Assignment\_11\_MahaldarArti\_Exercise\_11.2.1

Arti Mahaldar

Nov 15, 2021

## R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

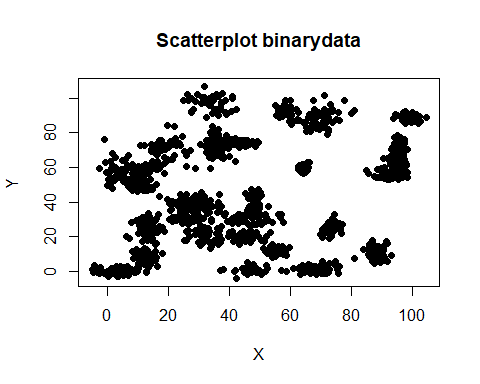
binarydata <- read.csv('C:/Users/Sandeep Raina/Documents/dsc520/data/binary-classifier-data.csv')  
  
#here is the structure of the data  
  
summary(binarydata)

## label x y   
## Min. :0.000 Min. : -5.20 Min. : -4.019   
## 1st Qu.:0.000 1st Qu.: 19.77 1st Qu.: 21.207   
## Median :0.000 Median : 41.76 Median : 44.632   
## Mean :0.488 Mean : 45.07 Mean : 45.011   
## 3rd Qu.:1.000 3rd Qu.: 66.39 3rd Qu.: 68.698   
## Max. :1.000 Max. :104.58 Max. :106.896

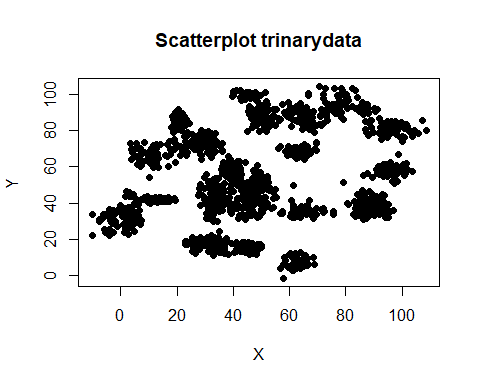
#trinary-classifier-data.csv  
  
trinarydata <- read.csv('C:/Users/Sandeep Raina/Documents/dsc520/data/trinary-classifier-data.csv')  
  
summary(trinarydata)

## label x y   
## Min. :0.000 Min. :-10.26 Min. : -1.541   
## 1st Qu.:0.000 1st Qu.: 31.15 1st Qu.: 35.906   
## Median :1.000 Median : 45.59 Median : 55.073   
## Mean :1.037 Mean : 48.86 Mean : 55.282   
## 3rd Qu.:2.000 3rd Qu.: 66.27 3rd Qu.: 77.403   
## Max. :2.000 Max. :108.56 Max. :104.293

#Plot binarydata  
x <- binarydata$x  
y <- binarydata$y  
  
plot(x, y, main="Scatterplot binarydata",xlab="X", ylab="Y", pch=19)



#PLot Trinarydata  
a <- trinarydata$x  
b <- trinarydata$y  
  
plot(a, b, main="Scatterplot trinarydata",xlab="X", ylab="Y", pch=19)



##Generate a random number that is 90% of the total number of rows in dataset.  
ran\_binarydata <- sample(1:nrow(binarydata), 0.9 \* nrow(binarydata))   
  
##the normalization function is created  
nor\_binarydata <-function(x) { (x -min(x))/(max(x)-min(x)) }  
  
##Run nomalization on first 4 coulumns of dataset because they are the predictors  
binarydata\_norm <- as.data.frame(lapply(binarydata[,c(2,3)], nor\_binarydata))  
  
summary(binarydata\_norm)

## x y   
## Min. :0.0000 Min. :0.0000   
## 1st Qu.:0.2275 1st Qu.:0.2274   
## Median :0.4278 Median :0.4386   
## Mean :0.4580 Mean :0.4421   
## 3rd Qu.:0.6522 3rd Qu.:0.6556   
## Max. :1.0000 Max. :1.0000

##extract training set  
binary\_train <- binarydata\_norm[ran\_binarydata,]   
##extract testing set  
binary\_test <- binarydata\_norm[-ran\_binarydata,]   
##extract 5th column of train dataset because it will be used as 'cl' argument in knn function.  
binary\_target\_category <- binarydata[ran\_binarydata,2]  
##extract 5th column if test dataset to measure the accuracy  
binary\_test\_category <- binarydata[-ran\_binarydata,2]  
  
##load the package class  
library(class)  
##run knn function  
pr <- knn(binary\_train,binary\_test,cl=binary\_target\_category,k=5)  
##create confusion matrix  
tab <- table(pr,binary\_test\_category)  
##this function divides the correct predictions by total number of predictions that tell us how accurate teh model is.  
  
accuracy <- function(pr){sum(diag(pr)/(sum(rowSums(pr)))) \* 100}  
accuracy(tab)

## [1] 0

#TrinaryData set  
  
  
##Generate a random number that is 90% of the total number of rows in dataset.  
ran\_trinarydata <- sample(1:nrow(trinarydata), 0.9 \* nrow(trinarydata))   
  
##the normalization function is created  
nor\_trinarydata <-function(x) { (x -min(x))/(max(x)-min(x)) }  
  
##Run nomalization on first 4 coulumns of dataset because they are the predictors  
trinarydata\_norm <- as.data.frame(lapply(trinarydata[,c(2,3)], nor\_trinarydata))  
  
summary(trinarydata\_norm)

## x y   
## Min. :0.0000 Min. :0.0000   
## 1st Qu.:0.3485 1st Qu.:0.3538   
## Median :0.4701 Median :0.5349   
## Mean :0.4976 Mean :0.5369   
## 3rd Qu.:0.6441 3rd Qu.:0.7459   
## Max. :1.0000 Max. :1.0000

##extract training set  
trinary\_train <- trinarydata\_norm[ran\_trinarydata,]   
##extract testing set  
trinary\_test <- trinarydata\_norm[-ran\_trinarydata,]   
##extract 2nd column of train dataset because it will be used as 'cl' argument in knn function.  
trinary\_target\_category <- trinarydata[ran\_trinarydata,2]  
##extract 5th column if test dataset to measure the accuracy  
trinary\_test\_category <- trinarydata[-ran\_trinarydata,2]  
  
##load the package class  
library(class)  
##run knn function  
pr <- knn(trinary\_train,trinary\_test,cl=trinary\_target\_category,k=3)  
##create confusion matrix  
tab <- table(pr,trinary\_test\_category)  
##this function divides the correct predictions by total number of predictions that tell us how accurate teh model is.  
  
accuracy <- function(x){sum(diag(x)/(sum(rowSums(x)))) \* 100}  
accuracy(tab)

## [1] 0

Note that the echo = FALSE parameter was added to the code chunk to prevent printing of the R code that generated the plot.

## References

install.packages(“knitr”)