ASSIGNMENT_2_FML - 811287455

Sudarshan rayapati

2023-09-30

Problem Statement

Universal bank is a young bank growing rapidly in terms of overall customer acquisition. The majority of these customers are liability customers (depositors) with varying sizes of relationship with the bank. The customer base of asset customers (borrowers) is quite small, and the bank is interested in expanding this base rapidly in more loan business. In particular, it wants to explore ways of converting its liability customers to personal loan customers.

A campaign that the bank ran last year for liability customers showed a healthy conversion rate of over 9% success. This has encouraged the retail marketing department to devise smarter campaigns with better target marketing. The goal is to use k-NN to predict whether a new customer will accept a loan offer. This will serve as the basis for the design of a new campaign.

The file UniversalBank.csv contains data on 5000 customers. The data include customer demographic information (age, income, etc.), the customer's relationship with the bank (mortgage, securities account, etc.), and the customer response to the last personal loan campaign (Personal Loan). Among these 5000 customers, only 480 (= 9.6%) accepted the personal loan that was offered to them in the earlier campaign.

Partition the data into training (60%) and validation (40%) sets

---

installing the pacakges "class","caret","e1071"

calling the libraries "class","caret","e1071"

```
library(class)
library(caret)
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

```
library(e1071)
```

Reading the bank csv file

```r
x<-read.csv("/Users/sudarshan/Desktop/FML/dataset/UniversalBank.csv")
dim(x)
```

```
## [1] 5000   14
```

```r
t(t(names(x))) #transpose of the dataframe
```

```
##       [,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"
```

droping the "id" and "zip" attributes for the dataset

```r
new_x <-x[,-c(1,5)]
dim(new_x)
```

```
## [1] 5000   12
```

converting education attribute from int to char

```r
new_x$Education <- as.factor(new_x$Education)
```

creating the dummy variables for the "education" attribute

```r
dumy <- dummyVars(~.,data=new_x)
the_neww <- as.data.frame(predict(dumy,new_x))
```

Partitioning the data into training (60%) and validation (40%) set and setting the seed as we need to re-run the code.

```r
set.seed(1)
train.df <- sample(row.names(the_neww), 0.6*dim(the_neww)[1])
valid.df <- setdiff(row.names(the_neww),train.df)
t.df <- the_neww[train.df,]
v.df<- the_neww[valid.df,]
t(t(names(t.df)))
```

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

```r
summary(t.df)
```

```
##       Age          Experience        Income           Family
##  Min.   :23.00   Min.   :-3.00   Min.   :  8.00   Min.   :1.000
##  1st Qu.:36.00   1st Qu.:10.00   1st Qu.: 39.00   1st Qu.:1.000
##  Median :45.00   Median :20.00   Median : 63.00   Median :2.000
##  Mean   :45.43   Mean   :20.19   Mean   : 73.08   Mean   :2.388
##  3rd Qu.:55.00   3rd Qu.:30.00   3rd Qu.: 98.00   3rd Qu.:3.000
##  Max.   :67.00   Max.   :43.00   Max.   :224.00   Max.   :4.000
##      CCAvg         Education.1      Education.2     Education.3
##  Min.   : 0.000   Min.   :0.0000   Min.   :0.000   Min.   :0.0000
##  1st Qu.: 0.700   1st Qu.:0.0000   1st Qu.:0.000   1st Qu.:0.0000
##  Median : 1.500   Median :0.0000   Median :0.000   Median :0.0000
##  Mean   : 1.915   Mean   :0.4173   Mean   :0.285   Mean   :0.2977
##  3rd Qu.: 2.500   3rd Qu.:1.0000   3rd Qu.:1.000   3rd Qu.:1.0000
##  Max.   :10.000   Max.   :1.0000   Max.   :1.000   Max.   :1.0000
##     Mortgage      Personal.Loan     Securities.Account   CD.Account
##  Min.   :  0.00   Min.   :0.00000   Min.   :0.0000     Min.   :0.00000
##  1st Qu.:  0.00   1st Qu.:0.00000   1st Qu.:0.0000     1st Qu.:0.00000
##  Median :  0.00   Median :0.00000   Median :0.0000     Median :0.00000
##  Mean   : 57.34   Mean   :0.09167   Mean   :0.1003     Mean   :0.05367
##  3rd Qu.:102.00   3rd Qu.:0.00000   3rd Qu.:0.0000     3rd Qu.:0.00000
##  Max.   :635.00   Max.   :1.00000   Max.   :1.0000     Max.   :1.00000
##      Online         CreditCard
##  Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.0000
##  Median :1.0000   Median :0.0000
##  Mean   :0.5847   Mean   :0.2927
##  3rd Qu.:1.0000   3rd Qu.:1.0000
##  Max.   :1.0000   Max.   :1.0000
```

```r
cat("The size of the training dataset is:",nrow(t.df))
```

```
## The size of the training dataset is: 3000
```

```r
summary(v.df)
```

```
##       Age         Experience        Income           Family     
##  Min.   :23.0   Min.   :-3.00   Min.   :  8.00   Min.   :1.00  
##  1st Qu.:35.0   1st Qu.:10.00   1st Qu.: 39.00   1st Qu.:1.00  
##  Median :45.0   Median :20.00   Median : 64.00   Median :2.00  
##  Mean   :45.2   Mean   :19.97   Mean   : 74.81   Mean   :2.41  
##  3rd Qu.:55.0   3rd Qu.:30.00   3rd Qu.: 99.00   3rd Qu.:3.00  
##  Max.   :67.0   Max.   :43.00   Max.   :218.00   Max.   :4.00  
##      CCAvg          Education.1     Education.2     Education.3   
##  Min.   : 0.000   Min.   :0.000   Min.   :0.000   Min.   :0.000  
##  1st Qu.: 0.700   1st Qu.:0.000   1st Qu.:0.000   1st Qu.:0.000  
##  Median : 1.600   Median :0.000   Median :0.000   Median :0.000  
##  Mean   : 1.973   Mean   :0.422   Mean   :0.274   Mean   :0.304  
##  3rd Qu.: 2.600   3rd Qu.:1.000   3rd Qu.:1.000   3rd Qu.:1.000  
##  Max.   :10.000   Max.   :1.000   Max.   :1.000   Max.   :1.000  
##     Mortgage       Personal.Loan    Securities.Account   CD.Account    
##  Min.   :  0.00   Min.   :0.0000   Min.   :0.0000     Min.   :0.0000  
##  1st Qu.:  0.00   1st Qu.:0.0000   1st Qu.:0.0000     1st Qu.:0.0000  
##  Median :  0.00   Median :0.0000   Median :0.0000     Median :0.0000  
##  Mean   : 55.24   Mean   :0.1025   Mean   :0.1105     Mean   :0.0705  
##  3rd Qu.: 97.25   3rd Qu.:0.0000   3rd Qu.:0.0000     3rd Qu.:0.0000  
##  Max.   :617.00   Max.   :1.0000   Max.   :1.0000     Max.   :1.0000  
##      Online        CreditCard   
##  Min.   :0.000   Min.   :0.000  
##  1st Qu.:0.000   1st Qu.:0.000  
##  Median :1.000   Median :0.000  
##  Mean   :0.615   Mean   :0.296  
##  3rd Qu.:1.000   3rd Qu.:1.000  
##  Max.   :1.000   Max.   :1.000
```

```r
cat("The size of the validation dataset is:",nrow(v.df))
```

```
## The size of the validation dataset is: 2000
```

normalizing the dataset

```r
train.norm <- t.df[,-10]
valid.norm <- v.df[,-10]

norm <- preProcess(t.df[,-10],method=c("center","scale"))

train.norm <- predict(norm,t.df[,-10])
valid.norm <- predict(norm,v.df[,-10])
```

Questions

Consider the following customer:

1. 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, and Credit Card = 1. Perform a k-NN classification with all predictors except ID and ZIP code using k = 1. Remember to transform categorical predictors with more than two categories into dummy variables first. Specify the success class as 1 (loan acceptance), and use the default cutoff value of 0.5. How would this customer be classified?

Creating new customer data

```
new.cust <- 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
)

# Normalize the new customer dataset
cust.norm <- predict(norm, new.cust)
```

Performing kNN classification

```
pred1 <- class::knn(train = train.norm,
                       test = cust.norm,
                       cl = t.df$Personal.Loan, k = 1)

pred1
```

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

2.What is a choice of k that balances between overfitting and ignoring the predictor information?

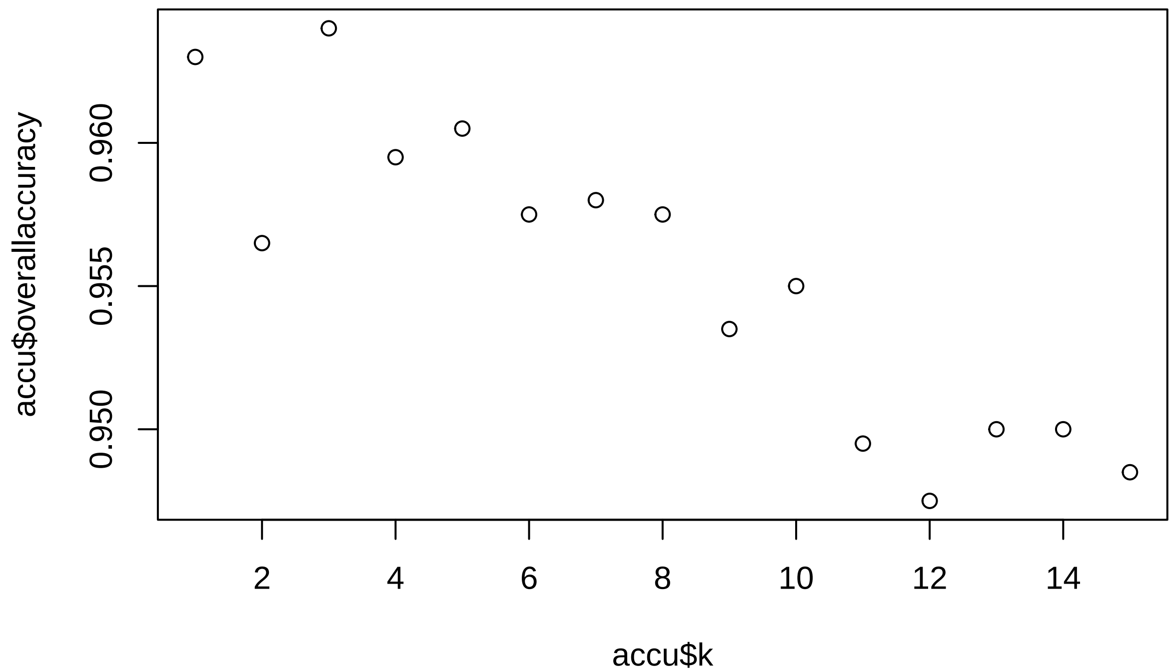
```
# Calculate the accuracy for each value of k
# Set the range of k values to consider
accu      <- data.frame(k = seq(1, 15, 1), overallaccuracy = rep(0, 15))
for(i in 1:15) {
  kn <- class::knn(train = train.norm,
                   test = valid.norm,
                   cl = t.df$Personal.Loan, k = i)
  accu[i, 2] <- confusionMatrix(kn,
                  as.factor(v.df$Personal.Loan),positive = "1")$overall[1]
}

which(accu[,2] == max(accu[,2]))
```

```
## [1] 3
```

```
accu
```

```
##     k overallaccuracy
## 1   1          0.9630
## 2   2          0.9565
## 3   3          0.9640
## 4   4          0.9595
## 5   5          0.9605
## 6   6          0.9575
## 7   7          0.9580
## 8   8          0.9575
## 9   9          0.9535
## 10 10          0.9550
## 11 11          0.9495
## 12 12          0.9475
## 13 13          0.9500
## 14 14          0.9500
## 15 15          0.9485
```

```
plot(accu$k,accu$overallaccuracy)
```



3. Show the confusion matrix for the validation data that results from using the best k.

confusion matrix

```
predt <- class::knn(train = train.norm,
                   test = valid.norm,
                   cl =  t.df$Personal.Loan, k=3)

confusionMatrix(predt,as.factor(v.df$Personal.Loan))
```

```
## 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
##
```

4. Consider the following customer: 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 and CreditCard = 1. Classify the customer using the best k.

now creating the 2nd new customer dataset

```
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)


#Normalizing the 2nd customer dataset

cust_2 <- predict(norm , customer2)
```

Question-5: Repeating the process by partitioning the data into three parts - 50%, 30%, 20%,Apply the k-NN method with the k chosen above. Compare the confusion matrix of the test set with that of the training and validation sets. Comment on the differences and their reason.

```
set.seed(123)
Train_In <- sample(row.names(the_neww), .5*dim(the_neww)[1])#create train index
```

```r
#create validation index
Va_In <- sample(setdiff(row.names(the_neww),Train_In),.3*dim(the_neww)[1])

Test_In =setdiff(row.names(the_neww),union(Train_In,Va_In))#create test index

train.d <- the_neww[Train_In,]

cat("The size of the  new training dataset is:", nrow(train.d))
```

```
## The size of the  new training dataset is: 2500
```

```r
valid.d <- the_neww[Va_In, ]
cat("The size of the new validation dataset is:", nrow(valid.d))
```

```
## The size of the new validation dataset is: 1500
```

```r
test.d <- the_neww[Test_In, ]
cat("The size of the new test dataset is:", nrow(test.d))
```

```
## The size of the new test dataset is: 1000
```

Data Normalizing

```r
norm.val <- preProcess(train.d[, -10], method=c("center", "scale"))
train.norm <- predict(norm.val, train.d[, -10])
valid.norm <- predict(norm.val, valid.d[, -10])
test.norm <- predict(norm.val, test.d[,-10])
```

Performing kNN and creating confusion matrix on training, testing, validation data

```r
pred31 <- class::knn(train = train.norm,
                     test = test.norm,
                     cl = train.d$Personal.Loan, k=3)

confusionMatrix(pred31,as.factor(test.d$Personal.Loan))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 890  38
##          1   2  70
##
##                Accuracy : 0.96
##                  95% CI : (0.9459, 0.9713)
##     No Information Rate : 0.892
```

```
##     P-Value [Acc > NIR] : 4.095e-15
##
##                   Kappa : 0.7568
##
##  Mcnemar's Test P-Value : 3.130e-08
##
##             Sensitivity : 0.9978
##             Specificity : 0.6481
##          Pos Pred Value : 0.9591
##          Neg Pred Value : 0.9722
##              Prevalence : 0.8920
##          Detection Rate : 0.8900
##    Detection Prevalence : 0.9280
##       Balanced Accuracy : 0.8230
##
##        'Positive' Class : 0
##
```

```
pred41 <- class::knn(train = train.norm,
                    test = valid.norm,
                    cl = train.d$Personal.Loan, k=3)

confusionMatrix(pred41,as.factor(valid.d$Personal.Loan))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1350   58
##          1    7   85
##
##                Accuracy : 0.9567
##                  95% CI : (0.9451, 0.9664)
##     No Information Rate : 0.9047
##     P-Value [Acc > NIR] : 2.347e-14
##
##                   Kappa : 0.7011
##
##  Mcnemar's Test P-Value : 5.584e-10
##
##             Sensitivity : 0.9948
##             Specificity : 0.5944
##          Pos Pred Value : 0.9588
##          Neg Pred Value : 0.9239
##              Prevalence : 0.9047
##          Detection Rate : 0.9000
##    Detection Prevalence : 0.9387
##       Balanced Accuracy : 0.7946
##
##        'Positive' Class : 0
##
```

```r
pred51 <- class::knn(train = train.norm,
                     test = train.norm,
                     cl = train.d$Personal.Loan, k=3)

confusionMatrix(pred51,as.factor(train.d$Personal.Loan))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 2267   57
##          1    4  172
##
##                Accuracy : 0.9756
##                  95% CI : (0.9688, 0.9813)
##     No Information Rate : 0.9084
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8364
##
##  Mcnemar's Test P-Value : 2.777e-11
##
##             Sensitivity : 0.9982
##             Specificity : 0.7511
##          Pos Pred Value : 0.9755
##          Neg Pred Value : 0.9773
##              Prevalence : 0.9084
##          Detection Rate : 0.9068
##    Detection Prevalence : 0.9296
##       Balanced Accuracy : 0.8747
##
##        'Positive' Class : 0
##
```

“‘