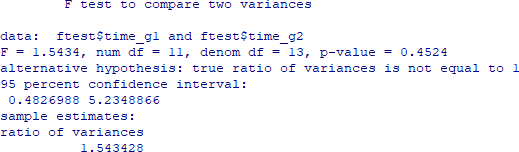
**PRACTICAL 01: ANALYSIS OF VARIANCE**

**Code:**

ftest<-read.csv(file.choose(),sep=",",header=T) var.test(ftest$time\_g1,ftest$time\_g2,alternative = "two.sided")

**Output:**

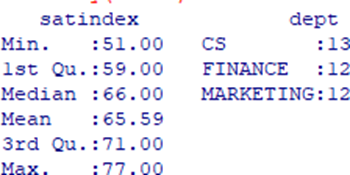
1. **One-way ANOVA**

**Code:** data1<-read.csv(file.choose(),sep = ",",header = T) names(data1)

**Output:**

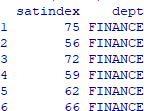
**Code:** summary(data1)

**Output:**

****

**Code:** head(data1)

**Output:**



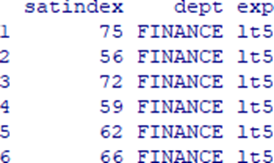
**Code:** anv<-aov(formula = satindex~dept,data=data1) summary(anv)

**Output:**

1. **Two-way ANOVA**

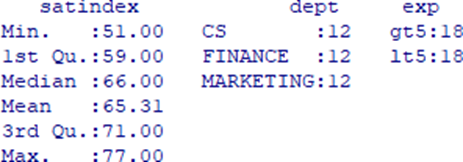
**Code:** head(data2)

**Output:**



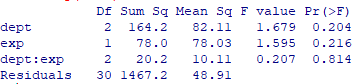
**Code:** summary(data2)

Output:



**Code:** anv1<-aov(formula = satindex~ dept+exp+dept\*exp,data = data2) summary(anv1)

**Output:**



**PRACTICAL 02: DECISION TREE - PART 01**

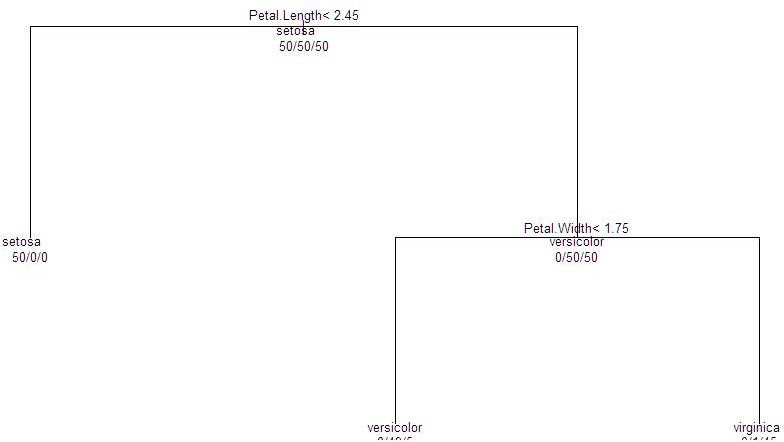
* **Code:**

mydata<-data.frame(iris) attach(mydata) install.packages("rpart") library(rpart)

model<-rpart(Species ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width, data=mydata, method="class")

plot(model) text(model,use.n=TRUE,all=TRUE,cex=0.8)

**Output:**



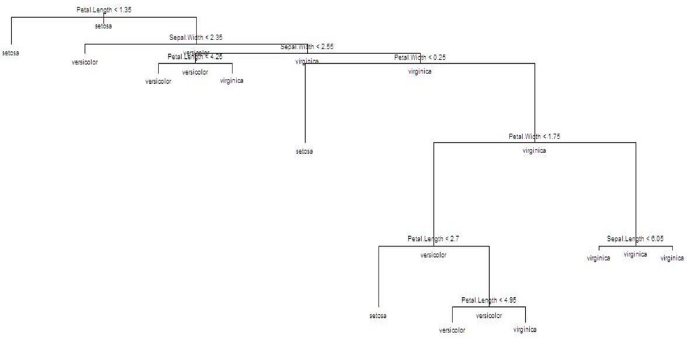
* **Code:**

install.packages("tree") library(tree)

model1<-tree(Species ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width, data=mydata, method="class", split="gini")

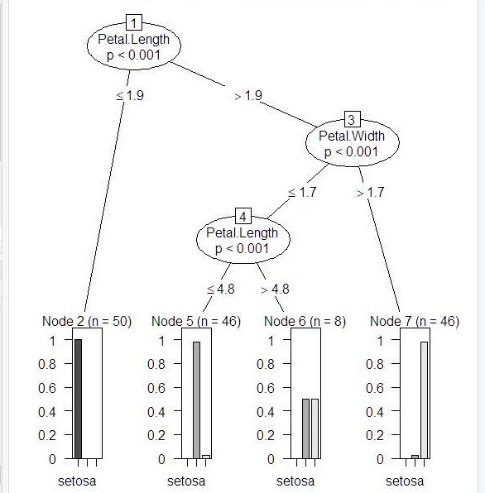
plot(model1) text(model1,all=TRUE,cex=0.6)

**Output:**

****

* **Code:**

install.packages("party") library(party)

model2<-ctree(Species ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width, data=mydata)

plot(model2)

**Output:**

**PRACTICAL 02: DECISION TREE - PART 02**

* **Code:**

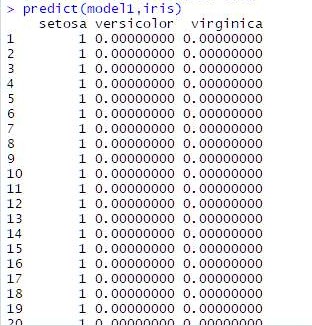
library(tree)

mydata<-data.frame(iris) attach(mydata)

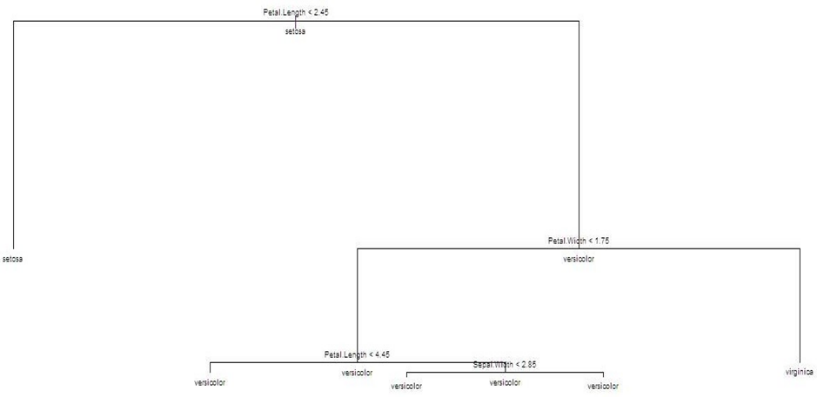
model1<-tree(Species ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width, data=mydata, method="class", control = tree.control(nobs = 150, mincut = 10))

plot(model1)

text(model1,all=TRUE,cex=0.6)

predict(model1,iris)

**Output:**

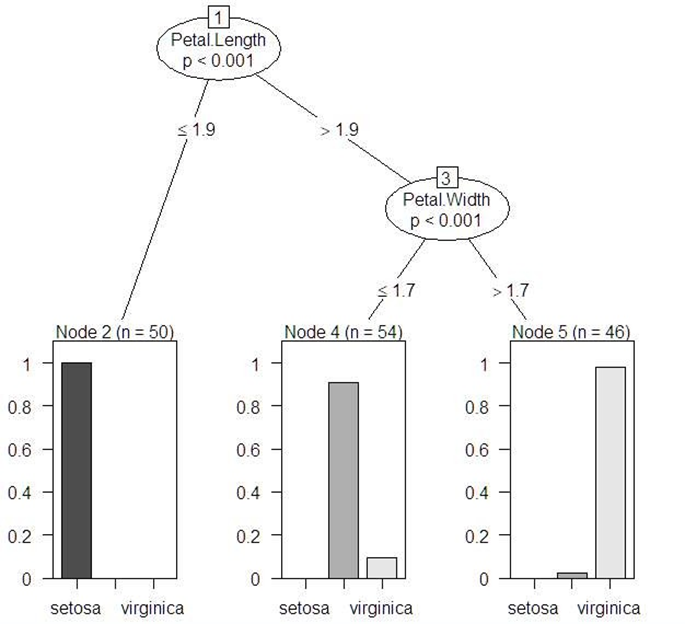


* **Code:**

model2<-ctree(Species ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width, data=mydata, controls = ctree\_control(maxdepth=2))

plot(model2)

**Output:**

****

**PRACTICAL 03: LOGISTIC REGRESSION**

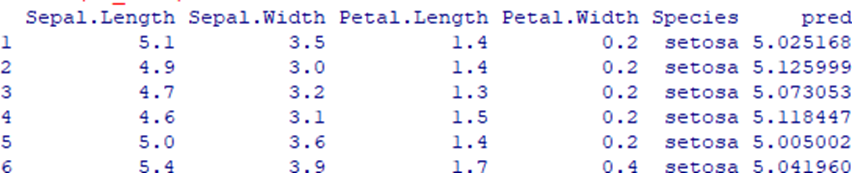
**Code:**

library(datasets)

ir\_data<- iris

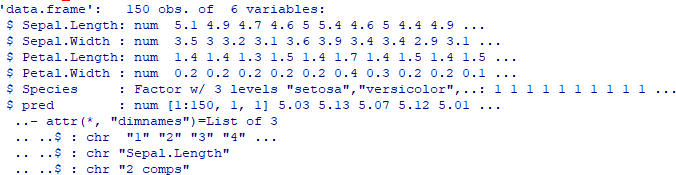
head(ir\_data)

**Output:**



**Code:**

str(ir\_data)

**Output:**

**Code:**

levels(ir\_data$Species)

**Output:**



**Code:**

sum(is.na(ir\_data))

**Output:**



**Code:**

ir\_data<-ir\_data[1:100,] set.seed(100)

samp<-sample(1:100,80) ir\_test<-ir\_data[samp,] ir\_ctrl<-ir\_data[-samp,]

install.packages("ggplot")

library(ggplot)

install.packages("ggplot2")

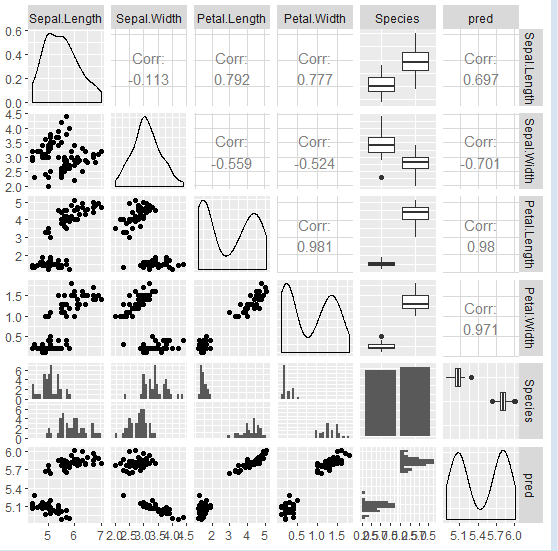
library(ggplot2)

install.packages("GGally")

library(GGally)

ggpairs(ir\_test)

**Output:**

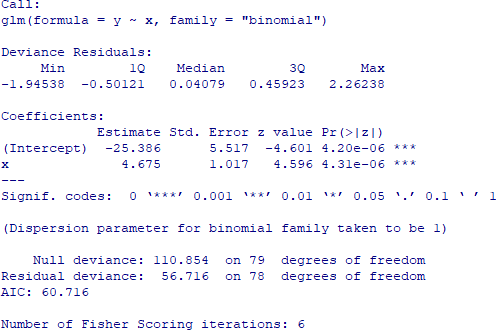


**Code:**

y<-ir\_test$Species; x<-ir\_test$Sepal.Length glfit<-glm(y~x, family = 'binomial')

summary(glfit)

**Output:**



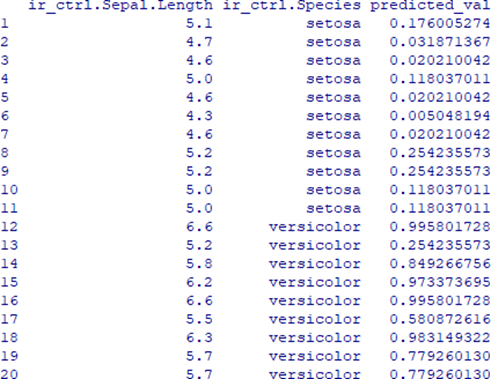
**Code:**

newdata<- data.frame(x=ir\_ctrl$Sepal.Length) predicted\_val<-predict(glfit, newdata, type="response")

prediction<-data.frame(ir\_ctrl$Sepal.Length, ir\_ctrl$Species,predicted\_val)

prediction

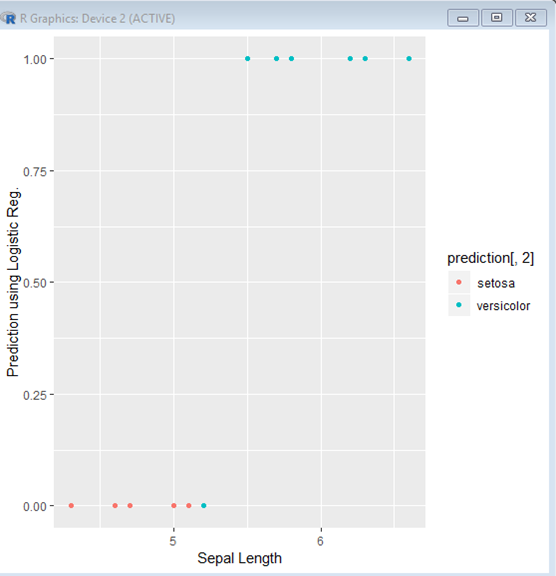
**Output:**



**Code:**

qplot(prediction[,1], round(prediction[,3]), col=prediction[,2], xlab = 'Sepal Length', ylab = 'Prediction using Logistic Reg.')

**Output:**



**PRACTICAL 04: TIME-SERIES FORECASTING**

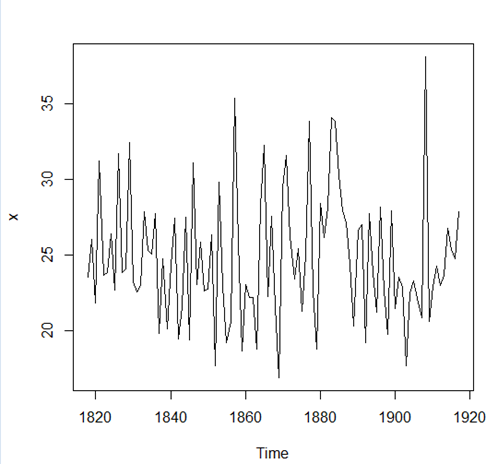
**Code:**

rain=read.csv("rain.csv")

raints=ts(rain,start=c(1818))

plot(raints)

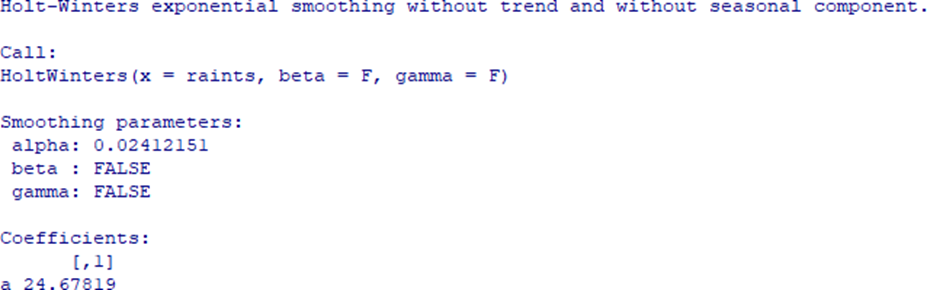
**Output:**



**Code:**

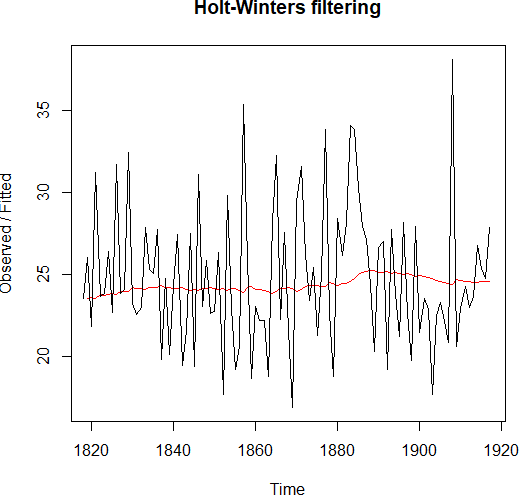
rainforecast=HoltWinters(raints,beta=F,gamma=F) rainforecast

**Output:**



**Code:**

plot(rainforecast)

**Output:**

**Code:**

names(rainforecast)

**Output:**

****

**Code:**

rainforecast$fitted

**Output:**

Time Series:

Start = 1819

End = 1917

Frequency = 1 xhat level

1819 23.56000 23.56000

1820 23.62054 23.62054

1821 23.57808 23.57808

...

...

...

...

1915 24.57541 24.57541

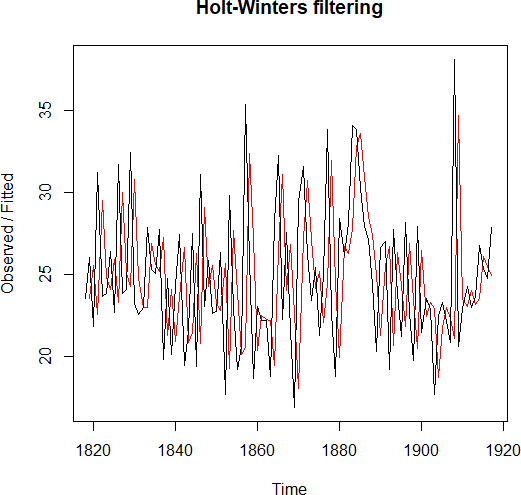
1916 24.59433 24.59433

1917 24.59905 24.59905

**Code:**

r2=HoltWinters(raints,alpha=0.8,beta=F,gamma=F) plot(r2)

**Output:**

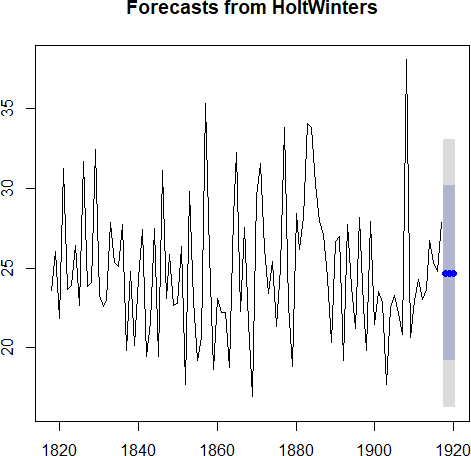


**Code:**

install library forecast

rf=forecast:::forecast.HoltWinters(rainforecast,h=3)

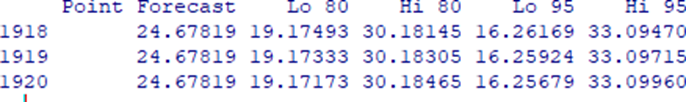
plot(rf)

**Output:**

**Code:**

rf

**Output:**



**PRACTICAL 05: TIME-SERIES FORECASTING**

**Code:**

# Import necessary modules

from sklearn.neighbors import KNeighborsClassifier

from sklearn.model\_selection import train\_test\_split

from sklearn.datasets import load\_iris

# Loading data

irisData = load\_iris()

# Create feature and target arrays

X = irisData.data

y = irisData.target

# Split into training and test set

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

knn = KNeighborsClassifier(n\_neighbors=7)

knn.fit(X\_train, y\_train)

# Predict on dataset which model has not seen before

print(knn.predict(X\_test))

print(knn.score(X\_test, y\_test))

**Output:**

[1 0 2 1 1 0 1 2 2 1 2 0 0 0 0 1 2 1 1 2 0 2 0 2 2 2 2 2 0 0]

0.9666666666666667