#importing packages that are required to execute

library(ISLR)

library(caret)

library(dplyr)

library(FNN)

library(gmodels)

#Importing data set into rstudio

Bankdata <- read.csv("UniversalBank.csv")

```{r}

#to find the structure of the data set.

str(Bankdata)

#understanding summary stats for the data set.

summary(Bankdata)

```

```{r}

#Creating a new data set NewBank by removing ID and ZIP.Code variables

NewBank <- Bankdata[ ,-c(1,5)]

#Reviewing the structure of the new dataset.

str(NewBank)

```

```{r}

#Converting variables to categorical values variables

NewBank$Education <- as.factor(NewBank$Education)

NewBank$Family <- as.factor(NewBank$Family)

#Creating the dummy variables for categorical values with more than 2 they are Education and Family.

dummy\_model1 <- dummyVars(~Family + Education, data = NewBank)

#Adding the dummy variables to "NewBank" and removing the initial "Education" and "Family" variables.

dv <- as.data.frame(predict(dummy\_model1, NewBank))

NewBank <- as.data.frame(c(NewBank, dv))

NewBank <- NewBank[, -c(4,6)]

#splitting the data set into 60% training and 40% test data using the "createDataPartition" function.

```

```{r}

#Setting the seed for randomized functions

set.seed(100319)

#Splitting the data into 60% training data and 40% test data

NewBank\_Index <- createDataPartition(NewBank$Age, p=0.4, list = F)

NewBank\_Test <- NewBank[NewBank\_Index,]

NewBank\_Train <- NewBank[-NewBank\_Index,]

#normalizing the training and testing data sets using the "preProcess" function.

```

```{r}

#Creating copy of the data set for normalization

NewBank\_Train\_Norm <- NewBank\_Train

NewBank\_Test\_Norm <- NewBank\_Test

#Using preProcess function for creating a model for centering and scaling the data

Norm\_Values <- preProcess(NewBank\_Train[, c(1:5)], method = c("center", "scale"))

#Replacing the numerical variables with normalized and centered data

NewBank\_Train\_Norm[, c(1:5)] <- predict(Norm\_Values, NewBank\_Train[, c(1:5)])

NewBank\_Test\_Norm[, c(1:5)] <- predict(Norm\_Values, NewBank\_Test[, c(1:5)])

```

```{r}

#Creating the KNN model with K = 1 and training and test data

knn2 <- knn(train = NewBank\_Train\_Norm[, -6], test = NewBank\_Test\_Norm[, -6], cl = NewBank\_Train\_Norm[, 6], k = 1, prob = TRUE)

```

```{r}

head(NewBank)

```

```{r}

#Creating the customer profile for the customer in question #1

customer <- data.frame("Age" = 40,

"Experience" = 10,

"Income" = 84,

"CCAvg" = 2,

"Mortgage" = 0,

"Securities.Account" = 0,

"CD.Account" = 0,

"Online" = 1,

"CreditCard" = 1,

"Family.1" = 0,

"Family.2" = 1,

"Family.3" = 0,

"Family.4" = 0,

"Education.1" = 0,

"Education.2" = 1,

"Education.3" = 0)

#Performing the preProcessing steps on the customer variable

customer[, c(1:5)] <- predict(Norm\_Values, customer[, c(1:5)])

#Running the KNN model on the customer variable

knn\_customer <- knn(train = NewBank\_Train\_Norm[, -6], test = customer, cl = NewBank\_Train\_Norm[, 6], k = 1, prob = TRUE)

#Returning the value predicted by model

as.data.frame(knn\_customer)

```

```{r}

#Creating a data frame with two columns k and accuracy

accuracy.df <- data.frame(k = seq(1, 30, 1), accuracy = rep(0, 30))

#Performing predictions for the values at various K values in KNN

for(i in 1:30) {

knn.pred <- knn(train = NewBank\_Train\_Norm[, -6], test = NewBank\_Test\_Norm[, -6],

cl = NewBank\_Train\_Norm[, 6], k = i)

accuracy.df[i, 2] <- confusionMatrix(as.factor(knn.pred), as.factor(NewBank\_Test\_Norm[, 6]))$overall[1]

}

#Displaying the results in a data frame

accuracy.df

#Rough plot of accuracy to view the trend in data

plot(x = accuracy.df$k,y = accuracy.df$accuracy, main = "Plot of Accuracy Values vs K", xlab = "K Value", ylab = "Accuracy")

```

```{r}

#Creating the KNN model with K = 1 and training and testing data

knn2 <- knn(train = NewBank\_Train\_Norm[, -6], test = NewBank\_Test\_Norm[, -6], cl = NewBank\_Train\_Norm[, 6], k = 1, prob = TRUE)

#Confusion Matrix

predicted <- as.factor(knn2)

actual <- as.factor(NewBank\_Test\_Norm[, 6])

confusionMatrix(predicted, actual, positive = "1")

```

```{r}

#Running the KNN model on customer variable with the K = 1

knn\_cust2 <- knn(train = NewBank\_Train\_Norm[, -6], test = customer, cl = NewBank\_Train\_Norm[, 6], k = 20, prob = TRUE)

#Returning the value predicted by our model

as.data.frame(knn\_cust2)

```

```{r}

#Setting the seed for randomized functions

set.seed(100619)

#Splitting the data into 50% training data, 30% validation data, and 20% testing data.

Bnk2\_Index <- createDataPartition(NewBank$Age, p=0.2, list = F)

Bnk2\_Test <- NewBank[Bnk2\_Index,]

Bnk2\_Remaining <- NewBank[-Bnk2\_Index,]

Bnk2\_Index <- createDataPartition(Bnk2\_Remaining$Age, p=0.625, list = F)

Bnk2\_Train <- Bnk2\_Remaining[Bnk2\_Index,]

Bnk2\_Validation <- Bnk2\_Train[-Bnk2\_Index,]

#The new partitioned data will now have to be normalized.

```

```{r}

#Create a copy of the data sets for normalization

Bnk2\_Train\_Norm <- Bnk2\_Train

Bnk2\_Test\_Norm <- Bnk2\_Test

Bnk2\_Validation\_Norm <- Bnk2\_Validation

#Use preProcess function to create a model to scale and centering the data

Norm\_Values <- preProcess(Bnk2\_Train[, c(1:5)], method = c("center", "scale"))

#Replace the numeric variables with normalized and centered data

Bnk2\_Train\_Norm[, c(1:5)] <- predict(Norm\_Values, Bnk2\_Train[, c(1:5)])

Bnk2\_Test\_Norm[, c(1:5)] <- predict(Norm\_Values, Bnk2\_Test[, c(1:5)])

Bnk2\_Validation\_Norm[, c(1:5)] <- predict(Norm\_Values, Bnk2\_Validation[, c(1:5)])

#The first confusion matrix is for the "test" data:

```

```{r}

#Creating the KNN model with K = 1

knn\_model\_test <- knn(train = Bnk2\_Train\_Norm[, -6], test = Bnk2\_Test\_Norm[, -6],

cl = Bnk2\_Train\_Norm[, 6], k = 1, prob = TRUE)

#Confusion Matrix

predicted\_test <- as.factor(knn\_model\_test)

actual\_test <- as.factor(Bnk2\_Test\_Norm[, 6])

confusionMatrix(predicted\_test, actual\_test, positive = "1")

#second confusion matrix is for the "validation" data:

```

```{r}

#Create the KNN model with K = 1

knn\_model\_validation <- knn(train = Bnk2\_Train\_Norm[, -6], test = Bnk2\_Validation\_Norm[, -6],

cl = Bnk2\_Train\_Norm[, 6], k = 1, prob = TRUE)

#Confusion Matrix

predicted\_validation <- as.factor(knn\_model\_validation)

actual\_validation <- as.factor(Bnk2\_Validation\_Norm[, 6])

confusionMatrix(predicted\_validation, actual\_validation, positive = "1")

#The third confusion matrix is for the "train" data:

{r}

#Creating the KNN model with K = 1

knn\_model\_train <- knn(train = Bnk2\_Train\_Norm[, -6], test = Bnk2\_Train\_Norm[, -6],

cl = Bnk2\_Train\_Norm[, 6], k = 1, prob = TRUE)

#Confusion Matrix

predicted\_train <- as.factor(knn\_model\_train)

actual\_train <- as.factor(Bnk2\_Train\_Norm[, 6])

confusionMatrix(predicted\_train, actual\_train, positive = "1")

OUTPUT:

