ML2

Gloria Stephen

2022-10-02

library(readr)  
library(caret)

## Loading required package: ggplot2

## Loading required package: lattice

UniversalBank <- read\_csv("C:/Users/idast/Downloads/UniversalBank.csv")

## Rows: 5000 Columns: 14

## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## dbl (14): ID, Age, Experience, Income, ZIP.Code, Family, CCAvg, Education, M...  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

UniversalBank$Personal.Loan =as.factor(UniversalBank$Personal.Loan)  
data1<-UniversalBank  
  
summary(UniversalBank)

## ID Age Experience Income ZIP.Code   
## Min. : 1 Min. :23.00 Min. :-3.0 Min. : 8.00 Min. : 9307   
## 1st Qu.:1251 1st Qu.:35.00 1st Qu.:10.0 1st Qu.: 39.00 1st Qu.:91911   
## Median :2500 Median :45.00 Median :20.0 Median : 64.00 Median :93437   
## Mean :2500 Mean :45.34 Mean :20.1 Mean : 73.77 Mean :93153   
## 3rd Qu.:3750 3rd Qu.:55.00 3rd Qu.:30.0 3rd Qu.: 98.00 3rd Qu.:94608   
## Max. :5000 Max. :67.00 Max. :43.0 Max. :224.00 Max. :96651   
## Family CCAvg Education Mortgage Personal.Loan  
## Min. :1.000 Min. : 0.000 Min. :1.000 Min. : 0.0 0:4520   
## 1st Qu.:1.000 1st Qu.: 0.700 1st Qu.:1.000 1st Qu.: 0.0 1: 480   
## Median :2.000 Median : 1.500 Median :2.000 Median : 0.0   
## Mean :2.396 Mean : 1.938 Mean :1.881 Mean : 56.5   
## 3rd Qu.:3.000 3rd Qu.: 2.500 3rd Qu.:3.000 3rd Qu.:101.0   
## Max. :4.000 Max. :10.000 Max. :3.000 Max. :635.0   
## 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

View(data1)

##removing unwanted data from dataset

data2<-data1[,-c(1,5)]  
View(data2)

#converting personal loan to factors

#data2$Personal.Loan =as.factor(data2$Personal.Loan)  
  
View(data2)

#creating the dummy variables

Education\_1 <- ifelse(data2$Education == 1, 1,0)  
Education\_2 <- ifelse(data2$Education == 2, 1,0)  
Education\_3 <- ifelse(data2$Education == 3, 1,0)

data3<-data.frame(Age=data2$Age,Experience=data2$Experience,Income=data2$Income,Family=data2$Family,CCAvg=data2$CCAvg, Education\_1=Education\_1,Education\_2=Education\_2,Education\_3=Education\_3,Personal.Loan=data2$Personal.Loan,Mortgage=data2$Mortgage,Securities.Account=data2$Securities.Account,CD.Account=data2$CD.Account,Online=data2$Online,CreditCard=data2$CreditCard)  
  
View(data3)

##creating a partition of 60:40

Train\_Index = createDataPartition(data3$Personal.Loan,p=0.6, list = FALSE)  
Train.df =data3[Train\_Index,]  
Validation.df=data3[-Train\_Index,]  
  
nrow(Train.df)

## [1] 3000

nrow(Validation.df)

## [1] 2000

#normalization of the data

Norm\_model <- preProcess(Train.df[,-(6:9)], method = c("center", "scale"))  
  
training\_norm<-predict(Norm\_model,Train.df)  
  
validation\_norm<-predict(Norm\_model,Validation.df)

#test data set

Test<-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)  
  
  
View(Test)  
  
test\_norm<-predict(Norm\_model,Test)

#knn algorithm in dataset

library(class)  
  
pred\_train<-training\_norm[,-9]  
label\_train<-training\_norm[,9]  
  
pred\_valid<-validation\_norm[,-9]  
label\_valid<-validation\_norm[,9]  
  
  
  
View(training\_norm)  
  
Model<-knn(pred\_train,test\_norm,cl=label\_train,k=1)  
  
Model

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

#since value of k=0, the customer will not accept the loan offer

#Finding the best value for k

set.seed(541)  
  
searchGrid <- expand.grid(k=seq(1:30))  
  
predicted <- train(Personal.Loan~.,training\_norm,method="knn",tuneGrid=searchGrid)  
  
predicted

## k-Nearest Neighbors   
##   
## 3000 samples  
## 13 predictor  
## 2 classes: '0', '1'   
##   
## No pre-processing  
## Resampling: Bootstrapped (25 reps)   
## Summary of sample sizes: 3000, 3000, 3000, 3000, 3000, 3000, ...   
## Resampling results across tuning parameters:  
##   
## k Accuracy Kappa   
## 1 0.9528646 0.6971321  
## 2 0.9468186 0.6567318  
## 3 0.9456644 0.6410080  
## 4 0.9455045 0.6330429  
## 5 0.9455891 0.6271350  
## 6 0.9466297 0.6283100  
## 7 0.9454040 0.6134105  
## 8 0.9446813 0.6054016  
## 9 0.9432419 0.5901438  
## 10 0.9425477 0.5817488  
## 11 0.9423726 0.5789005  
## 12 0.9413446 0.5669753  
## 13 0.9409838 0.5633790  
## 14 0.9405163 0.5562279  
## 15 0.9399798 0.5503329  
## 16 0.9397219 0.5461707  
## 17 0.9393179 0.5413481  
## 18 0.9385959 0.5335072  
## 19 0.9383829 0.5309314  
## 20 0.9376146 0.5229023  
## 21 0.9376866 0.5234405  
## 22 0.9373976 0.5195726  
## 23 0.9365388 0.5100577  
## 24 0.9362835 0.5057487  
## 25 0.9354130 0.4953579  
## 26 0.9355978 0.4974728  
## 27 0.9350530 0.4913619  
## 28 0.9346187 0.4864327  
## 29 0.9338252 0.4771988  
## 30 0.9334998 0.4729072  
##   
## Accuracy was used to select the optimal model using the largest value.  
## The final value used for the model was k = 1.

best\_k <- predicted$bestTune[[1]]  
  
#this saves the best value for k

#the confusion matrix

model<-predict(predicted,validation\_norm[,-9])  
  
confusionMatrix(model,label\_valid)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 0 1  
## 0 1782 62  
## 1 26 130  
##   
## Accuracy : 0.956   
## 95% CI : (0.9461, 0.9646)  
## No Information Rate : 0.904   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.7233   
##   
## Mcnemar's Test P-Value : 0.0001907   
##   
## Sensitivity : 0.9856   
## Specificity : 0.6771   
## Pos Pred Value : 0.9664   
## Neg Pred Value : 0.8333   
## Prevalence : 0.9040   
## Detection Rate : 0.8910   
## Detection Prevalence : 0.9220   
## Balanced Accuracy : 0.8314   
##   
## 'Positive' Class : 0   
##

#knn for new customer

Prediction\_new<-knn(pred\_train,test\_norm,cl=label\_train,k=best\_k)  
  
Prediction\_new

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

#k=0, new customer will not accept the loan offer

#5#splitting data to 50:30:20 ratio

set.seed(887)

part1<- createDataPartition(data3$Personal.Loan,p=.5,list=FALSE,times=1)

part2<- createDataPartition(data3$Personal.Loan,p=.3,list=FALSE,times=1)

part3<- createDataPartition(data3$Personal.Loan,p=.2,list=FALSE,times=1)

new\_train <- data3[part1, ]

new\_valid <- data3[part2, ]

new\_test <- data3[part3, ]

#normalization of the data

normalized <- preProcess(Train.df[,-(6:9)], method=c(“center”,“scale”))

#training data

normalized\_train <- predict(normalized,new\_train)

#the validation data

normalized\_valid<-predict(normalized,new\_valid)

#the test data created

normalized\_test<-predict(normalized,new\_test)

#running knn for train,validation and test data

pred\_train1=normalized\_train[,-9]

label\_train1=normalized\_train[,9]

pred\_valid1=normalized\_valid[,-9]

label\_valid1=normalized\_valid[,9]

pred\_test1=normalized\_test[,-9]

label\_test1=normalized\_test[,9]

training\_prediction<-knn(pred\_train1,pred\_train1,cl=label\_train1,k=best\_k)

validation\_prediction<-knn(pred\_train1,pred\_valid1,cl=label\_train1,k=best\_k)

test\_prediction<-knn(pred\_train1,pred\_test1,cl=label\_train1,k=best\_k)

#confusion matrix of training data

confusionMatrix(pred\_train1,label\_train1)

#confusion matrix of validation data

confusionMatrix(pred\_valid1,label\_valid1)

#confusion matrix of test data

confusionMatrix(pred\_test1,label\_test1)