Assignment 2

Rob DiVincenzo 10-01-2023 BA 64060

This is the submission for Assignment 2.

#Read data  
DF=read.csv("./UniversalBank.csv") # Read the Online Retail csv file  
#Load libraries  
library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

library(caret)

## Loading required package: ggplot2

## Loading required package: lattice

library(ISLR)  
library(FNN)  
library(gmodels)  
set.seed(15)  
  
#Select variables in need  
DF <- select(DF, Age, Experience, Income, Family, CCAvg, Education, Mortgage, Personal.Loan, Securities.Account, CD.Account, Online, CreditCard) # Select a subset of variables  
  
#Create dummy variables for Education (categorical variable with 2+ levels)  
DF$Education\_1 <- as.integer(DF$Education == 1)  
DF$Education\_2 <- as.integer(DF$Education == 2)  
DF$Education\_3 <- as.integer(DF$Education == 3)  
#Drop no longer needed Education field  
DF<-DF[,-6]  
  
#Partition Data into 60% Train, 40% val  
Train\_Index=createDataPartition(DF$Age,p=0.60, list=FALSE)  
Train\_Data = DF[Train\_Index,] # create the training data; we include all columns; note the index is row, column  
Val\_Data = DF[-Train\_Index,] # create the val set  
  
#Create copy of data for normalization  
train.norm.df <- Train\_Data  
val.norm.df <- Val\_Data  
  
# use preProcess() from the caret package to normalize data, ignore target variable personal loan  
norm.values <- preProcess(Train\_Data[,-7], method=c("center", "scale"))  
  
#Replace columns with normalized values  
train.norm.df[,-7] <- predict(norm.values, Train\_Data[,-7])  
val.norm.df[,-7] <- predict(norm.values, Val\_Data[,-7])  
  
#Train knn model on all variables except personal loan status since that's our target variable  
#nn<-knn(train = train.norm.df[,-7], val = val.norm.df[,-7], cl=train.norm.df$Personal.Loan,k=1)  
  
#Load a custom customer  
custom\_customer\_data <- data.frame(  
 Age = c(40),  
 Experience = c(10),  
 Income = c(84),  
 Family = c(2),  
 CCAvg = c(2),  
 Mortgage = c(0),  
 Securities.Account = c(0),  
 CD.Account = c(0),  
 Online = c(1),  
 CreditCard = c(1),  
 Education\_1 = c(0),  
 Education\_2 = c(1),  
 Education\_3 = c(0)  
)  
  
#Normalize customer data  
customer.norm.df <- custom\_customer\_data  
customer.norm.df <- predict(norm.values, custom\_customer\_data)  
  
  
  
#Question 1)  
  
classification <- knn(train = train.norm.df[,-7], test = customer.norm.df, cl = train.norm.df$Personal.Loan, k = 1)  
# Add the predictions to the new data  
new\_customer\_data <- customer.norm.df  
new\_customer\_data$LoanAcceptance <- classification  
# Print the new data with predicted classification  
print(new\_customer\_data)

## Age Experience Income Family CCAvg Mortgage  
## 1 -0.4684867 -0.880771 0.2074445 -0.3426499 0.0380226 -0.56064  
## Securities.Account CD.Account Online CreditCard Education\_1 Education\_2  
## 1 -0.3381309 -0.257007 0.8430359 1.552232 -0.841302 1.590517  
## Education\_3 LoanAcceptance  
## 1 -0.6580258 0

#Question 1 answer: This customer would be classified as not accepting the loan.  
  
  
#End question 1  
  
  
  
#Question 2)  
  
#train model for k  
set.seed(1111)  
#model<-train(Personal.Loan~Age+Experience+Income+Family+CCAvg+Mortgage+Securities.Account+CD.Account+Online+CreditCard+Education\_1+Education\_2+Education\_3, data=train.norm.df, method="knn")  
#model  
accuracy.df <- data.frame(k = seq(1, 14, 1), accuracy = rep(0, 14))  
  
# compute knn for different k on validation.  
for(i in 1:14) {  
knn.pred <- knn(train = train.norm.df[,-7], test = val.norm.df[,-7],   
 cl = train.norm.df$Personal.Loan, k = i)   
  
accuracy.df[i, 2] <- confusionMatrix(knn.pred, as.factor(val.norm.df$Personal.Loan))$overall[1]  
  
}  
accuracy.df

## k accuracy  
## 1 1 0.9624812  
## 2 2 0.9549775  
## 3 3 0.9634817  
## 4 4 0.9589795  
## 5 5 0.9644822  
## 6 6 0.9574787  
## 7 7 0.9604802  
## 8 8 0.9559780  
## 9 9 0.9584792  
## 10 10 0.9544772  
## 11 11 0.9554777  
## 12 12 0.9534767  
## 13 13 0.9559780  
## 14 14 0.9529765

#Question 2 answer: Looking at the accuracy.df print K=5 was the highest accuracy and will be used as k.  
  
#End question 2  
  
  
#Question 3) Show confusion matrix  
  
set.seed(112)  
  
predicted\_test\_labels <- knn(train = train.norm.df[,-7], test = val.norm.df[,-7], cl = train.norm.df$Personal.Loan, k = 5)  
  
CrossTable(x=val.norm.df$Personal.Loan,y=predicted\_test\_labels, prop.chisq = FALSE)

##   
##   
## Cell Contents  
## |-------------------------|  
## | N |  
## | N / Row Total |  
## | N / Col Total |  
## | N / Table Total |  
## |-------------------------|  
##   
##   
## Total Observations in Table: 1999   
##   
##   
## | predicted\_test\_labels   
## val.norm.df$Personal.Loan | 0 | 1 | Row Total |   
## --------------------------|-----------|-----------|-----------|  
## 0 | 1810 | 2 | 1812 |   
## | 0.999 | 0.001 | 0.906 |   
## | 0.963 | 0.017 | |   
## | 0.905 | 0.001 | |   
## --------------------------|-----------|-----------|-----------|  
## 1 | 69 | 118 | 187 |   
## | 0.369 | 0.631 | 0.094 |   
## | 0.037 | 0.983 | |   
## | 0.035 | 0.059 | |   
## --------------------------|-----------|-----------|-----------|  
## Column Total | 1879 | 120 | 1999 |   
## | 0.940 | 0.060 | |   
## --------------------------|-----------|-----------|-----------|  
##   
##

# End question 3  
  
#Question 4) try again with optimal k  
set.seed(26)  
  
classification <- knn(train = train.norm.df[,-7], test = customer.norm.df, cl = train.norm.df$Personal.Loan, k = 5)  
head(classification)

## [1] 0  
## Levels: 0

new\_customer\_data\_k5 <- customer.norm.df  
new\_customer\_data\_k5$LoanAcceptance <- classification  
# Print the new data with predicted classification  
print(new\_customer\_data\_k5)

## Age Experience Income Family CCAvg Mortgage  
## 1 -0.4684867 -0.880771 0.2074445 -0.3426499 0.0380226 -0.56064  
## Securities.Account CD.Account Online CreditCard Education\_1 Education\_2  
## 1 -0.3381309 -0.257007 0.8430359 1.552232 -0.841302 1.590517  
## Education\_3 LoanAcceptance  
## 1 -0.6580258 0

#Question 4 answer: The customer would still not be classified as accepting the loan  
  
# End question 4  
  
  
#Question 5)  
set.seed(60)  
  
#Re-partition Data into 50% Train, 30% validation 20% test  
Test\_Index\_2 = createDataPartition(DF$Age,p=0.2, list=FALSE) # 20% reserved for Test  
Test\_Data\_2 = DF[Test\_Index\_2,]  
TraVal\_Data\_2 = DF[-Test\_Index\_2,] # Validation and Training data is rest  
  
Train\_Index\_2 = createDataPartition(TraVal\_Data\_2$Age,p=0.75, list=FALSE) # 75% of remaining data as training  
Train\_Data\_2 = TraVal\_Data\_2[Train\_Index\_2,]  
Validation\_Data\_2 = TraVal\_Data\_2[-Train\_Index\_2,] # rest as validation  
  
#normalize  
train.norm.df\_2 <- Train\_Data\_2  
val.norm.df\_2 <- Validation\_Data\_2  
traval.norm.df\_2 <- TraVal\_Data\_2  
test.norm.df\_2 <- Test\_Data\_2  
  
norm.values\_2 <- preProcess(Train\_Data\_2[,-7], method=c("center", "scale"))  
  
train.norm.df\_2[,-7] <- predict(norm.values\_2, Train\_Data\_2[,-7])  
val.norm.df\_2[,-7] <- predict(norm.values\_2, Validation\_Data\_2[,-7])  
traval.norm.df\_2[,-7] <- predict(norm.values\_2, traval.norm.df\_2[,-7])  
test.norm.df\_2[,-7] <- predict(norm.values\_2, Test\_Data\_2[,-7])  
  
predicted\_train\_labels\_2 <- knn(train = train.norm.df\_2[,-7], test = train.norm.df\_2[,-7], cl = train.norm.df\_2$Personal.Loan, k = 5)  
predicted\_val\_labels\_2 <- knn(train = train.norm.df\_2[,-7], test = val.norm.df\_2[,-7], cl = train.norm.df\_2$Personal.Loan, k = 5)  
predicted\_test\_labels\_2 <- knn(train = train.norm.df\_2[,-7], test = test.norm.df\_2[,-7], cl = train.norm.df\_2$Personal.Loan, k = 5)  
  
#Train confusion matrix  
CrossTable(x=train.norm.df\_2$Personal.Loan,y=predicted\_train\_labels\_2, prop.chisq = FALSE)

##   
##   
## Cell Contents  
## |-------------------------|  
## | N |  
## | N / Row Total |  
## | N / Col Total |  
## | N / Table Total |  
## |-------------------------|  
##   
##   
## Total Observations in Table: 3000   
##   
##   
## | predicted\_train\_labels\_2   
## train.norm.df\_2$Personal.Loan | 0 | 1 | Row Total |   
## ------------------------------|-----------|-----------|-----------|  
## 0 | 2690 | 5 | 2695 |   
## | 0.998 | 0.002 | 0.898 |   
## | 0.968 | 0.023 | |   
## | 0.897 | 0.002 | |   
## ------------------------------|-----------|-----------|-----------|  
## 1 | 90 | 215 | 305 |   
## | 0.295 | 0.705 | 0.102 |   
## | 0.032 | 0.977 | |   
## | 0.030 | 0.072 | |   
## ------------------------------|-----------|-----------|-----------|  
## Column Total | 2780 | 220 | 3000 |   
## | 0.927 | 0.073 | |   
## ------------------------------|-----------|-----------|-----------|  
##   
##

#Val confusion Matrix  
CrossTable(x=val.norm.df\_2$Personal.Loan,y=predicted\_val\_labels\_2, prop.chisq = FALSE)

##   
##   
## Cell Contents  
## |-------------------------|  
## | N |  
## | N / Row Total |  
## | N / Col Total |  
## | N / Table Total |  
## |-------------------------|  
##   
##   
## Total Observations in Table: 999   
##   
##   
## | predicted\_val\_labels\_2   
## val.norm.df\_2$Personal.Loan | 0 | 1 | Row Total |   
## ----------------------------|-----------|-----------|-----------|  
## 0 | 919 | 2 | 921 |   
## | 0.998 | 0.002 | 0.922 |   
## | 0.967 | 0.041 | |   
## | 0.920 | 0.002 | |   
## ----------------------------|-----------|-----------|-----------|  
## 1 | 31 | 47 | 78 |   
## | 0.397 | 0.603 | 0.078 |   
## | 0.033 | 0.959 | |   
## | 0.031 | 0.047 | |   
## ----------------------------|-----------|-----------|-----------|  
## Column Total | 950 | 49 | 999 |   
## | 0.951 | 0.049 | |   
## ----------------------------|-----------|-----------|-----------|  
##   
##

#Test confusion matrix  
CrossTable(x=test.norm.df\_2$Personal.Loan,y=predicted\_test\_labels\_2, prop.chisq = FALSE)

##   
##   
## Cell Contents  
## |-------------------------|  
## | N |  
## | N / Row Total |  
## | N / Col Total |  
## | N / Table Total |  
## |-------------------------|  
##   
##   
## Total Observations in Table: 1001   
##   
##   
## | predicted\_test\_labels\_2   
## test.norm.df\_2$Personal.Loan | 0 | 1 | Row Total |   
## -----------------------------|-----------|-----------|-----------|  
## 0 | 901 | 3 | 904 |   
## | 0.997 | 0.003 | 0.903 |   
## | 0.960 | 0.048 | |   
## | 0.900 | 0.003 | |   
## -----------------------------|-----------|-----------|-----------|  
## 1 | 38 | 59 | 97 |   
## | 0.392 | 0.608 | 0.097 |   
## | 0.040 | 0.952 | |   
## | 0.038 | 0.059 | |   
## -----------------------------|-----------|-----------|-----------|  
## Column Total | 939 | 62 | 1001 |   
## | 0.938 | 0.062 | |   
## -----------------------------|-----------|-----------|-----------|  
##   
##

#Question 5 Answer:  
#Comparing the training and validation test confusion matrices to the test confusion matrix, it appears the train confusion matrix is more accurate. This result is likely because the prediction model is being applied against the same dataset by which it was trained. In order to get a true test of the model, the model must be applied to unseen data like in the validation data. The performance of the model on the validation data and the test data is much closer, which is to be expected since they both represent new data. Overall, the validation confusion matrix could be a truer reading of the model's performance since the dataset is larger (30% vs. 20%), it contains more samples.  
  
#End question 5