FML-ASSIGNMENT-2

LOKESH JETANGI

2023-02-20

#Importing the required packages   
library('caret')

## Loading required package: ggplot2

## 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 the data and original data name = Uni\_bank  
Uni\_bank <- read.csv("C:/Users/jetan/OneDrive/Desktop/UniversalBank.csv")  
head(Uni\_bank)

## 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

Uni\_bank$ID <- NULL  
Uni\_bank$ZIP.Code <- NULL  
summary(Uni\_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

set.seed(123)

#Creating a new Data set by removing the "ID" and "ZIP Code" columns and normalizing  
Uni\_bank$Personal.Loan = as.factor(Uni\_bank$Personal.Loan)  
Normalized\_model <- preProcess(Uni\_bank[, -8],method = c("center", "scale"))  
Bank\_normalized <- predict(Normalized\_model,Uni\_bank)  
summary(Bank\_normalized)

## Age Experience Income Family   
## Min. :-1.94871 Min. :-2.014710 Min. :-1.4288 Min. :-1.2167   
## 1st Qu.:-0.90188 1st Qu.:-0.881116 1st Qu.:-0.7554 1st Qu.:-1.2167   
## Median :-0.02952 Median :-0.009121 Median :-0.2123 Median :-0.3454   
## Mean : 0.00000 Mean : 0.000000 Mean : 0.0000 Mean : 0.0000   
## 3rd Qu.: 0.84284 3rd Qu.: 0.862874 3rd Qu.: 0.5263 3rd Qu.: 0.5259   
## Max. : 1.88967 Max. : 1.996468 Max. : 3.2634 Max. : 1.3973   
## CCAvg Education Mortgage Personal.Loan  
## Min. :-1.1089 Min. :-1.0490 Min. :-0.5555 0:4520   
## 1st Qu.:-0.7083 1st Qu.:-1.0490 1st Qu.:-0.5555 1: 480   
## Median :-0.2506 Median : 0.1417 Median :-0.5555   
## Mean : 0.0000 Mean : 0.0000 Mean : 0.0000   
## 3rd Qu.: 0.3216 3rd Qu.: 1.3324 3rd Qu.: 0.4375   
## Max. : 4.6131 Max. : 1.3324 Max. : 5.6875   
## Securities.Account CD.Account Online CreditCard   
## Min. :-0.3414 Min. :-0.2535 Min. :-1.2165 Min. :-0.6452   
## 1st Qu.:-0.3414 1st Qu.:-0.2535 1st Qu.:-1.2165 1st Qu.:-0.6452   
## Median :-0.3414 Median :-0.2535 Median : 0.8219 Median :-0.6452   
## Mean : 0.0000 Mean : 0.0000 Mean : 0.0000 Mean : 0.0000   
## 3rd Qu.:-0.3414 3rd Qu.:-0.2535 3rd Qu.: 0.8219 3rd Qu.: 1.5495   
## Max. : 2.9286 Max. : 3.9438 Max. : 0.8219 Max. : 1.5495

head(Uni\_bank)

## Age Experience Income Family CCAvg Education Mortgage Personal.Loan  
## 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 2 0 0  
## 5 35 8 45 4 1.0 2 0 0  
## 6 37 13 29 4 0.4 2 155 0  
## Securities.Account CD.Account Online CreditCard  
## 1 1 0 0 0  
## 2 1 0 0 0  
## 3 0 0 0 0  
## 4 0 0 0 0  
## 5 0 0 0 1  
## 6 0 0 1 0

#Creating Dummy Variables for Education  
Uni\_bank$high\_school <- ifelse(Uni\_bank$Education == "1", 1, 0)  
Uni\_bank$under\_graduation <- ifelse(Uni\_bank$Education == "2", 1, 0)  
Uni\_bank$graduation <- ifelse(Uni\_bank$Education == "3", 1, 0)  
str(Uni\_bank)

## 'data.frame': 5000 obs. of 15 variables:  
## $ 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 ...  
## $ 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 : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...  
## $ 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 ...  
## $ high\_school : num 1 1 1 0 0 0 0 0 0 0 ...  
## $ under\_graduation : num 0 0 0 1 1 1 1 0 1 0 ...  
## $ graduation : num 0 0 0 0 0 0 0 1 0 1 ...

# Remove original education variable from the data.  
Uni\_bank= subset(Uni\_bank, select = -c(Education) )  
str(Uni\_bank)

## 'data.frame': 5000 obs. of 14 variables:  
## $ 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 ...  
## $ 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 ...  
## $ Mortgage : int 0 0 0 0 0 155 0 0 104 0 ...  
## $ Personal.Loan : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...  
## $ 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 ...  
## $ high\_school : num 1 1 1 0 0 0 0 0 0 0 ...  
## $ under\_graduation : num 0 0 0 1 1 1 1 0 1 0 ...  
## $ graduation : num 0 0 0 0 0 0 0 1 0 1 ...

#question 1  
#partitioning the data into 60% for training and 40% for validating  
Train\_index <- createDataPartition(Uni\_bank$Personal.Loan, p = 0.6, list = FALSE)  
train.df = Bank\_normalized[Train\_index,]  
validation.df = Bank\_normalized[-Train\_index,]  
#Prediction   
ub.predict\_Norm = data.frame(Age = 40, Experience = 10, Income = 84, Family = 2, CCAvg = 2, high\_school = 0, under\_graduation = 1, graduation = 0, Mortgage = 0, Securities.Account =0, CD.Account = 0, Online = 1, CreditCard = 1)  
  
dim(train.df)

## [1] 3000 12

dim(ub.predict\_Norm)

## [1] 1 13

print(ub.predict\_Norm)

## Age Experience Income Family CCAvg high\_school under\_graduation graduation  
## 1 40 10 84 2 2 0 1 0  
## Mortgage Securities.Account CD.Account Online CreditCard  
## 1 0 0 0 1 1

prediction <- class::knn(train=as.data.frame(train.df),test=as.data.frame(ub.predict\_Norm[,-c(7)]),cl=train.df$Personal.Loan, k=1)  
  
print(prediction)

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

#question 2  
#picking the value of k that gives largest accuracy  
set.seed(123)  
Bankcontrol <- trainControl(method= "repeatedcv", number = 3, repeats = 2)  
searchGrid = expand.grid(k=1:12)  
knn.model = train(Personal.Loan~., data = train.df, method = 'knn', tuneGrid = searchGrid,trControl = Bankcontrol)  
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 0.9508333 0.6882579  
## 2 0.9481667 0.6666850  
## 3 0.9535000 0.6759337  
## 4 0.9511667 0.6580416  
## 5 0.9518333 0.6584044  
## 6 0.9518333 0.6585355  
## 7 0.9495000 0.6310891  
## 8 0.9485000 0.6254851  
## 9 0.9458333 0.6009763  
## 10 0.9450000 0.5942968  
## 11 0.9433333 0.5800019  
## 12 0.9410000 0.5584309  
##   
## Accuracy was used to select the optimal model using the largest value.  
## The final value used for the model was k = 3.

#Question 3  
#Confusion matrix for the Validation data  
predictions <- predict(knn.model,validation.df)  
confusionMatrix(predictions,validation.df$Personal.Loan)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 1800 65  
## 1 8 127  
##   
## Accuracy : 0.9635   
## 95% CI : (0.9543, 0.9713)  
## No Information Rate : 0.904   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.7575   
##   
## Mcnemar's Test P-Value : 5.59e-11   
##   
## Sensitivity : 0.9956   
## Specificity : 0.6615   
## Pos Pred Value : 0.9651   
## Neg Pred Value : 0.9407   
## Prevalence : 0.9040   
## Detection Rate : 0.9000   
## Detection Prevalence : 0.9325   
## Balanced Accuracy : 0.8285   
##   
## 'Positive' Class : 0   
##

#Question 4  
# creating data frame for prediction and using the best k value   
  
ub.predict\_Norm <- data.frame(Age = 40, Experience = 10, Income = 84, Family = 2, CCAvg = 2, high\_school = 0, under\_graduation = 1, graduation = 0, Mortgage = 0, Securities.Account =0, CD.Account = 0, Online = 1, CreditCard = 1)  
#Customers are classified using a trained model with the highest k value.  
numericVariables<-c("Age","Experience", "Income","Family","CCAvg","Mortgage")  
  
normValues<-preProcess(train.df[,numericVariables],  
 method=c("center","scale"))  
  
train\_norm<-predict(normValues,train.df)  
val\_norm<-predict(normValues,validation.df)  
  
dim(train\_norm)

## [1] 3000 12

dim(ub.predict\_Norm)

## [1] 1 13

prediction <- class::knn(train=as.data.frame(train\_norm),test=as.data.frame(ub.predict\_Norm[,-c(7)]),cl=train.df$Personal.Loan, k=3)  
print(prediction)

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

#Question 5  
#Splitting the data into 50% for training ,30% for validation, 20% for test  
train\_size = 0.5  
Train\_index = createDataPartition(Uni\_bank$Personal.Loan, p = 0.5, list = FALSE)  
train.df = Bank\_normalized[Train\_index,]  
test\_size = 0.2  
Test\_index = createDataPartition(Uni\_bank$Personal.Loan, p = 0.2, list = FALSE)  
Test.df = Bank\_normalized[Test\_index,]  
valid\_size = 0.3  
Validation\_index = createDataPartition(Uni\_bank$Personal.Loan, p = 0.3, list = FALSE)  
validation.df = Bank\_normalized[Validation\_index,]  
Testknn <- knn(train = train.df[,-8], test = Test.df[,-8], cl = train.df[,8], k =3)  
Validationknn <- knn(train = train.df[,-8], test = validation.df[,-8], cl = train.df[,8], k =3)  
Trainknn <- knn(train = train.df[,-8], test = train.df[,-8], cl = train.df[,8], k =3)  
confusionMatrix(Testknn, Test.df[,8])

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 900 32  
## 1 4 64  
##   
## Accuracy : 0.964   
## 95% CI : (0.9505, 0.9747)  
## No Information Rate : 0.904   
## P-Value [Acc > NIR] : 2.787e-13   
##   
## Kappa : 0.7615   
##   
## Mcnemar's Test P-Value : 6.795e-06   
##   
## Sensitivity : 0.9956   
## Specificity : 0.6667   
## Pos Pred Value : 0.9657   
## Neg Pred Value : 0.9412   
## Prevalence : 0.9040   
## Detection Rate : 0.9000   
## Detection Prevalence : 0.9320   
## Balanced Accuracy : 0.8311   
##   
## 'Positive' Class : 0   
##

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

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 2257 48  
## 1 3 192  
##   
## Accuracy : 0.9796   
## 95% CI : (0.9733, 0.9848)  
## No Information Rate : 0.904   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.8717   
##   
## Mcnemar's Test P-Value : 7.218e-10   
##   
## Sensitivity : 0.9987   
## Specificity : 0.8000   
## Pos Pred Value : 0.9792   
## Neg Pred Value : 0.9846   
## Prevalence : 0.9040   
## Detection Rate : 0.9028   
## Detection Prevalence : 0.9220   
## Balanced Accuracy : 0.8993   
##   
## 'Positive' Class : 0   
##

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

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 1352 43  
## 1 4 101  
##   
## Accuracy : 0.9687   
## 95% CI : (0.9585, 0.9769)  
## No Information Rate : 0.904   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.7946   
##   
## Mcnemar's Test P-Value : 2.976e-08   
##   
## Sensitivity : 0.9971   
## Specificity : 0.7014   
## Pos Pred Value : 0.9692   
## Neg Pred Value : 0.9619   
## Prevalence : 0.9040   
## Detection Rate : 0.9013   
## Detection Prevalence : 0.9300   
## Balanced Accuracy : 0.8492   
##   
## 'Positive' Class : 0   
##

#It is clear from the data above that the training accuracy is a little bit greater than the accuracy of the test and validation sets, indicating that the algorithm is performing as it should.