Assignment_2

Andrew Gutierrez

2022-10-02

Note that my responses to the 5 assignment prompts are actually contained in the 'A.Gutierrez Assignment 2 Responses' TXT file that is also included in my GitHub folder for this assignment.

First, I'll install the requisite 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(ggplot2)
library(dummy)
```

```
## dummy 0.1.3
```

```
## dummyNews()
```

```
library(FNN)
```

Then, I'll read the UniversalBank.csv file into a DataFrame in R:

```
##### Note: the below file path may need to be adjusted, as it currently references a local location on my laptop
UB = read.csv("C:\\Users\\gutiera9\\Documents\\MSBA KSU\\UniversalBank.csv",header=T,sep=",")

head(UB)
```

```
##   ID Age Experience Income ZIP.Code Family CCAvg Education Mortgage
## 1  1  25          1     49    91107      4   1.6         1        0
## 2  2  45         19     34    90089      3   1.5         1        0
## 3  3  39         15     11    94720      1   1.0         1        0
## 4  4  35          9    100    94112      1   2.7         2        0
## 5  5  35          8     45    91330      4   1.0         2        0
## 6  6  37         13     29    92121      4   0.4         2      155
##   Personal.Loan Securities.Account CD.Account Online CreditCard
## 1             0                  1          0      0          0
## 2             0                  1          0      0          0
## 3             0                  0          0      0          0
## 4             0                  0          0      0          0
## 5             0                  0          0      0          1
## 6             0                  0          0      1          0
```

Before proceeding further, I'll first split the categorical column "Education" into separate dummy variables corresponding to 'High School' (value = 1), 'Undergraduate' (value = 2), and 'Graduate' (value = 3).

```r
##### Convert Education column to categorical
UB$Education <- as.factor(UB$Education)

##### Rename the values in the Education column using the levels() function
levels(UB$Education) <- c('High School','Undergraduate','Graduate')
levels(UB$Education)
```

```
## [1] "High School"   "Undergraduate" "Graduate"
```

```r
##### Convert Education Column into dummy variables
dummy_model <- dummyVars(~Education, data=UB)
head(predict(dummy_model,UB))
```

```
##   Education.High School Education.Undergraduate Education.Graduate
## 1                     1                       0                  0
## 2                     1                       0                  0
## 3                     1                       0                  0
## 4                     0                       1                  0
## 5                     0                       1                  0
## 6                     0                       1                  0
```

```r
##### Combine original dataframe with new dummy variables, then remove the original Education column, along with the ID and ZIP code colum
df <- data.frame(predict(dummy_model,newdata = UB))
UB <- cbind(UB,df)
UB <- UB[-c(1,5,8)]
head(UB)
```

```
##   Age Experience Income Family CCAvg Mortgage Personal.Loan Securities.Account
## 1  25          1     49      4   1.6        0             0                  1
## 2  45         19     34      3   1.5        0             0                  1
## 3  39         15     11      1   1.0        0             0                  0
## 4  35          9    100      1   2.7        0             0                  0
## 5  35          8     45      4   1.0        0             0                  0
## 6  37         13     29      4   0.4      155             0                  0
##   CD.Account Online CreditCard Education.High.School Education.Undergraduate
## 1          0      0          0                    1                       0
## 2          0      0          0                    1                       0
```

```
## 3          0      0          0                    1                    0
## 4          0      0          0                    0                    1
## 5          0      0          1                    0                    1
## 6          0      1          0                    0                    1
##   Education.Graduate
## 1                  0
## 2                  0
## 3                  0
## 4                  0
## 5                  0
## 6                  0
```

Next, I'll split the data into a training set (comprising 60% of the total dataset), and a validation set (comprising the remaining 40%). Using the createDataPartition() function will ensure that the training and validation samples remained "stratified" - since only 9.6% of the customers in the dataset accepted the personal loan, we'll need to make sure that ratio of personal loan customers to non-personal loan customers stays constant in both samples.

```
Validation_Index = createDataPartition(UB$Personal.Loan,p=0.4,list=FALSE) # Set aside 40% for the Validation set
Validation_Data = UB[Validation_Index,]
Training_Data = UB[-Validation_Index,] # Remaining data becomes the Training set

print('Summary of Training Data Set: ')
```

```
## [1] "Summary of Training Data Set: "
```

```
summary(Training_Data)
```

```
##       Age          Experience        Income          Family
##  Min.   :23.00   Min.   :-3.00   Min.   :  8.00   Min.   :1.000
##  1st Qu.:35.00   1st Qu.:10.00   1st Qu.: 39.00   1st Qu.:1.000
##  Median :46.00   Median :20.00   Median : 64.00   Median :2.000
##  Mean   :45.39   Mean   :20.16   Mean   : 74.45   Mean   :2.378
##  3rd Qu.:55.00   3rd Qu.:30.00   3rd Qu.:101.00   3rd Qu.:3.000
##  Max.   :67.00   Max.   :43.00   Max.   :224.00   Max.   :4.000
##      CCAvg           Mortgage      Personal.Loan    Securities.Account
##  Min.   : 0.000   Min.   :  0.00   Min.   :0.0000   Min.   :0.0000
##  1st Qu.: 0.700   1st Qu.:  0.00   1st Qu.:0.0000   1st Qu.:0.0000
```

```
##  Median : 1.500   Median :  0.00   Median :0.0000   Median :0.0000
##  Mean   : 1.965   Mean   : 56.49   Mean   :0.1017   Mean   :0.1063
##  3rd Qu.: 2.600   3rd Qu.: 99.00   3rd Qu.:0.0000   3rd Qu.:0.0000
##  Max.   :10.000   Max.   :635.00   Max.   :1.0000   Max.   :1.0000
##    CD.Account          Online         CreditCard    Education.High.School
##  Min.   :0.00000   Min.   :0.0000   Min.   :0.000   Min.   :0.0000
##  1st Qu.:0.00000   1st Qu.:0.0000   1st Qu.:0.000   1st Qu.:0.0000
##  Median :0.00000   Median :1.0000   Median :0.000   Median :0.0000
##  Mean   :0.05933   Mean   :0.6003   Mean   :0.284   Mean   :0.4167
##  3rd Qu.:0.00000   3rd Qu.:1.0000   3rd Qu.:1.000   3rd Qu.:1.0000
##  Max.   :1.00000   Max.   :1.0000   Max.   :1.000   Max.   :1.0000
##  Education.Undergraduate Education.Graduate
##  Min.   :0.0000          Min.   :0.000
##  1st Qu.:0.0000          1st Qu.:0.000
##  Median :0.0000          Median :0.000
##  Mean   :0.2783          Mean   :0.305
##  3rd Qu.:1.0000          3rd Qu.:1.000
##  Max.   :1.0000          Max.   :1.000
```

```
print('Summary of Validation Data Set: ')
```

```
## [1] "Summary of Validation Data Set: "
```

```
summary(Validation_Data)
```

```
##       Age          Experience        Income          Family
##  Min.   :23.00   Min.   :-3.00   Min.   :  8.00   Min.   :1.000
##  1st Qu.:35.00   1st Qu.:10.00   1st Qu.: 39.00   1st Qu.:1.000
##  Median :45.00   Median :20.00   Median : 63.00   Median :2.000
##  Mean   :45.27   Mean   :20.03   Mean   : 72.76   Mean   :2.424
##  3rd Qu.:55.00   3rd Qu.:30.00   3rd Qu.: 94.00   3rd Qu.:4.000
##  Max.   :67.00   Max.   :43.00   Max.   :204.00   Max.   :4.000
##      CCAvg          Mortgage      Personal.Loan    Securities.Account
##  Min.   :0.000   Min.   :  0.00   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.670   1st Qu.:  0.00   1st Qu.:0.0000   1st Qu.:0.0000
##  Median :1.500   Median :  0.00   Median :0.0000   Median :0.0000
##  Mean   :1.898   Mean   : 56.52   Mean   :0.0875   Mean   :0.1015
##  3rd Qu.:2.500   3rd Qu.:102.00   3rd Qu.:0.0000   3rd Qu.:0.0000
```

```
##  Max.   :9.300   Max.   :601.00   Max.   :1.0000   Max.   :1.0000
##    CD.Account        Online         CreditCard    Education.High.School
##  Min.   :0.000   Min.   :0.0000   Min.   :0.000   Min.   :0.000
##  1st Qu.:0.000   1st Qu.:0.0000   1st Qu.:0.000   1st Qu.:0.000
##  Median :0.000   Median :1.0000   Median :0.000   Median :0.000
##  Mean   :0.062   Mean   :0.5915   Mean   :0.309   Mean   :0.423
##  3rd Qu.:0.000   3rd Qu.:1.0000   3rd Qu.:1.000   3rd Qu.:1.000
##  Max.   :1.000   Max.   :1.0000   Max.   :1.000   Max.   :1.000
##  Education.Undergraduate Education.Graduate
##  Min.   :0.000           Min.   :0.000
##  1st Qu.:0.000           1st Qu.:0.000
##  Median :0.000           Median :0.000
##  Mean   :0.284           Mean   :0.293
##  3rd Qu.:1.000           3rd Qu.:1.000
##  Max.   :1.000           Max.   :1.000
```

Before I can jump in to start running a k-NN analysis, I'll first need to normalize the training predictors using a range method, which will result in all variables being re-scaled from 0 to 1. I'll follow by using the normalized values from the training set to also normalize the validation set predictor values. To do this, I'll use CARET's preProcess() function.

```
##### Normalize Training Set Predictors, then use values to normalize Validation Set
normalizedValues <- preProcess(Training_Data[,c(1:6,8:14)], method=c("range"))
Training_Data[,c(1:6,8:14)] <- predict(normalizedValues, Training_Data[,c(1:6,8:14)])
Validation_Data[,c(1:6,8:14)]<- predict(normalizedValues, Validation_Data[,c(1:6,8:14)])

print('Summary of normalized Training Data Predictors: ')
```

```
## [1] "Summary of normalized Training Data Predictors: "
```

```
summary(Training_Data)
```

```
##       Age           Experience         Income           Family
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.2727   1st Qu.:0.2826   1st Qu.:0.1435   1st Qu.:0.0000
##  Median :0.5227   Median :0.5000   Median :0.2593   Median :0.3333
##  Mean   :0.5088   Mean   :0.5034   Mean   :0.3076   Mean   :0.4593
##  3rd Qu.:0.7273   3rd Qu.:0.7174   3rd Qu.:0.4306   3rd Qu.:0.6667
```

```
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##      CCAvg            Mortgage       Personal.Loan    Securities.Account
##  Min.   :0.0000   Min.   :0.00000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.0700   1st Qu.:0.00000   1st Qu.:0.0000   1st Qu.:0.0000
##  Median :0.1500   Median :0.00000   Median :0.0000   Median :0.0000
##  Mean   :0.1965   Mean   :0.08896   Mean   :0.1017   Mean   :0.1063
##  3rd Qu.:0.2600   3rd Qu.:0.15591   3rd Qu.:0.0000   3rd Qu.:0.0000
##  Max.   :1.0000   Max.   :1.00000   Max.   :1.0000   Max.   :1.0000
##    CD.Account          Online         CreditCard    Education.High.School
##  Min.   :0.00000   Min.   :0.0000   Min.   :0.000   Min.   :0.0000
##  1st Qu.:0.00000   1st Qu.:0.0000   1st Qu.:0.000   1st Qu.:0.0000
##  Median :0.00000   Median :1.0000   Median :0.000   Median :0.0000
##  Mean   :0.05933   Mean   :0.6003   Mean   :0.284   Mean   :0.4167
##  3rd Qu.:0.00000   3rd Qu.:1.0000   3rd Qu.:1.000   3rd Qu.:1.0000
##  Max.   :1.00000   Max.   :1.0000   Max.   :1.000   Max.   :1.0000
##  Education.Undergraduate Education.Graduate
##  Min.   :0.0000          Min.   :0.000
##  1st Qu.:0.0000          1st Qu.:0.000
##  Median :0.0000          Median :0.000
##  Mean   :0.2783          Mean   :0.305
##  3rd Qu.:1.0000          3rd Qu.:1.000
##  Max.   :1.0000          Max.   :1.000
```

```
print('Summary of normalized Validation Data Predictors: ')
```

```
## [1] "Summary of normalized Validation Data Predictors: "
```

```
summary(Validation_Data)
```

```
##       Age           Experience         Income           Family
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.2727   1st Qu.:0.2826   1st Qu.:0.1435   1st Qu.:0.0000
##  Median :0.5000   Median :0.5000   Median :0.2546   Median :0.3333
##  Mean   :0.5060   Mean   :0.5006   Mean   :0.2998   Mean   :0.4747
##  3rd Qu.:0.7273   3rd Qu.:0.7174   3rd Qu.:0.3981   3rd Qu.:1.0000
##  Max.   :1.0000   Max.   :1.0000   Max.   :0.9074   Max.   :1.0000
##      CCAvg            Mortgage       Personal.Loan    Securities.Account
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
```

```
##  1st Qu.:0.0670   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000
##  Median :0.1500   Median :0.0000   Median :0.0000   Median :0.0000
##  Mean   :0.1898   Mean   :0.0890   Mean   :0.0875   Mean   :0.1015
##  3rd Qu.:0.2500   3rd Qu.:0.1606   3rd Qu.:0.0000   3rd Qu.:0.0000
##  Max.   :0.9300   Max.   :0.9465   Max.   :1.0000   Max.   :1.0000
##    CD.Account        Online         CreditCard    Education.High.School
##  Min.   :0.000   Min.   :0.0000   Min.   :0.000   Min.   :0.000
##  1st Qu.:0.000   1st Qu.:0.0000   1st Qu.:0.000   1st Qu.:0.000
##  Median :0.000   Median :1.0000   Median :0.000   Median :0.000
##  Mean   :0.062   Mean   :0.5915   Mean   :0.309   Mean   :0.423
##  3rd Qu.:0.000   3rd Qu.:1.0000   3rd Qu.:1.000   3rd Qu.:1.000
##  Max.   :1.000   Max.   :1.0000   Max.   :1.000   Max.   :1.000
##  Education.Undergraduate Education.Graduate
##  Min.   :0.000           Min.   :0.000
##  1st Qu.:0.000           1st Qu.:0.000
##  Median :0.000           Median :0.000
##  Mean   :0.284           Mean   :0.293
##  3rd Qu.:1.000           3rd Qu.:1.000
##  Max.   :1.000           Max.   :1.000
```

Perform a k-NN classification with all predictors except ID and ZIP code using k = 1. Specify the success class as 1 (loan acceptance), and use the default cutoff value of 0.5.

Now that we have our normalized predictor variables, and our success class to serve as the labels (Personal Loan), we can proceed with running a k-NN model with k equal to 1.

```
##### Run KNN Function for K=1
predicted_val_labels <- knn(Training_Data[,c(1:6,8:14)],Validation_Data[,c(1:6,8:14)],cl=Training_Data[,7],k=1)

##### View class output for the first 100 records in the validation set
print(predicted_val_labels[1:100])
```

```
##   [1] 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
##  [38] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
##  [75] 0 0 1 1 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0
## Levels: 0 1
```

Consider the following customer: 1. Age = 40, Experience = 10, Income = 84, Family = 2, CCAvg = 2, Education_1 = 0, Education_2 = 1, Education_3 = 0, Mortgage = 0, Securities Account = 0, CD Account = 0, Online = 1, and Credit Card = 1.

How would this customer be classified?

```
##### Create variables for new customer
Age.new <- 40
Experience.new <- 10
Income.new <- 84
Family.new <- 2
CCAvg.new <- 2
Education.High.School.new <- 0
Education.Undergraduate.new <- 1
Education.Graduate.new <- 0
Mortgage.new <- 0
Securities.Account.new <- 0
CD.Account.new <- 0
Online.new <- 1
CreditCard.new <- 1

##### Create new dataframe for variables for the existing customer
NewCustomer = data.frame(Age = Age.new, Experience = Experience.new, Income = Income.new, Family = Family.new, CCAvg = CCAvg.new, Educatio

##### Normalize and view new dataframe
NewCustomer <- predict(normalizedValues,NewCustomer)
NewCustomer
```

```
##         Age Experience    Income    Family CCAvg Education.High.School
## 1 0.3863636  0.2826087 0.3518519 0.3333333   0.2                       0
##   Education.Undergraduate Education.Graduate Mortgage Securities.Account
## 1                       1                  0        0                   0
##   CD.Account Online CreditCard
## 1          0      1          1
```

```
##### Run the model again, testing on the new dataframe
predicted_val_labels.2 <- knn(Training_Data[,c(1:6,8:14)],NewCustomer,cl=Training_Data[,7],k=1)

##### Display the resulting class
print(predicted_val_labels.2[1])
```

```
## [1] 0
## Levels: 0
```

What is a choice of k that balances between overfitting and ignoring the predictor information?

```
##### Create dataframe containing all values of k 1 through 15
accuracy.df <- data.frame(k = seq(1, 15, 1), accuracy = rep(0, 15))

##### Loop through all values of K in the dataframe to find the ideal one
for(i in 1:15) {
  hypertune <- knn(train=Training_Data[,c(1:6,8:14)], test=Validation_Data[,c(1:6,8:14)],
                   cl = Training_Data[,7], k = i,prob=TRUE)
  accuracy.df[i, 2] <- confusionMatrix(hypertune, as.factor(Validation_Data[, 7]))$overall[1]
}
accuracy.df
```

```
##     k accuracy
## 1   1   0.9650
## 2   2   0.9520
## 3   3   0.9615
## 4   4   0.9545
## 5   5   0.9585
## 6   6   0.9525
## 7   7   0.9570
## 8   8   0.9490
## 9   9   0.9515
## 10 10   0.9455
## 11 11   0.9500
## 12 12   0.9420
## 13 13   0.9455
## 14 14   0.9400
## 15 15   0.9425
```

Show the confusion matrix for the validation data that results from using the best k.

```
library("gmodels")

##### Run KNN Function again with new value of k
predicted_val_labels.3 <- knn(Training_Data[,c(1:6,8:14)],Validation_Data[,c(1:6,8:14)],cl=Training_Data[,7],k=1,prob=TRUE)

##### Generate Confusion Matrix
CrossTable(x=Validation_Data[,7],y=predicted_val_labels.3,prop.chisq=FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  2000
##
##
##                      | predicted_val_labels.3
## Validation_Data[, 7] |         0 |         1 | Row Total |
## ---------------------|-----------|-----------|-----------|
##                    0 |      1803 |        22 |      1825 |
##                      |     0.988 |     0.012 |     0.912 |
##                      |     0.974 |     0.148 |           |
##                      |     0.901 |     0.011 |           |
## ---------------------|-----------|-----------|-----------|
##                    1 |        48 |       127 |       175 |
##                      |     0.274 |     0.726 |     0.087 |
##                      |     0.026 |     0.852 |           |
##                      |     0.024 |     0.064 |           |
## ---------------------|-----------|-----------|-----------|
##         Column Total |      1851 |       149 |      2000 |
##                      |     0.925 |     0.074 |           |
## ---------------------|-----------|-----------|-----------|
##
##
```

Consider the following customer: Age = 40, Experience = 10, Income = 84, Family = 2, CCAvg = 2, Education_1 = 0, Education_2 = 1, Education_3 = 0, Mortgage = 0, Securities Account = 0, CD Account = 0, Online = 1 and Credit Card = 1. Classify the customer using the best k.

```
##### Run the model again on the previous dataframe, but changing the value of k
predicted_val_labels.4 <- knn(Training_Data[,c(1:6,8:14)],NewCustomer,cl=Training_Data[,7],k=3)
```

```
##### Display the resulting class
print(predicted_val_labels.4[1])
```

```
## [1] 0
## Levels: 0
```

Repartition the data, this time into training, validation, and test sets (50% : 30% : 20%). Apply the k-NN method with the k chosen above. Compare the confusion matrix of the test set with that of the training and validation sets. Comment on the differences and their reason.

```
set.seed(15)

Test_Index = createDataPartition(UB$Personal.Loan,p=0.2, list=FALSE) # Set aside 20% for the Validation set
Test_Data = UB[Test_Index,]
TraVal_Data = UB[-Test_Index,] # Validation and Training data is rest

Train_Index.2 = createDataPartition(TraVal_Data$Personal.Loan,p=0.625, list=FALSE) # 62.5% of remaining data becomes the training set
Training_Data.2 = TraVal_Data[Train_Index.2,]
Validation_Data.2 = TraVal_Data[-Train_Index.2,] # rest as validation

##### View number of records in training set:
dim(Training_Data.2)[1]
```

```
## [1] 2500
```

```
##### View number of records in validation set:
dim(Validation_Data.2)[1]
```

```
## [1] 1500
```

```
##### View number of records in test set:
dim(Test_Data)[1]
```

```
## [1] 1000
```

```
##### Normalize values in all data sets
normalizedValues.2 <- preProcess(Training_Data.2[,c(1:6,8:14)], method=c("range"))
Training_Data.2[,c(1:6,8:14)] <- predict(normalizedValues.2, Training_Data.2[,c(1:6,8:14)])
Validation_Data.2[,c(1:6,8:14)] <- predict(normalizedValues.2, Validation_Data.2[,c(1:6,8:14)])
Test_Data[,c(1:6,8:14)] <- predict(normalizedValues.2, Test_Data[,c(1:6,8:14)])

print('Summary of Training Data:')
```

```
## [1] "Summary of Training Data:"
```

```
summary(Training_Data.2)
```

```
##       Age           Experience         Income           Family
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.2898   1st Qu.:0.2889   1st Qu.:0.1523   1st Qu.:0.0000
##  Median :0.5000   Median :0.5111   Median :0.2792   Median :0.3333
##  Mean   :0.5079   Mean   :0.5134   Mean   :0.3295   Mean   :0.4775
##  3rd Qu.:0.7273   3rd Qu.:0.7333   3rd Qu.:0.4454   3rd Qu.:1.0000
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##      CCAvg           Mortgage       Personal.Loan   Securities.Account
##  Min.   :0.0000   Min.   :0.00000   Min.   :0.000   Min.   :0.0000
##  1st Qu.:0.0700   1st Qu.:0.00000   1st Qu.:0.000   1st Qu.:0.0000
##  Median :0.1600   Median :0.00000   Median :0.000   Median :0.0000
##  Mean   :0.1943   Mean   :0.08663   Mean   :0.094   Mean   :0.1088
##  3rd Qu.:0.2500   3rd Qu.:0.15591   3rd Qu.:0.000   3rd Qu.:0.0000
##  Max.   :1.0000   Max.   :1.00000   Max.   :1.000   Max.   :1.0000
##    CD.Account       Online         CreditCard     Education.High.School
##  Min.   :0.00   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.00   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000
##  Median :0.00   Median :1.0000   Median :0.0000   Median :0.0000
##  Mean   :0.06   Mean   :0.5912   Mean   :0.2908   Mean   :0.4176
##  3rd Qu.:0.00   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000
##  Max.   :1.00   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##  Education.Undergraduate Education.Graduate
##  Min.   :0.0000          Min.   :0.0000
##  1st Qu.:0.0000          1st Qu.:0.0000
##  Median :0.0000          Median :0.0000
```

```
##  Mean   :0.2832          Mean   :0.2992
##  3rd Qu.:1.0000          3rd Qu.:1.0000
##  Max.   :1.0000          Max.   :1.0000
```

```
print('Summary of Validation Data:')
```

```
## [1] "Summary of Validation Data:"
```

```
summary(Validation_Data.2)
```

```
##       Age            Experience         Income           Family
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.2727   1st Qu.:0.2889   1st Qu.:0.1574   1st Qu.:0.0000
##  Median :0.5227   Median :0.5333   Median :0.2893   Median :0.3333
##  Mean   :0.5120   Mean   :0.5177   Mean   :0.3412   Mean   :0.4562
##  3rd Qu.:0.7273   3rd Qu.:0.7333   3rd Qu.:0.4721   3rd Qu.:0.6667
##  Max.   :1.0000   Max.   :1.0222   Max.   :1.0660   Max.   :1.0000
##      CCAvg            Mortgage       Personal.Loan     Securities.Account
##  Min.   :0.0000   Min.   :0.00000   Min.   :0.00000   Min.   :0.0000
##  1st Qu.:0.0700   1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.0000
##  Median :0.1500   Median :0.00000   Median :0.00000   Median :0.0000
##  Mean   :0.1963   Mean   :0.09015   Mean   :0.09933   Mean   :0.1007
##  3rd Qu.:0.2600   3rd Qu.:0.16063   3rd Qu.:0.00000   3rd Qu.:0.0000
##  Max.   :1.0000   Max.   :0.91496   Max.   :1.00000   Max.   :1.0000
##    CD.Account          Online         CreditCard     Education.High.School
##  Min.   :0.00000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.00000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000
##  Median :0.00000   Median :1.0000   Median :0.0000   Median :0.0000
##  Mean   :0.06067   Mean   :0.5987   Mean   :0.2967   Mean   :0.4213
##  3rd Qu.:0.00000   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000
##  Max.   :1.00000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##  Education.Undergraduate Education.Graduate
##  Min.   :0.0000          Min.   :0.000
##  1st Qu.:0.0000          1st Qu.:0.000
##  Median :0.0000          Median :0.000
##  Mean   :0.2707          Mean   :0.308
##  3rd Qu.:1.0000          3rd Qu.:1.000
##  Max.   :1.0000          Max.   :1.000
```

```
print('Summary of Test Data:')
```

```
## [1] "Summary of Test Data:"
```

```
summary(Test_Data)
```

```
##       Age           Experience          Income           Family
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.2727   1st Qu.:0.2667   1st Qu.:0.1574   1st Qu.:0.0000
##  Median :0.5000   Median :0.5111   Median :0.2843   Median :0.3333
##  Mean   :0.5009   Mean   :0.5071   Mean   :0.3339   Mean   :0.4493
##  3rd Qu.:0.7273   3rd Qu.:0.7333   3rd Qu.:0.4569   3rd Qu.:0.6667
##  Max.   :1.0000   Max.   :1.0222   Max.   :1.0964   Max.   :1.0000
##      CCAvg            Mortgage       Personal.Loan   Securities.Account
##  Min.   :0.0000   Min.   :0.00000   Min.   :0.000   Min.   :0.000
##  1st Qu.:0.0600   1st Qu.:0.00000   1st Qu.:0.000   1st Qu.:0.000
##  Median :0.1500   Median :0.00000   Median :0.000   Median :0.000
##  Mean   :0.1887   Mean   :0.09307   Mean   :0.096   Mean   :0.099
##  3rd Qu.:0.2500   3rd Qu.:0.16693   3rd Qu.:0.000   3rd Qu.:0.000
##  Max.   :0.9300   Max.   :0.92756   Max.   :1.000   Max.   :1.000
##    CD.Account        Online         CreditCard    Education.High.School
##  Min.   :0.000   Min.   :0.000   Min.   :0.000   Min.   :0.00
##  1st Qu.:0.000   1st Qu.:0.000   1st Qu.:0.000   1st Qu.:0.00
##  Median :0.000   Median :1.000   Median :0.000   Median :0.00
##  Mean   :0.061   Mean   :0.608   Mean   :0.298   Mean   :0.42
##  3rd Qu.:0.000   3rd Qu.:1.000   3rd Qu.:1.000   3rd Qu.:1.00
##  Max.   :1.000   Max.   :1.000   Max.   :1.000   Max.   :1.00
##  Education.Undergraduate Education.Graduate
##  Min.   :0.000           Min.   :0.000
##  1st Qu.:0.000           1st Qu.:0.000
##  Median :0.000           Median :0.000
##  Mean   :0.289           Mean   :0.291
##  3rd Qu.:1.000           3rd Qu.:1.000
##  Max.   :1.000           Max.   :1.000
```

```
##### Run k-NN function with ideal value of k using training data
predicted_train_labels.2 <- knn(Training_Data.2[,c(1:6,8:14)],Training_Data.2[,c(1:6,8:14)],cl=Training_Data.2[,7],k=3,prob=TRUE)

##### Run k-NN function again using validation data
predicted_val_labels.5 <- knn(Training_Data.2[,c(1:6,8:14)],Validation_Data.2[,c(1:6,8:14)],cl=Training_Data.2[,7],k=3,prob=TRUE)

##### Run k-NN function one more time using test data
predicted_test_labels <- knn(Training_Data.2[,c(1:6,8:14)],Test_Data[,c(1:6,8:14)],cl=Training_Data.2[,7],k=3,prob=TRUE)


##### Generate Confusion Matrix for training data
print('Training Data Confusion Matrix:')
```

```
## [1] "Training Data Confusion Matrix:"
```

```
CrossTable(x=Training_Data.2[,7],y=predicted_train_labels.2,prop.chisq=FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  2500
##
##
##                      | predicted_train_labels.2
## Training_Data.2[, 7] |         0 |         1 | Row Total |
## ---------------------|-----------|-----------|-----------|
##                    0 |      2258 |         7 |      2265 |
##                      |     0.997 |     0.003 |     0.906 |
##                      |     0.975 |     0.038 |           |
```

```
##                       |     0.903 |     0.003 |           |
## ----------------------|-----------|-----------|-----------|
##                     1 |        59 |       176 |       235 |
##                       |     0.251 |     0.749 |     0.094 |
##                       |     0.025 |     0.962 |           |
##                       |     0.024 |     0.070 |           |
## ----------------------|-----------|-----------|-----------|
##          Column Total |      2317 |       183 |      2500 |
##                       |     0.927 |     0.073 |           |
## ----------------------|-----------|-----------|-----------|
##
##
```

```
##### Generate Confusion Matrix for validation data
print('Validation Data Confusion Matrix:')
```

```
## [1] "Validation Data Confusion Matrix:"
```

```
CrossTable(x=Validation_Data.2[,7],y=predicted_val_labels.5,prop.chisq=FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  1500
##
##
##                       | predicted_val_labels.5
## Validation_Data.2[, 7] |         0 |         1 | Row Total |
## ----------------------|-----------|-----------|-----------|
##                     0 |      1340 |        11 |      1351 |
```

```
##                       |     0.992 |     0.008 |     0.901 |
##                       |     0.959 |     0.107 |           |
##                       |     0.893 |     0.007 |           |
## ----------------------|-----------|-----------|-----------|
##                     1 |        57 |        92 |       149 |
##                       |     0.383 |     0.617 |     0.099 |
##                       |     0.041 |     0.893 |           |
##                       |     0.038 |     0.061 |           |
## ----------------------|-----------|-----------|-----------|
##          Column Total |      1397 |       103 |      1500 |
##                       |     0.931 |     0.069 |           |
## ----------------------|-----------|-----------|-----------|
##
##
```

```
##### Generate Confusion Matrix for test data
print('Test Data Confusion Matrix:')
```

```
## [1] "Test Data Confusion Matrix:"
```

```
CrossTable(x=Test_Data[,7],y=predicted_test_labels,prop.chisq=FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  1000
##
##
##                | predicted_test_labels
## Test_Data[, 7] |         0 |         1 | Row Total |
```

```
## ---------------|-----------|-----------|-----------|
##              0 |       897 |         7 |       904 |
##                |     0.992 |     0.008 |     0.904 |
##                |     0.958 |     0.109 |           |
##                |     0.897 |     0.007 |           |
## ---------------|-----------|-----------|-----------|
##              1 |        39 |        57 |        96 |
##                |     0.406 |     0.594 |     0.096 |
##                |     0.042 |     0.891 |           |
##                |     0.039 |     0.057 |           |
## ---------------|-----------|-----------|-----------|
##   Column Total |       936 |        64 |      1000 |
##                |     0.936 |     0.064 |           |
## ---------------|-----------|-----------|-----------|
##
##
```