Assignment 2 K-NN

Jyoti Phogat

2022-10-06

###Project Background:

*Liability customers - Majority - Depositors

*Asset customers - Small - Borrowers

*Campaign of last year - conversion rate of 9.6% [Among the 5000 customers, only 480 (= 9.6%) accepted the personal loan that was offered to them in the earlier campaign.]

*Goal : use k-NN to predict whether a new customer will accept a loan offer.

- Data (rows): 5000 customers

*Success class as 1 (loan acceptance)

####Packages used

```r
install.packages("psych")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.2'
## (as 'lib' is unspecified)
```

```r
library(psych)  #for creating dummies
install.packages("caret")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.2'
## (as 'lib' is unspecified)
```

```r
library(caret)  #for data partition, normalize data
```

```
## Loading required package: ggplot2

##
## Attaching package: 'ggplot2'

## The following objects are masked from 'package:psych':
##
##     %+%, alpha

## Loading required package: lattice
```

```r
install.packages("FNN")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.2'
## (as 'lib' is unspecified)
```

```r
library(FNN)    #for Perfoming knn classification
install.packages("class")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.2'
## (as 'lib' is unspecified)
```

```
library(class)
```

```
##
## Attaching package: 'class'

## The following objects are masked from 'package:FNN':
##
##     knn, knn.cv
```

```
install.packages("dplyr")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.2'
## (as 'lib' is unspecified)
```

```
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
```

####Data Exploration

```
#loading the data in R
originaldata <- read.csv("UniversalBank.csv")
```

```
#Eliminating variables [id & zip code] from the dataset
df=subset(originaldata, select=-c(ID, ZIP.Code ))
```

```
#creating dummies
#library(psych)
dummy_Education <- as.data.frame(dummy.code(df$Education))
names(dummy_Education) <- c("Education_1", "Education_2","Education_3") #renaming dummy variable
df_without_education <- subset(df, select=-c(Education)) #eliminating education variable

UBank_data <- cbind(df_without_education, dummy_Education) #main dataset
```

####Data Partition

```
#Partitioning the data into Traning(60%) and Validation(40%)
#library(caret)
set.seed(2019)
Train_Index     = createDataPartition(UBank_data$Age, p= 0.6 , list=FALSE)
Train_Data      = UBank_data[Train_Index,]  #3001 observations

Validation_Data = UBank_data[-Train_Index,] #1999 observations
```

####Genearting Test Data

```
Test_Data <- data.frame(Age=40 , Experience=10, Income = 84, Family = 2, CCAvg = 2, Education_1 = 0, Edu
```

####Data Normalization

```r
# Copy the original data
train.norm.df    <- Train_Data
valid.norm.df    <- Validation_Data
test.norm.df     <- Test_Data
maindata.norm.df <- UBank_data

head(maindata.norm.df)
```

```
##   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_1 Education_2 Education_3
## 1          0      0          0           1           0           0
## 2          0      0          0           1           0           0
## 3          0      0          0           1           0           0
## 4          0      0          0           0           0           1
## 5          0      0          1           0           0           1
## 6          0      1          0           0           0           1
```

```r
# use preProcess() from the caret package to normalize .
norm.values <- preProcess(Train_Data[,-7], method=c("center", "scale"))

train.norm.df[,-7] <- predict(norm.values, Train_Data[,-7])  #Training Data
valid.norm.df [,-7]<- predict(norm.values, Validation_Data[,-7])#Validation Data
test.norm.df <- predict(norm.values, Test_Data)#Test Data
maindata.norm.df[,-7] <- predict(norm.values,UBank_data[,-7]) #Training + Validation data

head(maindata.norm.df)
```

```
##           Age  Experience     Income     Family      CCAvg   Mortgage
## 1 -1.77433045 -1.66194702 -0.5412194  1.3938471 -0.2037850 -0.5610795
## 2 -0.02864873 -0.09511012 -0.8671501  0.5222207 -0.2609914 -0.5610795
## 3 -0.55235324 -0.44329609 -1.3669105 -1.2210321 -0.5470231 -0.5610795
## 4 -0.90148959 -0.96557506  0.5669449 -1.2210321  0.4254849 -0.5610795
## 5 -0.90148959 -1.05262156 -0.6281343  1.3938471 -0.5470231 -0.5610795
## 6 -0.72692141 -0.61738908 -0.9757937  1.3938471 -0.8902612  0.9377628
##   Personal.Loan Securities.Account CD.Account     Online CreditCard Education_1
## 1             0          2.9352543 -0.2510627 -1.2138946 -0.6471267   1.1964328
## 2             0          2.9352543 -0.2510627 -1.2138946 -0.6471267   1.1964328
## 3             0         -0.3405725 -0.2510627 -1.2138946 -0.6471267   1.1964328
## 4             0         -0.3405725 -0.2510627 -1.2138946 -0.6471267  -0.8355394
## 5             0         -0.3405725 -0.2510627 -1.2138946  1.5447776  -0.8355394
## 6             0         -0.3405725 -0.2510627  0.8235202 -0.6471267  -0.8355394
##   Education_2 Education_3
## 1  -0.6460905  -0.6455725
## 2  -0.6460905  -0.6455725
## 3  -0.6460905  -0.6455725
## 4  -0.6460905   1.5484966
## 5  -0.6460905   1.5484966
## 6  -0.6460905   1.5484966
```

####Perfoming k-NN classification , using k = 1

```r
#library(FNN)
set.seed(2019)
prediction <- knn(train = train.norm.df[,-7], test = valid.norm.df[,-7],
          cl = train.norm.df[,7], k = 1, prob=TRUE)
actual= valid.norm.df$Personal.Loan
prediction_prob = attr(prediction,"prob")



table(prediction,actual)
```

```
##           actual
## prediction    0    1
##          0 1792   58
##          1   23  126
```

```r
mean(prediction==actual)
```

```
## [1] 0.9594797
```

```r
#library(class)
NROW(train.norm.df)
```

```
## [1] 3001
```

```r
sqrt(3001)
```

```
## [1] 54.78138
```

####Generating loop to find best k

```r
#library(caret)
#library(FNN)

accuracy.df <- data.frame(k = seq(1, 60, 1), accuracy = rep(0, 60))

# compute knn for different k on validation.
for(i in 1:60) {
prediction <- knn(train = train.norm.df[,-7], test = valid.norm.df[-7],
          cl = train.norm.df[,7], k = i, prob=TRUE)

accuracy.df[i,2] <- mean(prediction==actual)


}
accuracy.df
```

```
##     k  accuracy
## 1   1 0.9594797
## 2   2 0.9539770
## 3   3 0.9594797
## 4   4 0.9594797
## 5   5 0.9564782
## 6   6 0.9564782
## 7   7 0.9544772
## 8   8 0.9554777
```

```
## 9    9 0.9519760
## 10  10 0.9514757
## 11  11 0.9529765
## 12  12 0.9504752
## 13  13 0.9479740
## 14  14 0.9479740
## 15  15 0.9479740
## 16  16 0.9474737
## 17  17 0.9479740
## 18  18 0.9479740
## 19  19 0.9479740
## 20  20 0.9464732
## 21  21 0.9459730
## 22  22 0.9464732
## 23  23 0.9444722
## 24  24 0.9449725
## 25  25 0.9439720
## 26  26 0.9434717
## 27  27 0.9429715
## 28  28 0.9419710
## 29  29 0.9409705
## 30  30 0.9414707
## 31  31 0.9409705
## 32  32 0.9399700
## 33  33 0.9409705
## 34  34 0.9404702
## 35  35 0.9394697
## 36  36 0.9389695
## 37  37 0.9389695
## 38  38 0.9379690
## 39  39 0.9374687
## 40  40 0.9369685
## 41  41 0.9359680
## 42  42 0.9344672
## 43  43 0.9349675
## 44  44 0.9344672
## 45  45 0.9344672
## 46  46 0.9334667
## 47  47 0.9329665
## 48  48 0.9329665
## 49  49 0.9309655
## 50  50 0.9324662
## 51  51 0.9304652
## 52  52 0.9304652
## 53  53 0.9294647
## 54  54 0.9304652
## 55  55 0.9299650
## 56  56 0.9314657
## 57  57 0.9304652
## 58  58 0.9299650
## 59  59 0.9294647
## 60  60 0.9299650
```

#####Answer 2: The value of k we choose is 3 as it provides the best result [i.e the choice of k that balances between overfitting and ignoring the predictor information]

####Validation data results using best k value [i.e: k = 3]

```
#library(FNN)
set.seed(2019)
prediction <- knn(train = train.norm.df[,-7], test = valid.norm.df[,-7],
          cl = train.norm.df[,7], k = 3, prob=TRUE)
actual= valid.norm.df$Personal.Loan
prediction_prob = attr(prediction,"prob")



#Answer 3: confusion matrix for the best k value =3
table(prediction,actual)
```

```
##           actual
## prediction    0    1
##          0 1808   74
##          1    7  110
```

```
#accuracy of the best k=3
mean(prediction==actual)
```

```
## [1] 0.9594797
```

```

#library(FNN)
prediction_test <- knn(train = maindata.norm.df[,-7], test = Test_Data,
          cl = maindata.norm.df[,7], k = 1, prob=TRUE)


head(prediction_test)
```

Classifying the customer using the best k [perfominng k-NN classification on test data]

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

#####Answer 4: k-NN model predicted that the new customer will accept a loan offer [loan accepted]

####Question 5

```
#Repartitiong the data

#Partitioning the data into Traning(50%) ,Validation(30%), Test(20%)
#library(dplyr)
#library(caret)
set.seed(2019)

Test_Index_1 = createDataPartition(UBank_data$Age, p= 0.2 , list=FALSE) #20% test data
Test_Data_1  = UBank_data [Test_Index_1,]

Rem_DATA = UBank_data[-Test_Index_1,] #80% remaining data [training + validation]

Train_Index_1 = createDataPartition(Rem_DATA$Age, p= 0.5 , list=FALSE)
Train_Data_1 = Rem_DATA[Train_Index_1,] #Training data
```

```r
Validation_Data_1 = Rem_DATA[-Train_Index_1,] #Validation data
```

```r
#Data Normalization


# Copy the original data
train.norm.df_1 <- Train_Data_1
valid.norm.df_1 <- Validation_Data_1
test.norm.df_1 <- Test_Data_1
rem_data.norm.df_1 <- Rem_DATA

# use preProcess() from the caret package to normalize Sales and Age.
norm.values_1 <- preProcess(Train_Data_1[-7], method=c("center", "scale"))

train.norm.df_1[-7] <- predict(norm.values_1, Train_Data_1[-7])  #Training Data
valid.norm.df_1[-7] <- predict(norm.values_1, Validation_Data_1[-7])#Validation Data
test.norm.df_1[-7] <- predict(norm.values_1, test.norm.df_1[-7]) #Test Data
test.norm.df_1[-7] <- predict(norm.values_1, Test_Data_1[-7])
rem_data.norm.df_1[-7] <- predict(norm.values_1,Rem_DATA[-7]) #Training + Validation data

head(test.norm.df_1)
```

```
##           Age Experience     Income     Family      CCAvg   Mortgage
## 16  1.2829377  0.8686599 -1.1472922 -1.1935339 -0.2821052 -0.5513052
## 23 -1.4358939 -1.3293661 -0.2927939 -1.1935339 -0.4484833  1.9632437
## 33  0.6690080  0.6928178 -0.7414055 -0.3258248 -0.7812396  1.3152638
## 36  0.2304868  0.3411336  0.1130929  0.5418844 -0.7257802 -0.5513052
## 41  1.0198250  1.0445019  0.1771802  0.5418844 -0.2266458 -0.5513052
## 43 -1.1727812 -1.1535240  1.2025783  1.4095935 -0.5039427  3.4332877
##    Personal.Loan Securities.Account CD.Account     Online CreditCard
## 16             0         -0.3387769 -0.2635981  0.8342689  1.5381222
## 23             0         -0.3387769 -0.2635981  0.8342689 -0.6498183
## 33             0         -0.3387769 -0.2635981 -1.1980549 -0.6498183
## 36             0         -0.3387769 -0.2635981 -1.1980549 -0.6498183
## 41             0          2.9503192 -0.2635981 -1.1980549 -0.6498183
## 43             1         -0.3387769 -0.2635981  0.8342689 -0.6498183
##    Education_1 Education_2 Education_3
## 16  -0.8825926   1.5605153  -0.6095541
## 23   1.1324590  -0.6404936  -0.6095541
## 33  -0.8825926   1.5605153  -0.6095541
## 36   1.1324590  -0.6404936  -0.6095541
## 41  -0.8825926   1.5605153  -0.6095541
## 43  -0.8825926  -0.6404936   1.6397231
```

```r
#Perfoming k-NN classification on Training Data, k = 3


#library(FNN)
set.seed(2019)
prediction_Q5 <- knn(train = train.norm.df_1[,-7], test = valid.norm.df_1[,-7],
          cl = train.norm.df_1[,7], k = 3, prob=TRUE)
actual= valid.norm.df_1$Personal.Loan
prediction_prob = attr(prediction_Q5,"prob")
```

```r
table(prediction_Q5,actual)  #confusion matrix for the best k value =3
```

```
##              actual
## prediction_Q5    0    1
##             0 1797   89
##             1    8  105
```

```r
mean(prediction_Q5==actual)  #accuracy of the best k=3
```

```
## [1] 0.9514757
```

```r
#Perfoming k-NN classification on Test Data, k = 3


library(FNN)
set.seed(2019)
prediction_Q5 <- knn(train = rem_data.norm.df_1[,-7], test = test.norm.df_1[,-7],
          cl = rem_data.norm.df_1[,7], k = 3, prob=TRUE)
actual= test.norm.df_1$Personal.Loan
prediction_prob = attr(prediction_Q5,"prob")

table(prediction_Q5,actual)  #confusion matrix for the best k value =3
```

```
##              actual
## prediction_Q5   0   1
##             0 910  31
##             1   1  59
```

```r
mean(prediction_Q5==actual)  #accuracy of the best k=3
```

```
## [1] 0.968032
```

#####The model performed better in the test set, as it got enough data to learn from i.e 80% of the data, Whereas when we were working on training data it only learned from 50% of the data.