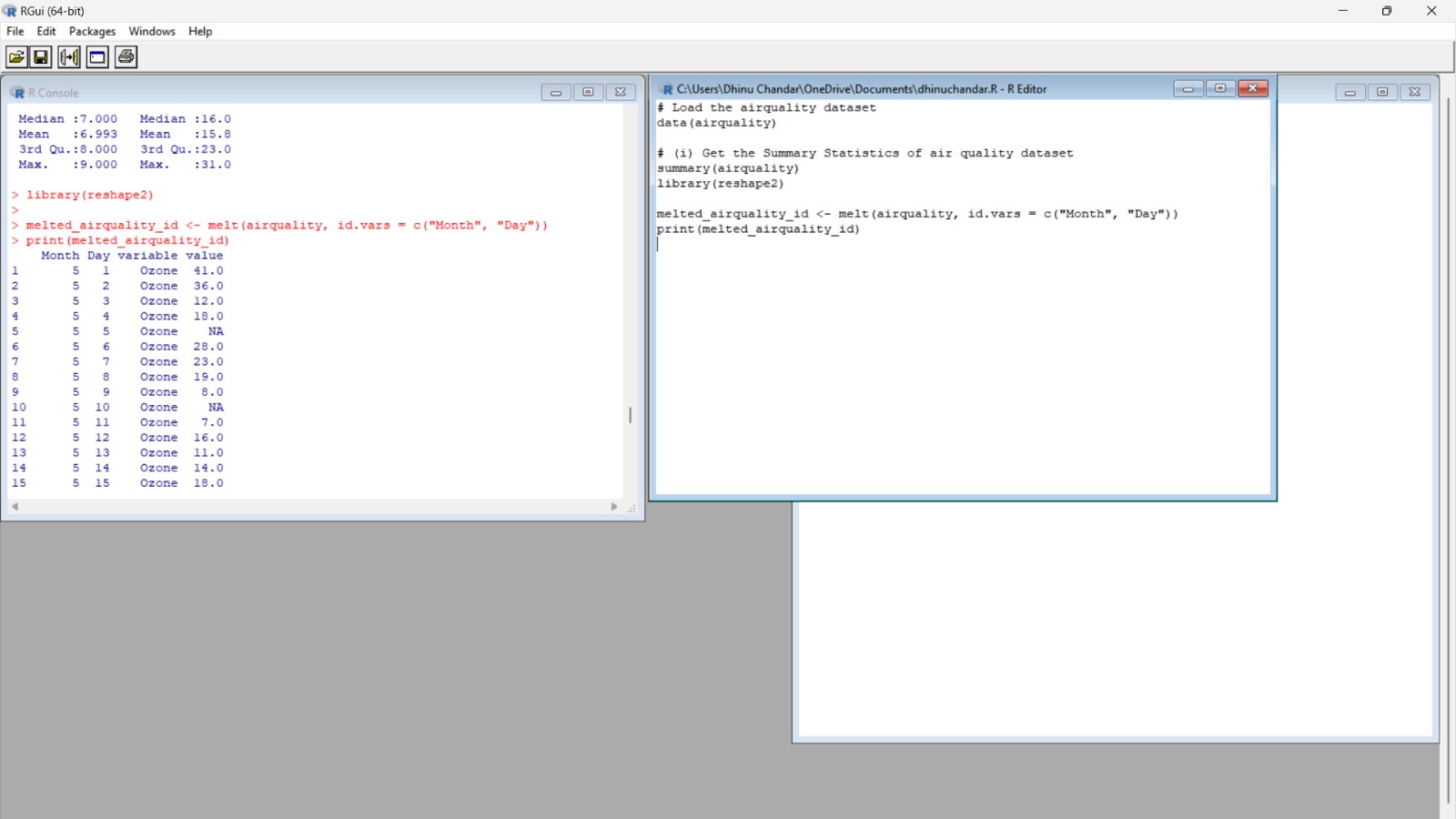
(iii)Melt airquality data and specify month and day to be “ID variables”?

CODE

Melt airquality data and specify month and day to be "ID variables"

melted\_airquality\_id <- melt(airquality, id.vars = c("Month", "Day"))

print(melted\_airquality\_id)



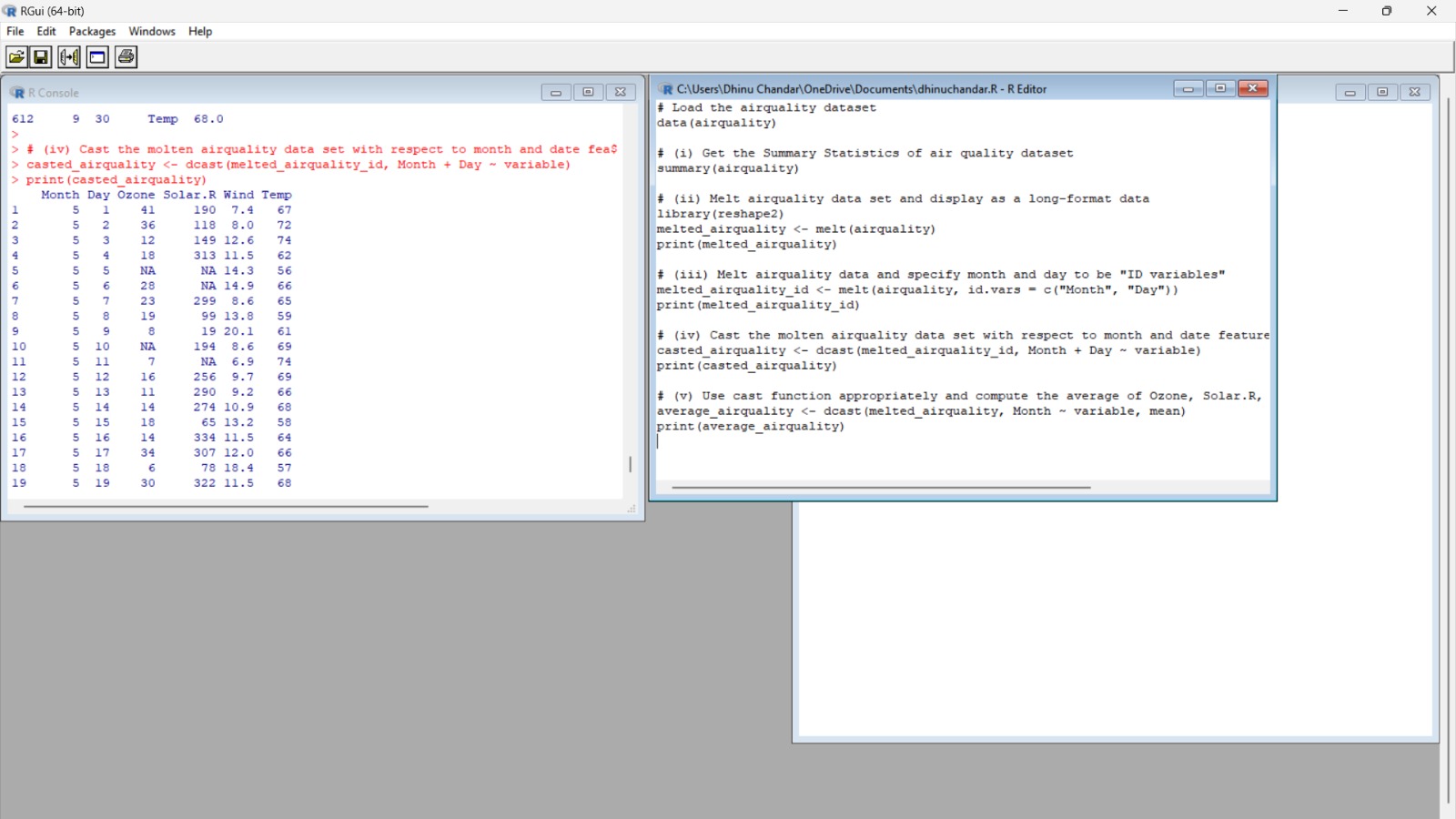
 (iv)Cast the molten airquality data set with respect to month and date features

CODE

Melt airquality data and specify month and day to be "ID variables"

melted\_airquality\_id <- melt(airquality, id.vars = c("Month", "Day"))

print(melted\_airquality\_id)



 (v) Use cast function appropriately and compute the average of Ozone, Solar.R , Wind

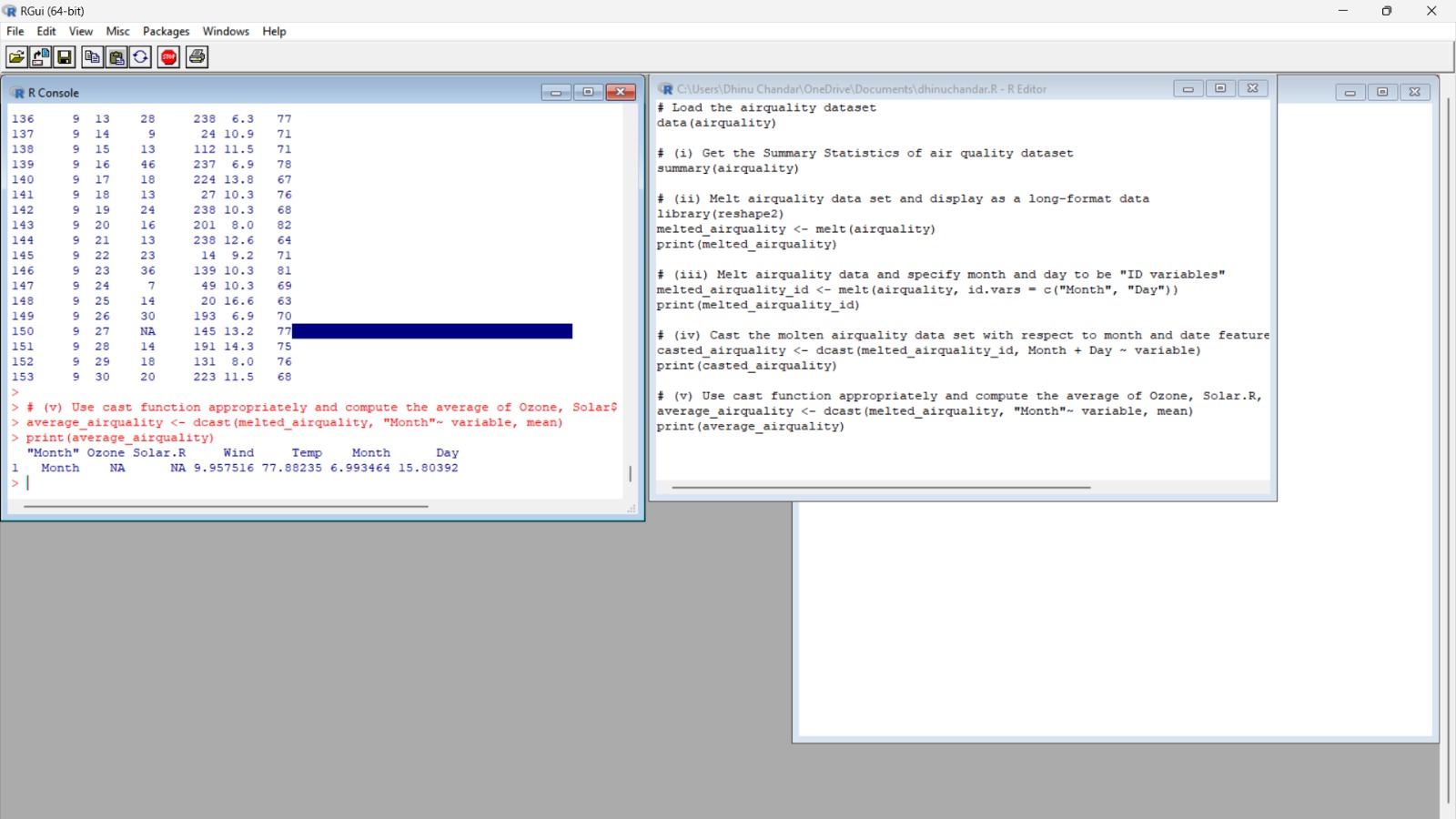
and temperature per month?

CODE

Use cast function appropriately and compute the average of Ozone, Solar.R, Wind, and temperature per month

average\_airquality <- dcast(melted\_airquality, Month ~ variable, mean)

print(average\_airquality)



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.

CODE

# Load airquality dataset

data(airquality)

# Find missing values

na\_count <- colMeans(is.na(airquality)) \* 100

na\_count

# Replace missing values with mean of the feature if less than 10%

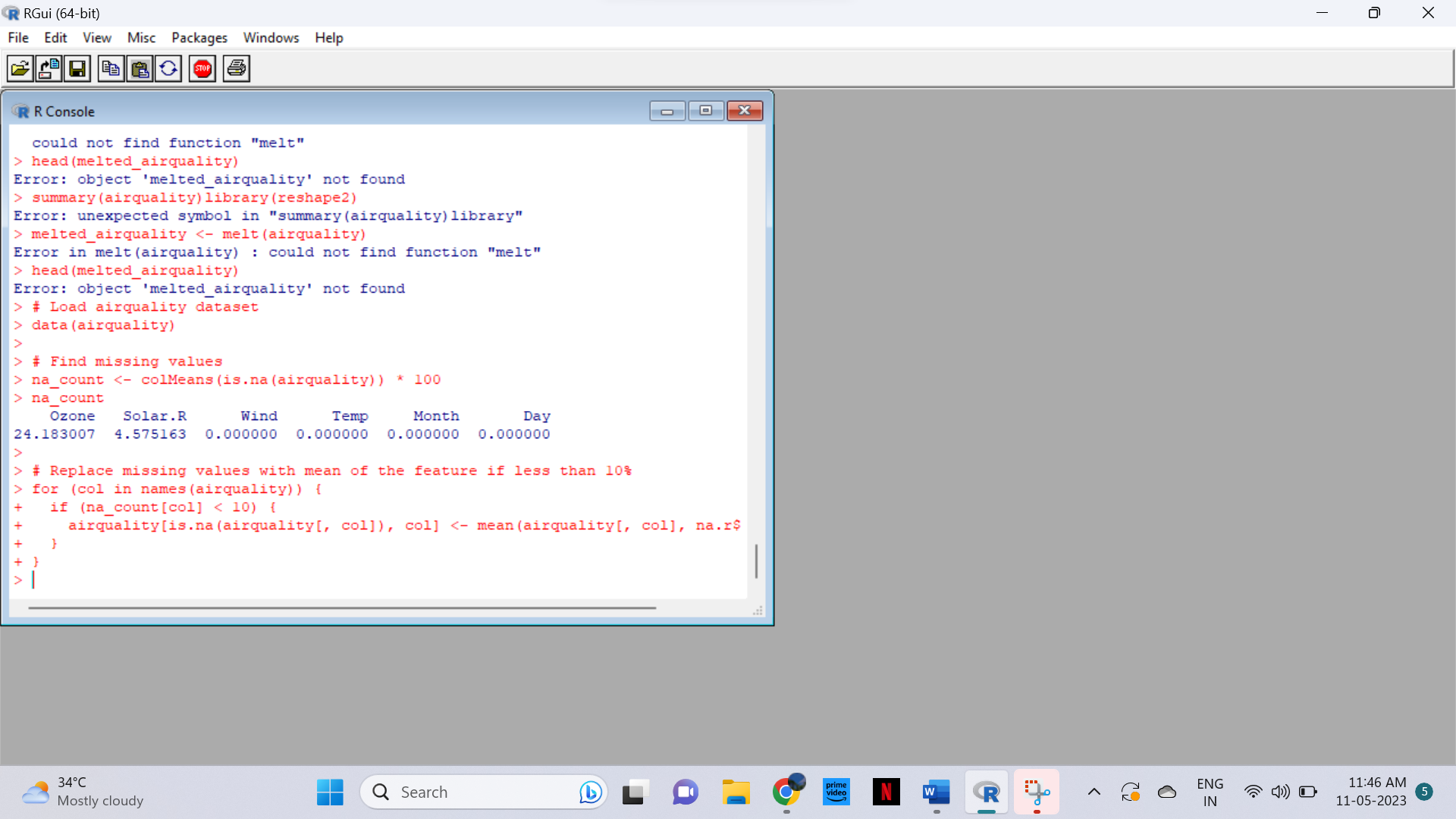
for (col in names(airquality)) {

if (na\_count[col] < 10) {

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

}

}



   (ii) Apply a linear regression algorithm using Least Squares Method on “Ozone” and

“Solar.R”

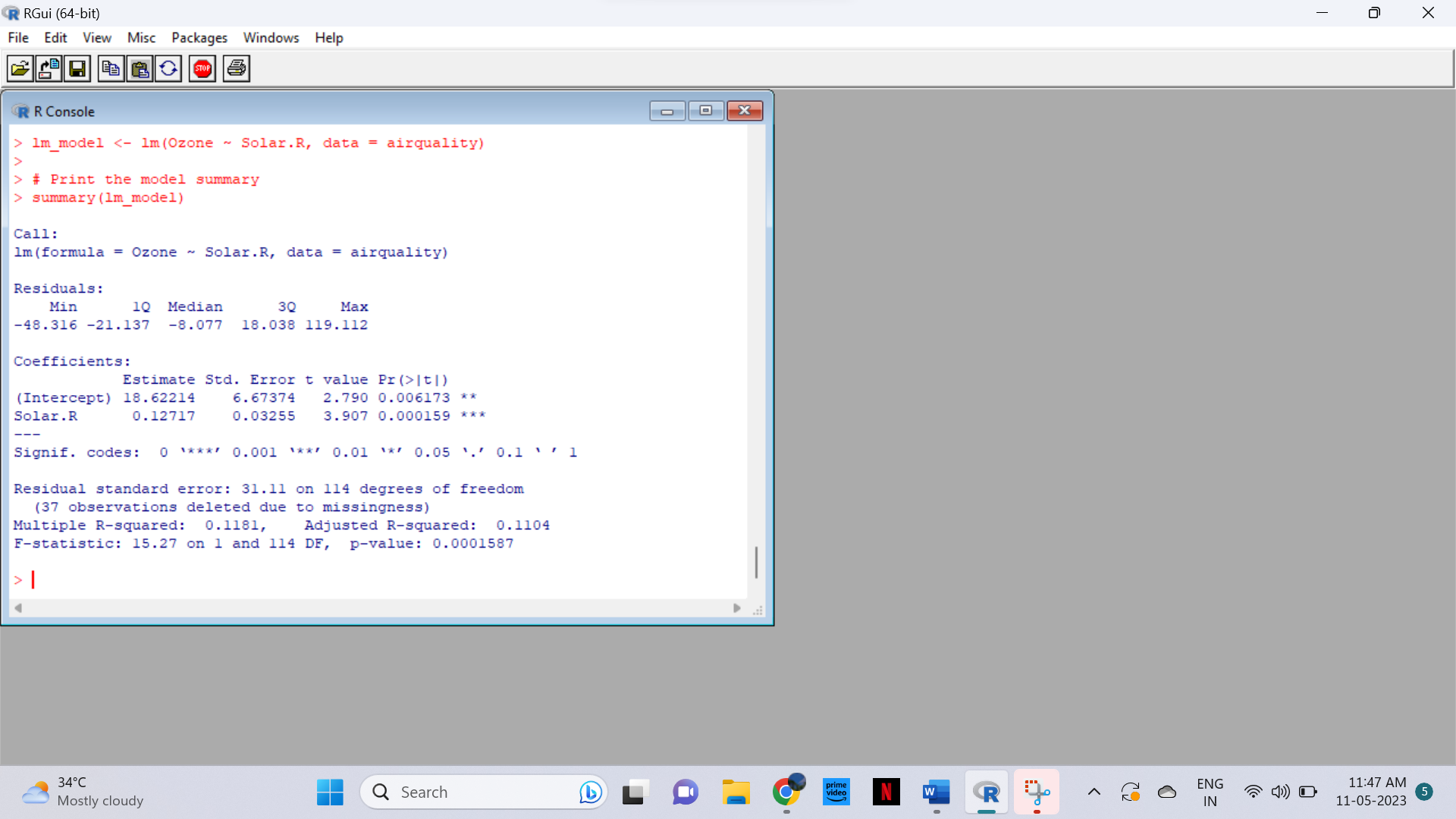
CODE

# Apply linear regression on "Ozone" and "Solar.R"

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

# Print the model summary

summary(lm\_model)



   (iii)Plot Scatter plot between Ozone and Solar and add regression line created by

above model

CODE

# Fit linear regression model

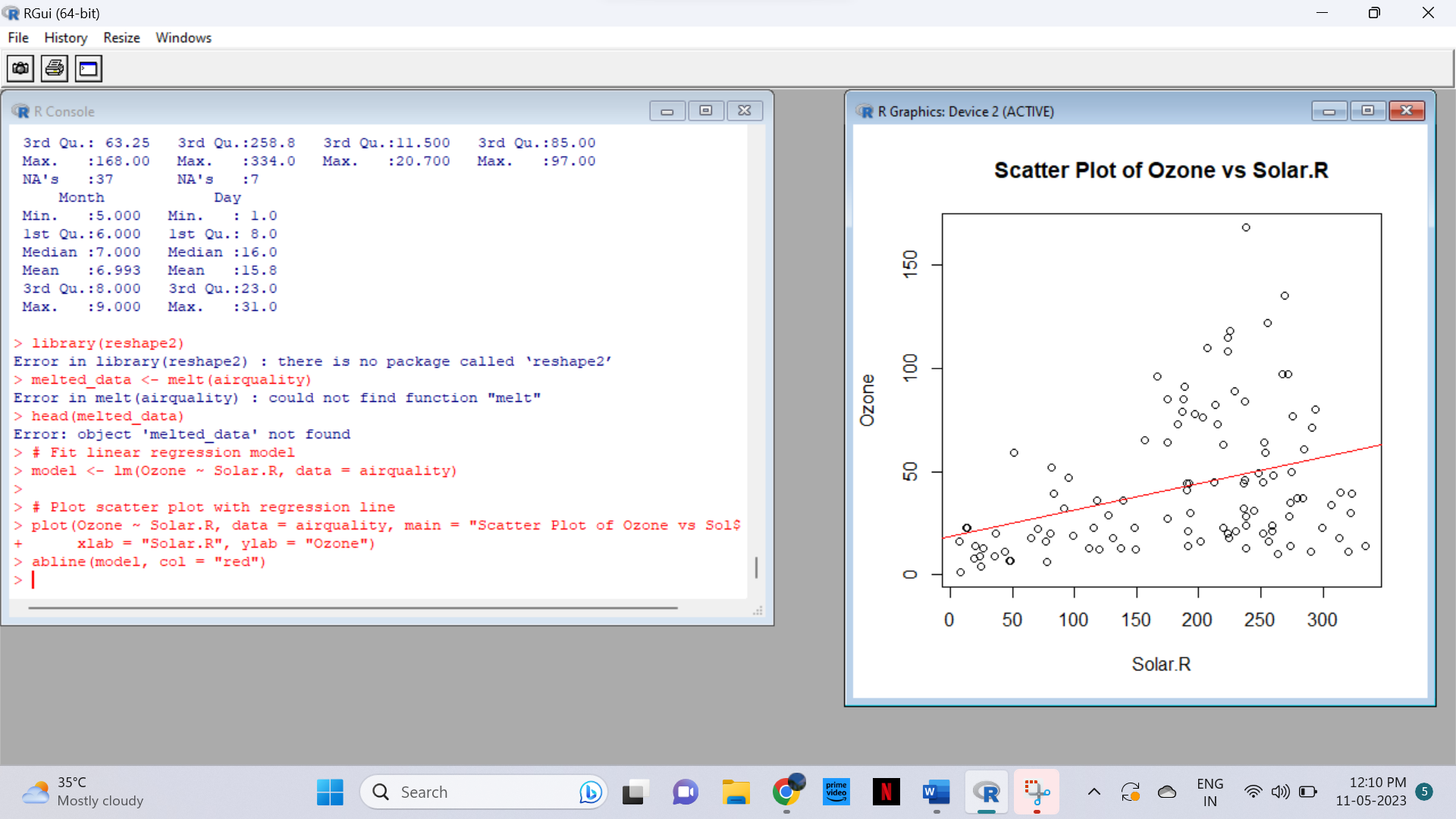
model <- lm(Ozone ~ Solar.R, data = airquality)

# Plot scatter plot with regression line

plot(Ozone ~ Solar.R, data = airquality, main = "Scatter Plot of Ozone vs Solar.R",

xlab = "Solar.R", ylab = "Ozone")

abline(model, col = "red")



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&quot;, &quot;Time&quot;, &quot;Diet&quot;   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

# Load ChickWeight dataset

data(ChickWeight)

# Order by weight and group by diet

ordered\_data <- ChickWeight %>%

arrange(diet, weight) %>%

group\_by(diet)

# Extract last 6 records

last\_six <- tail(ordered\_data, 6)

last\_six

# Load reshape2 library

library(reshape2)

# Melting function

melted\_data <- melt(ChickWeight, id.vars = c("Chick", "Time", "Diet"))

# Cast function to display mean of weight grouped by Diet

mean\_data <- cast(melted\_data, Diet ~ variable, mean)

mean\_data

# Cast function to display mode of weight grouped by Diet

mode\_data <- cast(melted\_data, Diet ~ variable, mode)

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

# Box plot for "weight" grouped by "Diet"

boxplot(weight ~ Diet, data = ChickWeight, main = "Weight by Diet", xlab = "Diet", ylab = "Weight")

# Histogram for "weight" features belonging to Diet-1 category

hist(ChickWeight$weight[ChickWeight$Diet == 1], main = "Weight for Diet-1", xlab = "Weight", col = "lightblue")

# Scatter plot for "weight" vs "Time" grouped by Diet

plot(ChickWeight$Time, ChickWeight$weight, xlab = "Time", ylab = "Weight", col = ChickWeight$Diet, main = "Weight vs Time by Diet")

legend("topright", legend = levels(factor(ChickWeight$Diet)), col = 1:length(levels(factor(ChickWeight$Diet))), pch = 1, title = "Diet")

