Assignment\_2

Hemani Panchmatiya

2025-09-22

#Introduction: This dataset is to predict the number of new customers who has accepted personal loan offered by the UnverisalBank. Following are the steps taken using K-NN

#Load the packages

library(class)  
library(caret)

## Loading required package: ggplot2

## Loading required package: lattice

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(ggplot2)

#Import dataset

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

#Remove predictors ID and Zip code

UniversalBank <- subset(UniversalBank, select = -c(ID, ZIP.Code))

#Transforming Categorical predictors to dummmies

UniversalBank <- fastDummies::dummy\_cols(UniversalBank, select\_columns = "Education", remove\_first\_dummy = FALSE, remove\_selected\_columns = TRUE)

#seperate predictors (x) and target (Y- personal.loan) & Normalize the predictors

num\_vars <- UniversalBank %>% select(-Personal.Loan)  
preproc <- preProcess(num\_vars, method = c("center", "scale"))  
bank\_norm <- predict(preproc, num\_vars)  
head(bank\_norm)

## Age Experience Income Family CCAvg Mortgage  
## 1 -1.77423939 -1.66591186 -0.5381750 1.3972742 -0.1933661 -0.5554684  
## 2 -0.02952064 -0.09632058 -0.8640230 0.5259383 -0.2505855 -0.5554684  
## 3 -0.55293627 -0.44511864 -1.3636566 -1.2167334 -0.5366825 -0.5554684  
## 4 -0.90188002 -0.96831574 0.5697084 -1.2167334 0.4360473 -0.5554684  
## 5 -0.90188002 -1.05551525 -0.6250678 1.3972742 -0.5366825 -0.5554684  
## 6 -0.72740814 -0.61951767 -0.9726390 1.3972742 -0.8799989 0.9684153  
## Securities.Account CD.Account Online CreditCard Education\_1 Education\_2  
## 1 2.9286223 -0.2535149 -1.2164961 -0.6452498 1.1769533 -0.6244752  
## 2 2.9286223 -0.2535149 -1.2164961 -0.6452498 1.1769533 -0.6244752  
## 3 -0.3413892 -0.2535149 -1.2164961 -0.6452498 1.1769533 -0.6244752  
## 4 -0.3413892 -0.2535149 -1.2164961 -0.6452498 -0.8494814 1.6010244  
## 5 -0.3413892 -0.2535149 -1.2164961 1.5494774 -0.8494814 1.6010244  
## 6 -0.3413892 -0.2535149 0.8218687 -0.6452498 -0.8494814 1.6010244  
## Education\_3  
## 1 -0.6548999  
## 2 -0.6548999  
## 3 -0.6548999  
## 4 -0.6548999  
## 5 -0.6548999  
## 6 -0.6548999

#CBind the file

bank\_file <- cbind(bank\_norm, Personal.Loan = UniversalBank$Personal.Loan)  
head(bank\_file)

## Age Experience Income Family CCAvg Mortgage  
## 1 -1.77423939 -1.66591186 -0.5381750 1.3972742 -0.1933661 -0.5554684  
## 2 -0.02952064 -0.09632058 -0.8640230 0.5259383 -0.2505855 -0.5554684  
## 3 -0.55293627 -0.44511864 -1.3636566 -1.2167334 -0.5366825 -0.5554684  
## 4 -0.90188002 -0.96831574 0.5697084 -1.2167334 0.4360473 -0.5554684  
## 5 -0.90188002 -1.05551525 -0.6250678 1.3972742 -0.5366825 -0.5554684  
## 6 -0.72740814 -0.61951767 -0.9726390 1.3972742 -0.8799989 0.9684153  
## Securities.Account CD.Account Online CreditCard Education\_1 Education\_2  
## 1 2.9286223 -0.2535149 -1.2164961 -0.6452498 1.1769533 -0.6244752  
## 2 2.9286223 -0.2535149 -1.2164961 -0.6452498 1.1769533 -0.6244752  
## 3 -0.3413892 -0.2535149 -1.2164961 -0.6452498 1.1769533 -0.6244752  
## 4 -0.3413892 -0.2535149 -1.2164961 -0.6452498 -0.8494814 1.6010244  
## 5 -0.3413892 -0.2535149 -1.2164961 1.5494774 -0.8494814 1.6010244  
## 6 -0.3413892 -0.2535149 0.8218687 -0.6452498 -0.8494814 1.6010244  
## Education\_3 Personal.Loan  
## 1 -0.6548999 0  
## 2 -0.6548999 0  
## 3 -0.6548999 0  
## 4 -0.6548999 0  
## 5 -0.6548999 0  
## 6 -0.6548999 0

#Split the data into train and Validation

set.seed(456)  
Index\_Train <- createDataPartition(bank\_file$Personal.Loan, p=0.6, list = FALSE)  
Train <- bank\_file[Index\_Train, ]  
Valid <- bank\_file[-Index\_Train, ]

#Create new customer profile

new\_customer <- 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)

#Normalize new customer data

new\_customer\_norm <- predict(preproc, new\_customer)  
head(new\_customer\_norm)

## Age Experience Income Family CCAvg Mortgage  
## 1 -0.4657003 -0.8811162 0.2221371 -0.3453975 0.0355115 -0.5554684  
## Securities.Account CD.Account Online CreditCard Education\_1 Education\_2  
## 1 -0.3413892 -0.2535149 0.8218687 1.549477 -0.8494814 1.601024  
## Education\_3  
## 1 -0.6548999

#KNN when K=1

customer\_pred <- knn(train = Train %>% select(-Personal.Loan),  
 test = new\_customer\_norm,  
 cl = Train$Personal.Loan,  
 k = 1)

#Prediction

cat("Prediction for new customer:", customer\_pred , "\n")

## Prediction for new customer: 1

#Define the k values to test

k\_values <- seq(1, 15, 2)   
accuracy <- c()  
for(i in k\_values){  
 pred\_valid <- knn(  
 train = Train %>% select(-Personal.Loan),  
 test = Valid %>% select(-Personal.Loan),  
 cl = Train$Personal.Loan,  
 k = i  
 )  
 conf\_matrix <- confusionMatrix(pred\_valid, factor(Valid$Personal.Loan))  
 accuracy <- c(accuracy, conf\_matrix$overall["Accuracy"])  
}  
best\_k <- k\_values[which.max(accuracy)]  
cat("Best k:", best\_k, "\n")

## Best k: 3

cat("Validation accuracy for best k:", max(accuracy), "\n")

## Validation accuracy for best k: 0.9565

#Plot the best of K

plot(k\_values, accuracy, type = "b", col = "blue", Xlab = "k\_values", ylab= "accuracy")

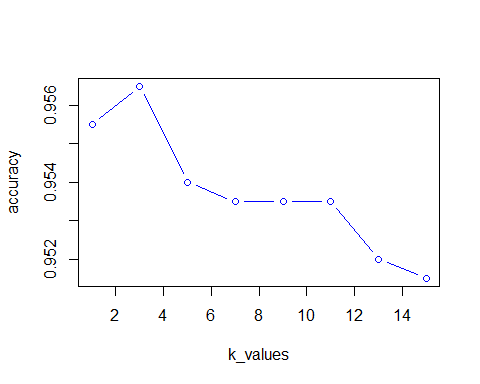
## Warning in plot.window(...): "Xlab" is not a graphical parameter

## Warning in plot.xy(xy, type, ...): "Xlab" is not a graphical parameter

## Warning in axis(side = side, at = at, labels = labels, ...): "Xlab" is not a  
## graphical parameter  
## Warning in axis(side = side, at = at, labels = labels, ...): "Xlab" is not a  
## graphical parameter

## Warning in box(...): "Xlab" is not a graphical parameter

## Warning in title(...): "Xlab" is not a graphical parameter



#Confusion matrix of validation set using best of k

pred\_valid <- knn(  
 train = Train %>% select(-Personal.Loan),  
 test = Valid %>% select(-Personal.Loan),  
 cl = Train$Personal.Loan,  
 k = 3  
 )  
 conf\_matrix <- confusionMatrix(pred\_valid, factor(Valid$Personal.Loan))  
print(conf\_matrix)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 1798 76  
## 1 11 115  
##   
## Accuracy : 0.9565   
## 95% CI : (0.9466, 0.965)  
## No Information Rate : 0.9045   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.703   
##   
## Mcnemar's Test P-Value : 6.813e-12   
##   
## Sensitivity : 0.9939   
## Specificity : 0.6021   
## Pos Pred Value : 0.9594   
## Neg Pred Value : 0.9127   
## Prevalence : 0.9045   
## Detection Rate : 0.8990   
## Detection Prevalence : 0.9370   
## Balanced Accuracy : 0.7980   
##   
## 'Positive' Class : 0   
##

#Using best of K = 3 for new customer dataset

customer\_pred <- knn(train = Train %>% select(-Personal.Loan),  
 test = new\_customer\_norm,  
 cl = Train$Personal.Loan,  
 k = 3)

#Prediction

cat("Prediction for new customer:", customer\_pred , "\n")

## Prediction for new customer: 1

#Spliting the data into Train (50%)

set.seed(456)  
Index\_Train <- createDataPartition(bank\_file$Personal.Loan, p=0.5, list = FALSE)  
Train <- bank\_file[Index\_Train, ]

#Spliting the data into Validation (30%) and test(20%)

Remaining <- bank\_file[-Index\_Train, ]  
Index\_valid <- createDataPartition(Remaining$Personal.Loan, p = 0.6, list = FALSE)   
Test <- Remaining[-Index\_valid, ]  
Valid <- Remaining[Index\_valid, ]

#Prediction and confusion matrix using K = 3 on validation set

pred\_valid <- knn(  
 train = Train %>% select(-Personal.Loan),  
 test = Valid %>% select(-Personal.Loan),  
 cl = Train$Personal.Loan,  
 k = 3  
 )  
 conf\_matrix <- confusionMatrix(pred\_valid, factor(Valid$Personal.Loan))  
print(conf\_matrix)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 1342 57  
## 1 9 92  
##   
## Accuracy : 0.956   
## 95% CI : (0.9444, 0.9658)  
## No Information Rate : 0.9007   
## P-Value [Acc > NIR] : 1.380e-15   
##   
## Kappa : 0.713   
##   
## Mcnemar's Test P-Value : 7.238e-09   
##   
## Sensitivity : 0.9933   
## Specificity : 0.6174   
## Pos Pred Value : 0.9593   
## Neg Pred Value : 0.9109   
## Prevalence : 0.9007   
## Detection Rate : 0.8947   
## Detection Prevalence : 0.9327   
## Balanced Accuracy : 0.8054   
##   
## 'Positive' Class : 0   
##

#Prediction and confusion matrix using K = 3 on training set

pred\_valid <- knn(  
 train = Train %>% select(-Personal.Loan),  
 test = Train %>% select(-Personal.Loan),  
 cl = Train$Personal.Loan,  
 k = 3  
 )  
 conf\_matrix <- confusionMatrix(pred\_valid, factor(Train$Personal.Loan))  
print(conf\_matrix)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 2253 50  
## 1 6 191  
##   
## Accuracy : 0.9776   
## 95% CI : (0.971, 0.983)  
## No Information Rate : 0.9036   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.86   
##   
## Mcnemar's Test P-Value : 9.132e-09   
##   
## Sensitivity : 0.9973   
## Specificity : 0.7925   
## Pos Pred Value : 0.9783   
## Neg Pred Value : 0.9695   
## Prevalence : 0.9036   
## Detection Rate : 0.9012   
## Detection Prevalence : 0.9212   
## Balanced Accuracy : 0.8949   
##   
## 'Positive' Class : 0   
##

#Prediction and confusion matrix using K = 3 on test set

pred\_valid <- knn(  
 train = Train %>% select(-Personal.Loan),  
 test = Test %>% select(-Personal.Loan),  
 cl = Train$Personal.Loan,  
 k = 3  
 )  
 conf\_matrix <- confusionMatrix(pred\_valid, factor(Test$Personal.Loan))  
print(conf\_matrix)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 904 35  
## 1 6 55  
##   
## Accuracy : 0.959   
## 95% CI : (0.9448, 0.9704)  
## No Information Rate : 0.91   
## P-Value [Acc > NIR] : 1.580e-09   
##   
## Kappa : 0.7072   
##   
## Mcnemar's Test P-Value : 1.226e-05   
##   
## Sensitivity : 0.9934   
## Specificity : 0.6111   
## Pos Pred Value : 0.9627   
## Neg Pred Value : 0.9016   
## Prevalence : 0.9100   
## Detection Rate : 0.9040   
## Detection Prevalence : 0.9390   
## Balanced Accuracy : 0.8023   
##   
## 'Positive' Class : 0   
##

##Summary: The accuracy is higher for training set (97.8%) while compared to validation (95.6%) and test (95.9%), likely because the model fits well on training data. Since the accuracy for validation and test is very similar, it indicates that the model generalizes well and is not overfitting. Also, sensitivity is higher for all three ( training, validation & test ) than of specificity, meaning that the classifier performs well in identifying the class with no loans and weaker in identifying class with loan customers.