

**American International University-Bangladesh (AIUB)**

**Course: INTRODUCTION TO DATA SCIENCE [D]**

**Faculty Name: TOHEDUL ISLAM**

**Student Name: Shanto Kumar Basak**

**ID: 20-42945-1**

**Submitted Date: 30th April 2023**

**Using KNN from Diabetics Dataset**

**INTRODUCTION:**

On the basis of supervised datasets, KNN (K-Nearest Neighbor) is one of the most well-liked machine learning algorithms. By choosing the Kth neighbor, this procedure is then used. Calculate the Euclidean distance after picking the neighbors. The number of data points in each category among these k neighbors were counted. The category for which the number of neighbors is highest should receive the new data points. Next, the model is prepared. With the help of the KNN algorithm and the R programming language, the project's goal was to create a model for predicting diabetic patients from data on Kaggle.

**METHODOLOGY:**

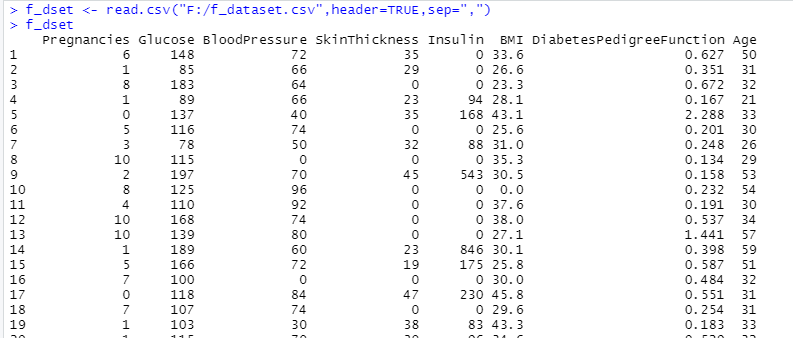
The main data was found from Kaggle (link- <https://www.kaggle.com/datasets/mathchi/diabetes-data-set>). Several constraints were placed on the selection of these instances from a larger database. In particular, all patients here are females at least 21 years old of Pima Indian heritage.

* Pregnancies: Number of times pregnant
* Glucose: Plasma glucose concentration 2 hours in an oral glucose tolerance test
* Blood Pressure: Diastolic blood pressure (mm Hg)
* Skin Thickness: Triceps skin fold thickness (mm)
* Insulin: 2-Hour serum insulin (mu U/ml)
* BMI: Body mass index (weight in kg/ (height in m) ^2)
* Diabetes Pedigree Function: Diabetes pedigree function
* Age: Age (years)
* Outcome: Class variable (0 or 1)

1. **Print the Data Set**

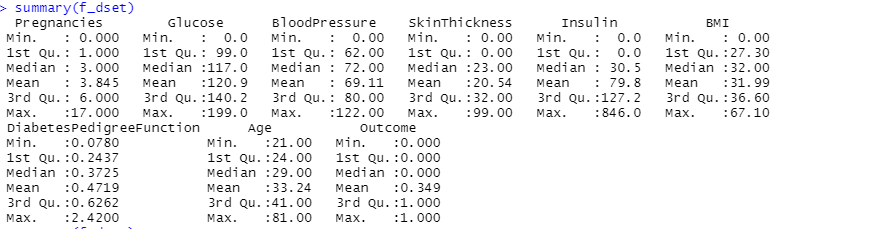
f\_dset <- read.csv("F:/f\_dataset.csv",header=TRUE,sep=",")

f\_dset

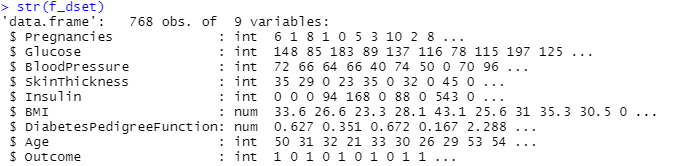


**2) An overview of each attribute in the data set**

summary(f\_dset)

****

str(f\_dset)

****

**3) Change to mean value in which case the attributes value is 0**

f\_dset$Pregnancies[f\_dset$Pregnancies ==0] = mean(f\_dset$Pregnancies,)

f\_dset$Glucose [f\_dset$Glucose ==0] = mean(f\_dset$Glucose,)

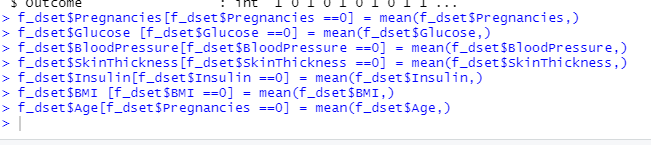
f\_dset$BloodPressure[f\_dset$BloodPressure ==0] = mean(f\_dset$BloodPressure,)

f\_dset$SkinThickness[f\_dset$SkinThickness ==0] = mean(f\_dset$SkinThickness,)

f\_dset$Insulin[f\_dset$Insulin ==0] = mean(f\_dset$Insulin,)

f\_dset$BMI [f\_dset$BMI ==0] = mean(f\_dset$BMI,)

f\_dset$Age[f\_dset$Pregnancies ==0] = mean(f\_dset$Age,)



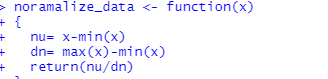
**Load the library of class**



**4) Data Normalization**

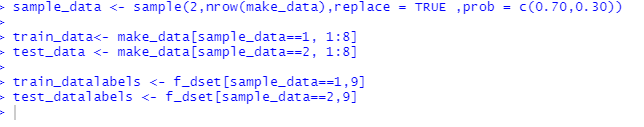
Normalization is a very important part of KNN. It is hard to make the current accuracy of is the data is not in well-shaped.It shapes the data in 0 to 1. The main math of the normalization is

= (value-min(value))/(max(value)-min(value))



**5) Data Splitting**

Data splitting basically involves splitting the data set into training and testing data set.There ware taken 70% data in train dataset and 30% data on the test dataset for making the confusion matrix so randomly.

****

**6) Confusion Matrix**

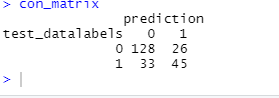
A confusion matrix is a table that is often used to describe the performance of a classification model or "classifier" on a set of test data for which the true values are known.

prediction= knn(train = train\_data,test = test\_data,train\_datalabels,k=5)

con\_matrix= table(test\_datalabels,prediction)

con\_matrix

**True Negative =128, True Positive=45, False positive =26, False Negative= 33**



**7) Here is the Some Accuracy**

Accuracy= function(con\_matrix)

{

sum=0

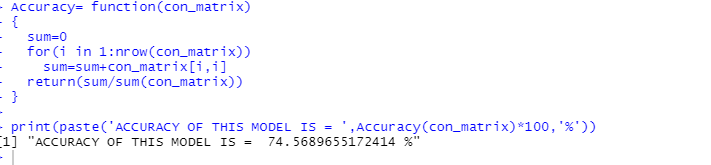
for(i in 1:nrow(con\_matrix))

sum=sum+con\_matrix[i,i]

return(sum/sum(con\_matrix))

}

print(paste('ACCURACY OF THIS MODEL IS = ',Accuracy(con\_matrix)\*100,'%'))

****

**8) Calculate accuracy with various K values between 1 and 100**

list\_k <- c(1:100)

arr\_k\_result <-c()

for(i in 1: length(list\_k))

{

prediction= knn(train = train\_data,test = test\_data,train\_datalabels,k=i)

con\_matrix= table(test\_datalabels,prediction)

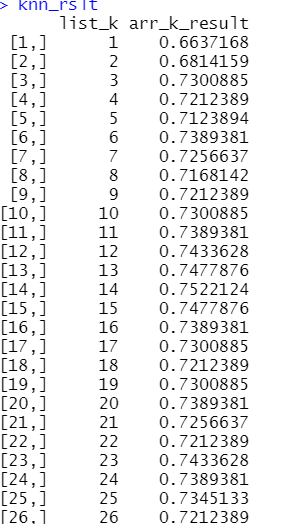
arr\_k\_result[i]<-Accuracy(con\_matrix)

}

knn\_rslt <- cbind(list\_k ,arr\_k\_result)

colnames(knn\_rslt)<-c("Value of k"," Accuracy")

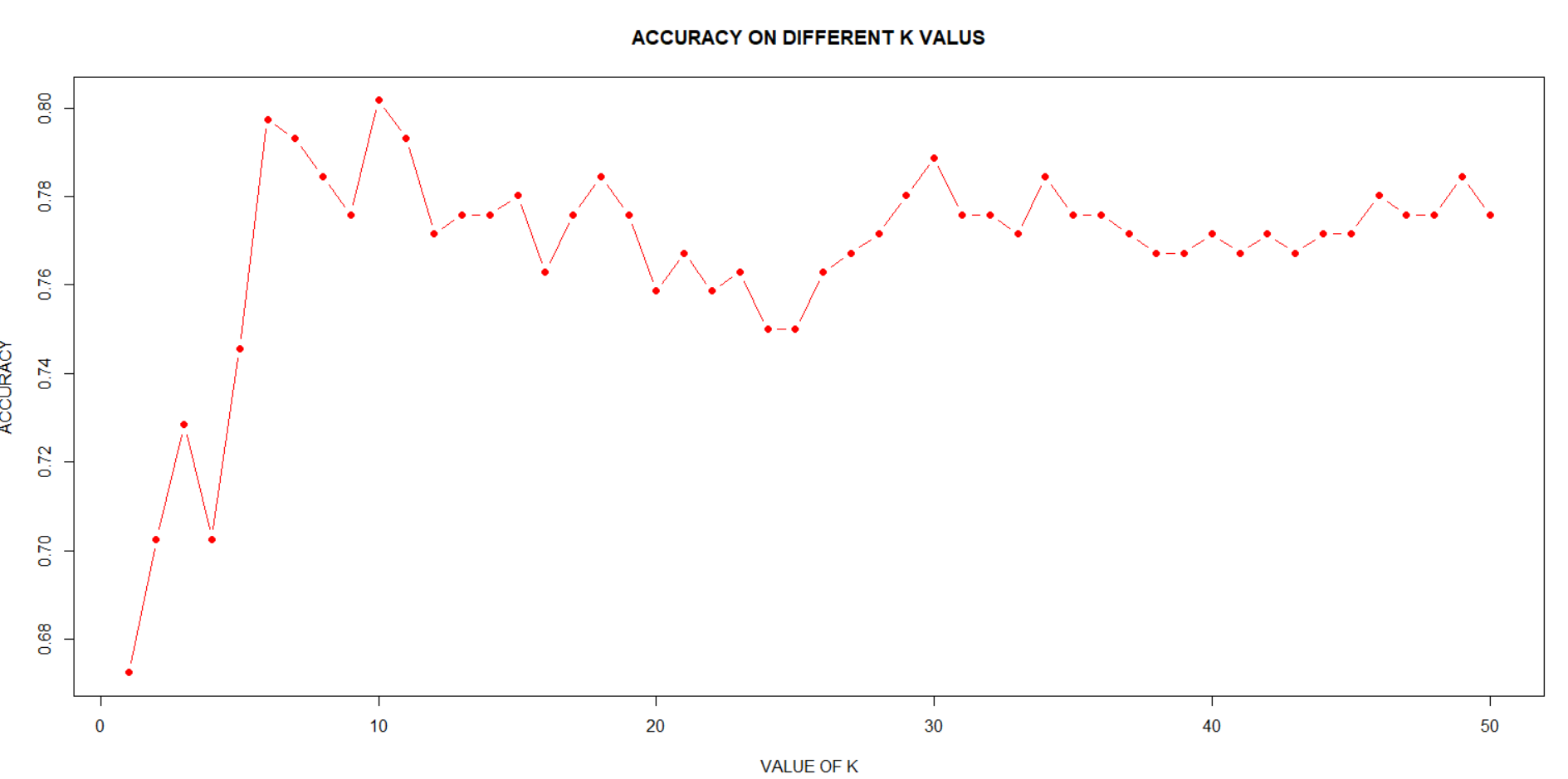
knn\_rslt <- as.data.frame(knn\_rslt)

****

**9) All-accuracy scatterplot with the K value**

knn\_rslt

plot(knn\_rslt$`Value of k`,knn\_rslt$` Accuracy`,type="b",pch=16, col="red", lwd=1, xlab="VALUE OF K", ylab="ACCURACY", main="ACCURACY ON DIFFERENT K VALUS")



**DISCUSSION:**

A dataset contained a total of 730 data. And there predicted the data by 70% train and 30% test dataset selected randomly. The forecast would be more accurate and useful in this situation if there were more data and qualities provided.