**SAVEETHA SCHOOL OF ENGINEERING**

**SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES**

**ITA 0443 - STATISTICS WITH R PROGRAMMING FOR REAL TIME PROBLEM**

**DAY 4– LAB MANUAL**

**Name:** Bala Manohar. J

**Reg no:** 192124133

**LINEAR REGRESSION ANALYSIS IN R**

**Exercise**

**1. Using linear regression analysis establish a relationship between height and weight of a**

**person using the input vector given below.**

**# Values of height**

**151, 174, 138, 186, 128, 136, 179, 163, 152, 131**

**# Values of weight.**

**63, 81, 56, 91, 47, 57, 76, 72, 62, 48**

**Predict the weight of a person with height 170. Visualize the regression graphically.**

**Step-01:**

CREATE RELATIONSHIP MODEL & GET THE COEFFICIENTS

**Source Code:**

x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)

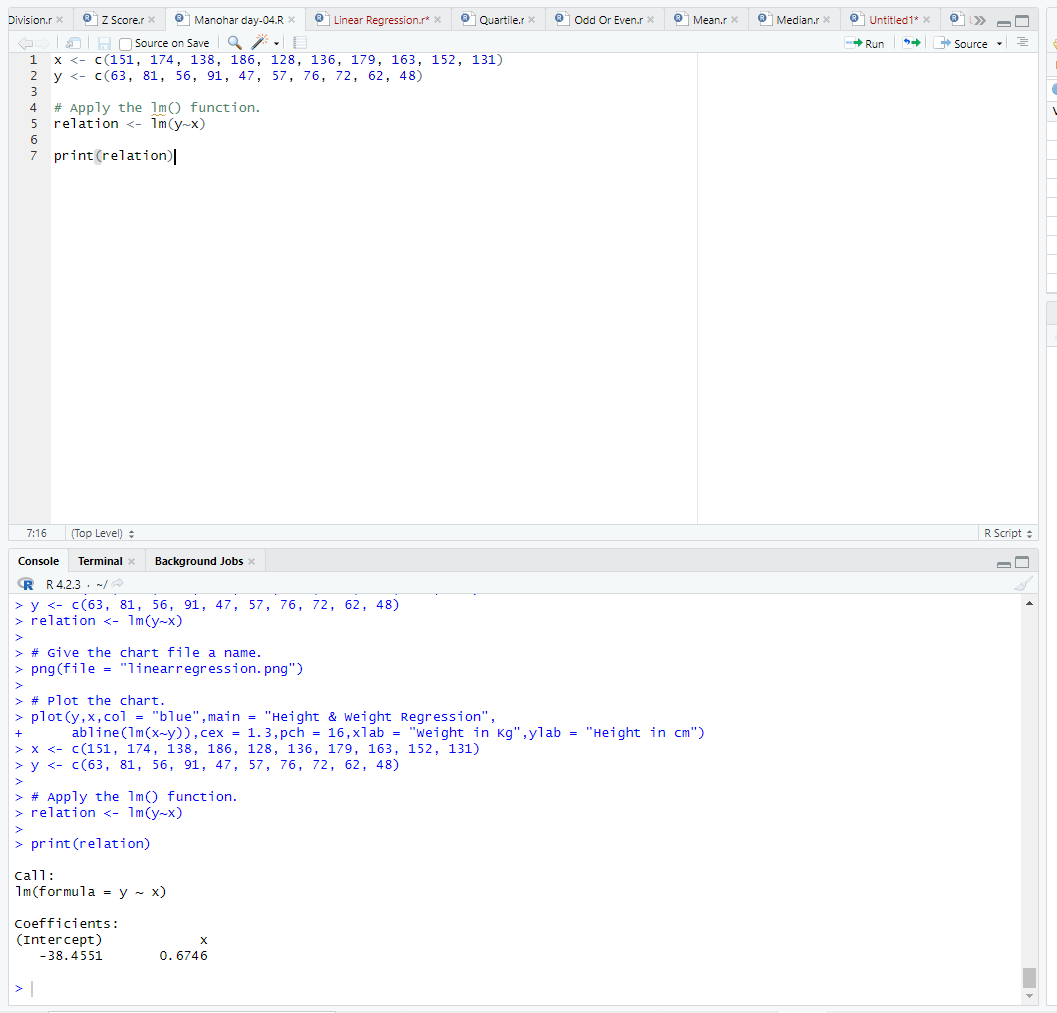
y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)

# Apply the lm() function.

relation <- lm(y~x)

print(relation)

**OUTPUT:**



**Step-02:**

### GET THE SUMMARY OF THE RELATIONSHIP

**Source Code:**

x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)

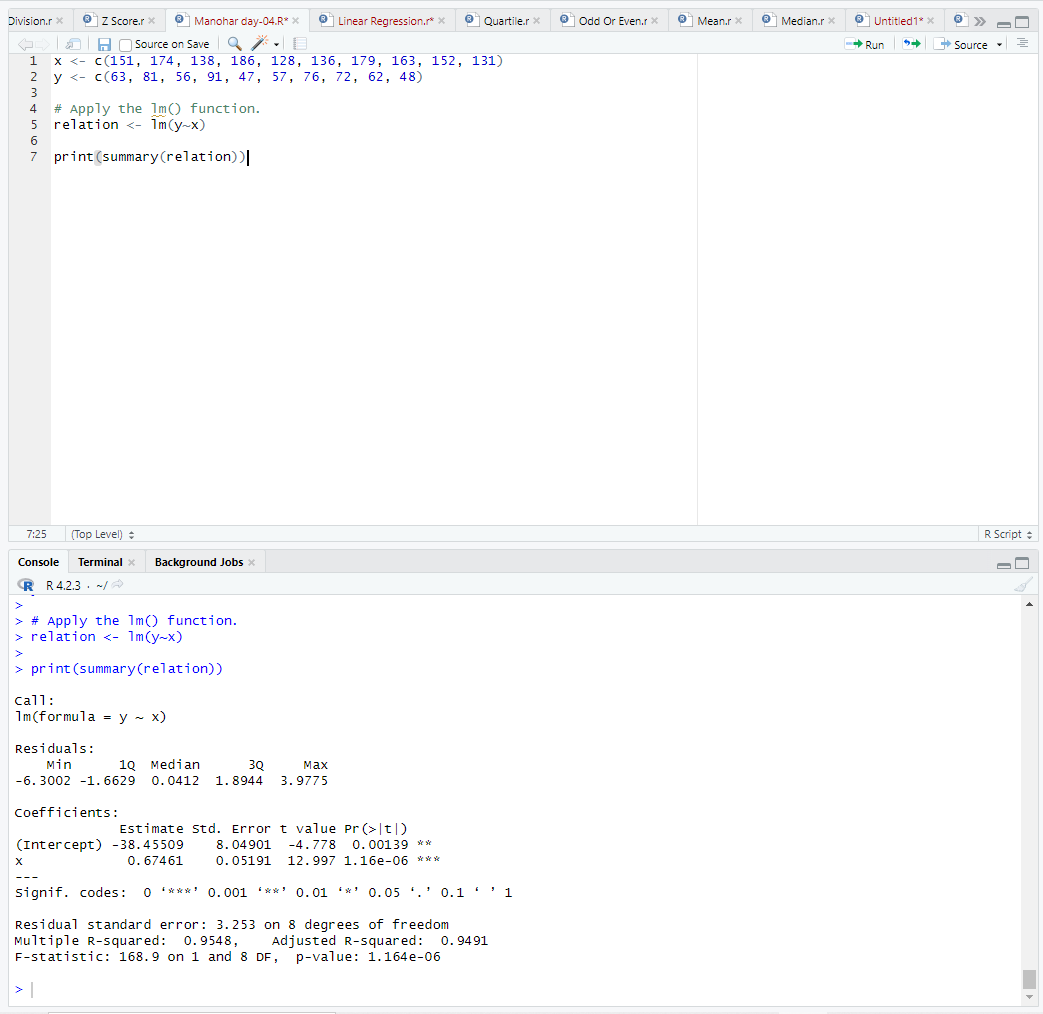
y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)

# Apply the lm() function.

relation <- lm(y~x)

print(summary(relation))

**OUTPUT**:



**Step-03:**

## **predict() Function**

**Source Code:**

# The predictor vector.

x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)

# The resposne vector.

y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)

# Apply the lm() function.

relation <- lm(y~x)

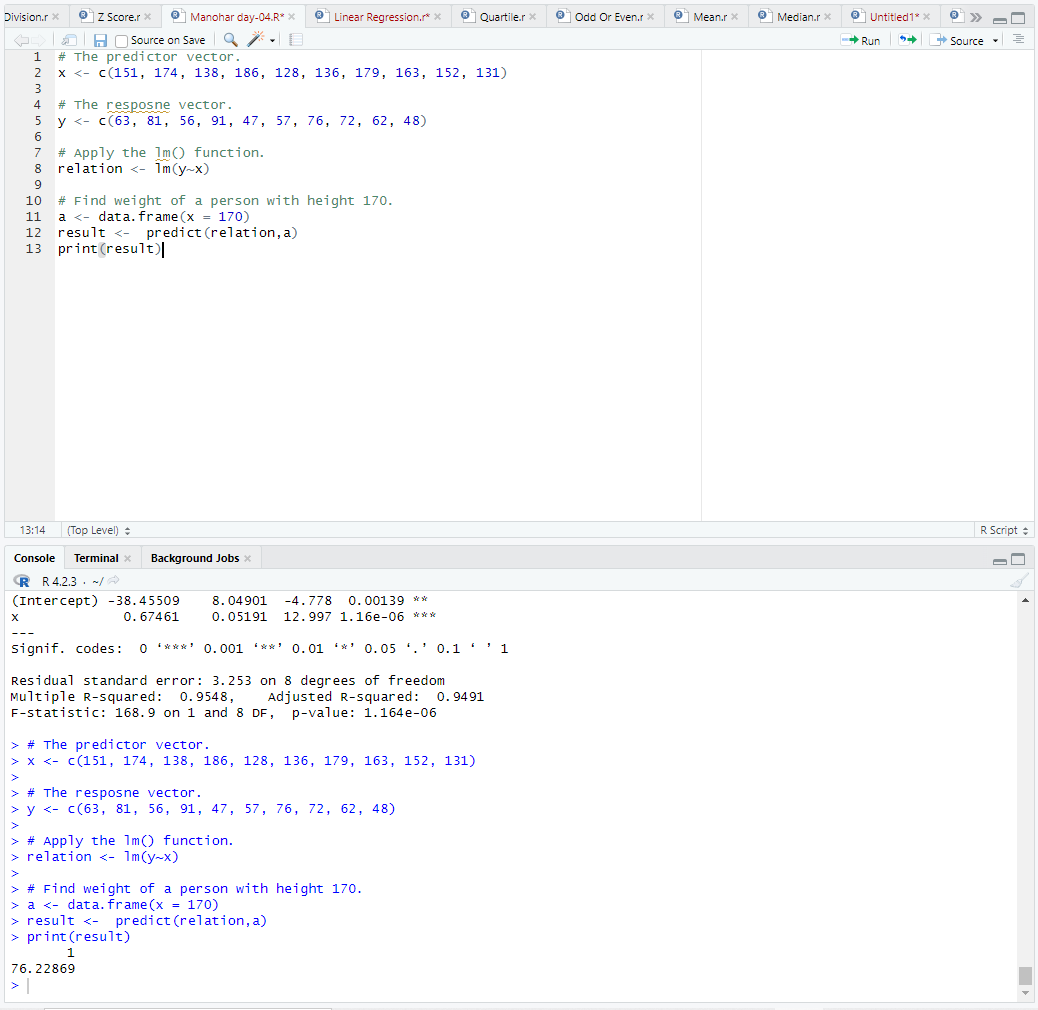
# Find weight of a person with height 170.

a <- data.frame(x = 170)

result <- predict(relation,a)

print(result)

**OUTPUT:**



**Step-04:**

### VISUALIZE THE REGRESSION GRAPHICALLY

**Source Code:**

install.packages("ggplot2")

library(ggplot2)

height <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)

weight <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)

data <- data.frame(height, weight)

model <- lm(weight ~ height, data = data)

new\_height <- data.frame(height = 170)

predicted\_weight <- predict(model, newdata = new\_height)

predicted\_weight

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

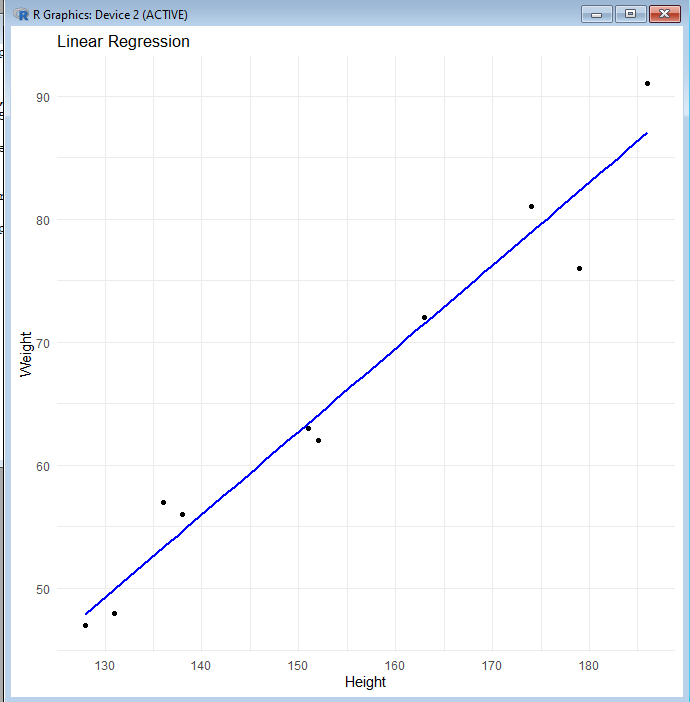
geom\_point() +

geom\_smooth(method = "lm", se = FALSE, color = "blue") +

labs(x = "Height", y = "Weight", title = "Linear Regression") +

theme\_minimal()

**OUTPUT:**



**2. Download the Dataset &quot;water&quot; From Rdataset Link.Find out whether there is a linear relation between attributes&quot;mortality&quot; and&quot;hardness&quot; by plot function.Fit the Data into the Linear Regression model.Predict the mortality for the hardness=88**

**Step-01:**

Download the Dataset Water

**Source Code:**

# Load the necessary packages

library(Ecdat)

library(ggplot2)

# Download the water dataset from Ecdat

data("water")

**Step-02:**

Check the Summary and statistics

**Source Code:**

# Check the structure of the dataset

str(water)

# Check the summary statistics of the dataset

summary(water)

**Step-03:**

Check for the Linear Relationship Between “Morality” and “Hardness” with the Help Of Scatter plot.

**Source Code:**

# Create a scatter plot to check for linear relationship between "mortality" and "hardness"

ggplot(water, aes(x=hardness, y=mortality)) +

geom\_point() +

labs(title="Scatter plot of Mortality and Hardness", x="Hardness", y="Mortality")

**Step-04:**

Fill The Data into a Linear Regression Model

**Source Code:**

# Fit the data into a linear regression model

model <- lm(mortality ~ hardness, data=water)

# Check the summary statistics of the model

summary(model)

**Step-05:**

Predict the Morality for the Hardness of 88

**Source Code:**

# Predict the mortality for the hardness of 88

new\_data <- data.frame(hardness=88)

prediction <- predict(model, new\_data)

prediction

**OUTPUT:**

1

196.978

**MULTIPLE REGRESSION ANALYSIS IN R**

**Exercise:**

**3.Generate a multiple regression model using the built in dataset mtcars.It gives a comparisonbetween different car models in terms of mileage per gallon (mpg), cylinderdisplacement(&quot;disp&quot;), horse power(&quot;hp&quot;), weight of the car(&quot;wt&quot;) and some more parameters.**

**Establish the relationship between &quot;mpg&quot; as a response variable with &quot;disp&quot;,&quot;hp&quot; and &quot;wt&quot; as predictor variables. Predict the mileage of the car with dsp=221,hp=102 and wt=2.91.**

**Step-01:**

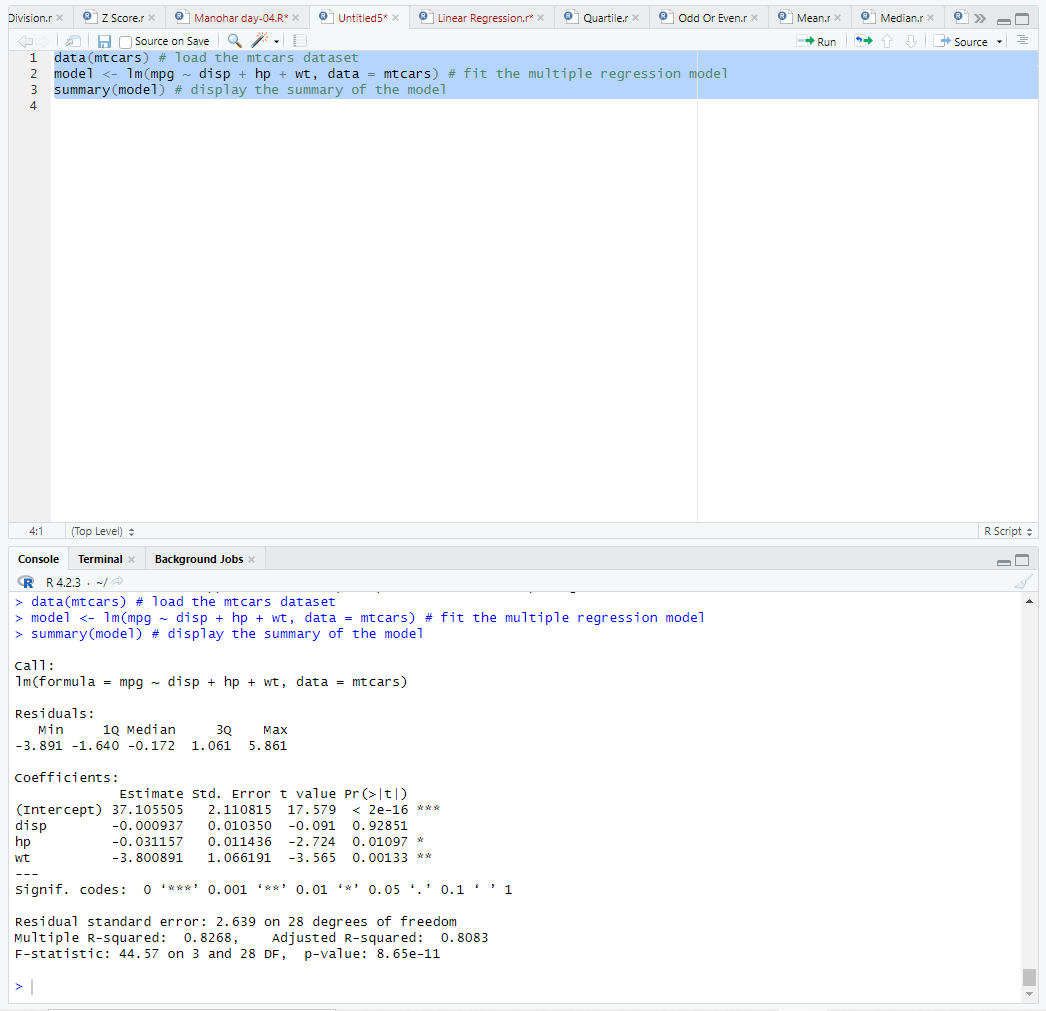
**Source Code:**

data(mtcars) # load the mtcars dataset

model <- lm(mpg ~ disp + hp + wt, data = mtcars) # fit the multiple regression model

summary(model) # display the summary of the model

**OUTPUT:**



**Step-02:**

To predict the mileage of a car with "disp"=221, "hp"=102, and "wt"=2.91, we can create a data frame with these values and use the predict function. Here's the code to do that:

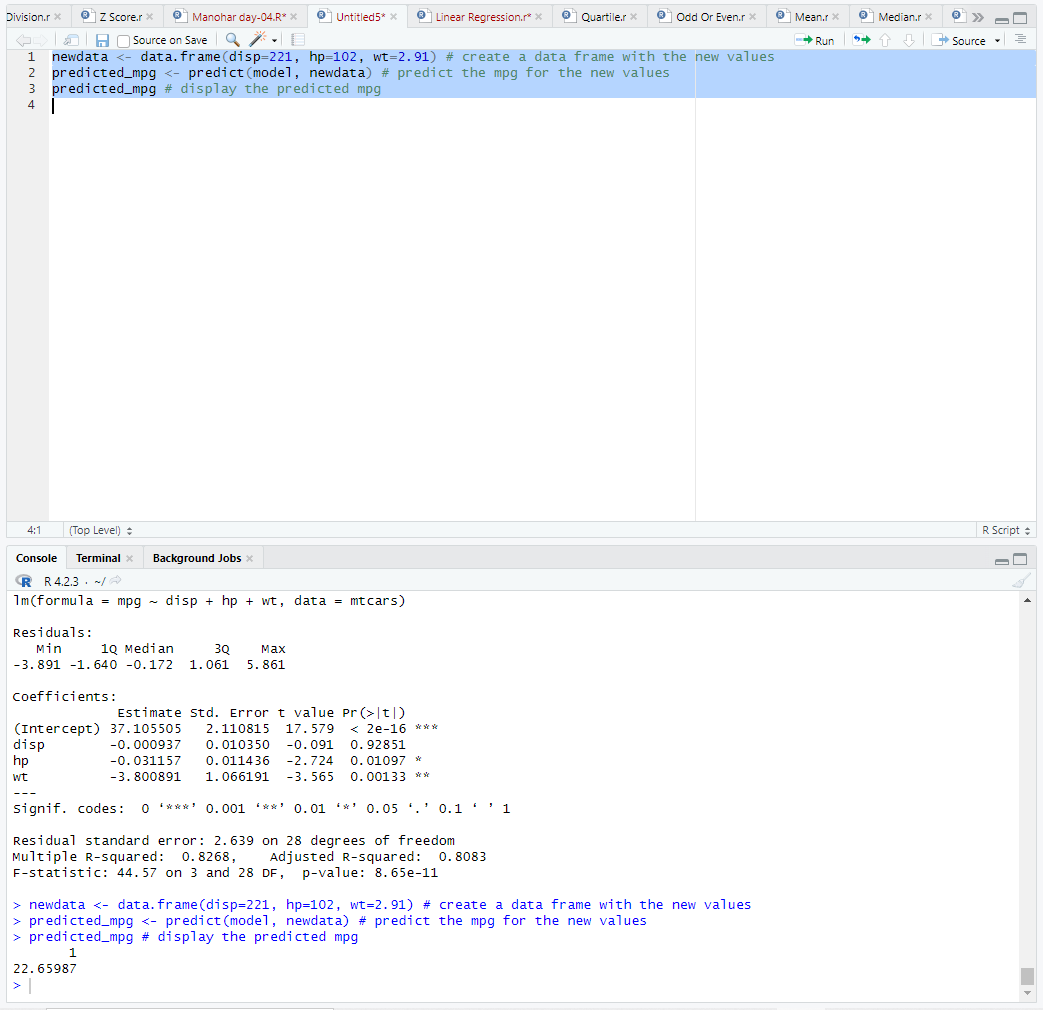
**Source Code:**

newdata <- data.frame(disp=221, hp=102, wt=2.91) # create a data frame with the new values

predicted\_mpg <- predict(model, newdata) # predict the mpg for the new values

predicted\_mpg # display the predicted mpg

**OUTPUT:**



**4. Consider the data set &quot;delivery&quot; available in the R environment. It gives a deliverytime(“delTime”)of production materials(number of productions “n.prod”) with the given distance(“distance”) to reach the destination place.**

**a)Create the model to establish the relationship between &quot;delTime&quot; as a response**

**variable with &quot;n.prod&quot; and &quot;distance&quot; as predictor variables.**

**b)Predict the delTime for the given number of production(“n.prod”)=9 and distance(“distance”)=450**

**Step-01:**

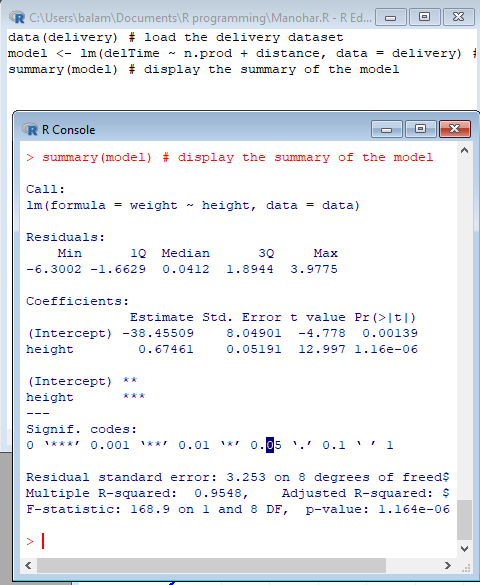
**Source Code:**

data(delivery) # load the delivery dataset

model <- lm(delTime ~ n.prod + distance, data = delivery) # fit the linear regression model

summary(model) # display the summary of the model

**OUTPUT:**

****

**Step-02:**

**Source Code:**

newdata <- data.frame(n.prod=9, distance=450) # create a data frame with the new values

predicted\_delTime <- predict(model, newdata) # predict the delivery time for the new values

predicted\_delTime # display the predicted delivery time

**OUTPUT:**

