Practical 2

Aim: Implement Decision tree classification techniques

install.packages("party")

library("party")

print(head(readingSkills))

library("party")

str(iris)

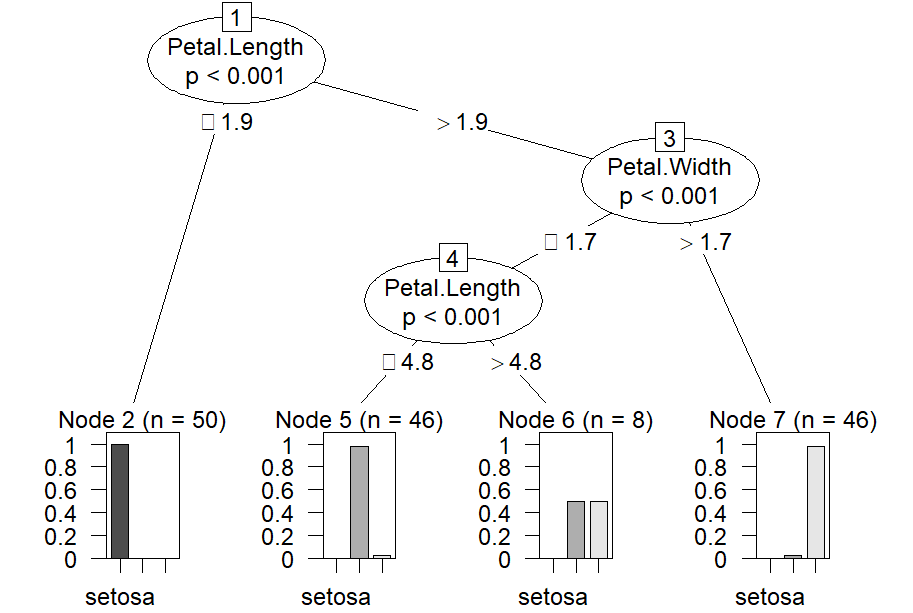
iris\_ctree<- ctree(Species ~ Sepal.Length + Sepal.Width +Petal.Length + Petal.Width, data=iris)

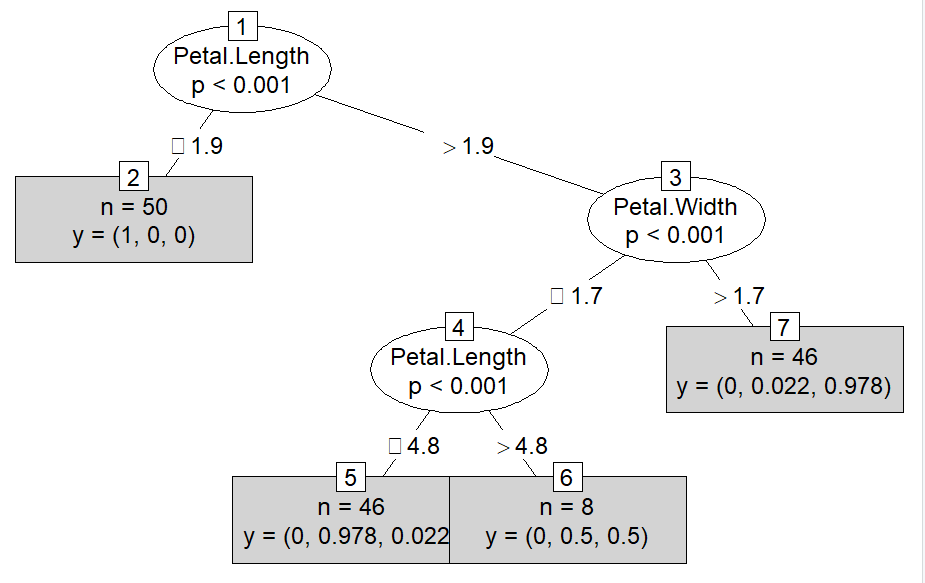
print(iris\_ctree)

plot(iris\_ctree)

plot(iris\_ctree, type="simple")

output:-





Practical 3

Aim: Classification using SVM

dataset = read.csv('Social\_Network\_Ads.csv')

dataset = dataset[3:5]

install.packages('caTools')

library(caTools)

set.seed(123)

split = sample.split(dataset$Purchased, SplitRatio = 0.75)

training\_set = subset(dataset, split == **TRUE**)

test\_set = subset(dataset, split == **FALSE**)

training\_set[-3] = scale(training\_set[-3])

test\_set[-3] = scale(test\_set[-3])

install.packages('e1071')

library(e1071)

classifier = svm(formula = Purchased ~ .,

                 data = training\_set,

                 type = 'C-classification',

                 kernel = 'linear')

classifier

y\_pred = predict(classifier, newdata = test\_set[-3])

y\_pred

cm = table(test\_set[, 3], y\_pred)

cm

set = training\_set

X1 = seq(min(set[, 1]) - 1, max(set[, 1]) + 1, by = 0.01)

X2 = seq(min(set[, 2]) - 1, max(set[, 2]) + 1, by = 0.01)

grid\_set = expand.grid(X1, X2)

colnames(grid\_set) = c('Age', 'EstimatedSalary')

y\_grid = predict(classifier, newdata = grid\_set)

plot(set[, -3],

     main = 'SVM (Training set)',

     xlab = 'Age', ylab = 'Estimated Salary',

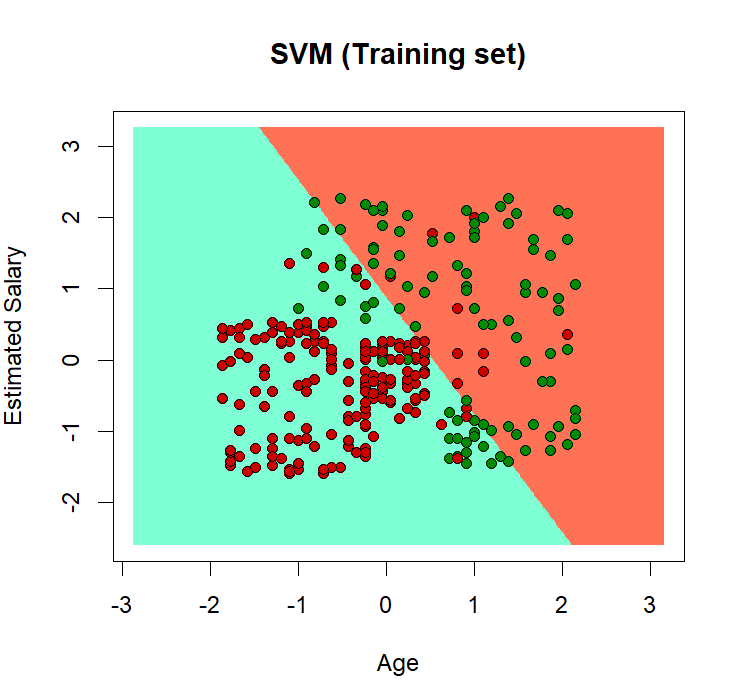
     xlim = range(X1), ylim = range(X2))

contour(X1, X2, matrix(as.numeric(y\_grid), length(X1), length(X2)), add = **TRUE**)

points(grid\_set, pch = '.', col = ifelse(y\_grid == 1, 'coral1', 'aquamarine'))

points(set, pch = 21, bg = ifelse(set[, 3] == 1, 'green4', 'red3'))

output:-



PRACTICAL NO : 4 Aim: Implement an application that stores big data in Hbase / MongoDB and manipulate it using R / Python

Practical 5

Aim: write program in R of Naive bayes theorem

install.packages("e1071")

install.packages("caTools")

install.packages("caret")

library(e1071)

library(caTools)

library(caret)

split <**-** sample.split(iris, SplitRatio **=** 0.7)

train\_cl <**-** subset(iris, split **==** "TRUE")

test\_cl <**-** subset(iris, split **==** "FALSE")

train\_scale <**-** scale(train\_cl[, 1:4])

test\_scale <**-** scale(test\_cl[, 1:4])

set.seed(120)

classifier\_cl <**-** naiveBayes(Species ~ ., data **=** train\_cl)

classifier\_cl

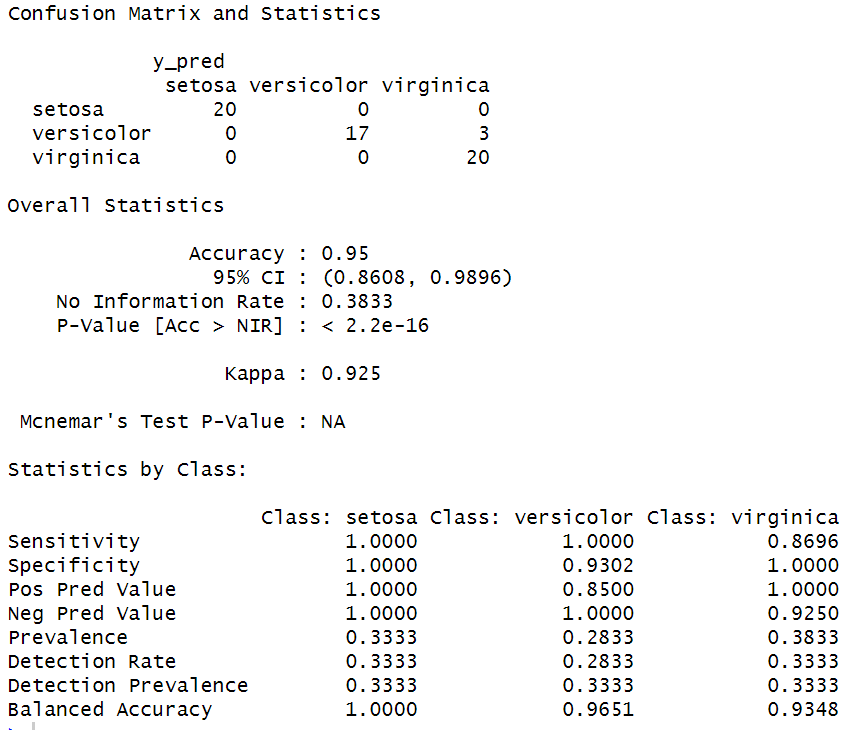
y\_pred <**-** predict(classifier\_cl, newdata **=** test\_cl)

cm <**-** table(test\_cl$Species, y\_pred)

cm

confusionMatrix(cm)

output:-



Practical 6

Aim: : WAP showing implementation of Regression model.

IQ <- rnorm(40, 30, 2)

IQ <- sort(IQ)

result <- c(0, 0, 0, 1, 0, 0, 0, 0, 0, 1,

1, 0, 0, 0, 1, 1, 0, 0, 1, 0,

0, 0, 1, 0, 0, 1, 1, 0, 1, 1,

1, 1, 1, 0, 1, 1, 1, 1, 0, 1)

df <- as.data.frame(cbind(IQ, result))

print(df)

png(file="LogisticRegressionGFG.png")

plot(IQ, result, xlab = "IQ Level",

ylab = "Probability of Passing")

g = glm(result~IQ, family=binomial, df)

curve(predict(g, data.frame(IQ=x), type="resp"), add=**TRUE**)

points(IQ, fitted(g), pch=30)

summary(g)

dev.off()

output:-

