Prob -> Prepare the model to classify a glass type

Soln ->

> #KNN algos p1

> # predicting glass

> # Read the dataset

> glass <- read.csv(file.choose())

> View(glass)

> #First colum in dataset is id which is not required so we will be taking out

> #glass <- glass[-1]

>

>

> table(glass$Type)

1 2 3 5 6 7

70 76 17 13 9 29

>

> # Replace B with Benign and M with Malignant. Diagnosis is factor with 2 levels that is B and M. We also replacing these two entery with Benign and Malignat

> glass$Type <- factor(glass$Type)

>

> # table or proportation of enteries in the datasets. What % of entry is Bengin and % of entry is Malignant

> round(prop.table(table(glass$Type))\*100,1)

1 2 3 5 6 7

32.7 35.5 7.9 6.1 4.2 13.6

>

> #Create a function to normalize the data

> norm <- function(x){

+ return((x-min(x))/(max(x)-min(x)))

+ }

>

> #we need to take random data splitting because the given data has output in sequence which will prove -

> #if we use sequential splitting

> train\_index <- sample(1:nrow(glass), 0.7 \* nrow(glass))

> test\_index <- setdiff(1:nrow(glass), train\_index)

>

> train<- glass[train\_index,-10]

> train\_label<-glass[train\_index,"Type"]

>

> test<-glass[test\_index,-10]

> test\_label<-glass[test\_index,"Type"]

>

> #Apply the normalization function

>

> train\_n<-as.data.frame(lapply(train[1:9], norm))

> test\_n<-as.data.frame(lapply(test[1:9], norm))

>

>

>

>

> # Build a KNN model on taining dataset

> library("class")

> # Building the KNN model on training dataset and also need labels which we are including c1. Once we build the preduction model

> # we have to test on test dataset

> test\_acc <- NULL

> train\_acc <- NULL

> for (i in seq(3,200,2))

+ {

+ train\_glass\_pred <- knn(train=train\_n,test=train\_n,cl=train\_label,k=i)

+ train\_acc <- c(train\_acc,mean(train\_glass\_pred==train\_label))

+ t\_a<-c()

+ test\_glass\_pred <- knn(train = train\_n, test = test\_n, cl = train\_label, k=i)

+ test\_acc <- c(test\_acc,mean(test\_glass\_pred==test\_label))

+ }

There were 50 or more warnings (use warnings() to see the first 50)

>

>

> # Testing Accuracy

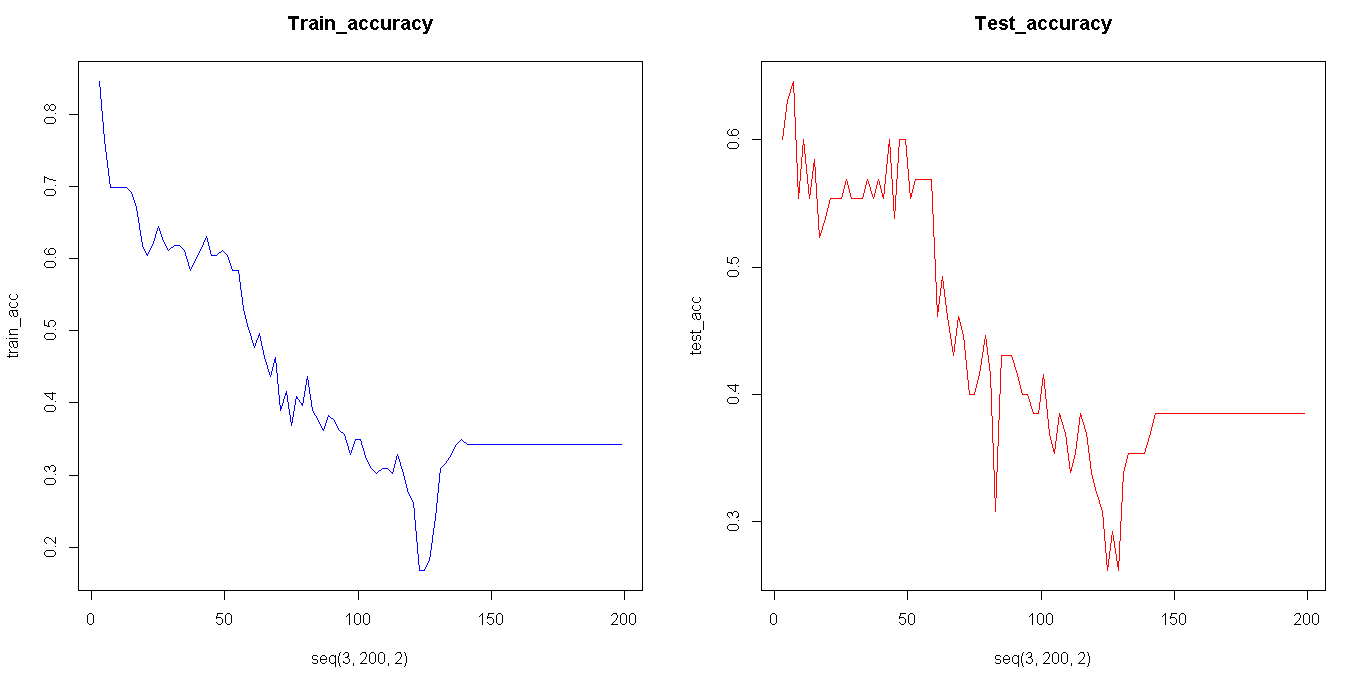
>

> # Plotting 2 different graphs on same window

> par(mfrow=c(1,2)) # c(1,2) => indicates 1 row and 2 columns

> plot(seq(3,200,2),train\_acc,type="l",main="Train\_accuracy",col="blue")

> plot(seq(3,200,2),test\_acc,type="l",main="Test\_accuracy",col="red")

> 

> acc\_neigh\_df <- data.frame(list(train\_acc=train\_acc,test\_acc=test\_acc,neigh=seq(3,200,2)))

> # Plotting 2 different graphs on same co-ordinate axis

> #install.packages("ggplot2")

> library(ggplot2)

> ggplot(acc\_neigh\_df,aes(x=neigh))+

+ geom\_line(aes(y=train\_acc,colour="train\_acc"),lwd=1.5)+

+ geom\_line(aes(y=test\_acc,colour="test\_acc"),lwd=1.5)+

+ scale\_fill\_manual(" ",breaks=c("train\_acc","test\_acc"),values = c("train\_acc"="green","test\_acc"="red"))

> 

> #from the graph max accuracy can be acheved at k = 50

> glass\_pred\_te <- knn(train = train\_n, test = test\_n, cl = train\_label, k=50)

> accuracy\_test<- sum(glass\_pred\_te==test\_label)/65

> glass\_pred\_tr <- knn(train = train\_n, test = train\_n, cl = train\_label, k=50)

> accuracy\_train<- sum(glass\_pred\_tr==train\_label)/149

> accuracy\_test

[1] 0.5846154

> accuracy\_train

[1] 0.6107383

# model seems to be underfitting this problem can only be solved if we have high quantity and quality of observation

Prob 2 -> Prepare the model to classify the animal type

Soln ->

> #KNN zoo prob

>

> #KNN algos p1

> # predicting glass

> # Read the dataset

> zoo1 <- read.csv(file.choose())

> View(zoo1)

> #First colum in dataset is id which is not required so we will be taking out

> zoo<-zoo1[-1]

>

>

> table(zoo$type)

1 2 3 4 5 6 7

41 20 5 13 4 8 10

>

> #factorizing the type

> zoo$Type <- factor(zoo$type)

>

> # table or proportation of enteries in the datasets. What % of entry is Bengin and % of entry is Malignant

> round(prop.table(table(zoo$type))\*100,1)

1 2 3 4 5 6 7

40.6 19.8 5.0 12.9 4.0 7.9 9.9

>

> #Create a function to normalize the data

> # norm <- function(x){

> # return((x-min(x))/(max(x)-min(x)))

> # }

>

> train\_index <- sample(1:nrow(zoo), 0.7 \* nrow(zoo))

> test\_index <- setdiff(1:nrow(zoo), train\_index)

>

> train<- zoo[train\_index,-10]

> train\_label<-zoo[train\_index,"Type"]

>

> test<-zoo[test\_index,-10]

> test\_label<-zoo[test\_index,"Type"]

>

>

>

> train\_n<-train[1:17]

> test\_n<-test[1:17]

>

>

>

>

> # Build a KNN model on taining dataset

> library("class")

> # Building the KNN model on training dataset and also need labels which we are including c1. Once we build the preduction model

> # we have to test on test dataset

> test\_acc <- NULL

> train\_acc <- NULL

> for (i in seq(3,200,2))

+ {

+ train\_zoo\_pred <- knn(train=train\_n,test=train\_n,cl=train\_label,k=i)

+ train\_acc <- c(train\_acc,mean(train\_zoo\_pred==train\_label))

+ t\_a<-c()

+ test\_zoo\_pred <- knn(train = train\_n, test = test\_n, cl = train\_label, k=i)

+ test\_acc <- c(test\_acc,mean(test\_zoo\_pred==test\_label))

+ }

There were 50 or more warnings (use warnings() to see the first 50)

>

>

> # Testing Accuracy

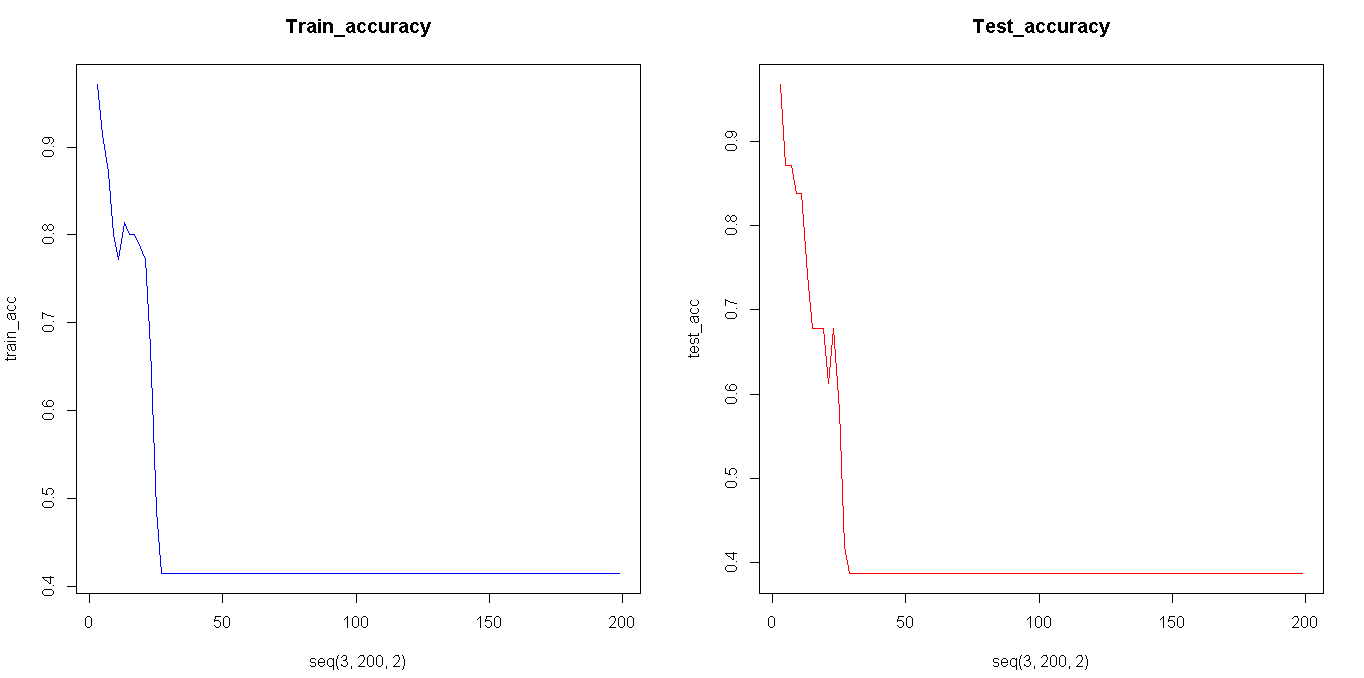
>

> # Plotting 2 different graphs on same window

> par(mfrow=c(1,2)) # c(1,2) => indicates 1 row and 2 columns

> plot(seq(3,200,2),train\_acc,type="l",main="Train\_accuracy",col="blue")

> plot(seq(3,200,2),test\_acc,type="l",main="Test\_accuracy",col="red")

> 

> acc\_neigh\_df <- data.frame(list(train\_acc=train\_acc,test\_acc=test\_acc,neigh=seq(3,200,2)))

> # Plotting 2 different graphs on same co-ordinate axis

> #install.packages("ggplot2")

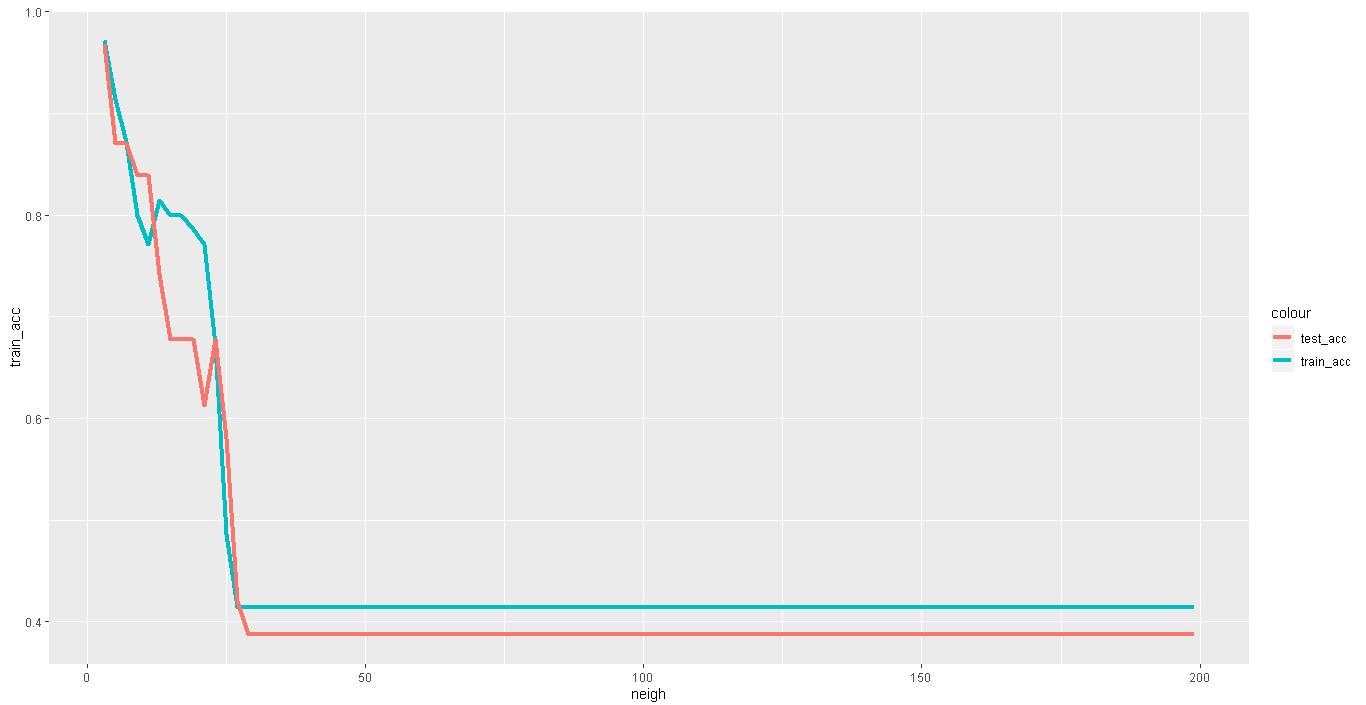
> library(ggplot2)

> ggplot(acc\_neigh\_df,aes(x=neigh))+

+ geom\_line(aes(y=train\_acc,colour="train\_acc"),lwd=1.5)+

+ geom\_line(aes(y=test\_acc,colour="test\_acc"),lwd=1.5)+

+ scale\_fill\_manual(" ",breaks=c("train\_acc","test\_acc"),values = c("train\_acc"="green","test\_acc"="red"))

> 

> #from the graph max accuracy can be acheved at k = 4

> zoo\_pred\_te <- knn(train = train\_n, test = test\_n, cl = train\_label, k=4)

> accuracy\_test<- sum(zoo\_pred\_te==test\_label)/31

> zoo\_pred\_tr <- knn(train = train\_n, test = train\_n, cl = train\_label, k=4)

> accuracy\_train<- sum(zoo\_pred\_tr==train\_label)/70

> accuracy\_test

[1] 0.9354839

> accuracy\_train

[1] 0.9428571