Assignment_2_KNN_811289717

SAKSHI

2023-09-24

Summary

Questions - Answers

1. How would this customer be classified? This new customer would be classified as 0, does not take the personal loan
2. The best K is 3.
3. Matrix printed below.
4. This new customer would be classified as 0, does not take the personal loan.
5. By comparing the confusion matrix of test with that of training and validation, the train set has highest sensitivity of around 75% as compared to other sets.(62-63%). This means the train set can be overfitted or very closely fitted whilst other two distributes evenly. Hence , k=3 is a perfect number to get balance between under and over fit data.

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

---

Before heading into the question, we have to import , clean and normalize the dataset.

Importing and Cleaning the data

Loading the required libraries

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

```
Loading required package: ggplot2

Loading required package: lattice
```

```r
library(e1071)
```

Importing and Reading the data

```r
UB <- read.csv("/Users/sakshibansal/Documents/UniversalBank.csv")
dim(UB)
```

```
[1] 5000   14
```

```r
#Using t function to create a transpose of dataset.
t(t(names(UB)))
```

```
      [,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"
```

Dropping ID and ZIP.Code

```r
UB_2 <- UB[,-c(1,5)]
```

Creating dummy variables for Education

```r
#Converting Education into factor.
UB_2$Education <- as.factor(UB_2$Education)

#Now converting Education into dummy variables
Edu_dum <- dummyVars(~.,data = UB_2)
UB_3 <- as.data.frame(predict(Edu_dum,UB_2))
```

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

```r
set.seed(1)
data_train <- sample(row.names(UB_3),  0.6*dim(UB_3)[1])
data_valid <- setdiff(row.names(UB_3),data_train)
Edu_train <- UB_3[data_train,]
Edu_valid <- UB_3[data_valid,]
```

Printing the sizes of training and validation datasets.

```r
print(paste("The size of the training dataset is:", nrow(Edu_train)))
```

```
[1] "The size of the training dataset is: 3000"
```

```r
print(paste("The size of the validation dataset is:", nrow(Edu_valid)))
```

```
[1] "The size of the validation dataset is: 2000"
```

Normalizing the data now.

```r
Edu_train.norm <- Edu_train[,-10]  #Personal loan is the 10th Variable
Edu_valid.norm <- Edu_valid[,-10]

Edu_norm <- preProcess(Edu_train[, -10], method = c("center", "scale"))
Edu_train.norm <- predict(Edu_norm, Edu_train[,-10])
Edu_valid.norm <- predict(Edu_norm, Edu_valid[,-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?

```r
# We have converted all categorical variables into dummy variables
# Now creating a new sample
new_data <- 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 first customer

new_data <- predict(Edu_norm, new_data)
```

Now, let’s predict using knn-method

```r
Edu_pred <- class::knn(train = Edu_train.norm,
                       test = new_data,
                       cl = Edu_train$Personal.Loan, k=1)

Edu_pred
```

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

---

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

information?

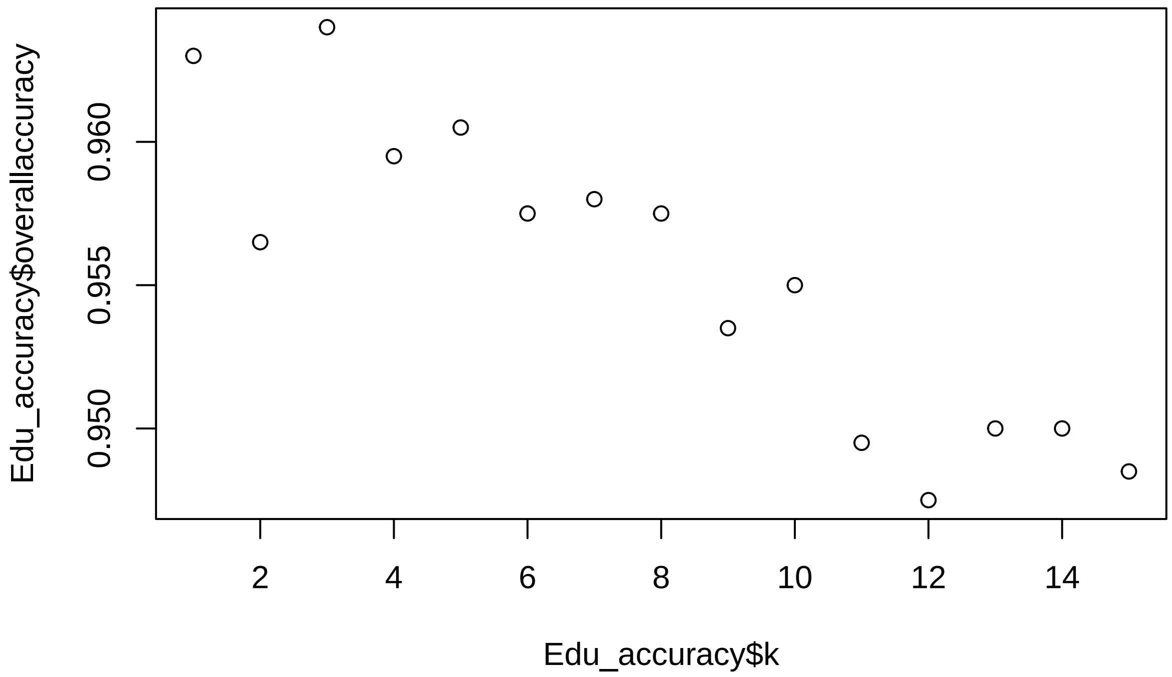
```r
# Calculating the accuracy for each value of k
# Setting the range of k values to consider

 Edu_accuracy <- data.frame(k = seq(1, 15, 1), overallaccuracy = rep(0, 15))
for(i in 1:15) {
  Edu_2 <- class::knn(train = Edu_train.norm,
                         test = Edu_valid.norm,
                         cl = Edu_train$Personal.Loan, k = i)
  Edu_accuracy[i, 2] <- confusionMatrix(Edu_2,
                                       as.factor(Edu_valid$Personal.Loan),
                                       positive = "1")$overall[1]
}

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

```
[1] 3
```

```r
plot(Edu_accuracy$k,Edu_accuracy$overallaccuracy)
```



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

```
Edu_pred2 <- class::knn(train = Edu_train.norm,
                        test = Edu_valid.norm,
                        cl = Edu_train$Personal.Loan, k=3)

confusionMatrix(Edu_pred2,as.factor(Edu_valid$Personal.Loan), positive= "1")$table
```

```
          Reference
Prediction    0    1
         0 1786   63
         1    9  142
```

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.

```
new_data2 <- 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 second customer

new_data2 <- predict(Edu_norm, new_data2)
```

Now, let's predict using knn-method

```
Edu_pred3 <- class::knn(train = Edu_train.norm,
                       test = new_data2,
                       cl = Edu_train$Personal.Loan, k=3)

Edu_pred3
```

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

---

5. Repartition the data, this time into training, validation, and test sets (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.

```
#Repartitioning the data into training , validation and test sets (50%:30%:20%)
#and setting the seed since we need to re-run the code.
set.seed(2)
Edu_train2 <- sample(row.names(UB_3),0.5*dim(UB_3)[1])
Edu_subset <- setdiff(row.names(UB_3),Edu_train2)
Edu_subset2 <- UB_3[Edu_subset,]
Edu_valid2 <- sample(row.names(Edu_subset2),0.6*dim(Edu_subset2)[1])
Edu_test2 <- setdiff(row.names(Edu_subset2),Edu_valid2)

#Making Datasets
Edu.train.d <- UB_3[Edu_train2,]
Edu.valid.d <- UB_3[Edu_valid2,]
Edu.test.d <- UB_3[Edu_test2,]

## Printing the sizes of training and validation datasets.
print(paste("The size of the training dataset is:", nrow(Edu.train.d)))
```

```
[1] "The size of the training dataset is: 2500"
```

```
print(paste("The size of the validation dataset is:", nrow(Edu.valid.d)))
```

```
[1] "The size of the validation dataset is: 1500"
```

```
print(paste("The size of the test dataset is:", nrow(Edu.test.d)))
```

```
[1] "The size of the test dataset is: 1000"
```

Normalizing the data now.

```
Edu.train.d.norm <- Edu.train.d[-10] #Personal loan is the 10th variable.
Edu.valid.d.norm <- Edu.valid.d[-10] #Personal loan is the 10th variable.
Edu.test.d.norm <- Edu.test.d[-10] #Personal loan is the 10th variable.

Edu_norm2 <- preProcess(Edu.train.d[,-10] , method = c("center","scale"))
Edu.train.d.norm <- predict(Edu_norm2,Edu.train.d[,-10])
Edu.valid.d.norm <- predict(Edu_norm2,Edu.valid.d[,-10])
Edu.test.d.norm <- predict(Edu_norm2,Edu.test.d[,-10])
```

Now, using knn-method for training and test dataset.

```
Edu_pred4 <- class::knn(train = Edu.train.d.norm,
                       test = Edu.test.d.norm,
                       cl = Edu.train.d$Personal.Loan, k=3)

confusionMatrix(Edu_pred4,as.factor(Edu.test.d$Personal.Loan), positive= "1")$table
```

```
          Reference
Prediction   0   1
         0 922  28
         1   4  46
```

Now, using knn-method for validation and testing dataset.

```
Edu_pred5 <- class::knn(train = Edu.valid.d.norm,
                       test = Edu.test.d.norm,
                       cl = Edu.valid.d$Personal.Loan, k=3)

confusionMatrix(Edu_pred5,as.factor(Edu.test.d$Personal.Loan), positive= "1")$table
```

```
          Reference
Prediction   0   1
         0 920  27
         1   6  47
```

Now, using knn-method for training and training dataset.

```
Edu_pred6 <- class::knn(train = Edu.train.d.norm,
                       test = Edu.train.d.norm,
                       cl = Edu.train.d$Personal.Loan, k=3)

confusionMatrix(Edu_pred6,as.factor(Edu.train.d$Personal.Loan), positive= "1")$table
```

```
          Reference
Prediction    0    1
         0 2246   61
         1    5  188
```