FML ASSIGNMENT2

DIVYA CHANDRASEKARAN

811284790

2023-09-30

The purpose of this assignment is to use k-NN for classification utilizing open data on 5000 customers from a financial institution to predict whether a liability customer would accept a personal loan offer.

Variables used in the Universal Bank dataset are the following:

• ID: Customer's identifier.

• Age: Customer's age.

• Experience: Number of customer's years of experience.

• Income: Annual income.

• Zip Code: Customer's location area.

• Family: Number of family members.

• CCAvg: Average spending on credit cards.

• Education: Highest customer's education level.

1 = High School

2 = Undergraduate.

3 = Graduate.

• Mortgage: Value of debt if the customer has a mortgage.

• Personal Loan: Indicates if the customer accepted or rejected the loan offered in last campaign.

1 = Accepted

0 = Rejected

• Securities Account: Indicates if the customer has security account.

1 = Yes.

0 = No.

• CD Account: Indicates if the customer has a Certificate of Deposit.

1 = Yes.

0 = No.

• Online: Indicates if the customer has Internet banking facilites.

1 = Yes.

0 = No.

• CredictCard: Indicates if the customer currently has credit cards.

1 = Yes.

0 = No.

```
#Install required packages
library(caret) #To split the dataset in training, validation, and testing.

library(class) #For classification of data
library(e1071) #For easy implementation of SVM
library(dplyr) #To select a subset of variables

library(readr) #To read files
library(gmodels) #To create the confusion matrix

#read in CSV file for training and testing
 ubank <- read.csv("C:/Users/spadd/OneDrive/Desktop/UniversalBank.csv")
 dim(ubank)

## [1] 5000   14

 t(t(names(ubank)))

##       [,1]
##  [1,] "ID"
##  [2,] "Age"
##  [3,] "Experience"
##  [4,] "Income"
##  [5,] "ZIP.Code"
##  [6,] "Family"
##  [7,] "CCAvg"
##  [8,] "Education"
```

```
##  [9,] "Mortgage"
## [10,] "Personal.Loan"
## [11,] "Securities.Account"
## [12,] "CD.Account"
## [13,] "Online"
## [14,] "CreditCard"
```

```
#SEEING THE DATA FRAME'S STRUCTURE
str(ubank)
```

```
## 'data.frame':    5000 obs. of  14 variables:
##  $ ID                : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Age               : int  25 45 39 35 35 37 53 50 35 34 ...
##  $ Experience        : int  1 19 15 9 8 13 27 24 10 9 ...
##  $ Income            : int  49 34 11 100 45 29 72 22 81 180 ...
##  $ ZIP.Code          : int  91107 90089 94720 94112 91330 92121 91711
93943 90089 93023 ...
##  $ Family            : int  4 3 1 1 4 4 2 1 3 1 ...
##  $ CCAvg             : num  1.6 1.5 1 2.7 1 0.4 1.5 0.3 0.6 8.9 ...
##  $ Education         : int  1 1 1 2 2 2 2 3 2 3 ...
##  $ Mortgage          : int  0 0 0 0 0 155 0 0 104 0 ...
##  $ Personal.Loan     : int  0 0 0 0 0 0 0 0 0 1 ...
##  $ Securities.Account: int  1 1 0 0 0 0 0 0 0 0 ...
##  $ CD.Account        : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Online            : int  0 0 0 0 0 1 1 0 1 0 ...
##  $ CreditCard        : int  0 0 0 0 1 0 0 1 0 0 ...
```

```
#DESCRIPTIVE STATISTICS
summary(ubank)
```

```
##        ID            Age          Experience       Income
ZIP.Code
##  Min.   :   1   Min.   :23.00   Min.   :-3.0   Min.   :  8.00   Min.   :
9307
##  1st Qu.:1251   1st Qu.:35.00   1st Qu.:10.0   1st Qu.: 39.00   1st
Qu.:91911
##  Median :2500   Median :45.00   Median :20.0   Median : 64.00   Median
:93437
##  Mean   :2500   Mean   :45.34   Mean   :20.1   Mean   : 73.77   Mean
:93153
##  3rd Qu.:3750   3rd Qu.:55.00   3rd Qu.:30.0   3rd Qu.: 98.00   3rd
Qu.:94608
##  Max.   :5000   Max.   :67.00   Max.   :43.0   Max.   :224.00   Max.
:96651
##      Family          CCAvg          Education        Mortgage
##  Min.   :1.000   Min.   : 0.000   Min.   :1.000   Min.   :  0.0
##  1st Qu.:1.000   1st Qu.: 0.700   1st Qu.:1.000   1st Qu.:  0.0
##  Median :2.000   Median : 1.500   Median :2.000   Median :  0.0
##  Mean   :2.396   Mean   : 1.938   Mean   :1.881   Mean   : 56.5
##  3rd Qu.:3.000   3rd Qu.: 2.500   3rd Qu.:3.000   3rd Qu.:101.0
##  Max.   :4.000   Max.   :10.000   Max.   :3.000   Max.   :635.0
```

```
##  Personal.Loan   Securities.Account   CD.Account         Online
##  Min.   :0.000   Min.   :0.0000     Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.000   1st Qu.:0.0000     1st Qu.:0.0000   1st Qu.:0.0000
##  Median :0.000   Median :0.0000     Median :0.0000   Median :1.0000
##  Mean   :0.096   Mean   :0.1044     Mean   :0.0604   Mean   :0.5968
##  3rd Qu.:0.000   3rd Qu.:0.0000     3rd Qu.:0.0000   3rd Qu.:1.0000
##  Max.   :1.000   Max.   :1.0000     Max.   :1.0000   Max.   :1.0000
##    CreditCard
##  Min.   :0.000
##  1st Qu.:0.000
##  Median :0.000
##  Mean   :0.294
##  3rd Qu.:1.000
##  Max.   :1.000

#DROP ID AND ZIP
ubank <- ubank[,-c(1,5)]

#ONLY EDUCATION NEEDS TO BE CONVERTED INTO FACTOR
ubank$Education <- as.factor(ubank$Education)

#After reviewing data, it appears all categorical variables are in binary
form except for EDUCATION.Therefore, we will need to convert to dummy before
implementing k-NN.

ubank$Education <- as.factor(ubank$Education)
dummy_model <- dummyVars(~., data=ubank) #this create dummy groups
head(predict(dummy_model,ubank))

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

ubank1 <- as.data.frame(predict(dummy_model, ubank))
```

```r
#We need to ensure that we are getting the same sample if we return the code

set.seed(1)
train_index <- sample(row.names(ubank1), 0.6*dim(ubank1)[1])
validate_index <- setdiff(row.names(ubank1), train_index)
train_ubank <- ubank1[train_index,]
Validate_ubank <- ubank1[validate_index,]
t(t(names(train_ubank)))
```

```
##       [,1]
##  [1,] "Age"
##  [2,] "Experience"
##  [3,] "Income"
##  [4,] "Family"
##  [5,] "CCAvg"
##  [6,] "Education.1"
##  [7,] "Education.2"
##  [8,] "Education.3"
##  [9,] "Mortgage"
## [10,] "Personal.Loan"
## [11,] "Securities.Account"
## [12,] "CD.Account"
## [13,] "Online"
## [14,] "CreditCard"
```

```r
#SPLITTING DATA INTO 60 PERCENT TRAINING AND 40 PERCENT VALIDATION
library(caTools)
set.seed(1)
split <- sample.split(ubank1, SplitRatio = 0.6)
training_set <- subset(ubank1, split == TRUE)
validation_set <- subset(ubank1, split == FALSE)

#PRINTING THE SIZES OF TRAINING AND VALIDATION SETS
print(paste("The size of the training set is:", nrow(training_set)))
```

```
## [1] "The size of the training set is: 2858"
```

```r
print(paste("The size of the validation set is:", nrow(validation_set)))
```

```
## [1] "The size of the validation set is: 2142"
```

```r
#Define success level of personal loan as 1. In "R" first level is failure
and second is success. In this case, the default is set to success.
ubank1$Personal.Loan <- as.factor(ubank1$Personal.Loan)
levels(ubank1$Personal.Loan)
```

```
## [1] "0" "1"
```

```r
#Normalize continuous variables used in modeling
train_ubank_norm <- train_ubank[,-10]
validate_ubank_norm <- Validate_ubank [,-10]

norm_values <- preProcess(train_ubank[, -10], method = c ("center", "scale"))
train_ubank_norm <- predict(norm_values, train_ubank[,-10])
validate_ubank_norm <- predict(norm_values, Validate_ubank[,-10])

train_predictors <-train_ubank_norm[, -10]
validate_predictors <- validate_ubank_norm[,-10]

train_labels <-train_ubank_norm[,12]
validate_labels <- validate_ubank_norm[,12]
```

```r
#Question1

#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. Perform a k-NN classification
with all predictors except ID and ZIP code using k = 1. Remember to transform
categorical predictors with more than two categories into dummy variables
first.

new.data <- data.frame(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, CreditCard = 1)
dim(new.data)

## [1]  1 13

#NORMALIZE THE NEW CUSTOMER
new.data.normalized <- new.data
new.data.normalized <- predict(norm_values, new.data.normalized)

#NOW, LET'S PREDICT USING KNN INTERPRETATION
predict_values_q1 <- class:: knn(train = train_ubank_norm,
                                 test = new.data.normalized,
                                 cl = train_ubank$Personal.Loan, k = 1)
predict_values_q1

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

#The Output below suggests that the model predicts that a person with these
criteria would not take out a personal loan.
```

```r
#QUESTION 2
```

Hyper tuning using Validation

#To determine k, we use the performance on the validation set.

#Here, we will vary the value of k from 1 to 14

initialize a data frame with two columns: k, and accuracy.

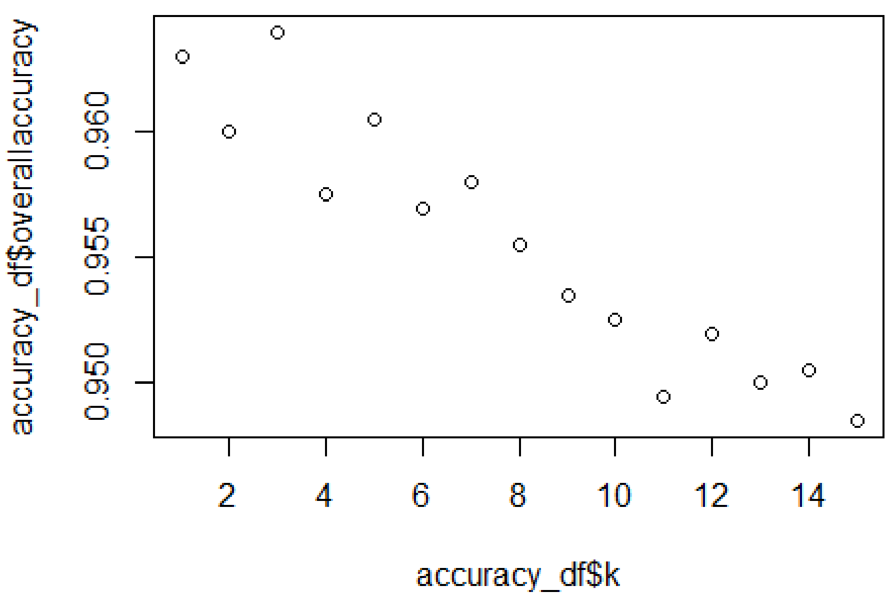
```r
accuracy_df <- data.frame(k = seq(1, 15, 1), overallaccuracy = rep(0,15))
for (i in 1:15) {
  knn_pred <- class:: knn(train = train_ubank_norm,
                          test = validate_ubank_norm,
                          cl = train_ubank$Personal.Loan, k=i)
  accuracy_df[i,2] <- confusionMatrix(knn_pred,

as.factor(Validate_ubank$Personal.Loan), positive = "1")$overall[1]

}
which(accuracy_df[,2]== max(accuracy_df[,2]))

## [1] 3

plot(accuracy_df$k, accuracy_df$overallaccuracy)
```



```
head(accuracy_df)

##   k accuracy
## 1 1   0.9615
## 2 2   0.9560
## 3 3   0.9680
## 4 4   0.9605
## 5 5   0.9650
## 6 6   0.9590
```

```
#QUESTION 3
#CONFUSION MATRIX
library(gmodels)
predicted_validate_labels_k5 <- knn(train_predictors, validate_predictors, cl
= train_labels, k=5)
conf_matrix <- CrossTable(x=validate_labels,
y=predicted_validate_labels_k5,prop.chisq = FALSE)

##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  2000
##
##
##                   | predicted_validate_labels_k5
##   validate_labels | -1.18626953935887 |  0.84269774574752 |         Row
Total |
## ------------------|-------------------|-------------------|---------------
----|
## -1.18626953935887 |               762 |                 8 |
770 |
##                   |             0.990 |             0.010 |
0.385 |
##                   |             0.999 |             0.006 |
|
```

```
##                   |              0.381 |              0.004 |
|
## ------------------|-------------------|-------------------|---------------
----|
##  0.84269774574752 |                  1 |               1229 |
1230 |
##                   |              0.001 |              0.999 |
0.615 |
##                   |              0.001 |              0.994 |
|
##                   |              0.000 |              0.615 |
|
## ------------------|-------------------|-------------------|---------------
----|
##      Column Total |                763 |               1237 |
2000 |
##                   |              0.382 |              0.619 |
|
## ------------------|-------------------|-------------------|---------------
----|
##
##
```

```r
#create probability set
set.seed(1234)
my_knnprob <-knn(train_predictors,
                 validate_predictors,
                 cl = train_labels, k=1, prob=TRUE)
class_prob <- attr(my_knnprob, 'prob')
# See the first rows
head(my_knnprob)
```

```
## [1] -1.18626953935887 -1.18626953935887 0.84269774574752  -
1.18626953935887
## [5] -1.18626953935887 0.84269774574752
## Levels: -1.18626953935887 0.84269774574752
```

```r
#Calculating accuracy
k1_accuracy <- (conf_matrix$t[2,2] + conf_matrix$t[1,1])/ sum(conf_matrix$t)
print(k1_accuracy)
```

```
## [1] 0.9955
```

```r
#Calculating recall
k1_recall <- conf_matrix$t[2,2]/ (conf_matrix$t[2,2] + conf_matrix$t[2,1])
print(k1_recall)
```

```
## [1] 0.999187
```

```
#Calculating Precision
k1_precision <- conf_matrix$t[2,2]/ (conf_matrix$t[2,2] + conf_matrix$t[1,2])
print(k1_precision)
```

```
## [1] 0.9935327
```

```
#Calculating Specificity
k1_specificity <- conf_matrix$t[1,1]/ (conf_matrix$t[1,1] +
conf_matrix$t[1,2])
print(k1_specificity)
```

```
## [1] 0.9896104
```

```
##THEREFORE, WE CAN SAY THAT THE MODEL IS LEARNING WELL
```

```
predicted_validate_labels_k5 <- knn(train_predictors, validate_predictors, cl
= train_labels, k =1)
CrossTable(x = validate_labels, y = predicted_validate_labels_k5, prop.chisq
= FALSE)
```

```
## 
## 
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
## 
## 
## Total Observations in Table:  2000 
## 
## 
##                   | predicted_validate_labels_k5 
##   validate_labels | -1.18626953935887 |  0.84269774574752 |         Row 
Total | 
## ------------------|-------------------|-------------------|---------------
----|
## -1.18626953935887 |               766 |                 4 |           
770 | 
##                   |             0.995 |             0.005 |           
0.385 | 
##                   |             0.997 |             0.003 |           
| 
##                   |             0.383 |             0.002 |           
|
```

```
## ------------------|------------------|------------------|------------------|
##  0.84269774574752 |                2 |             1228 |             1230 |
##                   |            0.002 |            0.998 |            0.615 |
##                   |            0.003 |            0.997 |                  |
##                   |            0.001 |            0.614 |                  |
## ------------------|------------------|------------------|------------------|
##      Column Total |              768 |             1232 |             2000 |
##                   |            0.384 |            0.616 |                  |
## ------------------|------------------|------------------|------------------|
##
##
```

```
PredictedTest_label4 <- knn(train_predictors, validate_predictors,
cl=train_labels, k=1)
CrossTable(x = validate_labels,y = PredictedTest_label4 ,prop.chisq = FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  2000
##
##
##                   | PredictedTest_label4
##   validate_labels | -1.18626953935887 |  0.84269774574752 |         Row Total |
## ------------------|------------------|------------------|------------------|
## -1.18626953935887 |              766 |                4 |              770 |
##                   |            0.995 |            0.005 |            0.385 |
##                   |            0.997 |            0.003 |                  |
```

```
##                   |             0.383 |             0.002 |
|
## ------------------|-------------------|-------------------|---------------
----|
##  0.84269774574752 |                 2 |              1228 |
1230 |
##                   |             0.002 |             0.998 |
0.615 |
##                   |             0.003 |             0.997 |
|
##                   |             0.001 |             0.614 |
|
## ------------------|-------------------|-------------------|---------------
----|
##      Column Total |               768 |              1232 |
2000 |
##                   |             0.384 |             0.616 |
|
## ------------------|-------------------|-------------------|---------------
----|
##
##
```

```
#question4
predict_values_q4 <- data.frame(
  "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,
  "Credit Card" = 1
)
# Set the column names of predict_values_q4 to match train_predictors
colnames(predict_values_q4) <- colnames(train_predictors)
predict_values_q4 <- predict_values_q4[,names(train_predictors)]
knn_prediction_q4 <- knn(train_predictors, predict_values_q4, cl =
train_labels, k = 5)
```

```r
#QUESTION5
#Create Partitioned data sets for training and validation. Use stratified
sampling with personal loan to ensure training and validation training sets
match to avoid under fitting
train_index2 <- createDataPartition(ubank1$Personal.Loan, p=.5, list = FALSE)
train_ubank_Q5 <- ubank1[train_index2,]
intermediate_ubank_Q5 <- ubank1[-train_index2,]

train_index3 <- createDataPartition(intermediate_ubank_Q5$Personal.Loan,
p=.6, list = FALSE)
validate_ubank_Q5 <- intermediate_ubank_Q5[train_index3,]
test_ubank_Q5 <- intermediate_ubank_Q5[-train_index3,]

# Normalize the training data using preProcess
train_ubank_Q5 <- train_ubank_Q5
validate_ubank_Q5 <- validate_ubank_Q5
test_ubank_Q5 <- test_ubank_Q5
norm_values_Q5 <- preProcess(train_ubank_Q5[, c(2:4, 6:7, 11)], method =
c("center", "scale"))
train_ubank_Q5[, c(2:4, 6:7, 11)] <- predict(norm_values_Q5, train_ubank_Q5[,
c(2:4, 6:7, 11)])

# Normalize the test data using the same normalization parameters as the
training data
test_ubank_Q5[, c(2:4, 6:7, 11)] <- predict(norm_values_Q5, test_ubank_Q5[,
c(2:4, 6:7, 11)])

#Create predictors and labels
names(train_ubank_Q5)

##  [1] "Age"                "Experience"         "Income"
##  [4] "Family"             "CCAvg"              "Education.1"
##  [7] "Education.2"        "Education.3"        "Mortgage"
## [10] "Personal.Loan"      "Securities.Account" "CD.Account"
## [13] "Online"             "CreditCard"

train_predictors_Q5 <-train_ubank_Q5[,c(2:4,6:11,13:14)]
validate_predictors_Q5 <- validate_ubank_Q5[,c(2:4,6:11,13:14)]
test_predictors_Q5 <- test_ubank_Q5[,c(2:4,6:11,13:14)]

train_labels_Q5 <-train_ubank_Q5[,12]
validate_labels_Q5 <- validate_ubank_Q5[,12]
test_labels_Q5 <- test_ubank_Q5[,12]
```

```r
#CONFUSION MATRIX FOR TRAINING DATA
predicted_train_labels_Q5 <- knn(train_predictors_Q5, train_predictors_Q5, cl
= train_labels_Q5, k = 5)
conf_matrix0 <- CrossTable(x = train_labels_Q5, y =
predicted_train_labels_Q5, prop.chisq = FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  2500
##
##
##                 | predicted_train_labels_Q5
## train_labels_Q5 |         0 |         1 | Row Total |
## ----------------|-----------|-----------|-----------|
##               0 |      2354 |         2 |      2356 |
##                 |     0.999 |     0.001 |     0.942 |
##                 |     0.956 |     0.053 |           |
##                 |     0.942 |     0.001 |           |
## ----------------|-----------|-----------|-----------|
##               1 |       108 |        36 |       144 |
##                 |     0.750 |     0.250 |     0.058 |
##                 |     0.044 |     0.947 |           |
##                 |     0.043 |     0.014 |           |
## ----------------|-----------|-----------|-----------|
##    Column Total |      2462 |        38 |      2500 |
##                 |     0.985 |     0.015 |           |
## ----------------|-----------|-----------|-----------|
##
##
```

```r
#CALCLATING ACCURACY FOR TRAINING DATA
k1_accuracy0 <- (conf_matrix0$t[2,2] + conf_matrix0$t[1,1])/
sum(conf_matrix0$t)
print(k1_accuracy0)
```

```
## [1] 0.956
```

```r
#CALCULATING RECALL FOR TRAINING DATA
k1_recall0 <- conf_matrix0$t[2,2]/ (conf_matrix0$t[2,2] +
conf_matrix0$t[2,1])
print(k1_recall0)
```

```
## [1] 0.25

#CALCULATING PRECISION FOR TRAINING DATA
k1_precision0 <- conf_matrix0$t[2,2]/ (conf_matrix0$t[2,2] +
conf_matrix0$t[1,2])
print(k1_precision0)

## [1] 0.9473684

#CALCULATING SPECIFICITY FOR TRAINING DATA
k1_specificity0 <- conf_matrix0$t[1,1]/ (conf_matrix0$t[1,1] +
conf_matrix0$t[1,2])
print(k1_specificity0)

## [1] 0.9991511

#CONFUSION MATRIX FOR VALIDATION DATA
predicted_validate_labels_Q5 <-
knn(train_predictors_Q5,validate_predictors_Q5, cl = train_labels_Q5, k = 5)
conf_matrix1 <- CrossTable(x = validate_labels_Q5, y =
predicted_validate_labels_Q5, prop.chisq = FALSE)

##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  1500
##
##
##                    | predicted_validate_labels_Q5
## validate_labels_Q5 |         0 |         1 | Row Total |
## -------------------|-----------|-----------|-----------|
##                  0 |       972 |       422 |      1394 |
##                    |     0.697 |     0.303 |     0.929 |
##                    |     0.919 |     0.955 |           |
##                    |     0.648 |     0.281 |           |
## -------------------|-----------|-----------|-----------|
##                  1 |        86 |        20 |       106 |
##                    |     0.811 |     0.189 |     0.071 |
##                    |     0.081 |     0.045 |           |
##                    |     0.057 |     0.013 |           |
## -------------------|-----------|-----------|-----------|
##       Column Total |      1058 |       442 |      1500 |
##                    |     0.705 |     0.295 |           |
```

```
## -------------------|-----------|-----------|-----------|
##
##

#CALCULATING ACCURACY FOR VALIDATION DATA
k1_accuracy2 <- (conf_matrix1$t[2,2] + conf_matrix1$t[1,1])/
sum(conf_matrix1$t)
print(k1_accuracy2)

## [1] 0.6613333

#CALCULATING RECALL FOR VALIDATION DATA
k1_recall2 <- conf_matrix1$t[2,2]/ (conf_matrix1$t[2,2] +
conf_matrix1$t[2,1])
print(k1_recall2)

## [1] 0.1886792

#CALCULATING PRECISION FOR VALIDATION DATA
k1_precision2 <- conf_matrix1$t[2,2]/ (conf_matrix1$t[2,2] +
conf_matrix1$t[1,2])
print(k1_precision2)

## [1] 0.04524887

#CALCULATING SPECIFICITY FOR VALIDATION DATA
k1_specificity2 <- conf_matrix1$t[1,1]/ (conf_matrix1$t[1,1] +
conf_matrix1$t[1,2])
print(k1_specificity2)

## [1] 0.697274

# Create a new variable for our probability
set.seed(1234)
my_knnprob2 <-knn(train_predictors_Q5,
  validate_predictors_Q5,
  cl = train_labels_Q5, k=1, prob=TRUE )
class_prob2 <-attr(my_knnprob2, 'prob')
# See the first rows
head(class_prob2)

## [1] 1 1 1 1 1 1

#CONFUSION MATRIX FOR TESTING DATA
predicted_test_labels_Q5 <- knn(train_predictors_Q5,test_predictors_Q5, cl =
train_labels_Q5, k = 5)
conf_matrix2 <- CrossTable(x = test_labels_Q5, y = predicted_test_labels_Q5,
prop.chisq = FALSE)

##
##
##    Cell Contents
```

```
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  1000
##
##
##                | predicted_test_labels_Q5
## test_labels_Q5 |         0 |         1 | Row Total |
## ---------------|-----------|-----------|-----------|
##              0 |       943 |         5 |       948 |
##                |     0.995 |     0.005 |     0.948 |
##                |     0.957 |     0.333 |           |
##                |     0.943 |     0.005 |           |
## ---------------|-----------|-----------|-----------|
##              1 |        42 |        10 |        52 |
##                |     0.808 |     0.192 |     0.052 |
##                |     0.043 |     0.667 |           |
##                |     0.042 |     0.010 |           |
## ---------------|-----------|-----------|-----------|
##   Column Total |       985 |        15 |      1000 |
##                |     0.985 |     0.015 |           |
## ---------------|-----------|-----------|-----------|
##
##
```

```
#CALCULATING ACCURACY FOR TESTING DATA
k1_accuracy3 <- (conf_matrix2$t[2,2] + conf_matrix2$t[1,1])/
sum(conf_matrix2$t)
print(k1_accuracy3)
```

```
## [1] 0.953
```

```
#CALCULATING RECALL FOR TESTING DATA
k1_recall3 <- conf_matrix2$t[2,2]/ (conf_matrix2$t[2,2] +
conf_matrix2$t[2,1])
print(k1_recall3)
```

```
## [1] 0.1923077
```

```
#CALCULATING PRECISION FOR TESTING DATA
k1_precision3 <- conf_matrix2$t[2,2]/ (conf_matrix2$t[2,2] +
conf_matrix2$t[1,2])
print(k1_precision3)
```

```
## [1] 0.6666667
```

```r
#CALCULATING SPECIFICITY FOR TESTING DATA
k1_specificity3 <- conf_matrix2$t[1,1]/ (conf_matrix2$t[1,1] +
conf_matrix2$t[1,2])
print(k1_specificity3)

## [1] 0.9947257

#Create a new variable for our probability
set.seed(1234)
my_knnprob3 <-knn(train_predictors_Q5,
                  test_predictors_Q5,
                  cl = train_labels_Q5, k = 1, prob = TRUE)
class_prob3<-attr(my_knnprob3, 'prob')
# See the first rows
head(class_prob3)

## [1] 1 1 1 1 1 1

#In conclusion, the new customer is going to be classify as accepting the
personal loan form the Universal
#Bank from the new marketing campaign.
#When looking at indicators for model performance we can see that most of
these metrics are very close between the test data set and those for
validation and training. This would indicate that we did not under fit our
data. However, the test data set does perform slightly worse than train and
validate data sets. However, given that this difference is so small we can
conclude the model was not over fitted. This means that we can have
confidence in our parameters and hyper parameters and therefore our models
ability to accurately predict a personal loan on unseen sets of data.
```