Week 11

Raj, Ponnam

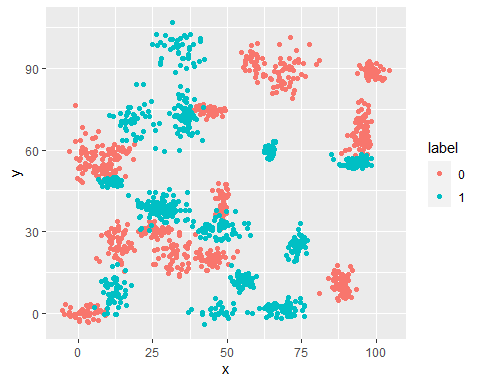
March 5th, 2022

# a. Plot the data from each dataset using a scatter plot.

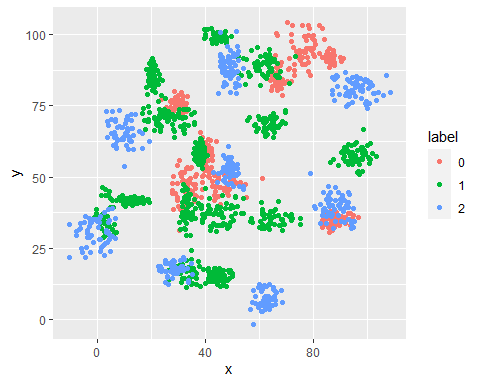
library('ggplot2')  
library('class')

## Warning: package 'class' was built under R version 4.0.3

# working directory path to access binary-classifier-data.CSV file from local drive  
setwd("/Users/rajponnam/Documents/ds\_course/dsc520/data")  
  
# Read the files `binary-classifier-data.csv` and trinary-classifier-data.csv to `binary\_df` and 'trinary\_df'  
binary\_df <- read.csv("binary-classifier-data.csv")  
trinary\_df <- read.csv("trinary-classifier-data.csv")  
  
temp\_binary\_df <- binary\_df  
temp\_binary\_df$label <- as.factor(temp\_binary\_df$label)  
temp\_trinary\_df <- trinary\_df  
temp\_trinary\_df$label <- as.factor(temp\_trinary\_df$label)  
  
# Plot the data dataset using a scatter plot for both files  
ggplot(temp\_binary\_df, aes(x=x, y=y, color=label)) + geom\_point()



ggplot(temp\_trinary\_df, aes(x=x, y=y, color=label)) + geom\_point()



# b.In this problem, you will determine which points are nearest by calculating the Euclidean distance between two points.

library ('TSdist')

## Warning: package 'TSdist' was built under R version 4.0.3

## Loading required package: proxy

## Warning: package 'proxy' was built under R version 4.0.3

##   
## Attaching package: 'proxy'

## The following objects are masked from 'package:stats':  
##   
## as.dist, dist

## The following object is masked from 'package:base':  
##   
## as.matrix

## Registered S3 method overwritten by 'quantmod':  
## method from  
## as.zoo.data.frame zoo

## Loaded TSdist v3.7. See ?TSdist for help, citation("TSdist") for use in publication.

EuclideanDistance(binary\_df$x,binary\_df$y)

## [1] 1411.959

EuclideanDistance(trinary\_df$x,trinary\_df$y)

## [1] 1357.734

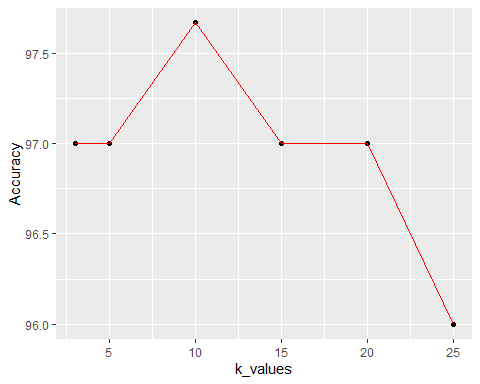
# Fit a k nearest neighbors model for each dataset for k=3, k=5, k=10, k=15, k=20, and k=25. Compute the accuracy of the resulting models for each value of k. Plot the results in a graph where the x-axis is the different values of k and the y-axis is the accuracy of the model.

knn for binary

# Split data to use 80% of data to train the model and 20% of data to test the model  
data\_split\_binary <- sample(1:nrow(binary\_df), 0.8 \* nrow(binary\_df))  
train\_binary <- binary\_df[data\_split\_binary,]  
test\_binary <- binary\_df[-data\_split\_binary,]  
  
binary\_glm <- glm(label ~ x + y, data=binary\_df, family = binomial)  
  
# extract 1st column of train dataset because it will be used as 'cl' argument in knn function.  
target\_category <- binary\_df[data\_split\_binary,1]  
  
# extract 1st column if test dataset to measure the accuracy  
test\_category <- binary\_df[-data\_split\_binary,1]  
  
k\_values<- c(3, 5, 10, 15, 20, 25)  
Accuracy <- NULL  
for (i in 1:length(k\_values))  
{  
 test\_pred <- knn(train\_binary,test\_binary,cl=target\_category,k=k\_values[i])  
 confmatrix <- table(test\_category,test\_pred)  
 accuracy <- (confmatrix[[1,1]] + confmatrix[[2,2]]) / sum(confmatrix)  
 Accuracy <- c(Accuracy, round((accuracy \* 100), digits=2))  
}  
binary\_results\_df <- data.frame(k\_values, Accuracy)  
binary\_results\_df

## k\_values Accuracy  
## 1 3 97.00  
## 2 5 97.00  
## 3 10 97.67  
## 4 15 97.00  
## 5 20 97.00  
## 6 25 96.00

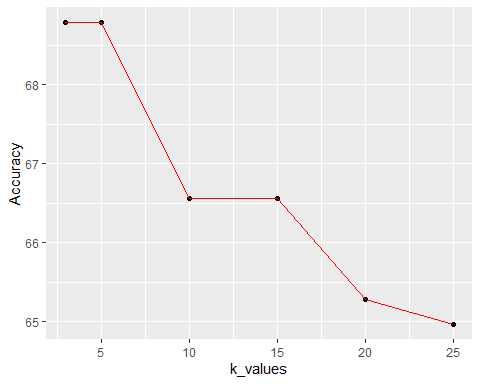
ggplot(binary\_results\_df, aes(x=k\_values, y=Accuracy)) + geom\_point() + geom\_line(colour="red")

 knn for trinary

# Split data to use 80% of data to train the model and 20% of data to test the model  
data\_split\_trinary <- sample(1:nrow(trinary\_df), 0.8 \* nrow(trinary\_df))  
train\_trinary <- trinary\_df[data\_split\_trinary,]  
test\_trinary <- trinary\_df[-data\_split\_trinary,]  
  
trinary\_glm <- glm(label ~ x + y, data=trinary\_df, family = poisson)  
  
# extract 1st column of train dataset because it will be used as 'cl' argument in knn function.  
target\_category <- trinary\_df[data\_split\_trinary,1]  
  
# extract 1st column if test dataset to measure the accuracy  
test\_category <- trinary\_df[-data\_split\_trinary,1]  
  
k\_values<- c(3, 5, 10, 15, 20, 25)  
Accuracy <- NULL  
for (i in 1:length(k\_values))  
{  
 test\_pred <- knn(train\_trinary,test\_trinary,cl=target\_category,k=k\_values[i])  
 confmatrix <- table(test\_category,test\_pred)  
 accuracy <- (confmatrix[[1,1]] + confmatrix[[2,2]]) / sum(confmatrix)  
 Accuracy <- c(Accuracy, round((accuracy \* 100), digits=2))  
}  
trinary\_results\_df <- data.frame(k\_values, Accuracy)  
trinary\_results\_df

## k\_values Accuracy  
## 1 3 68.79  
## 2 5 68.79  
## 3 10 66.56  
## 4 15 66.56  
## 5 20 65.29  
## 6 25 64.97

ggplot(trinary\_results\_df, aes(x=k\_values, y=Accuracy)) + geom\_point() + geom\_line(colour="red")



# C. Looking back at the plots of the data, do you think a linear classifier would work well on these datasets?

Linear classifier will not work for both binary and trinary datasets because data points are not grouped around a line. Data points are spread all over the graph.