Experiment No. : 1. Date : .....

1. Write a program to implement Random forest using any data sets

```
data(iris)
str(iris)

install.packages("caTools")
install.packages("randomForest")

library(caTools)
library(randomforest)

split ← sample.split(iris, SplitRatio = 0.7)
split

train ← subset(iris, split == "TRUE")
test ← subset(iris, split == "FALSE")

set.seed(120)
classifier_RF = randomForest(x = train[-5],
                             y = train$species,
                             ntree = 500)
classifier_RF
~~classifier_RF, newdata = test[-5]~~

y_pred = predict(classifier_RF, newdata = test[-5])

confusion_mtx = table(test[,5], y_pred)
confusion_mtx
```

Experiment No. :..... Date :..

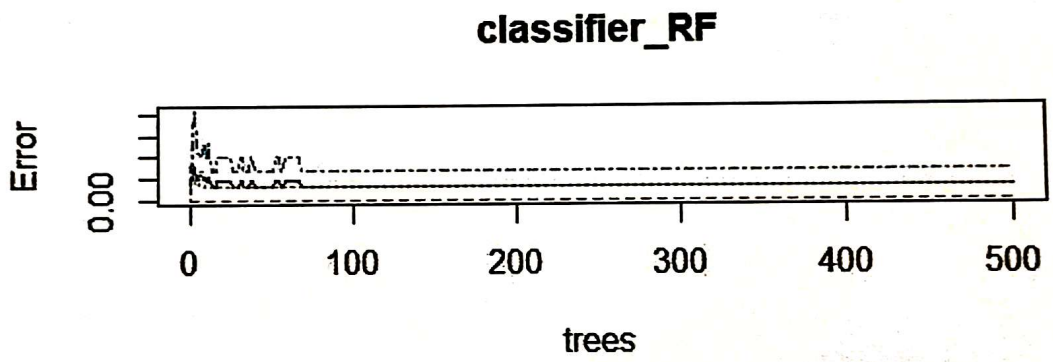
```
plot(classifier_RF)
importance(classifier_RF)
varImpPlot(classifier_RF)
```

28/9/22

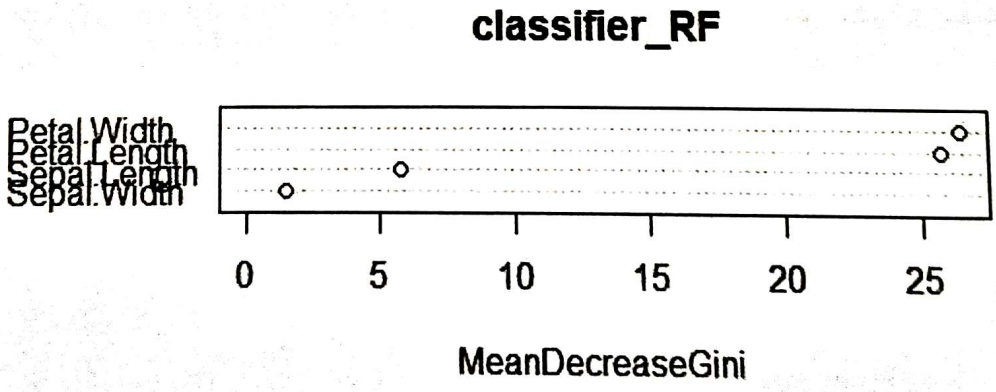
10/10

```
> classifier_RF

Call:
 randomForest(x = train[-5], y = train$species, ntree = 500)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 2

        OOB estimate of  error rate: 3.33%
Confusion matrix:
           setosa versicolor virginica class.error
setosa         30          0         0  0.00000000
versicolor      0         29         1  0.03333333
virginica       0          2        28  0.06666667
```

	setosa	versicolor	virginica
setosa	20	0	0
versicolor	0	20	0
virginica	0	3	17




```
> importance(classifier_RF)
             MeanDecreaseGini
Sepal.Length         5.772467
Sepal.Width          1.419426
Petal.Length        25.728012
Petal.Width         26.374384
```



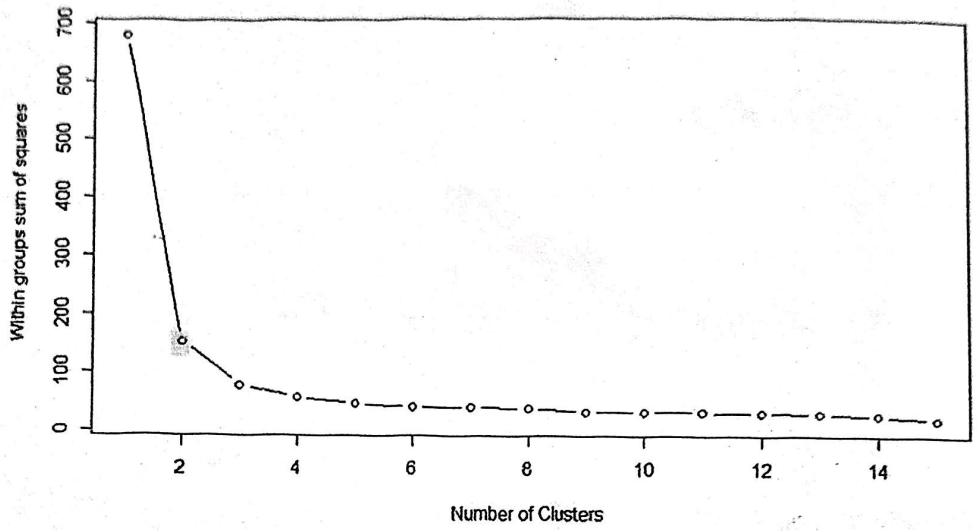
Experiment No. : 2 Date :

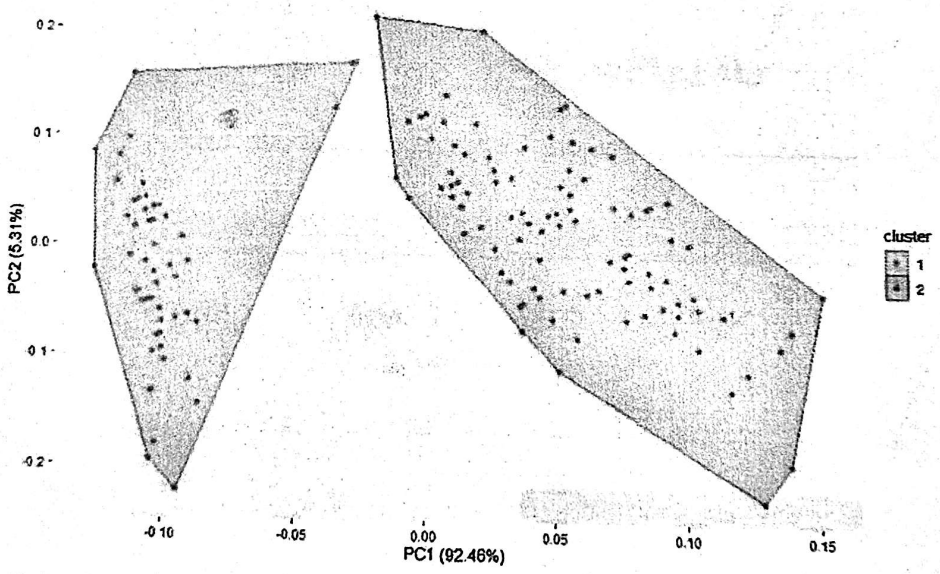
3. Implement any clustering technique

Hierarchial clustering

```
#install.packages("dplyr")
# install.packages("ggplot2")
# install.packages("ggfortify")
library("ggplot2")
library("dplyr")
library("ggfortify")

summary(iris)
head(iris)
data <- select(iris, c(1:4))

wssplot <- function(data, nc=15, seed=1234){
        wss <-(nrow(data)-1) * sum(apply(data,2,var))
        for(i in 2:nc){
                set.seed(seed)
                wss[i] <- sum(kmeans(data, centers=i)$withinss)
        plot(1:nc, wss, type="b", xlab="No. of clusters",
                        ylab="within groups sum of squares")

        wss
}

Kmean <- kmeans(data,2)
Kmean$centers
autoplot(Kmean, data, frame = TRUE)
```

10/10


Within groups sum of squares
0
100
200
300
400
500
600
700
2
4
6
8
10
12
14
Number of Clusters


PC2 (5.31%)
0.2
0.1
0.0
-0.1
-0.2
-0.10
-0.05
0.00
0.05
0.10
0.15
PC1 (92.46%)
cluster
1
2

Experiment No. : 3 Date :

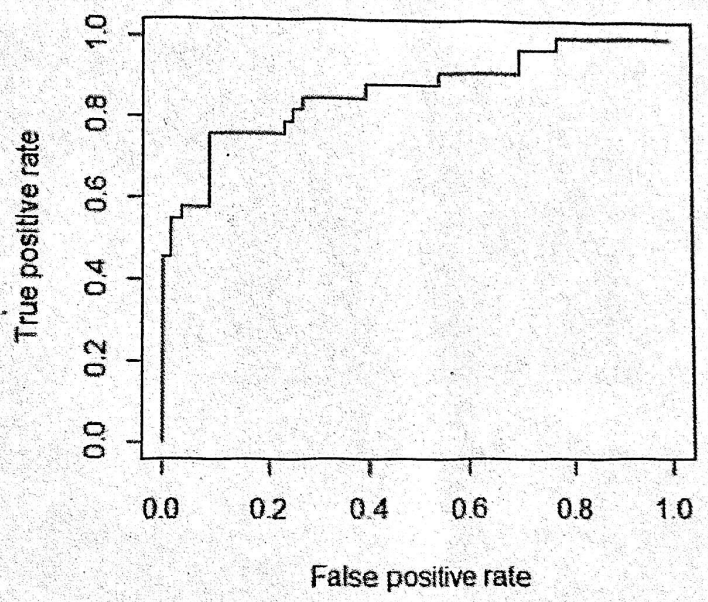
4. Implement linear and logistic regression

Logistic Regression

```
library(dyplr)
summary(mtcars)

install.packages("ROCR")

library(caTools)
library(ROCR)

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

train_reg <- subset(mtcars, split == "TRUE")
test_reg <- subset(mtcars, split == "FALSE")

logistic_model <- glm(vs ~ wt + disp, data = train_reg,
                      family = "binomial")

logistic_model
summary(logistic_model)
predict_reg <- predict(logistic_model, test_reg,
                       type = "response")

predict_reg
```

Experiment No. :..... Date :.....

```
predict_reg <- ifelse(predict_reg > 0.5, 1, 0)

table(test_reg$vs, predict_reg)

missing_classerr <- mean(predict_reg != test_reg$vs)
print(paste('Accuracy =', 1 - missing_classerr))

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

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(0.6, 0.4, auc, title = "AUC", box.lty = 1)
```

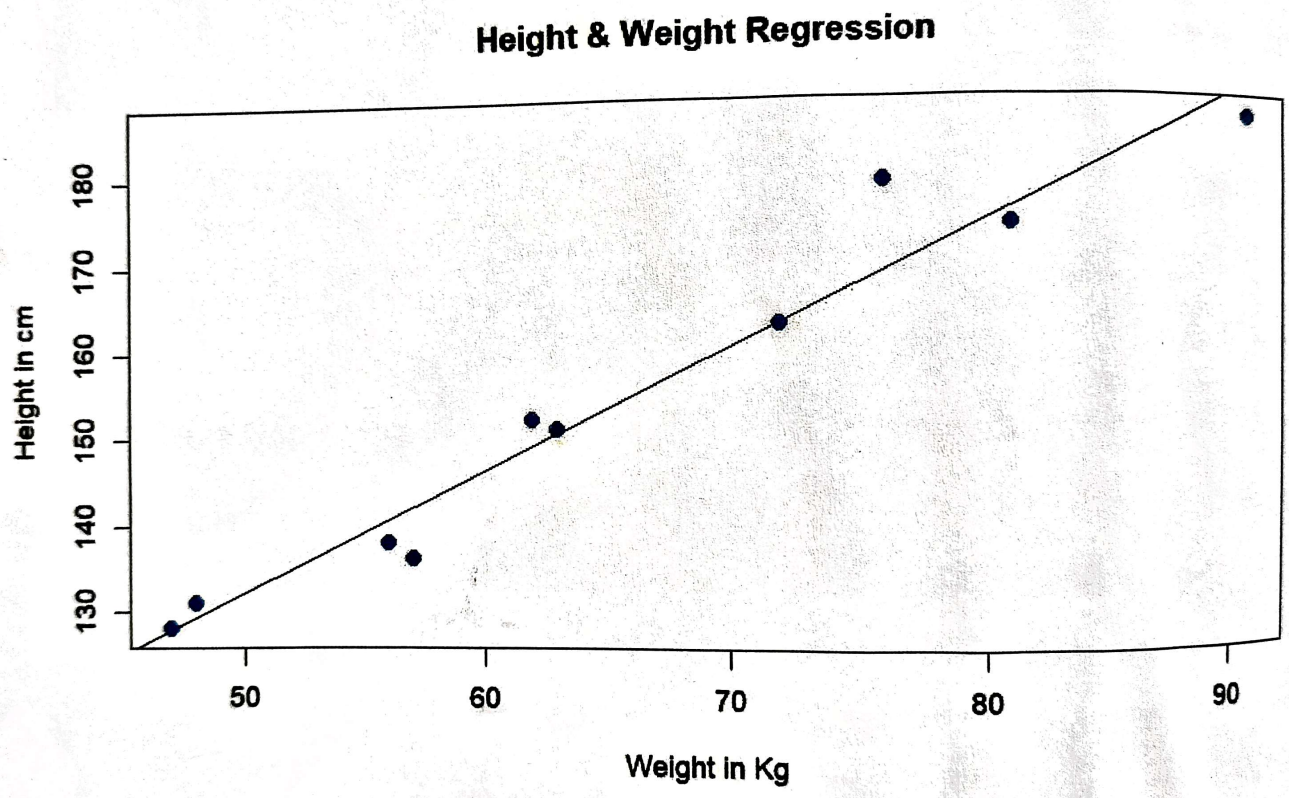

True positive rate
0.0
0.2
0.4
0.6
0.8
1.0
False positive rate

Experiment No. :..... Date :.....

```
#Linear Regression

Advertising <- read.csv("c:/users/ABC/Downloads/
                         Advertising.csv")
view(Advertising)
summary(Advertising)

head(Advertising)
install.packages("ggplot2")

library(ggplot2)

ggplot(Advertising, aes(x=TV, y=sales)) +
                    geom_point() + stat_smooth()

cor(Advertising$sales, Advertising$TV)

model <- lm(sales~TV, data = Advertising)
model

ggplot(Advertising, aes(TV, sales)) + geom_point()+
                    stat_smooth(method = lm)

summary(model)

confit(model)
Sigma(model) * 100/ mean(Advertising$sales)
```

10/10

8/10/22


Height & Weight Regression
Height in cm
130
140
150
160
170
180
50
60
70
80
90
Weight in Kg