FML\_Assignment\_2

2023-10-02

#solutions  
  
# Load the required libraries first  
  
library(readr)  
library(caret)

## Loading required package: ggplot2

## Loading required package: lattice

library(psych)

##   
## Attaching package: 'psych'

## The following objects are masked from 'package:ggplot2':  
##   
## %+%, alpha

# And installing required package: ggplot2,lattice  
  
library(class)  
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(FNN)

##   
## Attaching package: 'FNN'

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

#Read the dataset  
  
mydata <- read.csv("C:/Users/navat/Downloads/UniversalBank.csv")  
  
# Eliminating variables [id & zip code] from the dataset  
  
dataframe<-mydata[, -c(1,5)]  
View(dataframe)  
  
# Creating dummy variable for education  
  
dummy\_Edu <- as.data.frame(dummy.code(dataframe$Education))  
  
# Then rename the variable names of dummmy  
  
names(dummy\_Edu) <- c("Education\_1", "Education\_2","Education\_3")  
  
# Eliminating education column from the dataset  
  
dataframe\_no\_edu <- dataframe[,-c(6)]   
  
Ub\_df <- cbind(dataframe\_no\_edu, dummy\_Edu)  
  
names(Ub\_df)[8] ="Securities.Account"  
names(Ub\_df)[9] ="CD.Account"  
names(Ub\_df)[7] ="Personal.Loan"  
View(Ub\_df)  
  
# Partitioning the data to 60% Training and 40% Validation  
  
set.seed(1)  
  
training\_ind <- sample(row.names(Ub\_df),0.6\*dim(Ub\_df)[1])   
testing\_ind <- setdiff(row.names(Ub\_df), training\_ind)   
training\_df <- Ub\_df[training\_ind, ]  
validation\_df <- Ub\_df[testing\_ind, ]  
  
# Solution 1)  
  
# Creating the new customer data  
new\_cust\_df = data.frame(Age=40 , Experience=10, Income = 84, Family = 2, CCAvg = 2, Mortgage = 0, Securities.Account = 0, CD.Account = 0, Online = 1, CreditCard = 1, Education\_1 = 0, Education\_2 = 1, Education\_3 = 0)  
  
# Normalising the data  
  
norm <- preProcess(training\_df[, -c(7)], method=c("center", "scale"))  
training\_df[, -c(7)] <- predict(norm, training\_df[, -c(7)])  
validation\_df[, -c(7)] <- predict(norm, validation\_df[, -c(7)])  
new\_cust\_df <- predict(norm, new\_cust\_df)  
  
# Perform the classification for Knn  
pred\_cl <- knn(train = training\_df[,-c(7)],test = new\_cust\_df, cl = training\_df[,7], k=1, prob=TRUE)  
knn.attributes <- attributes(pred\_cl)  
  
#We will be getting the probability value as 1  
  
knn.attributes[3]

## $prob  
## [1] 1

actual= validation\_df$Personal.Loan  
  
prediction\_prob = attr(pred\_cl,"prob")  
mean(pred\_cl==actual)

## [1] 0.8975

# Solution 2)  
  
accuracy.dataframe <- data.frame(k = seq(1, 60, 1), accuracy = rep(0, 60))  
for(i in 1:60) {  
 prediction <- knn(train = training\_df[,-7], test = validation\_df[-7],cl = training\_df[,7], k = i, prob=TRUE)   
 accuracy.dataframe[i,2] <- mean(pred\_cl==actual)}  
  
View(accuracy.dataframe)  
  
# Solution 3)  
  
set.seed(123)  
pred\_cl <- knn(train = training\_df[,-7], test = validation\_df[,-7],cl = training\_df[,7], k = 3, prob=TRUE)   
confusionMatrix(pred\_cl, as.factor(validation\_df[,7]))

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 1786 63  
## 1 9 142  
##   
## Accuracy : 0.964   
## 95% CI : (0.9549, 0.9717)  
## No Information Rate : 0.8975   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.7785   
##   
## Mcnemar's Test P-Value : 4.208e-10   
##   
## Sensitivity : 0.9950   
## Specificity : 0.6927   
## Pos Pred Value : 0.9659   
## Neg Pred Value : 0.9404   
## Prevalence : 0.8975   
## Detection Rate : 0.8930   
## Detection Prevalence : 0.9245   
## Balanced Accuracy : 0.8438   
##   
## 'Positive' Class : 0   
##

# Solution 4)  
  
new\_customer\_df= 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)  
  
pred\_new <- knn(train = training\_df[,-7],test = new\_customer\_df, cl = training\_df[,7], k=3, prob=TRUE)  
pred\_new

## [1] 1  
## attr(,"prob")  
## [1] 1  
## attr(,"nn.index")  
## [,1] [,2] [,3]  
## [1,] 2721 939 2146  
## attr(,"nn.dist")  
## [,1] [,2] [,3]  
## [1,] 90.49831 90.53126 90.53372  
## Levels: 1

# Solution 5)  
  
set.seed(1)  
training\_ind <- sample(rownames(Ub\_df), 0.5\*dim(Ub\_df)[1])  
  
set.seed(1)  
validation\_ind <- sample(setdiff(rownames(Ub\_df),training\_ind), 0.3\*dim(Ub\_df)[1])  
testing\_ind = setdiff(rownames(Ub\_df), union(training\_ind, validation\_ind))  
  
training\_df <- Ub\_df[training\_ind, ]  
validation\_df <- Ub\_df[validation\_ind, ]  
testing\_df <- Ub\_df[testing\_ind, ]  
  
#Normalizing data  
  
norm.values <- preProcess(training\_df[, -c(7)], method=c("center", "scale"))  
training\_df[, -c(7)] <- predict(norm.values, training\_df[, -c(7)])  
validation\_df[, -c(7)] <- predict(norm.values, validation\_df[, -c(7)])  
testing\_df[,-c(7)] <- predict(norm.values, testing\_df[,-c(7)])  
  
# Perform kNN on training, testing, validation data  
  
testknn <- knn(train = training\_df[,-c(7)],test = testing\_df[,-c(7)], cl = training\_df[,7], k=3, prob=TRUE)  
  
validknn <- knn(train = training\_df[,-c(7)],test = validation\_df[,-c(7)], cl = training\_df[,7], k=3, prob=TRUE)  
  
trainknn <- knn(train = training\_df[,-c(7)],test = training\_df[,-c(7)], cl = training\_df[,7], k=3, prob=TRUE)  
  
# Calculating confusion matrix  
  
confusionMatrix(testknn, as.factor(testing\_df[,7]))

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 889 35  
## 1 3 73  
##   
## Accuracy : 0.962   
## 95% CI : (0.9482, 0.973)  
## No Information Rate : 0.892   
## P-Value [Acc > NIR] : 4.592e-16   
##   
## Kappa : 0.7732   
##   
## Mcnemar's Test P-Value : 4.934e-07   
##   
## Sensitivity : 0.9966   
## Specificity : 0.6759   
## Pos Pred Value : 0.9621   
## Neg Pred Value : 0.9605   
## Prevalence : 0.8920   
## Detection Rate : 0.8890   
## Detection Prevalence : 0.9240   
## Balanced Accuracy : 0.8363   
##   
## 'Positive' Class : 0   
##

# Confusion Matrix and Statistics  
  
confusionMatrix(validknn, as.factor(validation\_df[,7]))

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 1353 42  
## 1 7 98  
##   
## Accuracy : 0.9673   
## 95% CI : (0.957, 0.9757)  
## No Information Rate : 0.9067   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.7826   
##   
## Mcnemar's Test P-Value : 1.191e-06   
##   
## Sensitivity : 0.9949   
## Specificity : 0.7000   
## Pos Pred Value : 0.9699   
## Neg Pred Value : 0.9333   
## Prevalence : 0.9067   
## Detection Rate : 0.9020   
## Detection Prevalence : 0.9300   
## Balanced Accuracy : 0.8474   
##   
## 'Positive' Class : 0   
##

confusionMatrix(trainknn, as.factor(training\_df[,7]))

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 2263 54  
## 1 5 178  
##   
## Accuracy : 0.9764   
## 95% CI : (0.9697, 0.982)  
## No Information Rate : 0.9072   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.8452   
##   
## Mcnemar's Test P-Value : 4.129e-10   
##   
## Sensitivity : 0.9978   
## Specificity : 0.7672   
## Pos Pred Value : 0.9767   
## Neg Pred Value : 0.9727   
## Prevalence : 0.9072   
## Detection Rate : 0.9052   
## Detection Prevalence : 0.9268   
## Balanced Accuracy : 0.8825   
##   
## 'Positive' Class : 0   
##