**VIT-AP UNIVERSITY, ANDHRA PRADESH**

**CSE2047 – Data Analytics - Lab Sheet : 10**

**Academic year:** 2020-2021  **Branch/ Class:** B.Tech/M.Tech

**Semester:** Fall  **Date:**

**Faculty Name:** Prof. S.Gopikrishnan  **School:** SCOPE

**Student name:** Valiveti Manikanta bhuvanesh **Reg. no.: 19BCD7088**

**LAB 10**

**Clustering and Classification Algorithm implementation using R.**

* + - 1. **Use iris dataset for k means clustering in R**

df <- iris[, -5]

set.seed(240)

kmeans.re <- kmeans(df, centers = 3, nstart = 20)

kmeans.re

kmeans.re$cluster

plot(df[c("Sepal.Length", "Sepal.Width")])

plot(df[c("Sepal.Length", "Sepal.Width")], col = kmeans.re$cluster)

plot(df[c("Sepal.Length", "Sepal.Width")], col = kmeans.re$cluster, main = "K-means with 3 clusters")

kmeans.re$centers

kmeans.re$centers[, c("Sepal.Length", "Sepal.Width")]

points(kmeans.re$centers[, c("Sepal.Length", "Sepal.Width")], col = 1:3, pch = 8, cex = 3)

y\_kmeans <- kmeans.re$cluster

clusplot(df[, c("Sepal.Length", "Sepal.Width")],y\_kmeans,lines = 0,shade = TRUE,color = TRUE,labels = 2,

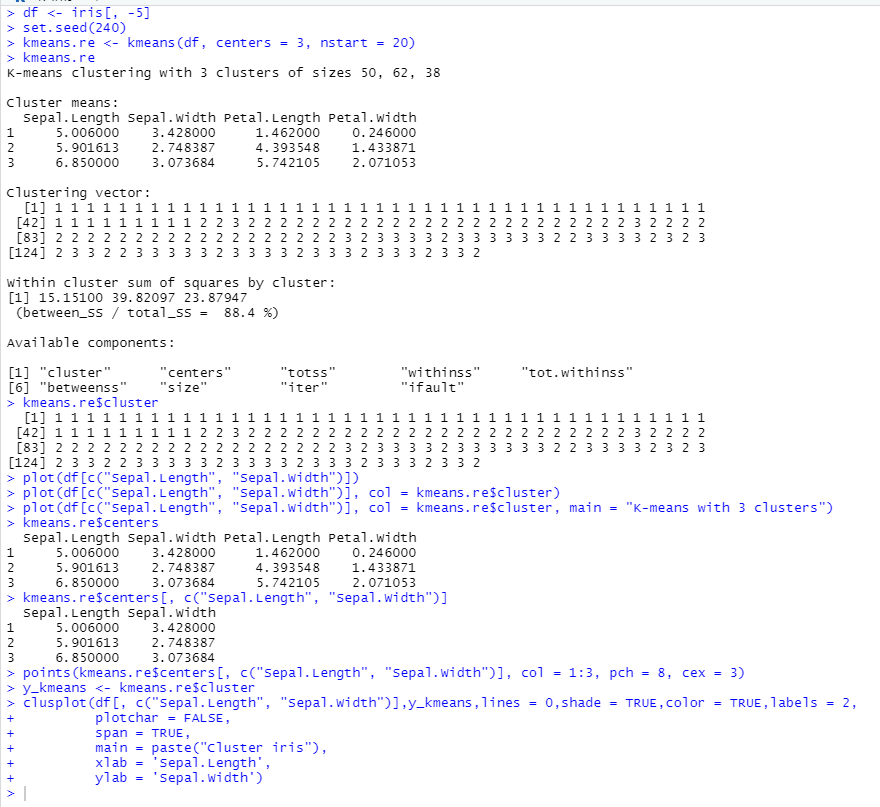
plotchar = FALSE,

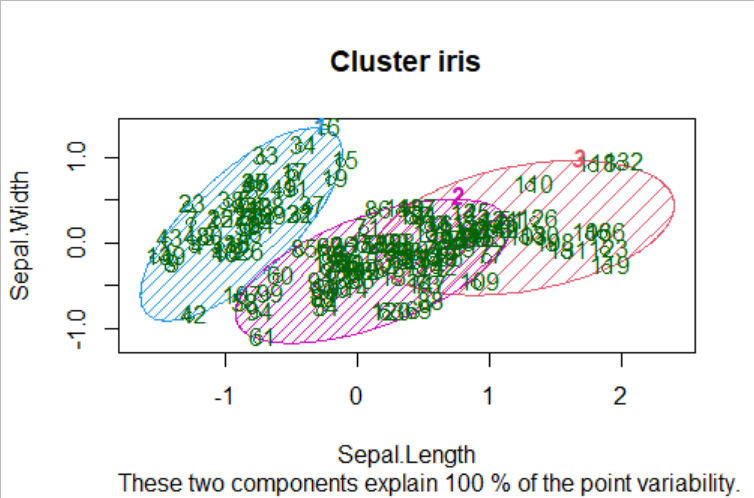
span = TRUE,

main = paste("Cluster iris"),

xlab = 'Sepal.Length',

ylab = 'Sepal.Width')





* + - 1. **Use readingSkills dataset for all classification practice which is default in party package**
         1. **Logistic regression**

df<-readingSkills[c(1:105), ]

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

split

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

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

logistic\_model <- glm(nativeSpeaker ~ age + shoeSize + score, data = train\_reg, family = "binomial")

summary(logistic\_model)

predict\_reg <- predict(logistic\_model,test\_reg, type = "response")

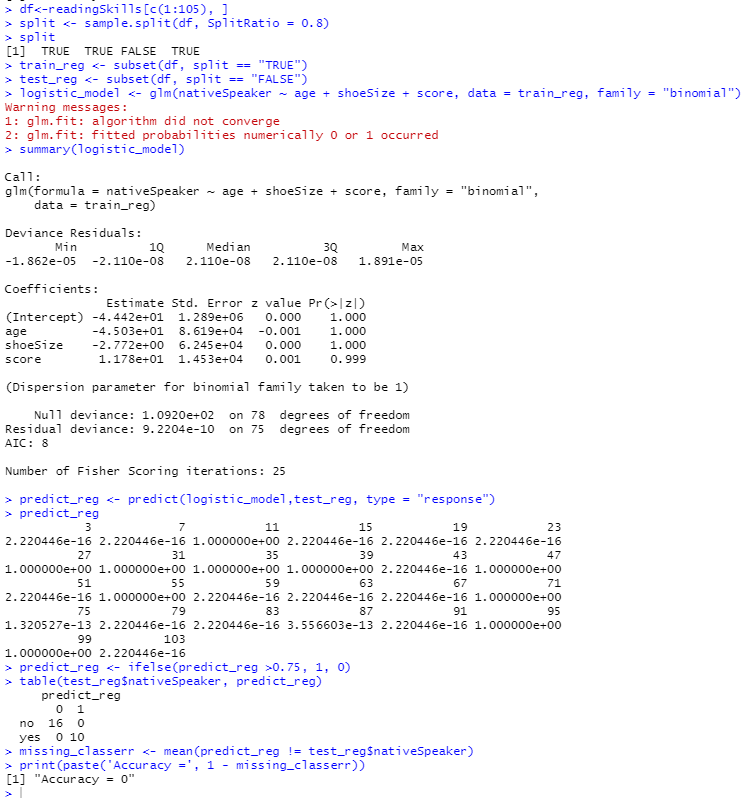
predict\_reg

predict\_reg <- ifelse(predict\_reg >0.75, 1, 0)

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

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

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



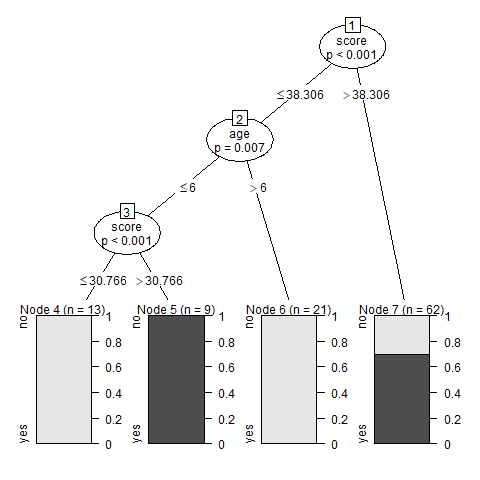
* + - * 1. **Decision trees**

png(file = "decision\_tree.png")

output.tree <- ctree(nativeSpeaker ~ age + shoeSize + score,data = df)

plot(output.tree)

dev.off()



* + - * 1. **Support Vector Machines (iris data – default)**

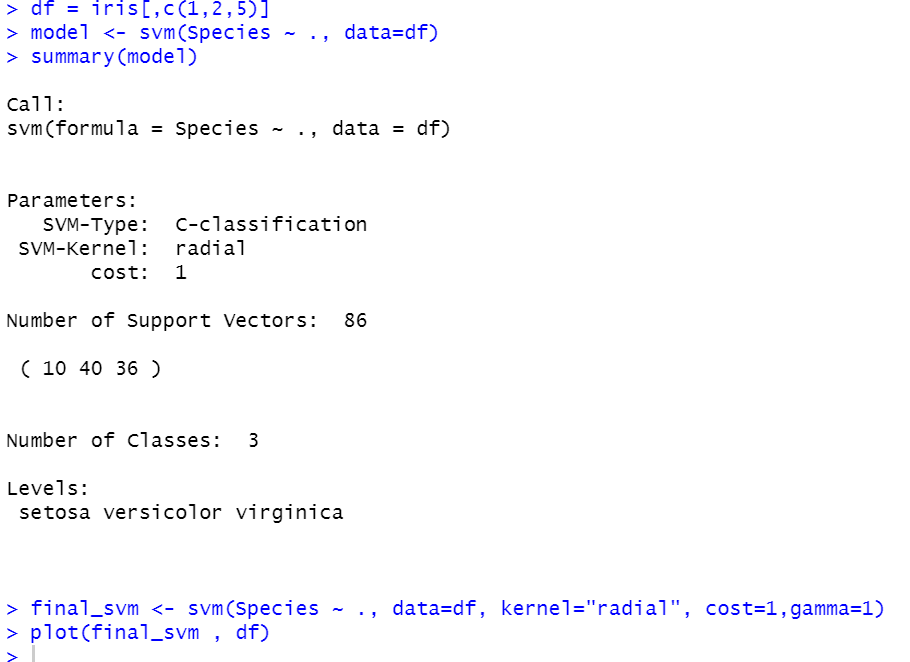
df = iris[,c(1,2,5)]

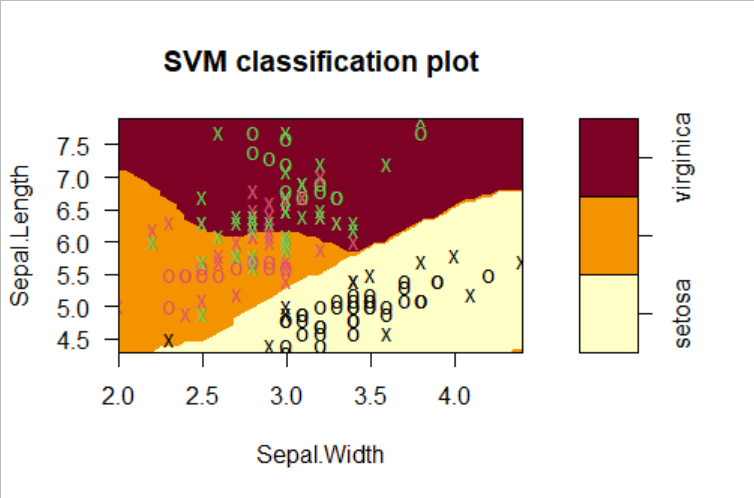
model <- svm(Species ~ ., data=df)

summary(model)

final\_svm <- svm(Species ~ ., data=df, kernel="radial", cost=1,gamma=1)

plot(final\_svm , df)





* + - * 1. **Naive Bayes Classifier (use hsbdata.csv)**

df<-hsb

set.seed(7267166)

trainIndex=createDataPartition(df$prog, p=0.7)$Resample1

train=df[trainIndex, ]

test=df[-trainIndex, ]

print(table(df$prog))

NBclassfier=naiveBayes(prog~science+socst, data=train)

print(NBclassfier)

Print=function(model){

trainPred=predict(model, newdata = train, type = "class")

trainTable=table(train$prog, trainPred)

testPred=predict(NBclassfier, newdata=test, type="class")

testTable=table(test$prog, testPred)

trainAcc=(trainTable[1,1]+trainTable[2,2]+trainTable[3,3])/sum(trainTable)

testAcc=(testTable[1,1]+testTable[2,2]+testTable[3,3])/sum(testTable)

message("Contingency Table for Training Data")

print(trainTable)

message("Contingency Table for Test Data")

print(testTable)

message("Accuracy")

print(round(cbind(trainAccuracy=trainAcc, testAccuracy=testAcc),3))

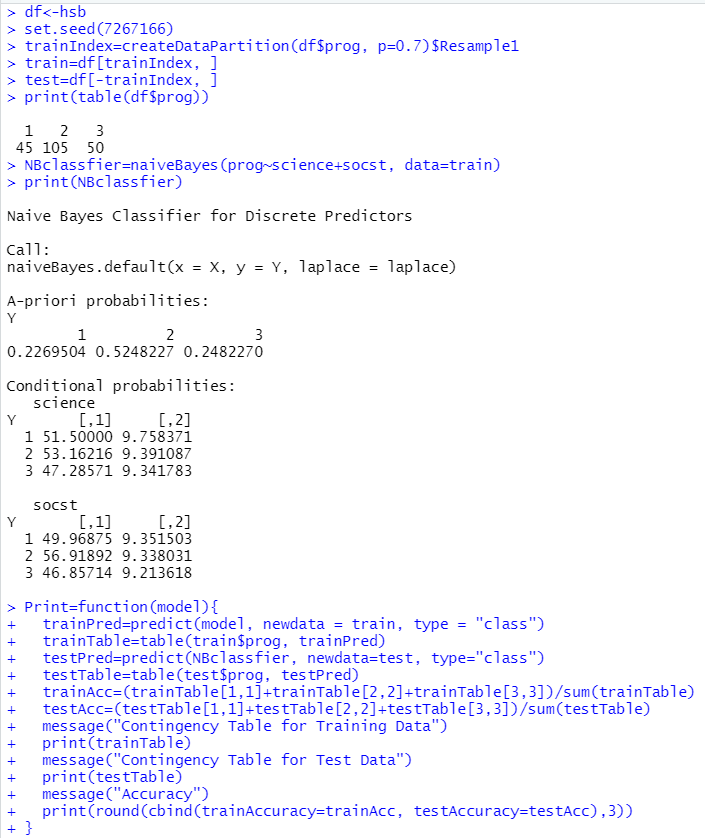
}

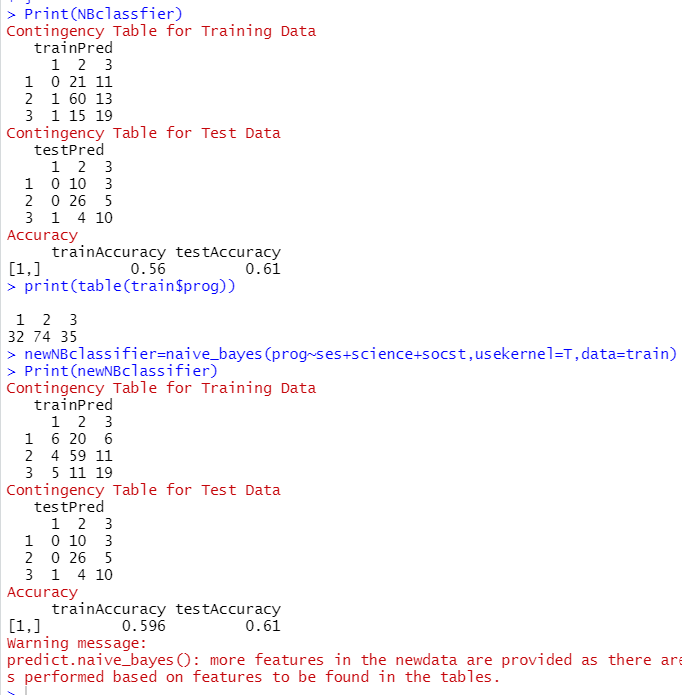
Print(NBclassfier)

print(table(train$prog))

newNBclassifier=naive\_bayes(prog~ses+science+socst,usekernel=T,data=train)

Print(newNBclassifier)





* + - * 1. **k-Nearest Neighbour (iris data)**

df = iris

df = df[-c(1,8)]

iris\_tr\_feat <- df[,1:4]

set.seed(1)

train\_pred <- knn(iris\_tr\_feat, iris\_tr\_feat, df$Species, k=3)

train\_pred[1:10]

accuracy <- mean(train\_pred == df$Species)

cat("Training Accuracy: ", accuracy, sep='')

* + - 1. **Use winequality dataset for all classification practice and use quality as predictor variable**
         1. **Logistic regression**

df <- read.csv('winequality.csv')

df <- df[,c(1,9,11,12)]

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

split

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

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

logistic\_model <- glm( quality ~ fixed.acidity+pH + alcohol,data = df)

logistic\_model

summary(logistic\_model)

predict\_reg <- predict(logistic\_model,test\_reg, type = "response")

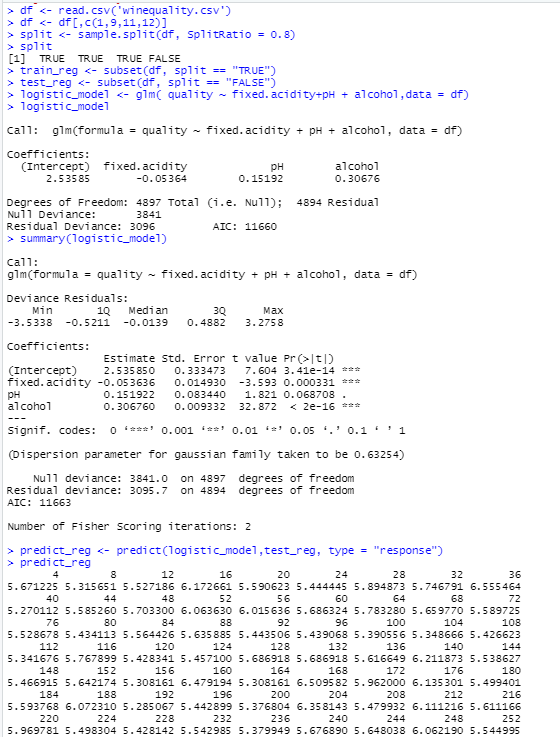
predict\_reg

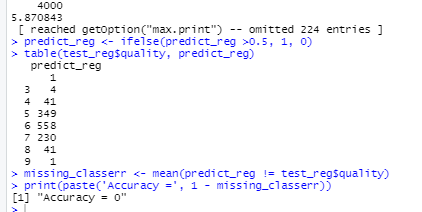
predict\_reg <- ifelse(predict\_reg >0.5, 1, 0)

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

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

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





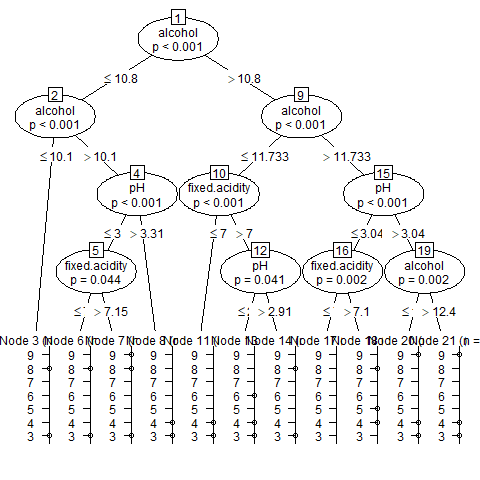
* + - * 1. **Decision trees**

png(file = "decision\_tree1.png")

output.tree <- ctree(quality ~ fixed.acidity + pH + alcohol,data = df)

plot(output.tree)

dev.off()



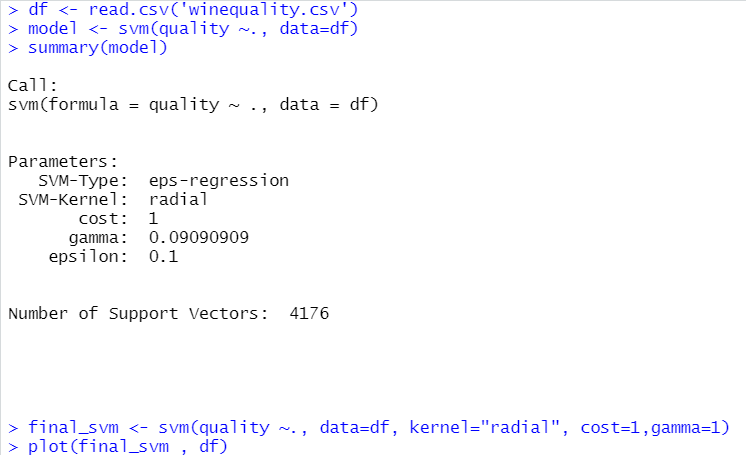
* + - * 1. **Support Vector Machines**

model <- svm(quality ~., data=df)

summary(model)

final\_svm <- svm(quality ~., data=df, kernel="radial", cost=1,gamma=1)

plot(final\_svm , df)



* + - * 1. **Naive Bayes Classifier**

split <- sample.split(df, SplitRatio = 0.7)

trainl <- subset(df, split == "TRUE")

testl <- subset(df, split == "FALSE")

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

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

set.seed(120)

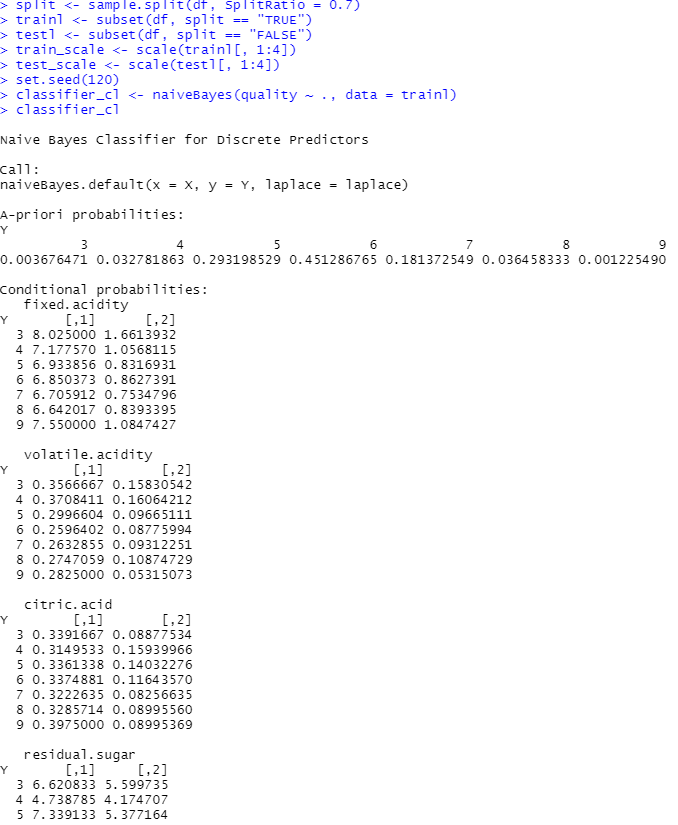
classifier\_cl <- naiveBayes(quality ~ ., data = trainl)

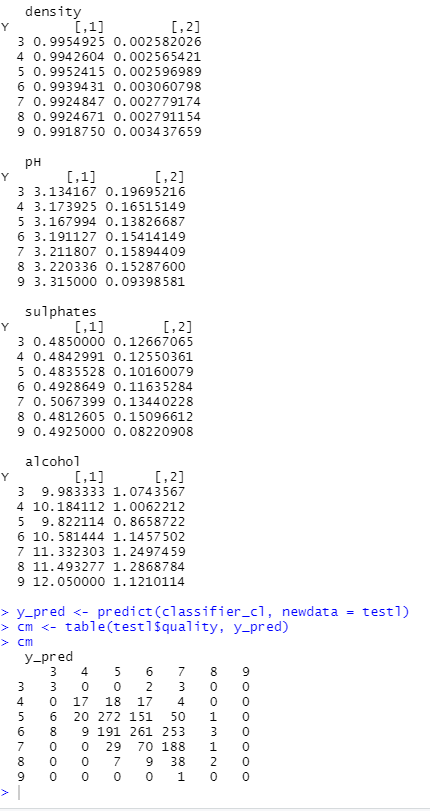
classifier\_cl

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

cm <- table(testl$quality, y\_pred)

cm





* + - * 1. **k-Nearest Neighbour**

df1<- df[,1:4]

set.seed(1)

train\_pred <- knn(df1, df1, df$quality, k=3)

train\_pred[1:10]

accuracy <- mean(train\_pred == df$quality)

cat("Training Accuracy: ", accuracy, sep='')

