Assignment-2-FML

Dev

2023-09-29

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. Confusion matrix is shown below
4. For k=3, it can be predicted that customer will not take a personal loan.
5. All three confusion matrices are displayed below. We can observe that the training set has highest sensitivity of approximately 96% which means that the data is over-fitting. The other two matrices are perfectly fitted as they are around 70% sensitive and generalised properly. Hence, k=3 is a good value to get accuracy.

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.

Data Import and Cleaning

First, load the required libraries

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

```
Loading required package: ggplot2

Loading required package: lattice
```

```r
library(e1071)
```

Read the data.

```r
setwd("/Users/devmarwah/Downloads")
universal.df <- read.csv("UniversalBank.csv")
dim(universal.df)
```

```
[1] 5000   14
```

```r
 # The t function creates a transpose of the dataframe
```

Dropping ID and ZIP

```r
universal.df <- universal.df[,-c(1,5)]
```

Making education as dummy variable as it is the only attribute with three categorical variables. Also, splitting data into 60% training and 40% validation datasets.

```r
# Only Education needs to be converted to factor
universal.df$Education <- as.factor(universal.df$Education)

# Now, convert Education to Dummy Variables

dmy<- dummyVars(~., data = universal.df) # This creates the dummy groups
universal_dum.df <- as.data.frame(predict(dmy,universal.df))


set.seed(1)  # Important to ensure that we get the same sample if we rerun the code
train.index <- sample(row.names(universal_dum.df), 0.6*dim(universal_dum.df)[1])
valid.index <- setdiff(row.names(universal_dum.df), train.index)
train.df <- universal_dum.df[train.index,]
valid.df <- universal_dum.df[valid.index,]


cat("The size of the training set is:", nrow(train.df))
```

```
The size of the training set is: 3000
```

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

```
The size of the validation set is: 2000
```

Normalizing data-

```r
#Dropping 10th attribute since it is to be predicted
norm <- preProcess(train.df[, -10], method=c("center", "scale"))
train.norm.df <- predict(norm, train.df[, -10])
valid.norm.df <- predict(norm, valid.df[, -10])
```

Questions

Q-1: 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 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
# Making new data a data frome and then normalizing it
new.customer.df <- 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

new.customer.df <- predict(norm, new.customer.df)
```

Now, let us predict using knn

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

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

Hence, customer will not take a personal loan.

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

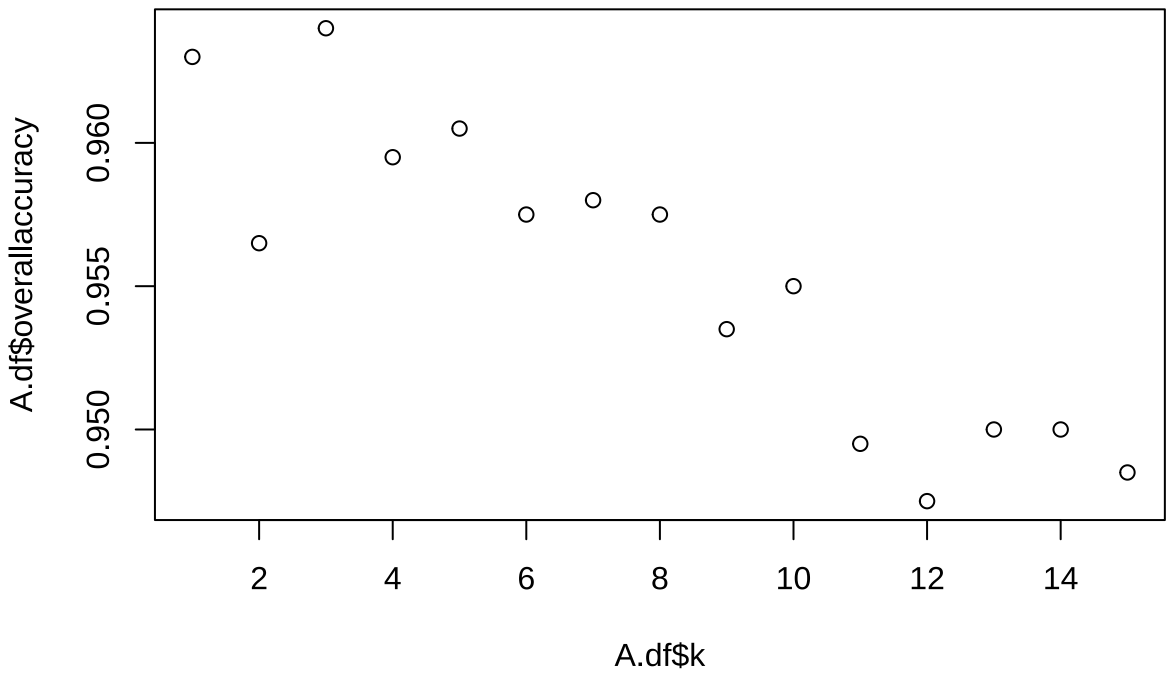
```r
# Calculating the accuracy for different values of  k
# Set the range of k values to consider

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

which(A.df[,2] == max(A.df[,2]))
```

```
[1] 3
```

```r
plot(A.df$k,A.df$overallaccuracy)
```



Therefore, best value of K is 3.

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

Confusion matrix for k=3 can be shown as follows:

```r
pred3 <- class::knn(train = train.norm.df,
                       test = valid.norm.df,
                       cl = train.df$Personal.Loan, k = 3)

confusionMatrix(pred3,as.factor(valid.df$Personal.Loan),positive = "1")$table
```

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

Q-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.

Making dataframe of second customer-

```r
second.customer.df=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 second customer

second.customer.df <- predict(norm,second.customer.df)

#Predicting using knn:

pred_second=class::knn(train = train.norm.df,
                       test=second.customer.df,
                       cl=train.df$Personal.Loan,k=3)
pred_second
```

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

Hence, this new customer will also not take a personal loan.

Q-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.

Making sets-

```
#Making indexes -
train_5.index= sample(row.names(universal_dum.df),0.5*dim(universal_dum.df)[1])
subset.index = setdiff(row.names(universal_dum.df),train_5.index)
subset.df = universal_dum.df[subset.index,]
valid_5.index=sample(row.names(subset.df),0.6*dim(subset.df)[1]) # 60% of the subset would be 30% of to
test_5.index = setdiff(row.names(subset.df),valid_5.index) #Remaining would be 20% of total

#Making datasets -
train_5.df = universal_dum.df[train_5.index,]
valid_5.df = universal_dum.df[valid_5.index,]
test_5.df = universal_dum.df[test_5.index,]

#Checking size of sets to confirm correct splitting of data
cat("Size of new training set is:", nrow(train_5.df),"\n")
```

```
Size of new training set is: 2500
```

```
cat("Size of new validation set is:", nrow(valid_5.df), "\n" )
```

```
Size of new validation set is: 1500
```

```
cat("Size of new test set is:", nrow(test_5.df),"\n")
```

```
Size of new test set is: 1000
```

Normalizing datasets-

```
norm <- preProcess(train_5.df[, -10], method=c("center", "scale"))
train_5.norm.df <- predict(norm, train_5.df[, -10])
valid_5.norm.df <- predict(norm, valid_5.df[, -10])
test_5.norm.df  <- predict(norm,test_5.df[,-10])
```

Hence, our splitting of data is correct.

Applying knn algorithm to test and training set:

```
pred5_1 <- class::knn(train = train_5.norm.df,
                      test = test_5.norm.df,
                      cl = train_5.df$Personal.Loan, k = 3)
confusionMatrix(pred5_1,as.factor(test_5.df$Personal.Loan),positive="1")$table
```

```
          Reference
Prediction   0   1
         0 896  34
         1   4  66
```

Applying knn algorithm to train and valid set:

```
pred5_2 <- class::knn(train = valid_5.norm.df,
                       test = test_5.norm.df,
                       cl = valid_5.df$Personal.Loan, k = 3)
confusionMatrix(pred5_2,as.factor(test_5.df$Personal.Loan),positive="1")$table
```

```
          Reference
Prediction   0   1
         0 899  44
         1   1  56
```

Applying knn training set itself

```
pred5_2 <- class::knn(train = train_5.norm.df,
                       test = train_5.norm.df,
                       cl = train_5.df$Personal.Loan, k = 3)
confusionMatrix(pred5_2,as.factor(train_5.df$Personal.Loan),positive="1")$table
```

```
          Reference
Prediction    0    1
         0 2254   50
         1    6  190
```