```
---
title: "Exercise 11-1-2"
author: "Saurabh Shrestha"
date: "6/5/2021"
output: pdf_document
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
## Load the ggplot2 package
library(ggplot2)
library(plyr)
library(dplyr)
library(class)
install.packages("factoextra")
library(factoextra)
library(tidyverse)
library(cluster)

```

Set the working directory to the root of your DSC 520 directory
#setwd("/home/jdoe/Workspaces/dsc520")
setwd("C:/Users/Saurabh/Desktop/DSC 520")

Load the datasets
binary_df <- read.csv("C:/Users/Saurabh/Desktop/binary-classifier-
data.csv")
trinary_df <- read.csv("C:/Users/Saurabh/Desktop/trinary-classifier-
data.csv")

For binary_df dataset- knn
##Generate a random number that is 90% of the total number of rows in
dataset.
ran <- sample(2:nrow(binary_df), 0.9 * nrow(binary_df))

##the normalization function is created
nor <-function(x) { (x -min(x))/(max(x)-min(x))   }

##Run normalization on 2 columns of dataset because they are the
predictors
binary_df_norm <- as.data.frame(lapply(binary_df[,c(2,3)], nor))

summary(binary_df_norm)

##extract training set
binary_df_train <- binary_df_norm[ran,]
##extract testing set
binary_df_test <- binary_df_norm[-ran,]
##extract 1st column of train dataset because it will be used as 'cl'
argument in knn function.
binary_df_target_category <- binary_df[ran,1]
##extract 1st column if test dataset to measure the accuracy
binary_df_test_category <- binary_df[-ran,1]
```

```
##load the package class
library(class)
##run knn = 3
pr <-
knn(binary_df_train,binary_df_test,cl=binary_df_target_category,k=3)

##create confusion matrix
tab <- table(pr,binary_df_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)
# 97% accuracy

# k = 5
##run knn function
pr <-
knn(binary_df_train,binary_df_test,cl=binary_df_target_category,k=5)

##create confusion matrix
tab <- table(pr,binary_df_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)

# k = 10
##run knn function
pr <-
knn(binary_df_train,binary_df_test,cl=binary_df_target_category,k=10)

##create confusion matrix
tab <- table(pr,binary_df_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)

# k = 15
##run knn function
pr <-
knn(binary_df_train,binary_df_test,cl=binary_df_target_category,k=15)

##create confusion matrix
tab <- table(pr,binary_df_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)

# k = 20
##run knn function
pr <-
knn(binary_df_train,binary_df_test,cl=binary_df_target_category,k=20)

##create confusion matrix
tab <- table(pr,binary_df_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)
# k = 25
##run knn function
pr <-
knn(binary_df_train,binary_df_test,cl=binary_df_target_category,k=25)

##create confusion matrix
tab <- table(pr,binary_df_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)

# For trinary_df dataset knn
##Generate a random number that is 90% of the total number of rows in
dataset.
ran <- sample(2:nrow(trinary_df), 0.9 * nrow(trinary_df))

##the normalization function is created
nor <-function(x) { (x -min(x))/(max(x)-min(x))   }

##Run nomalization on 2 coulumns of dataset because they are the
predictors
trinary_df_norm <- as.data.frame(lapply(trinary_df[,c(2,3)], nor))

summary(trinary_df_norm)

# extract training set
trinary_df_train <- trinary_df_norm[ran,]

##extract testing set
trinary_df_test <- trinary_df_norm[-ran,]

##extract 1st column of train dataset because it will be used as argument
in knn function.
trinary_df_target_category <- trinary_df[ran,1]
```

```
##extract 1st column if test dataset to measure the accuracy
trinary_df_test_category <- trinary_df[-ran,1]

##load the package class
library(class)

##run knn function for k == 3
performr <-
knn(trinary_df_train,trinary_df_test,cl=trinary_df_target_category,k=3)

##create confusion matrix
tab <- table(performr,trinary_df_test_category)

##this function divides the correct predictions by total number of
predictions that tell us how accurate the model is.

accuracy <- function(x){sum(diag(x)/(sum(rowSums(x)))) * 100}
accuracy(tab)
# 85% approx accuracy when k = 3

#  k=5

performr <-
knn(trinary_df_train,trinary_df_test,cl=trinary_df_target_category,k=5)

##create confusion matrix
tab <- table(performr,trinary_df_test_category)

##this function divides the correct predictions by total number of
predictions that tell us how accurate the model is.

accuracy <- function(x){sum(diag(x)/(sum(rowSums(x)))) * 100}
accuracy(tab)
# 88% accuracy

# k=10

##run knn function for k ==13
performr <-
knn(trinary_df_train,trinary_df_test,cl=trinary_df_target_category,k=10)

##create confusion matrix
tab <- table(performr,trinary_df_test_category)

##this function divides the correct predictions by total number of
predictions that tell us how accurate the model is.

accuracy <- function(x){sum(diag(x)/(sum(rowSums(x)))) * 100}
accuracy(tab)
# accuracy 87% approx

# k=15
```

```
performr <-
knn(trinary_df_train,trinary_df_test,cl=trinary_df_target_category,k=15)

##create confusion matrix
tab <- table(performr,trinary_df_test_category)

##this function divides the correct predictions by total number of
predictions that tell us how accurate the model is.

accuracy <- function(x){sum(diag(x)/(sum(rowSums(x)))) * 100}
accuracy(tab)
# k=20

##run knn function for k == 20
performr <-
knn(trinary_df_train,trinary_df_test,cl=trinary_df_target_category,k=20)

##create confusion matrix
tab <- table(performr,trinary_df_test_category)

##this function divides the correct predictions by total number of
predictions that tell us how accurate the model is.

accuracy <- function(x){sum(diag(x)/(sum(rowSums(x)))) * 100}
accuracy(tab)

# k=25

performr <-
knn(trinary_df_train,trinary_df_test,cl=trinary_df_target_category,k=15)

##create confusion matrix
tab <- table(performr,trinary_df_test_category)

##this function divides the correct predictions by total number of
predictions that tell us how accurate the model is.

accuracy <- function(x){sum(diag(x)/(sum(rowSums(x)))) * 100}
accuracy(tab)


#Scatterplot for binary_df and trinary_df datasets
plot(binary_df$x, binary_df$y, main="Binary Scatterplot",
     xlab="X variable ", ylab="Y variable ")
plot(trinary_df$x, trinary_df$y, main="Trinary Scatterplot",
     xlab="X variable ", ylab="Y variable ")

# euclidean distance for binary_df
set.seed(123)
subset <- sample(1:50, 15)   # Take 15 random rows
df <- binary_df[subset, ]    # Subset the 15 rows
df.scaled <- scale(df)
dist.eucl <- dist(df.scaled, method = "euclidean")
dist.eucl
```

```
# Reformat as a matrix
# Subset the first 3 columns and rows and Round the values
round(as.matrix(dist.eucl)[2:3, 1:3], 1)

# euclidean distance for trinary_df
set.seed(123)
subset() <- sample(1:50, 15)   # Take 15 random rows
df <- trinary_df[subset(), ]    # Subset the 15 rows
df.scaled <- scale(df)
dist.eucl <- dist(df.scaled, method = "euclidean")
dist.eucl

# Reformat as a matrix
# Subset the first 3 columns and rows and Round the values
round(as.matrix(dist.eucl)[2:3, 1:3], 1)




# 2.Clustering

clustering_df <- read.csv("C:/Users/Saurabh/Desktop/clustering-data.csv")

plot(clustering_df$x, clustering_df$y, main="Clustering Scatterplot",
     xlab="X variable ", ylab="Y variable ")


# k-means algorithm from k=2 to k=12.

k2 <- kmeans(clustering_df, centers = 2, nstart = 25)
k3 <- kmeans(clustering_df, centers = 3, nstart = 25)
k4 <- kmeans(clustering_df, centers = 4, nstart = 25)
k5 <- kmeans(clustering_df, centers = 5, nstart = 25)
k6 <- kmeans(clustering_df, centers = 6, nstart = 25)
k7 <- kmeans(clustering_df, centers = 7, nstart = 25)
k8 <- kmeans(clustering_df, centers = 8, nstart = 25)
k9 <- kmeans(clustering_df, centers = 9, nstart = 25)
k10 <- kmeans(clustering_df, centers = 10, nstart = 25)
k11 <- kmeans(clustering_df, centers = 11, nstart = 25)
k12 <- kmeans(clustering_df, centers = 12, nstart = 25)

# plots to compare
p1 <- fviz_cluster(k2, geom = "point", data = clustering_df) + ggtitle("k
= 2")
p2 <- fviz_cluster(k3, geom = "point",  data = clustering_df) +
ggtitle("k = 3")
p3 <- fviz_cluster(k4, geom = "point",  data = clustering_df) +
ggtitle("k = 4")
p4 <- fviz_cluster(k5, geom = "point",  data = clustering_df) +
ggtitle("k = 5")
p5 <- fviz_cluster(k2, geom = "point", data = clustering_df) + ggtitle("k
= 6")
```

```
p6 <- fviz_cluster(k3, geom = "point",  data = clustering_df) +
ggtitle("k = 7")
p7 <- fviz_cluster(k4, geom = "point",  data = clustering_df) +
ggtitle("k = 8")
p8 <- fviz_cluster(k5, geom = "point",  data = clustering_df) +
ggtitle("k = 9")
p9 <- fviz_cluster(k2, geom = "point", data = clustering_df) + ggtitle("k
= 10")
p10 <- fviz_cluster(k3, geom = "point",  data = clustering_df) +
ggtitle("k = 11")
p11 <- fviz_cluster(k4, geom = "point",  data = clustering_df) +
ggtitle("k = 12")
library(gridExtra)
# scatter plot of the resultant clusters for each value of k
grid.arrange(p1, p2, p3, p4,p5,p6,p7,p8,p9,p10,p11, nrow = 2)

# e.Calculate this average distance from the center of each cluster
fviz_nbclust(clustering_df, kmeans, method = "silhouette")


# f.One way of determining the â€œrightâ€ number of clusters is to look
# at the graph of k versus average distance and finding the
set.seed(123)
fviz_nbclust(clustering_df, kmeans, method = "wss")
fviz_nbclust



##
```