**Assignment 3**

Dataset contains information about patients who have diabetes based on glucose level, bmi, bp and age. First, we load the dataset and get the summary of data.

A screenshot of a computer

Description automatically generated with low confidence

Create frequency table for each of the rows.Text, application

Description automatically generated

Next, we find out the number of NaN values. In this case there are no NaN values.A picture containing text

Description automatically generated

We use the filter function to remove any outliers based on mean (+-) standard deviation. Now the dataset is ready for modeling. We divide the data into test and train set to feed into the model and use feature scaling to normailze the data. Take 70% of data as train and the rest as text data. Then we test the model using the test data and try to find the best value of K. After using several values of K we see the best option for **K is 5**.

Text

Description automatically generated

data = read.csv("C:/Users/raiai/OneDrive - Seneca/COURSE/BDM 300/diabetes.csv")

head(data)

summary(data)

#Diabetes Outcome

# Yes = 1

# No = 0

**#Frequency table**

glucose = as.data.frame(table(data$Glucose))

bp = as.data.frame(table(data$BloodPressure))

bmi = as.data.frame(table(data$BMI))

age = as.data.frame(table(data$Age))

**#Couning null values**

table(is.na.data.frame(data))

**# Outlier filtering**

stan\_dev = sd(data$Glucose)

mean = mean(data$Glucose)

new = data %>% filter( between(Glucose, mean+stan\_dev, mean-stan\_dev))

**# Splitting data into train**

# and test data (train =70% , test= 30%)

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

train\_cl <- subset(data, split == "TRUE")

test\_cl <- subset(data, split == "FALSE")

**# Normalizing dataset**

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

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

**#KNN Model**

classifier\_knn <- knn(train = train\_scale,

test = test\_scale,

cl = train\_cl$Outcome,

k = 1)

classifier\_knn

**# Confusiin Matrix**

cm <- table(test\_cl$Outcome, classifier\_knn)

cm

# Model Evaluation - Choosing K

# Calculate out of Sample error

error <- mean(classifier\_knn != test\_cl$Outcome)

print(paste('Accuracy =', (1-error)\*100, '%'))

**# K = 5**

classifier\_knn <- knn(train = train\_scale,

test = test\_scale,

cl = train\_cl$Outcome,

k = 5)

error <- mean(classifier\_knn != test\_cl$Outcome)

print(paste('Accuracy =', (1-error)\*100, '%'))

**Using K-Means Algorithm**

Chart, line chart

Description automatically generated

Using the Elbow method to find the optimal number of clusters. In this case the number is 5. We then plot the cluster.

set.seed(6)

#Compute the sum of square error for different clusters

x = data[2:3]

values<-vector()

for (i in 1:10) values[i]<-sum(kmeans(x,i)$withinss)

#to plot the number of clusters and

plot(1:10,values,type = "b",main = paste('clusters of clients'),xlab = "Number of clusters",ylab = "Values")

#Optimal Clusters=5

set.seed(10)

kmeans<-kmeans(x,5,iter.max=300,nstart=10)

#visualizing the clusters

install.packages("cluster")

library(cluster)

clusplot(x, kmeans$cluster,

lines = 0,

shade = TRUE,

color = TRUE,

labels = 2,

plotchar = FALSE,

span = TRUE,

main = paste('Clusters of patients'),

xlab = 'Glucose',

ylab = 'BloodPressure')

Chart, diagram

Description automatically generated

Results Report:

1. The patients at cluster 1 have a moderate blood pressure and higher glucose
2. Cluster 2 has average glucose and BP level
3. Cluster 3 have very glucose but low BP
4. The patients at cluster 4 has high blood pressure but a wide range of glucose
5. Cluster 5 has low BP and glucose levels