**DT:10-02-23 LAB EXERCISES**

**ITAO443-STATISTICS WITH R-PROGRAMMING**

**NAME:v.penchala reddy**

**REG.NO:192011486**

**GITHUB LINK:-** [penchala720/ITA0443-R-Programming (github.com)](https://github.com/penchala720/ITA0443-R-Programming)

**FEB 10 2023 DAY 3 LAB ASSIGNMENT**

1. (i) Write a function in R programming to print generate Fibonacci sequence using

Recursion in R

 .

(ii) Find sum of natural numbers up-to 10, without formula using loop statement.

(iii) create a vector 1:10 and Find a square of each number and store that in a

separate list.

P1(ii) sum=0 n=5

for(i in 1:n){ sum=sum+i

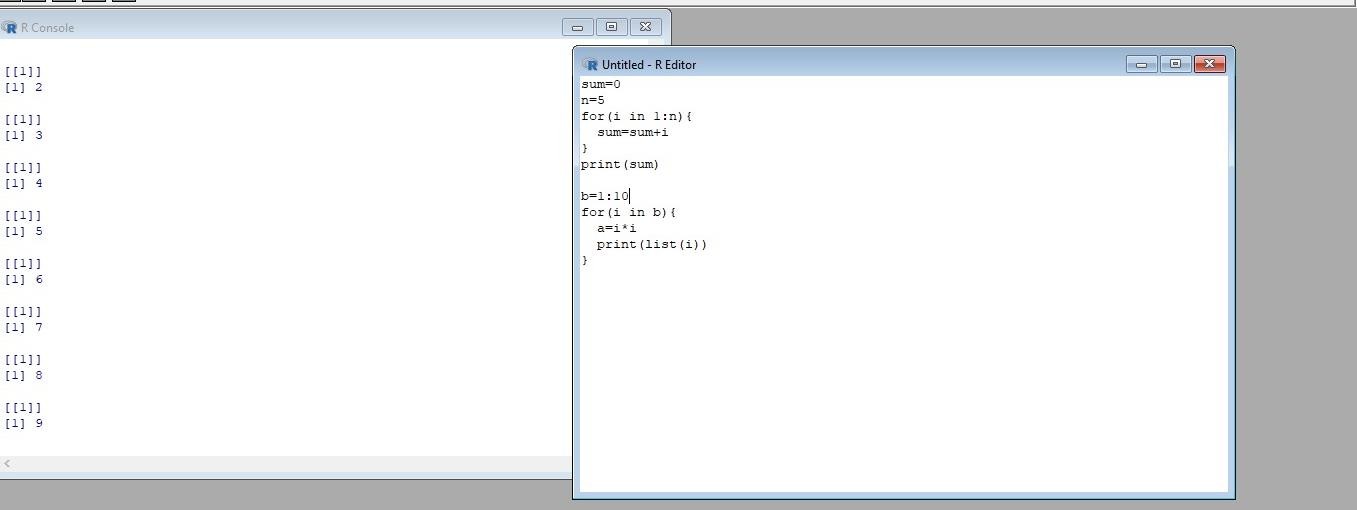
}

print(sum)

b=1:10

for(i in b){ a=i\*i print(list(i))

} OUTPUT:-

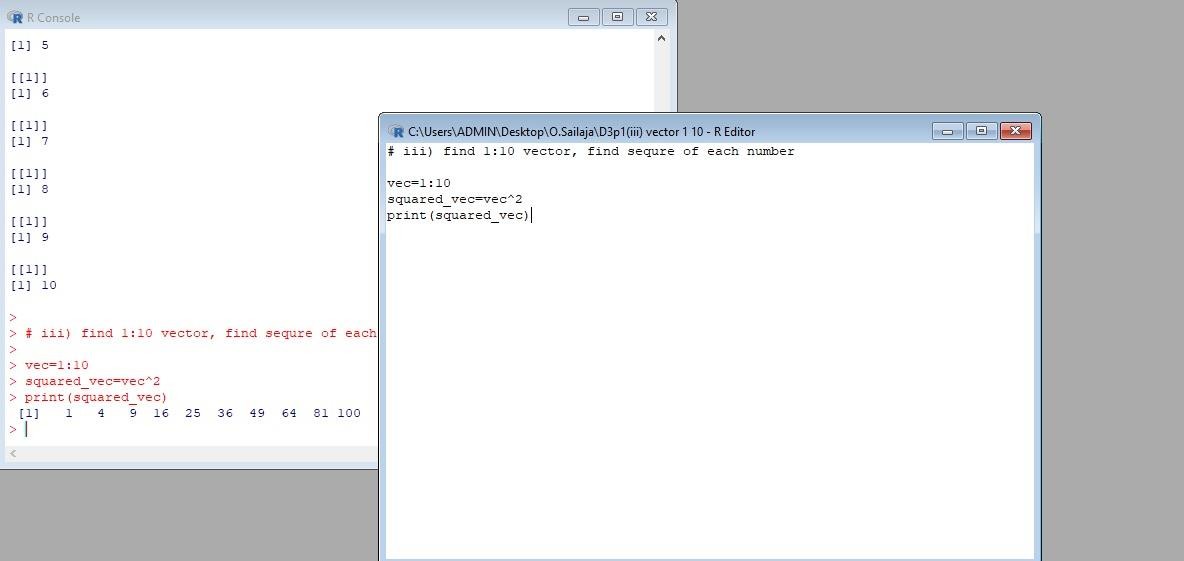


# iii) find 1:10 vector, find sequre of each number

vec=1:10

squared\_vec=vec^2 print(squared\_vec)

OUTPUT:-



2 question 2.    (motor trend car road test) comprises fuel consumption, performance and  10 aspects

of automobile

design for 32 automobiles. It comes pre-installed  with  package in R.

 (i)Find the dimension of the dataset

(ii)Give the statistical summary of the features.

(iii)Print the categorical features in Dataset

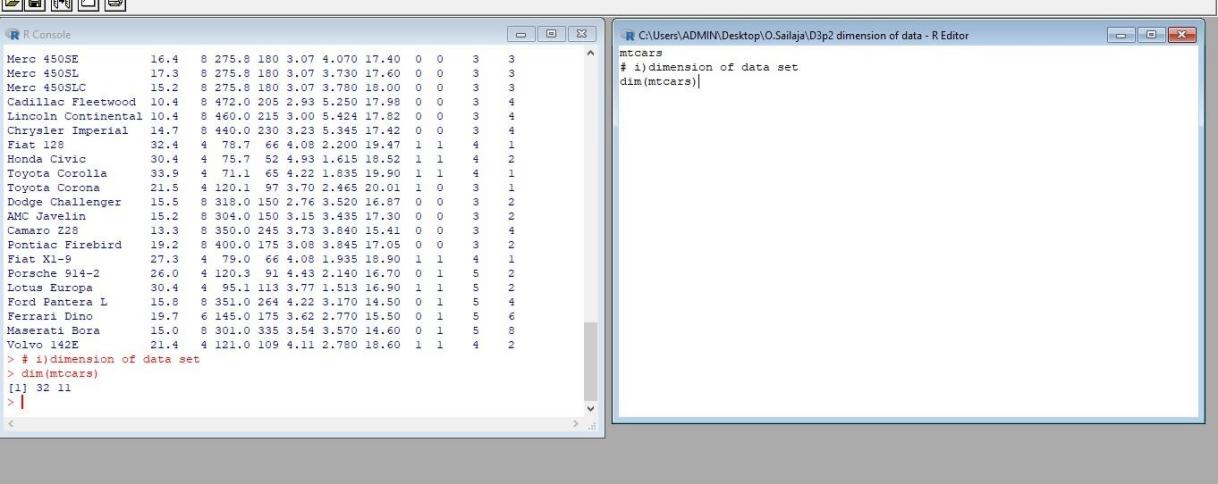
(iv)Find the average weight(wt) grouped by Engine shape(vs)

(v)Find the largest and smallest value of the variable weight with respect to Engine shape

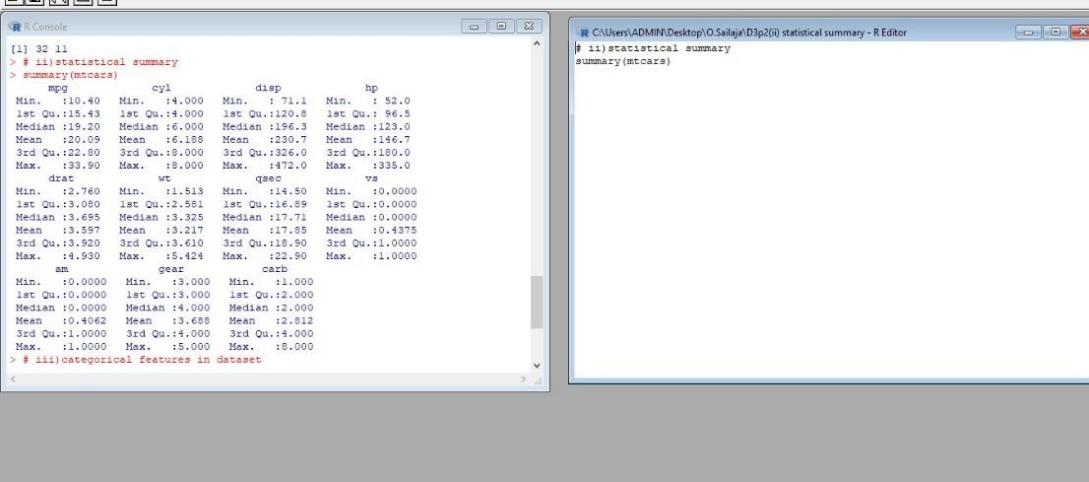
mtcars

# i)dimension of data set dim(mtcars)

Output:-

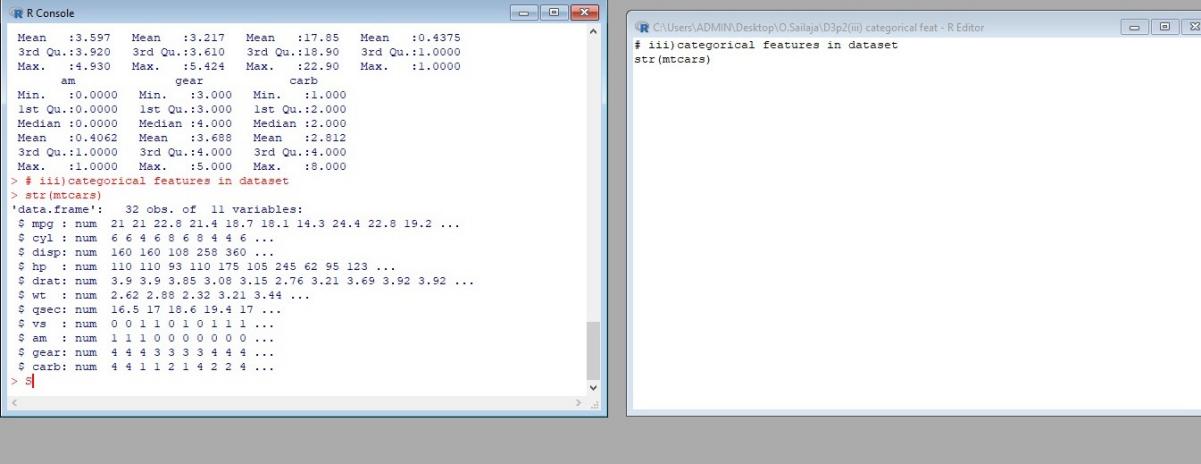


ii)statistical summary summary(mtcars) Output:-

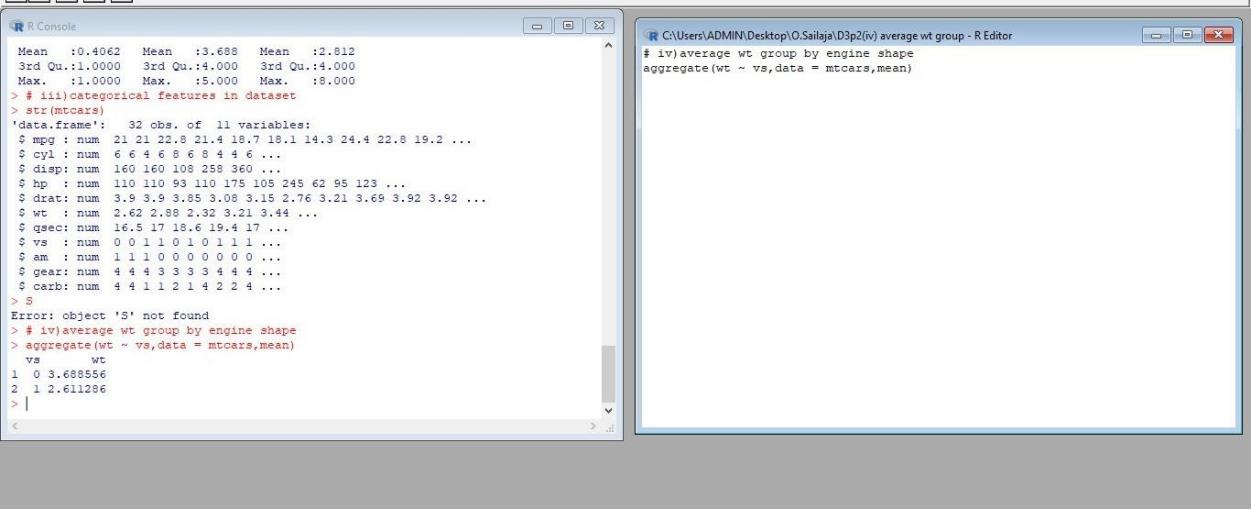


# iii)categorical features in dataset str(mtcars)

output:-



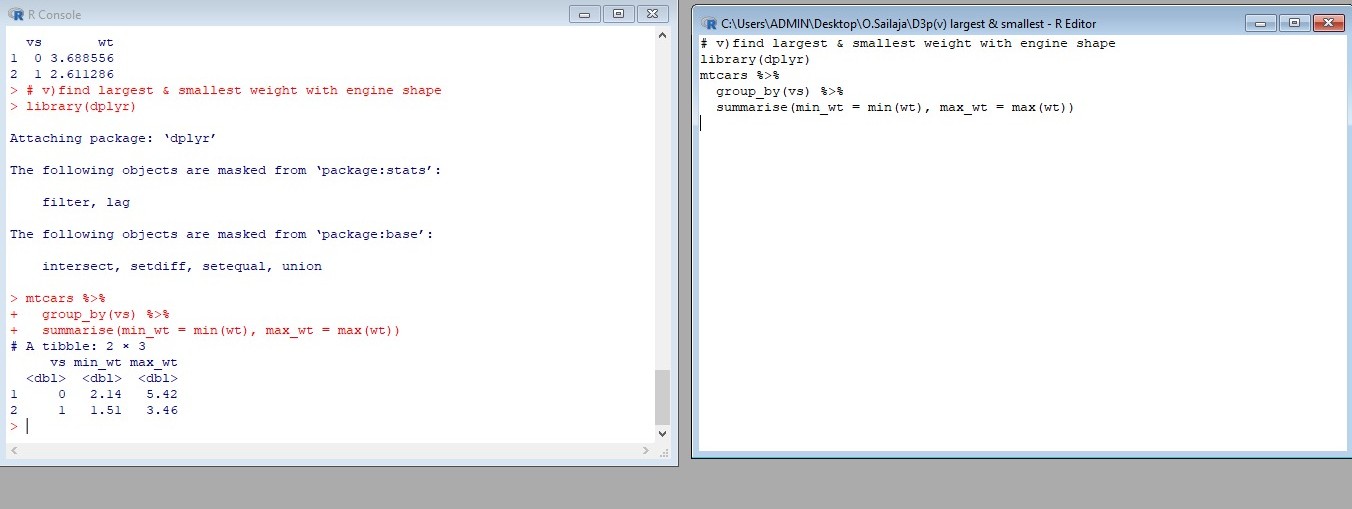
# iv)averagewt group by engine shape aggregate(wt ~ vs,data = mtcars,mean) output:-



# v)find largest & smallest weight with engine shape library(dplyr)

mtcars %>% group\_by(vs) %>%

summarise(min\_wt = min(wt), max\_wt = max(wt)) output:-



3 rd question 3.Use ggplot package to plot below EDA questions label the plot accordingly

(i)Create weight(wt) vs displacement(disp) scatter plot factor by  Engine Shape(vs)

(ii) Create horsepower(hp) vs mileage (mgp) scatter plot factor by  Engine Shape(vs)

(iv)In above plot , Separate columns according to cylinders(cyl) size

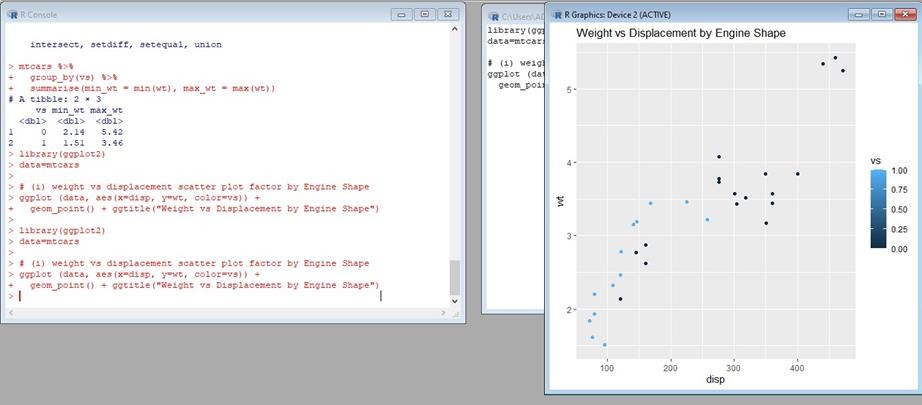
(v) Create histogram plot for horsepower (hp) with bin-width size of 5

library(ggplot2) data=mtcars

# (i) weight vs displacement scatter plot factor by Engine Shape ggplot (data, aes(x=disp, y=wt, color=vs)) +

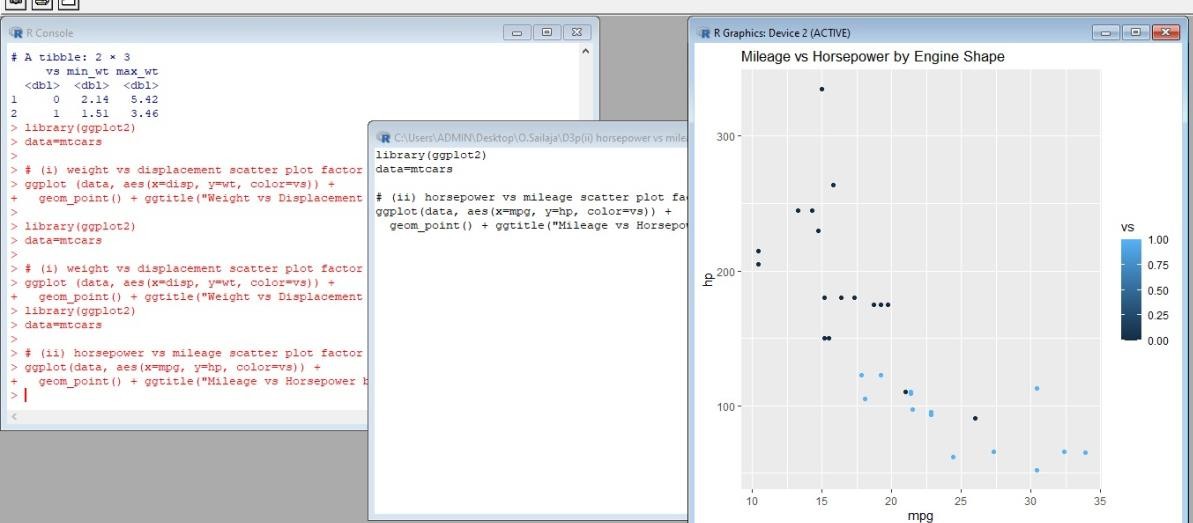
geom\_point() + ggtitle("Weight vs Displacement by Engine Shape")

output:-



# (ii) horsepower vs mileage scatter plot factor by Engine Shape ggplot(data, aes(x=mpg, y=hp, color=vs)) +

geom\_point() + ggtitle("Mileage vs Horsepower by Engine Shape") output:-



# (iii) horsepower vs mileage scatter plot factor by Cylinder Size ggplot(data, aes(x=mpg, y=hp, color=cyl)) +

geom\_point() + ggtitle("Mileage vs Horsepower by Cylinder Size")

4.Performing Logistic regression on dataset to predict the cars Engine shape(vs) .

(i)Do the EDA analysis and find the features which is impact the Engine shape and use

this for model.

(ii) Split the data set randomly with 80:20 ration to create train and test dataset and create

logistic model

(iii)Create the Confusion matrix among prediction and test data.

PROGRAM:-

# Installing the package

install.packages("caTools") # For Logistic regression

install.packages("ROCR") # For ROC curve to evaluate model

# Loading package

library(caTools)

library(ROCR)

# Splitting dataset

split <- sample.split(mtcars, SplitRatio = 0.8)

split

train\_reg <- subset(mtcars, split == "TRUE")

test\_reg <- subset(mtcars, split == "FALSE")

# Training model

logistic\_model <- glm(vs ~ wt + disp,

data = train\_reg,

family = "binomial")

logistic\_model

# Summary

summary(logistic\_model)

# Predict test data based on model

predict\_reg <- predict(logistic\_model,

test\_reg, type = "response")

predict\_reg

# Changing probabilities

predict\_reg <- ifelse(predict\_reg >0.5, 1, 0)

# Evaluating model accuracy

# using confusion matrix

table(test\_reg$vs, predict\_reg)

missing\_classerr <- mean(predict\_reg != test\_reg$vs)

print(paste('Accuracy =', 1 - missing\_classerr))

# ROC-AUC Curve

ROCPred <- prediction(predict\_reg, test\_reg$vs)

ROCPer <- performance(ROCPred, measure = "tpr",

x.measure = "fpr")

auc <- performance(ROCPred, measure = "auc")

auc <- auc@y.values[[1]]

auc

# Plotting curve

plot(ROCPer)

plot(ROCPer, colorize = TRUE,

print.cutoffs.at = seq(0.1, by = 0.1),

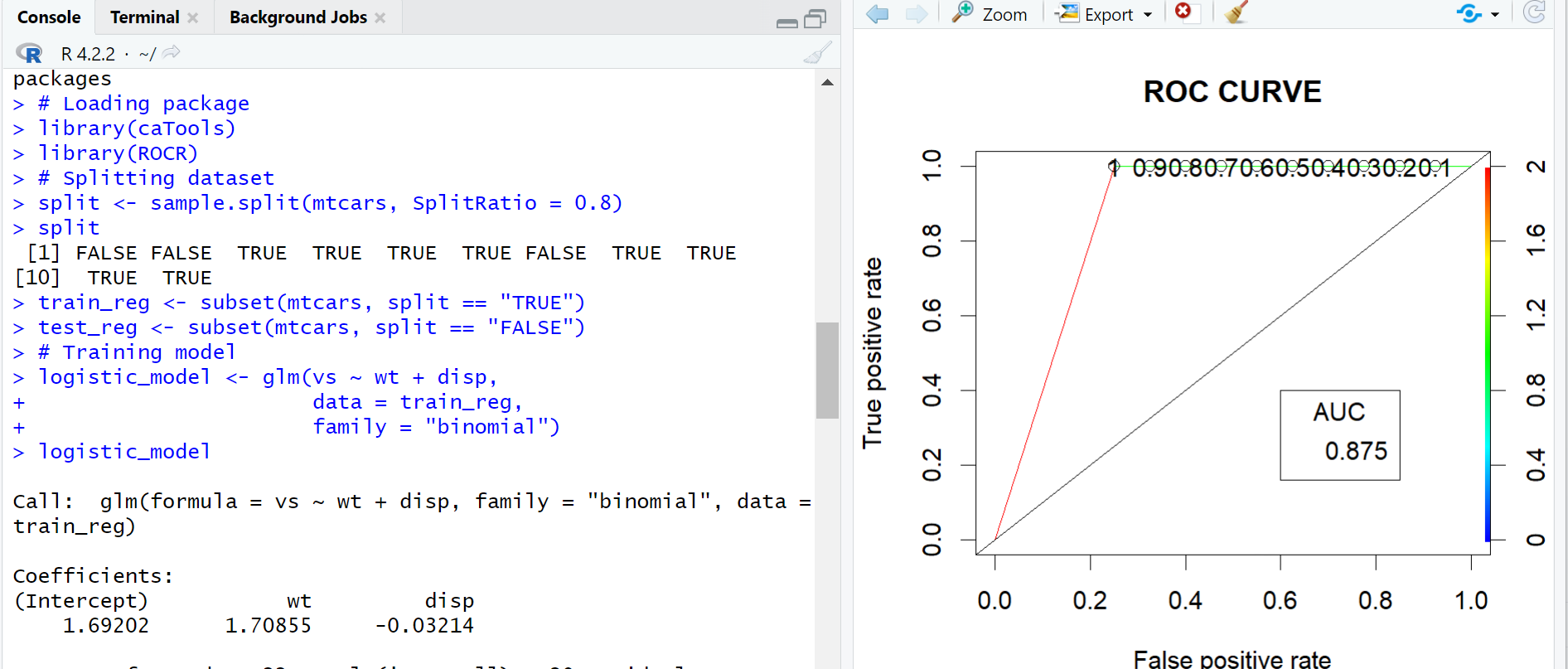
main = "ROC CURVE")

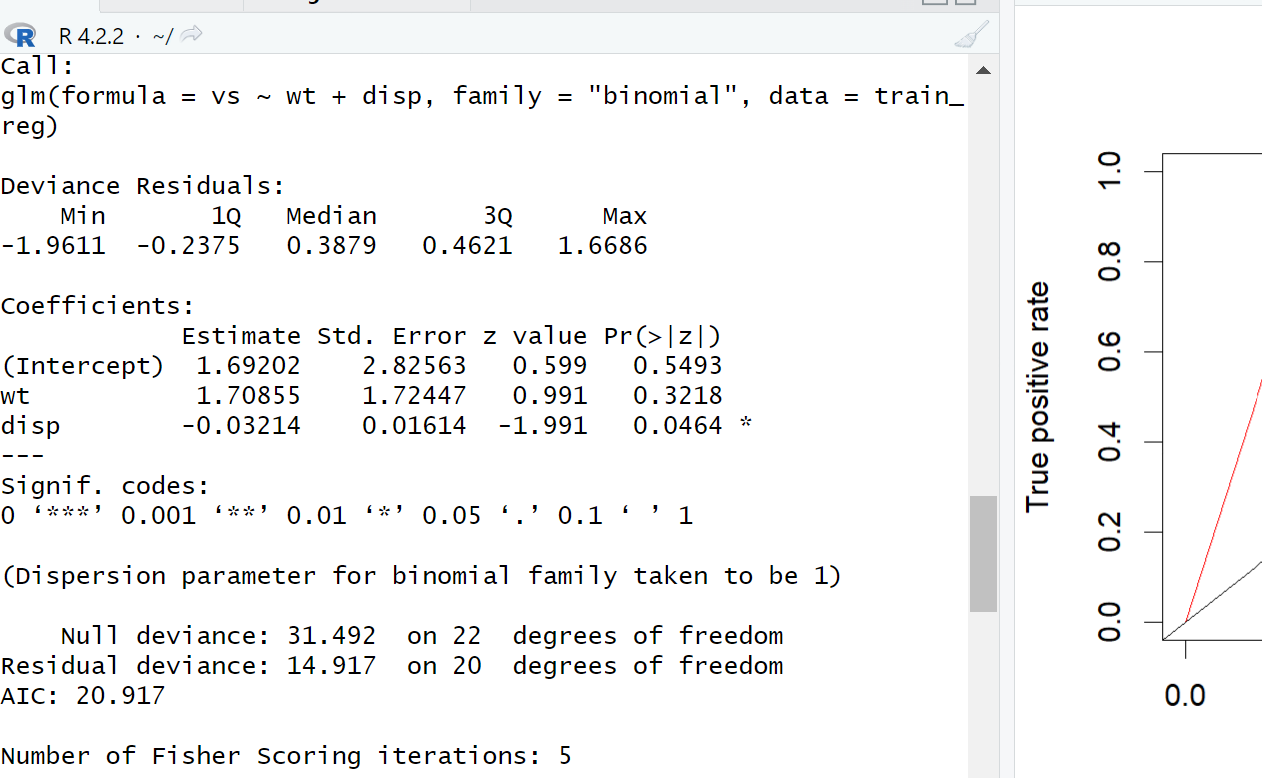
abline(a = 0, b = 1)

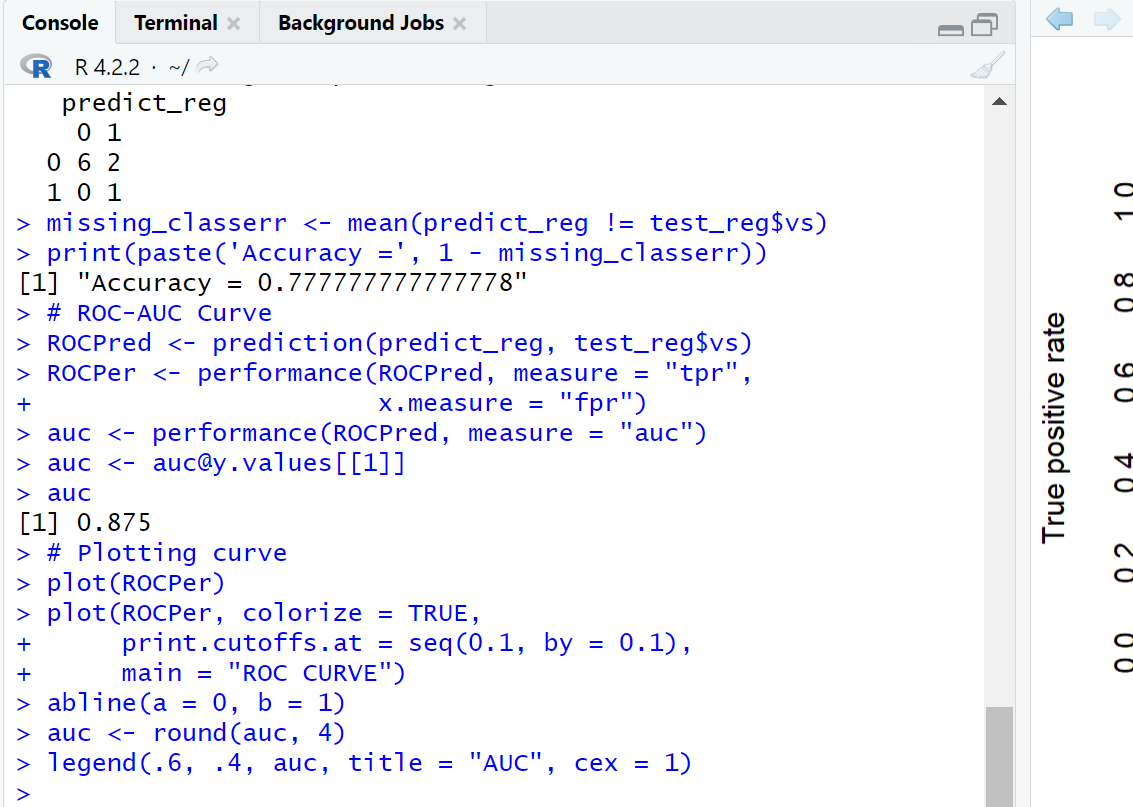
auc <- round(auc, 4)

legend(.6, .4, auc, title = "AUC", cex = 1)

OUTPUT:-







1. (I) Write R Program to create 15 x15 matrix filled with random numbers between -10 to 10, numbers can repeat.

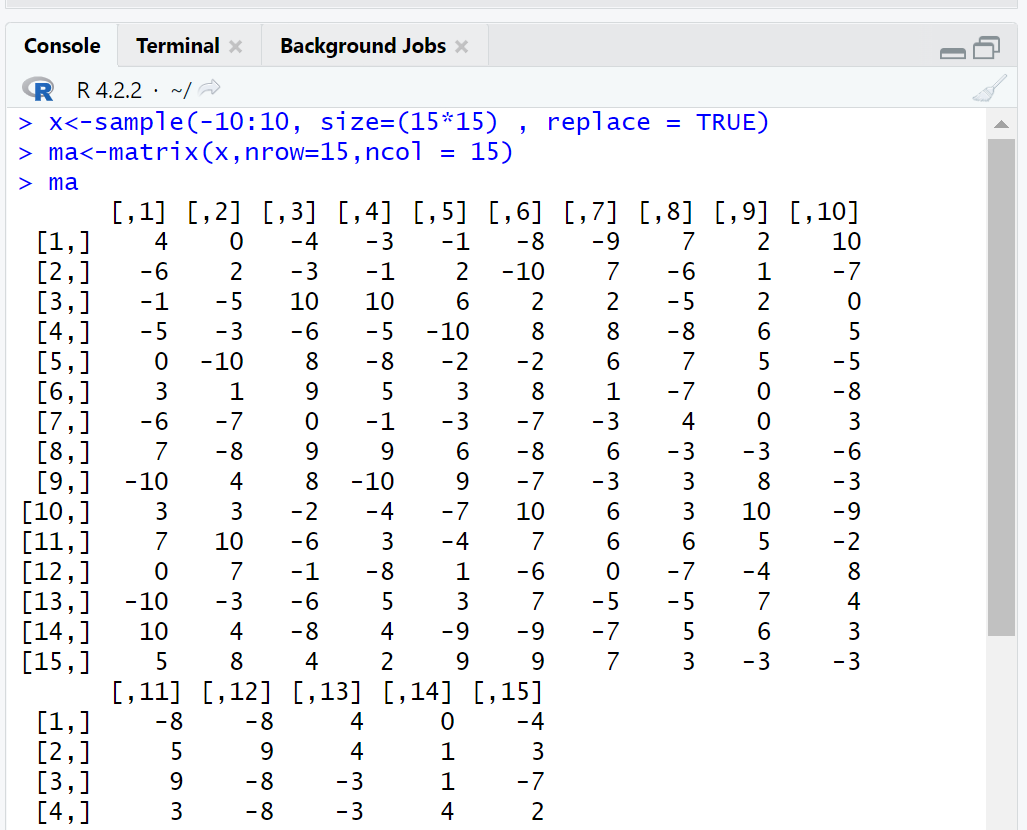
PROGRAM:-

x<-sample(-10:10, size=(15\*15) , replace = TRUE)

ma<-matrix(x,nrow=15,ncol = 15)

Ma

OUTPUT:-



(ii)Write R Program to display Lower Diagonal and upper Diagonal matrix

PROGRAM:-

x<-sample(-10:10, size=(15\*15) , replace = TRUE)

ma<-matrix(x,nrow=15,ncol = 15)

ma1<-ma

ma2<-ma

ma1[lower.tri(ma1)] <- 0

print("Lower diagonal matrix")

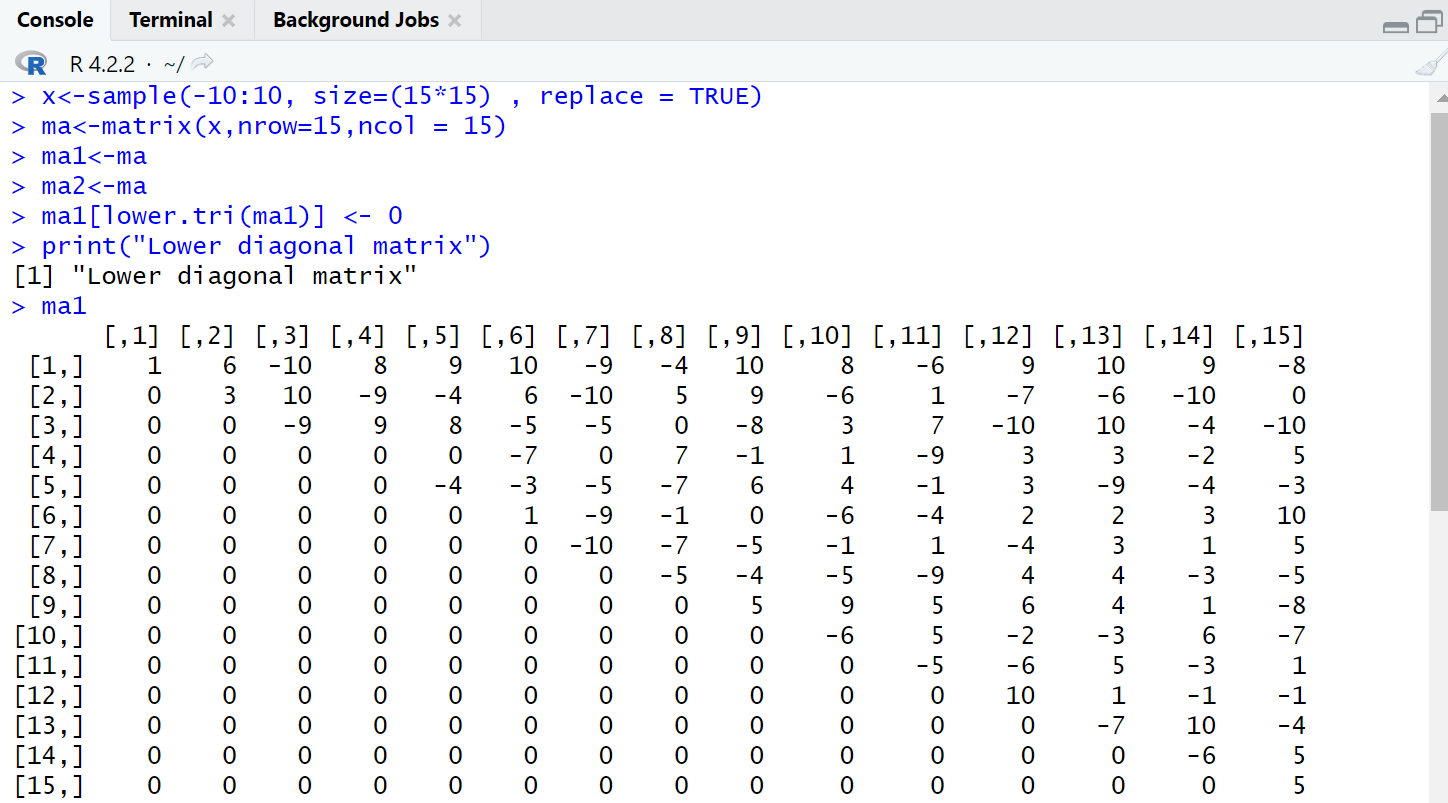
ma1

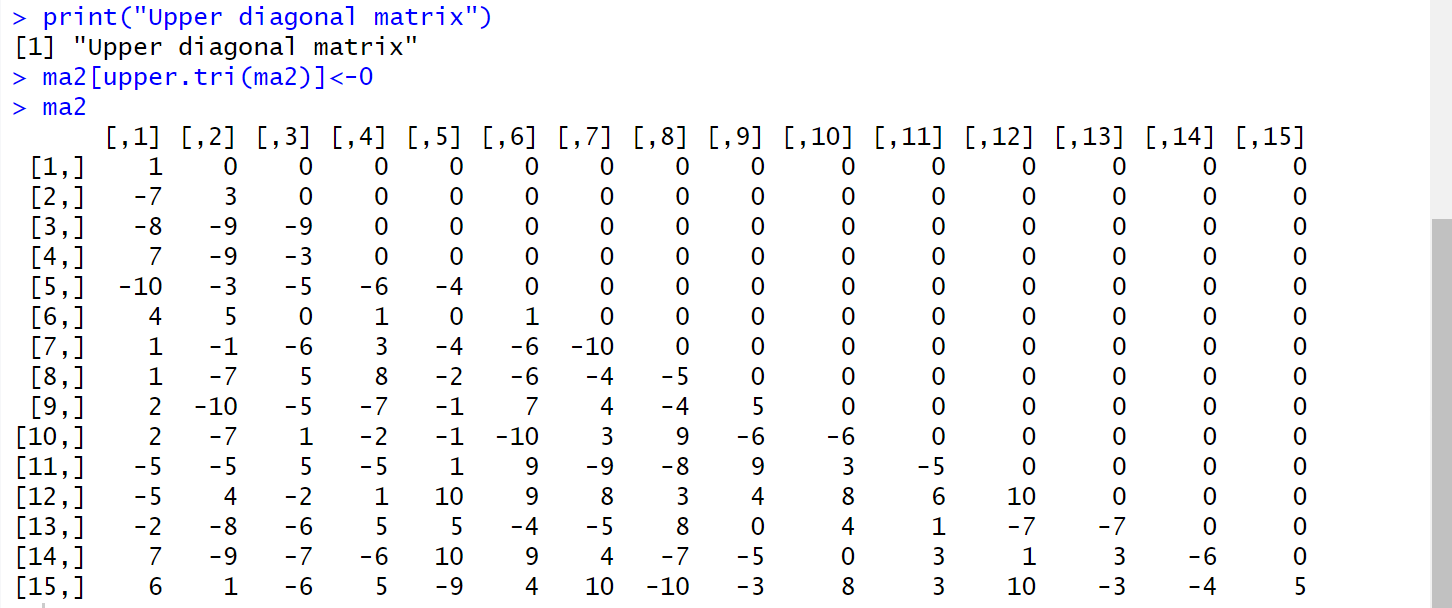
print("Upper diagonal matrix")

ma2[upper.tri(ma2)]<-0

ma2

OUTPUT:-





(iii)Write R Program to count 0’s in the matrix and check the matrix is sparse matrix or not

PROGRAM:-

x<-sample(-10:10, size=(15\*15) , replace = TRUE)

ma<-matrix(as.integer(x),nrow=15,ncol = 15)

y<-0

for(i in 1:15)

{

for(j in 1:15)

{

if(ma[i][j]==0)

{

y=y+1

}

}

}

paste("no of zeros in matrix equal to ",y)

z=y/(15\*15)

if(z>0.5)

{

print("The matrix is sparse matrix")

}

else

{

print("The matrix is not a sparse matarix")

}