**REPORT: DECISION TREE**

**PART 1:**  
  
*DECISION TREE:*  
A decision tree is a map of the possible outcomes of a series of related choices. It allows an individual or organization to weigh possible actions against one another based on their costs, probabilities, and benefits. They can be used either to drive informal discussion or to map out an algorithm that predicts the best choice mathematically.

*PRUNING:*  
The process of cutting off non-predictive parts of a model is called “pruning.” We change the model by deleting the child nodes of a branch node. By removing superfluous structure, pruning mechanisms reduce the size of a model and often improve its accuracy.

*ENTROPY:*

The entropy characterizes the impurity of an arbitrary collection of examples.

A decision tree is built top-down from a root node and involves partitioning the data into subsets that contain instances with similar values (homogenous) and we use entropy to calculate the homogeneity of a sample. If the sample is completely homogeneous the entropy is zero and if the sample is an equally divided it has entropy of one.

**Entropy(Set) = I(Set) = -Σki=1P(valuei).log2(P(valuei))**

Where **P(valuei)** is the probability of getting the **ith** value when randomly selecting one from the set

*INFORMATION GAIN:*

Information Gain is the expected reduction in entropy caused by partitioning the examples according to a given attribute. It is based on the decrease in entropy after a dataset is split on an attribute

**IG(attribute)= I(parent) – I(attribute)**

Where: **I(parent)** is the entropy of all data at parent node and  
**I(attribute)** is Child’s expected entropy for ‘size’ split

**MANUAL CALCULATION:**

**Step 1:** Calculate entropy of Target

ETarget=-[(5/16 log25/16)+( 5/16 log25/16)+ (6/16 log26/16)]

= 1.58

**Step 2:** Calculate entropy of Predictors

*Entropy of Demand:*

EHeavy= -[(3/7 log23/7)+(2/7log22/7)+ (2/7log22/7)]  
= 1.56

EModerate= -[(2/4log22/4)+( 1/4log21/4)+ (1/4log21/4)]  
= 1.5

Elow= -[(2/5log22/5)+ (3/5log23/5)]  
= 0.97

*Information Gain of Demand:*

IGDemand= 1.58-[(1.56\*7/16)+(1.5\*4/16)+(0.97\*5/16)]  
=0.22

*Entropy of Strategic:*

Eyes= -[(3/9 log23/9)+(3/9log23/9)+ (3/9log23/9)]  
=1.485

ENo= -[(2/6log22/6)+( 2/6log22/6)+ (3/6log23/6)]  
=1.545  
  
*Information Gain of Strategic:*

IGStrategic= 1.58-[(1.485\*9/16)+(1.545\*7/16)]  
=0.006

*Entropy of Campaign:*

Eaggressive= -[(4/9 log24/9)+(4/9log24/9)+ (1/9log21/9)]  
=1.54

Elowkey= -[(1/7log21/7)+( 1/7log21/7)+ (5/7log25/7)]  
=1.13  
  
*Information Gain of Campaign:*

IGCampaign= 1.58-[(1.54\*9/16)+(1.13\*7/16)]  
=0.293

**PART 2:**# Check if XLConnect is already installed:

any(grepl("XLConnect",

installed.packages()))

#If FALSE, install package XLConnect:

install.packages("XLConnectJars")

install.packages("XLConnect")

install.packages("rJava")

install.packages("rpart")

install.packages("rpart.plot")

#or (used this)

dyn.load('/Library/Java/JavaVirtualMachines/jdk1.8.0\_66.jdk/Contents/Home/jre/lib/server/libjvm.dylib')

library(XLConnectJars)

library(XLConnect)

library(rJava) ***(i) Implement Decision tree classification with Titanic dataset (predict Survived)***

#Read and display titanic train and test data from CSV:

library(XLConnect)

titanic\_train=read.csv("Desktop/Titanic\_train.csv")

titanic\_train

titanic\_test=read.csv("Desktop/Titanic\_test.csv")

titanic\_test

#Documentation for rpart and rpart.plot:

?rpart

?rpart.plot

?prp

#Display tree using the rpart library:

library(rpart)

library(rpart.plot)

tree\_model=rpart(Survived~Pclass+Sex+Age+SibSp+Parch+Fare+Embarked, data=titanic\_train, method="class")

tree\_model

prp(tree\_model)

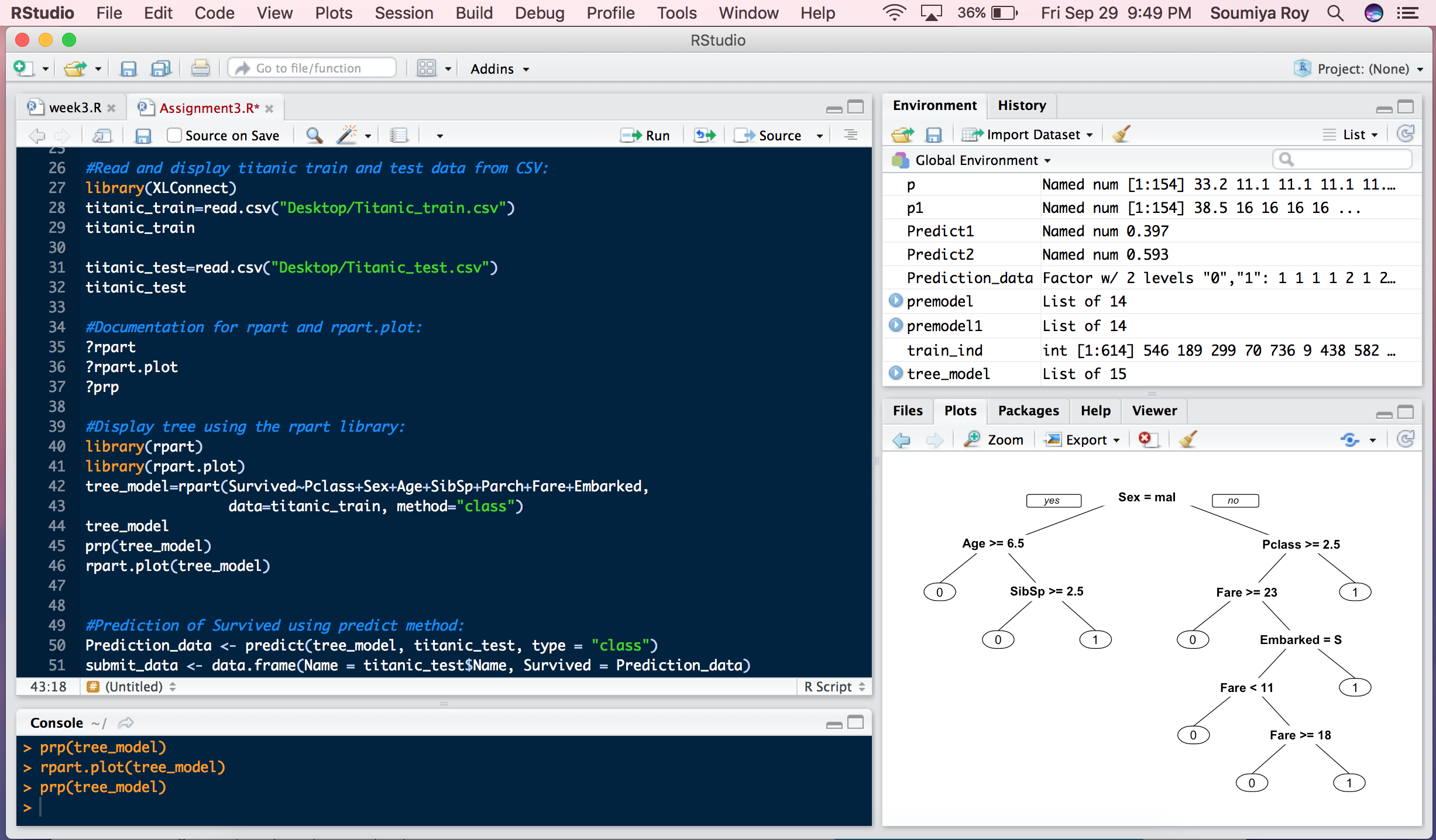
rpart.plot(tree\_model)

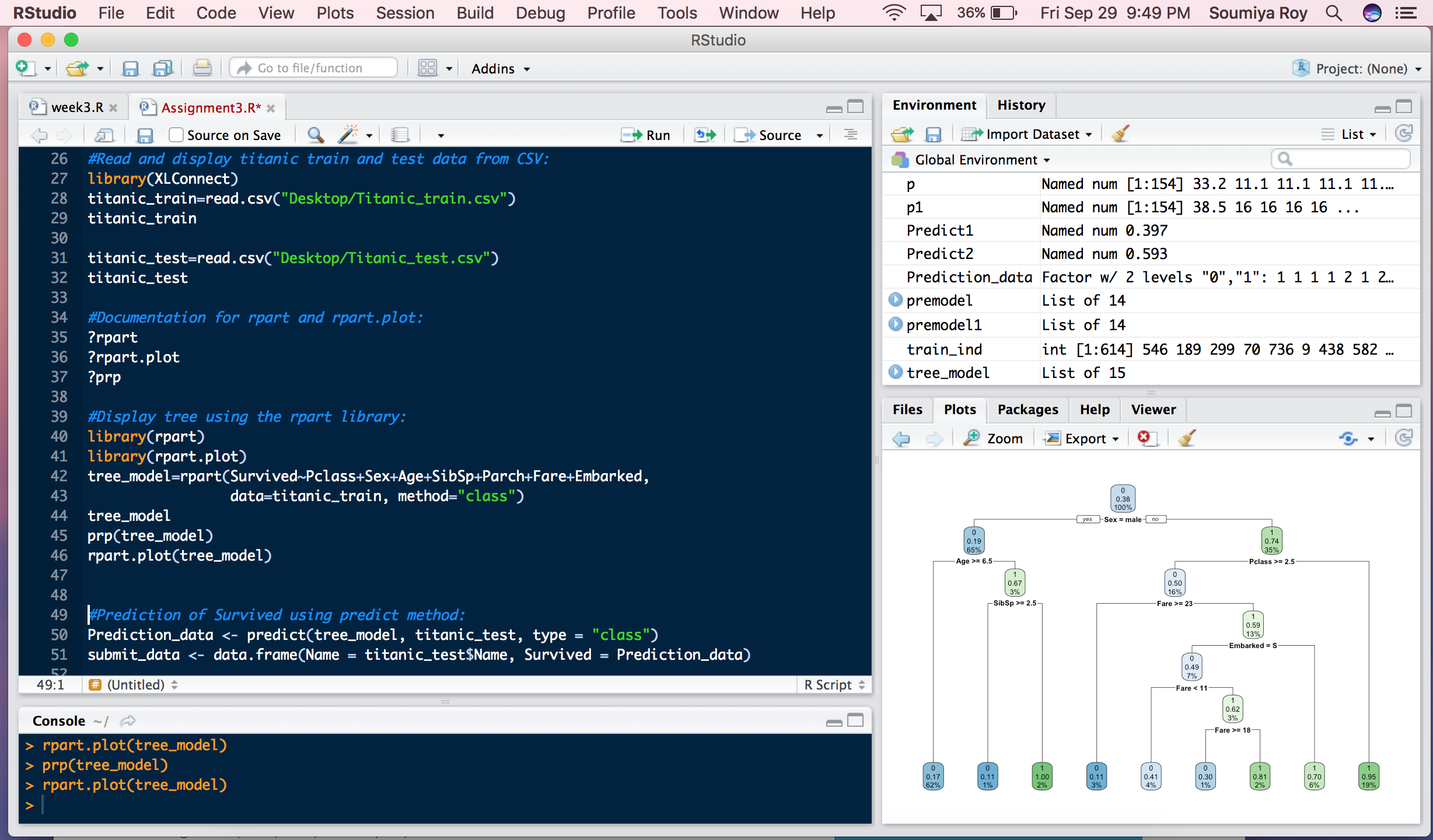
#Prediction of Survived using predict method:

Prediction\_data <- predict(tree\_model, titanic\_test, type = "class")

submit\_data <- data.frame(Name = titanic\_test$Name, Survived = Prediction\_data)

***Output:***

******

******

***(ii) Decision tree regression Energy Efficiency Dataset (outcome y1 or y2) in R***

#Read and display Energy efficiency data from Excel sheet:

library(XLConnect)

energy\_data = readWorksheetFromFile("Desktop/Energy\_Efficiency.xlsx",sheet = 1)

energy\_data

#Extract 80% value of the energy dataset:

energy\_train<-floor(0.8 \* nrow(energy\_data))

energy\_train

#Set the seed to 13:

set.seed(13)

train\_ind<-sample(seq\_len(nrow(energy\_data)),size = energy\_train)

train\_ind

train\_data = energy\_data[train\_ind,]

train\_data

test\_data = energy\_data[-train\_ind,]

test\_data

modelY2 = rpart(Y2~ X1+X2+X3+X4+X5+X6+X7+X8, data = train\_data, method="anova")

modelY2

modelY1 = rpart(Y1~ X1+X2+X3+X4+X5+X6+X7+X8, data = train\_data, method="anova")

modelY1

#Predict the model:

p1= predict(modelY2,test\_data,type = "vector")

p= predict(modelY1,test\_data,type = "vector")

#Plot the data model:

plot(p1,type='h',col="green",ylim=c(10,max(p1,p)))

lines(p,type='h',lty=2,col="red")

***Output:***

