Assignment 2

Hanish Bhogadi

2/20/2022

```
setwd("C:/Users/Hanish Bhogadi/Documents/64060_hbhogadi/Assignment 2")
```

```
#Calling libraries

library('caret')
```

```
## Loading required package: ggplot2
```

```
## Warning in register(): Can't find generic 'scale_type' in package ggplot2 to
## register S3 method.
```

```
## Loading required package: lattice
```

```
library('ISLR')
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('class')

#Importing Data
Bank <- read.csv("UniversalBank.csv")

#MakingID and ZIP code as NULL
Bank$ID <- NULL
Bank$ZIP.Code <- NULL
summary(Bank)
```

```
##       Age           Experience       Income           Family
##  Min.   :23.00   Min.   :-3.0   Min.   :  8.00   Min.   :1.000
##  1st Qu.:35.00   1st Qu.:10.0   1st Qu.: 39.00   1st Qu.:1.000
##  Median :45.00   Median :20.0   Median : 64.00   Median :2.000
##  Mean   :45.34   Mean   :20.1   Mean   : 73.77   Mean   :2.396
##  3rd Qu.:55.00   3rd Qu.:30.0   3rd Qu.: 98.00   3rd Qu.:3.000
##  Max.   :67.00   Max.   :43.0   Max.   :224.00   Max.   :4.000
##      CCAvg          Education        Mortgage     Personal.Loan
##  Min.   : 0.000   Min.   :1.000   Min.   :  0.0   Min.   :0.000
##  1st Qu.: 0.700   1st Qu.:1.000   1st Qu.:  0.0   1st Qu.:0.000
##  Median : 1.500   Median :2.000   Median :  0.0   Median :0.000
##  Mean   : 1.938   Mean   :1.881   Mean   : 56.5   Mean   :0.096
##  3rd Qu.: 2.500   3rd Qu.:3.000   3rd Qu.:101.0   3rd Qu.:0.000
##  Max.   :10.000   Max.   :3.000   Max.   :635.0   Max.   :1.000
##  Securities.Account   CD.Account         Online         CreditCard
##  Min.   :0.0000     Min.   :0.0000   Min.   :0.0000   Min.   :0.000
##  1st Qu.:0.0000     1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.000
##  Median :0.0000     Median :0.0000   Median :1.0000   Median :0.000
##  Mean   :0.1044     Mean   :0.0604   Mean   :0.5968   Mean   :0.294
##  3rd Qu.:0.0000     3rd Qu.:0.0000   3rd Qu.:1.0000   3rd Qu.:1.000
##  Max.   :1.0000     Max.   :1.0000   Max.   :1.0000   Max.   :1.000
```

```
Bank$Personal.Loan = as.factor(Bank$Personal.Loan)

summary(Bank)
```

```
##       Age           Experience       Income           Family
##  Min.   :23.00   Min.   :-3.0   Min.   :  8.00   Min.   :1.000
##  1st Qu.:35.00   1st Qu.:10.0   1st Qu.: 39.00   1st Qu.:1.000
##  Median :45.00   Median :20.0   Median : 64.00   Median :2.000
##  Mean   :45.34   Mean   :20.1   Mean   : 73.77   Mean   :2.396
##  3rd Qu.:55.00   3rd Qu.:30.0   3rd Qu.: 98.00   3rd Qu.:3.000
##  Max.   :67.00   Max.   :43.0   Max.   :224.00   Max.   :4.000
##      CCAvg          Education        Mortgage     Personal.Loan
##  Min.   : 0.000   Min.   :1.000   Min.   :  0.0   0:4520
##  1st Qu.: 0.700   1st Qu.:1.000   1st Qu.:  0.0   1: 480
##  Median : 1.500   Median :2.000   Median :  0.0
##  Mean   : 1.938   Mean   :1.881   Mean   : 56.5
##  3rd Qu.: 2.500   3rd Qu.:3.000   3rd Qu.:101.0
##  Max.   :10.000   Max.   :3.000   Max.   :635.0
##  Securities.Account   CD.Account         Online         CreditCard
##  Min.   :0.0000     Min.   :0.0000   Min.   :0.0000   Min.   :0.000
##  1st Qu.:0.0000     1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.000
##  Median :0.0000     Median :0.0000   Median :1.0000   Median :0.000
##  Mean   :0.1044     Mean   :0.0604   Mean   :0.5968   Mean   :0.294
##  3rd Qu.:0.0000     3rd Qu.:0.0000   3rd Qu.:1.0000   3rd Qu.:1.000
##  Max.   :1.0000     Max.   :1.0000   Max.   :1.0000   Max.   :1.000
```

```
#Normalization
Model_range_normalized <- preProcess(Bank[,-8],method = "range")
Bank_norm <- predict(Model_range_normalized,Bank)
summary(Bank_norm)
```

```
##       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 :0.2593   Median :0.3333
##  Mean   :0.5077   Mean   :0.5023   Mean   :0.3045   Mean   :0.4655
##  3rd Qu.:0.7273   3rd Qu.:0.7174   3rd Qu.:0.4167   3rd Qu.:0.6667
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##      CCAvg          Education         Mortgage       Personal.Loan
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.00000   0:4520
##  1st Qu.:0.0700   1st Qu.:0.0000   1st Qu.:0.00000   1: 480
##  Median :0.1500   Median :0.5000   Median :0.00000
##  Mean   :0.1938   Mean   :0.4405   Mean   :0.08897
##  3rd Qu.:0.2500   3rd Qu.:1.0000   3rd Qu.:0.15906
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.00000
##  Securities.Account   CD.Account         Online         CreditCard
##  Min.   :0.0000     Min.   :0.0000   Min.   :0.0000   Min.   :0.000
##  1st Qu.:0.0000     1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.000
##  Median :0.0000     Median :0.0000   Median :1.0000   Median :0.000
##  Mean   :0.1044     Mean   :0.0604   Mean   :0.5968   Mean   :0.294
##  3rd Qu.:0.0000     3rd Qu.:0.0000   3rd Qu.:1.0000   3rd Qu.:1.000
##  Max.   :1.0000     Max.   :1.0000   Max.   :1.0000   Max.   :1.000
```

```
View(Bank_norm)

#Data Partition into testing and training sets
Train_index <- createDataPartition(Bank$Personal.Loan, p = 0.6, list = FALSE)
train.df = Bank_norm[Train_index,]
validation.df = Bank_norm[-Train_index,]

#Predict k value
To_Predict = data.frame(Age = 40, Experience = 10, Income = 84, Family = 2, CCAvg = 2, Education = 1, M
print(To_Predict)
```

```
##   Age Experience Income Family CCAvg Education Mortgage Securities.Account
## 1  40         10     84      2     2         1        0                   0
##   CD.Account Online CreditCard
## 1          0      1          1
```

```
Prediction <- knn(train = train.df[,1:7,9:12], test = To_Predict[,1:7,9:12], cl = train.df$Personal.Loa
print(Prediction)
```

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

```
#Question 2 - Finding best value of k
set.seed(123)
Bankcontrol <- trainControl(method = "repeatedcv", number = 3, repeats = 2)
searchGrid = expand.grid(k=1:10)

knn.model = train(Personal.Loan~., data = train.df, method = 'knn', tuneGrid = searchGrid, trControl = 
knn.model
```

```
## k-Nearest Neighbors
```

```
##
## 3000 samples
##   11 predictor
##    2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (3 fold, repeated 2 times)
## Summary of sample sizes: 2000, 2000, 2000, 2000, 2000, 2000, ...
## Resampling results across tuning parameters:
##
##   k   Accuracy   Kappa
##    1  0.9531667  0.6917565
##    2  0.9463333  0.6468065
##    3  0.9521667  0.6627895
##    4  0.9483333  0.6294039
##    5  0.9468333  0.6088762
##    6  0.9445000  0.5884595
##    7  0.9421667  0.5632076
##    8  0.9415000  0.5569300
##    9  0.9408333  0.5465965
##   10  0.9371667  0.5062707
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 1.
```

```
#Question 3 - Confusion matrix from using the best k
predictions <- predict(knn.model, validation.df)
confusionMatrix(predictions, validation.df$Personal.Loan)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1790   50
##          1   18  142
##
##                Accuracy : 0.966
##                  95% CI : (0.9571, 0.9735)
##     No Information Rate : 0.904
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7883
##
##  Mcnemar's Test P-Value : 0.0001704
##
##             Sensitivity : 0.9900
##             Specificity : 0.7396
##          Pos Pred Value : 0.9728
##          Neg Pred Value : 0.8875
##              Prevalence : 0.9040
##          Detection Rate : 0.8950
##    Detection Prevalence : 0.9200
##       Balanced Accuracy : 0.8648
##
```

```
##        'Positive' Class : 0
##
```

```
#Question 4 - Classify the customer using the best k.
To_Predict_norm = data.frame(Age = 40, Experience = 10, Income = 84, family = 2, CCAvg = 2, Education =
To_Predict_norm = predict(Model_range_normalized, To_Predict)
predict(knn.model, To_Predict_norm)
```

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

```
#Question 5
#New split

train_size = 0.5
Train_index = createDataPartition(Bank$Personal.Loan, p = 0.5, list = FALSE)
train.df = Bank_norm[Train_index,]

test_size = 0.2
Test_index = createDataPartition(Bank$Personal.Loan, p = 0.2, list = FALSE)
Test.df = Bank_norm[Test_index,]

valid_size = 0.3
Validation_index = createDataPartition(Bank$Personal.Loan, p = 0.3, list = FALSE)
validation.df = Bank_norm[Validation_index,]


Testknn <- knn(train = train.df[,-8], test = Test.df[,-8], cl = train.df[,8], k =1)
Validationknn <- knn(train = train.df[,-8], test = validation.df[,-8], cl = train.df[,8], k =1)
Trainknn <- knn(train = train.df[,-8], test = train.df[,-8], cl = train.df[,8], k =1)

#Confusion Matrix
confusionMatrix(Testknn, Test.df[,8])
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 900  19
##          1   4  77
##
##                Accuracy : 0.977
##                  95% CI : (0.9657, 0.9854)
##     No Information Rate : 0.904
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8575
##
##  Mcnemar's Test P-Value : 0.003509
##
##             Sensitivity : 0.9956
##             Specificity : 0.8021
##          Pos Pred Value : 0.9793
```

```
##          Neg Pred Value : 0.9506
##              Prevalence : 0.9040
##          Detection Rate : 0.9000
##    Detection Prevalence : 0.9190
##       Balanced Accuracy : 0.8988
##
##        'Positive' Class : 0
##
```

```
confusionMatrix(Trainknn, train.df[,8])
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 2260    0
##          1    0  240
##
##                Accuracy : 1
##                  95% CI : (0.9985, 1)
##     No Information Rate : 0.904
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##
##  Mcnemar's Test P-Value : NA
##
##             Sensitivity : 1.000
##             Specificity : 1.000
##          Pos Pred Value : 1.000
##          Neg Pred Value : 1.000
##              Prevalence : 0.904
##          Detection Rate : 0.904
##    Detection Prevalence : 0.904
##       Balanced Accuracy : 1.000
##
##        'Positive' Class : 0
##
```

```
confusionMatrix(Validationknn, validation.df[,8])
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1350   23
##          1    6  121
##
##                Accuracy : 0.9807
##                  95% CI : (0.9724, 0.987)
##     No Information Rate : 0.904
##     P-Value [Acc > NIR] : < 2.2e-16
##
```

```
##                   Kappa : 0.8824
##
##  Mcnemar's Test P-Value : 0.002967
##
##             Sensitivity : 0.9956
##             Specificity : 0.8403
##          Pos Pred Value : 0.9832
##          Neg Pred Value : 0.9528
##              Prevalence : 0.9040
##          Detection Rate : 0.9000
##    Detection Prevalence : 0.9153
##       Balanced Accuracy : 0.9179
##
##        'Positive' Class : 0
##
```

```
#From the above, comparing confusion matrix of the test set with that of the training and validation se
```