FML\_Assignment2

Pavan Kumar Mekala

2024-02-25

## 1A: The new\_customer1 will be classified as 0 and have not taken the personal loan

## 2A: The best choice of K will be 3 which gives overall efficiency of 0

## 3A: If we use the value of K as 3 and set.seed(123) the confusion matrix will be

Reference

Prediction 0 1 0 1776 51 1 22 151 True positive = 151 True negative = 1776 False positive = 22 False negative = 51

## 4A: By using the value of K = 3, the customer will be classified as 0 which makes the customer to not take any personal loan

## 5A:

Training data: Accuracy = 97.36% Sensitivity = 74.90% Specificity = 99.87%

Validation data: Accuracy = 95.8% Sensitivity = 67.15% Specificity = 98.67%

For Testing data: Accuracy = 95.8% Sensitivity = 67.39% Specificity = 98.68%

## The library functions class and caret was loaded by using library() function

library(class)  
library(caret)

## Loading required package: ggplot2

## Loading required package: lattice

library(e1071)

##Importing the dataset UniversalBank(1).csv file into Rstudio

UniversalBank <- read.csv("UniversalBank (1).csv")  
head(UniversalBank)

## 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

dim(UniversalBank)

## [1] 5000 14

##Transposing the dataframe of UniversalBank by using function “t”

t(t(names(UniversalBank)))

## [,1]   
## [1,] "ID"   
## [2,] "Age"   
## [3,] "Experience"   
## [4,] "Income"   
## [5,] "ZIP.Code"   
## [6,] "Family"   
## [7,] "CCAvg"   
## [8,] "Education"   
## [9,] "Mortgage"   
## [10,] "Personal.Loan"   
## [11,] "Securities.Account"  
## [12,] "CD.Account"   
## [13,] "Online"   
## [14,] "CreditCard"

## Removing ID nad ZIP

UniversalBank <- UniversalBank[,-c(1,5)]  
head(UniversalBank)

## 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

dim(UniversalBank)

## [1] 5000 12

## Coverting Education as factor

UniversalBank$Education <- as.factor(UniversalBank$Education)

## Converting Education levels to dummy variables and creating dummy variables for factors

Dummy\_Education\_levels <- dummyVars(~., data = UniversalBank)  
Universal\_Bank <- as.data.frame(predict(Dummy\_Education\_levels,UniversalBank))

## Partition the data into training (60%) and validation (40%) sets and adding set.seed(123) for reproducibility and checked them by using dim function

set.seed(123)  
Number\_of\_rows <- nrow(Universal\_Bank)  
Training\_Universal\_Bank\_Index <- sample(row.names(Universal\_Bank), 0.6 \* Number\_of\_rows)  
Training\_Universal\_Bank <- Universal\_Bank[Training\_Universal\_Bank\_Index,]  
Validation\_Universal\_Bank\_Index <- setdiff(row.names(Universal\_Bank), Training\_Universal\_Bank\_Index)  
Validation\_Universal\_Bank <- Universal\_Bank[Validation\_Universal\_Bank\_Index,]  
dim(Training\_Universal\_Bank)

## [1] 3000 14

dim(Validation\_Universal\_Bank)

## [1] 2000 14

## PreProcessing the data by using preProcess() function and Normalizing the Training and Validation data predict() function

Trainining\_Norm\_Universal\_Bank <- Training\_Universal\_Bank [,-10]  
Validation\_Norm\_Universal\_Bank <- Validation\_Universal\_Bank [,-10]  
Normalization\_Values <- preProcess(Training\_Universal\_Bank[,-10], method=c("center","scale"))  
Trainining\_Norm\_Universal\_Bank <- predict(Normalization\_Values, Training\_Universal\_Bank[,-10])  
Validation\_Norm\_Universal\_Bank <- predict(Normalization\_Values, Validation\_Universal\_Bank[,-10])

## Creating the dataset and normalizing it by using predict() function

New\_customer1 <- 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  
)  
New\_customer1\_Norm <- New\_customer1  
New\_customer1\_Norm <- predict(Normalization\_Values, New\_customer1\_Norm)

## Performing knn classification as k=1 and predicting it by using class::knn() function

knn\_Prediction\_1 <- class::knn(train = Trainining\_Norm\_Universal\_Bank,  
 test = New\_customer1\_Norm,  
 cl = Training\_Universal\_Bank$Personal.Loan, k=1)  
knn\_Prediction\_1

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

## Calculating the accuracy of each value of k and checked it by using print(paste) function

Accuracy\_bank <- data.frame(k= seq(1, 20, 1), overallaccuracy = rep(0,20))  
for (i in 1:20){  
 knn\_prediction <- class::knn (train = Trainining\_Norm\_Universal\_Bank,  
 test = Validation\_Norm\_Universal\_Bank,  
 cl = Training\_Universal\_Bank$Personal.Loan, k=i)  
   
Accuracy\_bank[i, 2] <- confusionMatrix(knn\_prediction, as.factor(Validation\_Universal\_Bank$Personal.Loan),  
 positive = "1")$overall[1]  
}  
Best\_Value\_K <- which(Accuracy\_bank[,2] == max(Accuracy\_bank[,2]))  
Accuracy\_bank

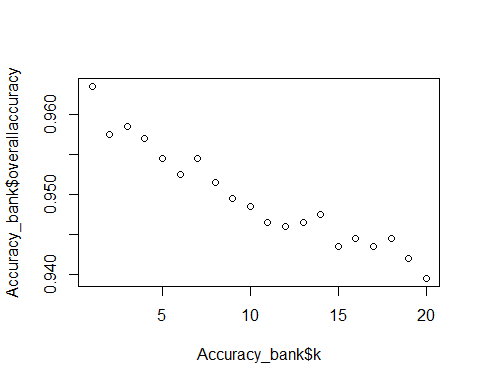
## k overallaccuracy  
## 1 1 0.9635  
## 2 2 0.9575  
## 3 3 0.9585  
## 4 4 0.9570  
## 5 5 0.9545  
## 6 6 0.9525  
## 7 7 0.9545  
## 8 8 0.9515  
## 9 9 0.9495  
## 10 10 0.9485  
## 11 11 0.9465  
## 12 12 0.9460  
## 13 13 0.9465  
## 14 14 0.9475  
## 15 15 0.9435  
## 16 16 0.9445  
## 17 17 0.9435  
## 18 18 0.9445  
## 19 19 0.9420  
## 20 20 0.9395

print(paste("Best value of k =", Best\_Value\_K))

## [1] "Best value of k = 1"

## Plotting the graph between the values of k accuracy

plot(Accuracy\_bank$k, Accuracy\_bank$overallaccuracy)



## Creating the confusion matrix of validation data for the best value of k and checked it by using print() function

knn\_Predicion\_2 <- class::knn(train = Trainining\_Norm\_Universal\_Bank,   
 test = Validation\_Norm\_Universal\_Bank,   
 cl = Training\_Universal\_Bank$Personal.Loan,k = Best\_Value\_K)  
Confusion\_matrix\_Data <- confusionMatrix(knn\_Predicion\_2, as.factor(Validation\_Universal\_Bank$Personal.Loan), positive = "1")  
print(Confusion\_matrix\_Data)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 1776 51  
## 1 22 151  
##   
## Accuracy : 0.9635   
## 95% CI : (0.9543, 0.9713)  
## No Information Rate : 0.899   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.7853   
##   
## Mcnemar's Test P-Value : 0.001049   
##   
## Sensitivity : 0.7475   
## Specificity : 0.9878   
## Pos Pred Value : 0.8728   
## Neg Pred Value : 0.9721   
## Prevalence : 0.1010   
## Detection Rate : 0.0755   
## Detection Prevalence : 0.0865   
## Balanced Accuracy : 0.8676   
##   
## 'Positive' Class : 1   
##

## Creating new dataset and normalizing the data

New\_customer2 <- 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  
)  
New\_Customer2\_Norm <- New\_customer2  
New\_Customer2\_Norm <- predict(Normalization\_Values, New\_Customer2\_Norm)

## Predicting the data by using knn Algorithm

knn\_Prediction\_3 <- class::knn(train = Trainining\_Norm\_Universal\_Bank,   
 test = New\_Customer2\_Norm,   
 cl = Training\_Universal\_Bank$Personal.Loan, k = Best\_Value\_K)  
knn\_Prediction\_3

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

## Splitting data again for training, validation and testing and setting the seed value as 134 and the dimensions by using dim() functions

set.seed(134)  
Training\_Universal\_Bank\_Index2 <- sample(row.names(Universal\_Bank), 0.5 \* Number\_of\_rows)  
Validation\_Universal\_Bank\_Index\_2 <- sample(setdiff(row.names(Universal\_Bank),Training\_Universal\_Bank\_Index2),  
 0.3 \* Number\_of\_rows)  
Testing\_Universal\_Bank\_Index <- setdiff(row.names(Universal\_Bank), c(Training\_Universal\_Bank\_Index2,Validation\_Universal\_Bank\_Index\_2))  
  
Training\_Universal\_Bank\_2 <- Universal\_Bank[Training\_Universal\_Bank\_Index2,]  
Validation\_Universal\_Bank2 <- Universal\_Bank[Validation\_Universal\_Bank\_Index\_2,]  
Testing\_Universal\_Bank <- Universal\_Bank[Testing\_Universal\_Bank\_Index,]  
dim(Training\_Universal\_Bank\_2)

## [1] 2500 14

dim(Validation\_Universal\_Bank2)

## [1] 1500 14

dim(Testing\_Universal\_Bank)

## [1] 1000 14

## Normalizing the data for Training, Validation and Testing by using predict() function and preprocessing it by using preProcess() function and checked it by using print() function

set.seed(134)  
Normalization\_Values2 <- preProcess(Training\_Universal\_Bank\_2[ ,-10], method = c("center", "scale"))  
Trainining\_Norm\_Universal\_Bank\_2 <- predict(Normalization\_Values2, Training\_Universal\_Bank\_2[ ,-10])  
Validation\_Norm\_Universal\_Bank\_2 <- predict(Normalization\_Values2, Validation\_Universal\_Bank2[ ,-10])  
Testing\_Norm\_Universal\_Bank\_2 <- predict(Normalization\_Values2, Testing\_Universal\_Bank[ ,-10])

## Confusion matrix for training data at k=3

knn\_Prediction\_Training <- class::knn(train = Trainining\_Norm\_Universal\_Bank\_2,   
 test = Trainining\_Norm\_Universal\_Bank\_2,   
 cl = Training\_Universal\_Bank\_2$Personal.Loan, k = 3)  
Confusion\_matrix\_Train <- confusionMatrix(knn\_Prediction\_Training, as.factor(Training\_Universal\_Bank\_2$Personal.Loan), positive = "1")  
print(Confusion\_matrix\_Train)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 2246 63  
## 1 3 188  
##   
## Accuracy : 0.9736   
## 95% CI : (0.9665, 0.9795)  
## No Information Rate : 0.8996   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.8365   
##   
## Mcnemar's Test P-Value : 3.803e-13   
##   
## Sensitivity : 0.7490   
## Specificity : 0.9987   
## Pos Pred Value : 0.9843   
## Neg Pred Value : 0.9727   
## Prevalence : 0.1004   
## Detection Rate : 0.0752   
## Detection Prevalence : 0.0764   
## Balanced Accuracy : 0.8738   
##   
## 'Positive' Class : 1   
##

##Confusion matrix for validation data at k=3 and chekced it by using print() function

knn\_Prediction\_Validation\_2 <- class::knn(train =Trainining\_Norm\_Universal\_Bank\_2,  
 test = Validation\_Norm\_Universal\_Bank\_2,  
 cl = Training\_Universal\_Bank\_2$Personal.Loan, k = Best\_Value\_K)  
Confusion\_matrix\_Validation\_2 <- confusionMatrix(knn\_Prediction\_Validation\_2, as.factor(Validation\_Universal\_Bank2$Personal.Loan), positive = "1")  
print(Confusion\_matrix\_Validation\_2)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 1345 45  
## 1 18 92  
##   
## Accuracy : 0.958   
## 95% CI : (0.9466, 0.9676)  
## No Information Rate : 0.9087   
## P-Value [Acc > NIR] : 1.639e-13   
##   
## Kappa : 0.7224   
##   
## Mcnemar's Test P-Value : 0.001054   
##   
## Sensitivity : 0.67153   
## Specificity : 0.98679   
## Pos Pred Value : 0.83636   
## Neg Pred Value : 0.96763   
## Prevalence : 0.09133   
## Detection Rate : 0.06133   
## Detection Prevalence : 0.07333   
## Balanced Accuracy : 0.82916   
##   
## 'Positive' Class : 1   
##

## Confusion matrix of testing data at k=3 and checked it by using print() function

knn\_Prediction\_Testing <- class::knn(train = Trainining\_Norm\_Universal\_Bank\_2,  
 test = Testing\_Norm\_Universal\_Bank\_2,  
 cl = Training\_Universal\_Bank\_2$Personal.Loan, k = Best\_Value\_K)  
Confusion\_matrix\_testing <- confusionMatrix(knn\_Prediction\_Testing, as.factor(Testing\_Universal\_Bank$Personal.Loan), positive = "1")  
print(Confusion\_matrix\_testing)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 896 30  
## 1 12 62  
##   
## Accuracy : 0.958   
## 95% CI : (0.9436, 0.9696)  
## No Information Rate : 0.908   
## P-Value [Acc > NIR] : 1.093e-09   
##   
## Kappa : 0.7244   
##   
## Mcnemar's Test P-Value : 0.008712   
##   
## Sensitivity : 0.6739   
## Specificity : 0.9868   
## Pos Pred Value : 0.8378   
## Neg Pred Value : 0.9676   
## Prevalence : 0.0920   
## Detection Rate : 0.0620   
## Detection Prevalence : 0.0740   
## Balanced Accuracy : 0.8303   
##   
## 'Positive' Class : 1   
##