Machine Learning Assignment 2

10-02-2022

####Activating Packages

```
library(psych)
library(caret)
```

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

```
##
## Attaching package: 'ggplot2'
```

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

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

```
library(FNN)
library(class)
```

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

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

```
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
```

###importing data

```
getwd()
```

```
## [1] "/Users/bharathreddy/Desktop"
```

```
setwd("/Users/bharathreddy/Desktop")
bharat <- read.csv("UniversalBank.csv")
```

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

```
#creating dummies

dummy_Education <- as.data.frame(dummy.code(df$Education))
names(dummy_Education) <- c("Education_1", "Education_2","Education_3")
df_without_education <- subset(df, select=-c(Education))

UBank_data <- cbind(df_without_education, dummy_Education)
```

###Data partition

```
set.seed(1234)
Train_Index     = createDataPartition(UBank_data$Personal.Loan, p= 0.6 , list=FALSE)
Train_Data      = UBank_data[Train_Index,]

Validation_Data = UBank_data[-Train_Index,]
```

###Generating test data

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

###Data Normalization

```
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
```

```
# 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 -1.76517597 -1.66184535 -0.5337999  1.3900576 -0.1858029 -0.5622826
## 2 -0.03138168 -0.09955276 -0.8590893  0.5257731 -0.2428548 -0.5622826
## 3 -0.55151996 -0.44672889 -1.3578664 -1.2027959 -0.5281144 -0.5622826
## 4 -0.89827882 -0.96749308  0.5721841 -1.2027959  0.4417682 -0.5622826
## 5 -0.89827882 -1.05428712 -0.6205437  1.3900576 -0.5281144 -0.5622826
## 6 -0.72489939 -0.62031695 -0.9675191  1.3900576 -0.8704259  0.9676351
##   Personal.Loan Securities.Account CD.Account     Online CreditCard Education_1
## 1             0          2.8635153 -0.2570526 -1.1911682 -0.6364096   1.1789719
## 2             0          2.8635153 -0.2570526 -1.1911682 -0.6364096   1.1789719
## 3             0         -0.3491047 -0.2570526 -1.1911682 -0.6364096   1.1789719
## 4             0         -0.3491047 -0.2570526 -1.1911682 -0.6364096  -0.8479139
## 5             0         -0.3491047 -0.2570526 -1.1911682  1.5707913  -0.8479139
## 6             0         -0.3491047 -0.2570526  0.8392322 -0.6364096  -0.8479139
##   Education_2 Education_3
## 1  -0.6477981  -0.6327928
## 2  -0.6477981  -0.6327928
## 3  -0.6477981  -0.6327928
## 4  -0.6477981   1.5797694
## 5  -0.6477981   1.5797694
## 6  -0.6477981   1.5797694
```

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

```
set.seed(1234)
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 1785   61
##          1   19  135
```

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

```
## [1] 0.96
```

```
accuracy.df <- data.frame(k = seq(1, 30, 1), accuracy = rep(0, 30))


for(i in 1:30) {
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   0.9600
## 2   2   0.9520
## 3   3   0.9590
## 4   4   0.9540
## 5   5   0.9540
## 6   6   0.9550
## 7   7   0.9515
## 8   8   0.9525
## 9   9   0.9495
## 10 10   0.9490
## 11 11   0.9485
## 12 12   0.9485
## 13 13   0.9465
## 14 14   0.9465
## 15 15   0.9460
## 16 16   0.9440
## 17 17   0.9435
## 18 18   0.9445
## 19 19   0.9430
## 20 20   0.9430
## 21 21   0.9430
## 22 22   0.9420
## 23 23   0.9420
## 24 24   0.9420
## 25 25   0.9420
## 26 26   0.9405
## 27 27   0.9425
## 28 28   0.9425
## 29 29   0.9400
## 30 30   0.9390
```

The value of k we choose is 1 as it is given in the question.

####Validation data results using best k value [i.e: k = 1]

```
set.seed(1234)
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 1785   61
##          1   19  135
```

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

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

k-NN model predicted that the new customer will accept a loan offer.

```
#Partitioning the data into Traning(50%) ,Validation(30%), Test(20%)
set.seed(1234)

Test_Index_1 = createDataPartition(UBank_data$Age, p= 0.2 , list=FALSE)
Test_Data_1  = UBank_data [Test_Index_1,]

Rem_DATA = UBank_data[-Test_Index_1,]
```

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

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

```
#Data Normalization

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

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])

head(test.norm.df_1)
```

```
##             Age   Experience     Income     Family        CCAvg   Mortgage
## 9  -0.90840439 -0.883582836  0.1435652  0.5333142 -0.780693325  0.4495336
## 28  0.05751618 -0.008054857  1.8189997 -1.2081200  0.234699617 -0.5532869
## 32 -0.46934959 -0.358266049 -0.9878972 -1.2081200  0.009056741 -0.5532869
## 40 -0.64497151 -0.620924443  0.1218063  1.4040313 -0.724282606  2.1948269
## 42 -0.99621536 -0.971135634 -0.3133715  0.5333142  0.178288898 -0.5532869
## 63 -0.29372767 -0.183160453 -1.1402094 -1.2081200 -0.555050449 -0.5532869
##    Personal.Loan Securities.Account CD.Account     Online CreditCard
## 9              0         -0.3360202 -0.2646808  0.8429167 -0.6350646
## 28             0         -0.3360202 -0.2646808  0.8429167  1.5738557
## 32             0         -0.3360202 -0.2646808  0.8429167 -0.6350646
## 40             0         -0.3360202 -0.2646808  0.8429167 -0.6350646
## 42             0         -0.3360202 -0.2646808 -1.1857637 -0.6350646
## 63             0         -0.3360202 -0.2646808 -1.1857637 -0.6350646
##    Education_1 Education_2 Education_3
## 9    -0.827392  -0.6607293    1.566207
## 28    1.208013  -0.6607293   -0.638166
## 32   -0.827392  -0.6607293    1.566207
## 40   -0.827392   1.5127224   -0.638166
## 42    1.208013  -0.6607293   -0.638166
## 63    1.208013  -0.6607293   -0.638166
```

```
#Perfoming k-NN classification on Training Data, k = 1
set.seed(1234)
prediction_Q5 <- knn(train = train.norm.df_1[,-7], test = valid.norm.df_1[,-7],
          cl = train.norm.df_1[,7], k = 1, prob=TRUE)
actual= valid.norm.df_1$Personal.Loan
prediction_prob = attr(prediction_Q5,"prob")

table(prediction_Q5,actual)  #confusion matrix for the best k value =1
```

```
##                actual
## prediction_Q5    0    1
##             0 1795   69
##             1   16  119
```

```
mean(prediction_Q5==actual)  #accuracy of the best k=1
```

```
## [1] 0.9574787
```

```
set.seed(1234)
prediction_Q5 <- knn(train = rem_data.norm.df_1[,-7], test = test.norm.df_1[,-7],
          cl = rem_data.norm.df_1[,7], k = 1, 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 =1
```

```
##              actual
## prediction_Q5   0   1
##             0 907  25
##             1  12  57
```