FML Assignment_2_KNN

Sri Naga Dattu Gummadi

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. What is a choice of k that balances between overfitting and ignoring the predictor information? The best K is 2
3. Show the confusion matrix for the validation data that results from using the best k.
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 Credit Card = 1. Classify the customer using the best k. The New customer is classified as 1 and hence he would get a personal loan
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. Train vs train test got the highest accuracy of 97% while the train vs validation and train vs test got 95% accuracy. Because the same data is being tested when train vs train is done whereas the test is done on different data when train vs validation and train vs test.

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

loading the required libraries

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

```
## Loading required package: ggplot2

## Loading required package: lattice
```

```r
library(e1071)
```

Read the data

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

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

```r
t(t(names(universal.df)))
```

```
##       [,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"
```

Drop ID and Zip code (unnecessary attributes)

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

Education needs to be converted to factor

```
universal.df$Education <- as.factor(universal.df$Education)
```

Converting Education into dummy varaibles

```
groups <- dummyVars(~., data = universal.df)
universal_m.df <- as.data.frame(predict(groups,universal.df))
```

```
set.seed(656)  # To ensure that we get the same sample when we rerun the code
train.index <- sample(row.names(universal_m.df), 0.6*dim(universal_m.df)[1])
valid.index <- setdiff(row.names(universal_m.df), train.index)
train.df <- universal_m.df[train.index,]
valid.df <- universal_m.df[valid.index,]
t(t(names(train.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"
```

Normalizing the data

```
train.norm.df <- train.df[,-10] # Note that Personal Income is the 10th variable
valid.norm.df <- valid.df[,-10]

norm.values <- preProcess(train.df[, -10], method=c("center", "scale"))
train.norm.df <- predict(norm.values, train.df[, -10])
valid.norm.df <- predict(norm.values, valid.df[, -10])
```

Question

1. 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?

```
# We have converted all categorical variables to dummy variables
# Let's create a new sample
new_customer <- 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.cust.norm <- predict(norm.values, new_customer)
```

Now predict using KNN

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

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

New customer would not get loan as knn.pred1 is 0.

---

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

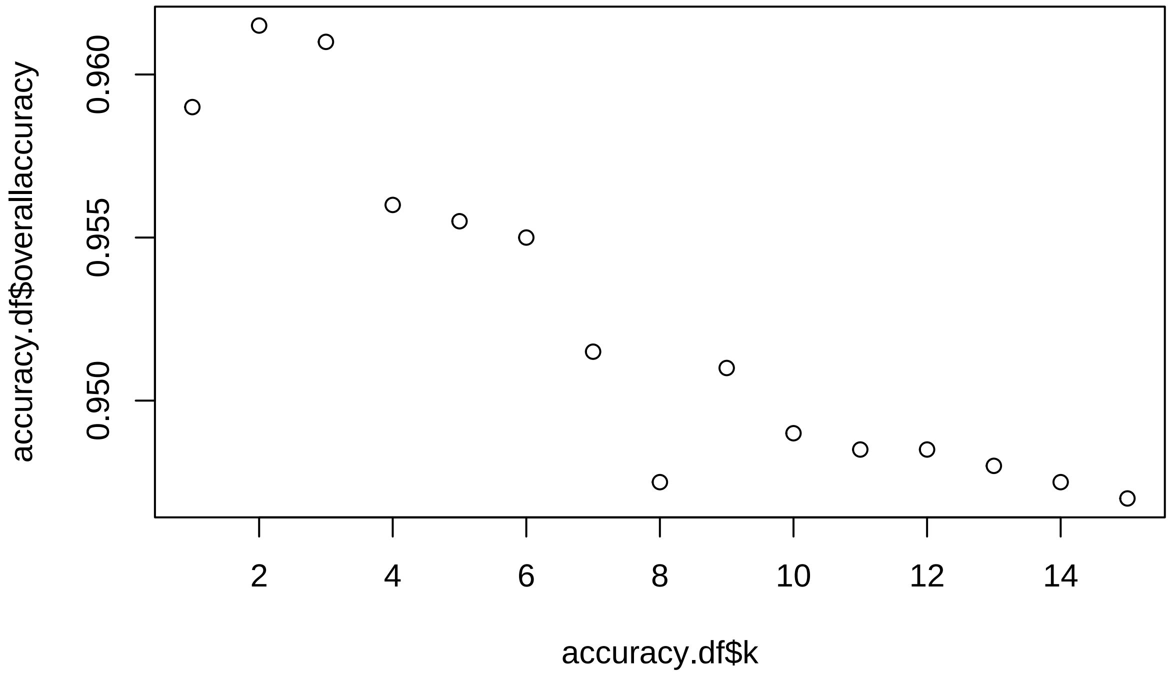
```
# Calculate the accuracy for each value of k
# Set the range of k values to consider

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

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

```
## [1] 2
```

```
plot(accuracy.df$k,accuracy.df$overallaccuracy)
```



# The best K in KNN value is 2

---

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

```
knn_pred_1 <- class::knn(train=train.norm.df,
                         test = valid.norm.df,
                         cl=train.df$Personal.Loan,k=2) #best k value taken from above
confusion_matrix_1 <- confusionMatrix(knn_pred_1, as.factor(valid.df$Personal.Loan),positive="1")
confusion_matrix_1
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1792   67
##          1   23  118
##
##                Accuracy : 0.955
##                  95% CI : (0.945, 0.9637)
##     No Information Rate : 0.9075
##     P-Value [Acc > NIR] : 5.538e-16
##
##                   Kappa : 0.6999
##
##  Mcnemar's Test P-Value : 5.826e-06
##
##             Sensitivity : 0.6378
##             Specificity : 0.9873
##          Pos Pred Value : 0.8369
##          Neg Pred Value : 0.9640
##              Prevalence : 0.0925
##          Detection Rate : 0.0590
##    Detection Prevalence : 0.0705
##       Balanced Accuracy : 0.8126
##
##        'Positive' Class : 1
##
```

---

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 Credit Card = 1. Classify the customer using the best k.

```r
knn_pred_2 <- class::knn(train=train.norm.df,
                         test = new_customer,
                         cl=train.df$Personal.Loan,k=2) #best k value taken from above
knn_pred_2
```

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

KNN_pred_2 classifies k as 1, so new customer get the personal loan

---

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.

```r
#Partitioning the data into training set, testing set and validation sets.
set.seed(656)
#Training Set is taken as 50%
train.size.t <- sample(row.names(universal_m.df),0.5*dim(universal_m.df)[1])

#Validation Set is taken as 30%
valid.size.v <- sample(setdiff(row.names(universal_m.df),train.size.t),0.3*dim(universal_m.df)[1])

#Testing Set is taken as 20%
test.size.t <- setdiff(row.names(universal_m.df),c(train.size.t,valid.size.v))
```

Assigning partitioned data into variables

```r
train.df.t <- universal_m.df[train.size.t,]
valid.df.v <- universal_m.df[valid.size.v,]
test.df.t <- universal_m.df[test.size.t,]
```

Normalizing the data

```r
train.norm.t <- train.df.t[,-10] # Note that Personal Income is the 10th variable
valid.norm.v <- valid.df.v[,-10]
test.norm.t <- test.df.t[,-10]


norm.values.n <- preProcess(train.df.t[, -10], method=c("center", "scale"))
train.norm.t <- predict(norm.values.n, train.df.t[, -10])
valid.norm.v <- predict(norm.values.n, valid.df.v[, -10])
test.norm.t <- predict(norm.values.n, test.df.t[, -10])
```

Applying the K value=2 obtained in above KNN classification

```r
knn_pred_3 <- class::knn(train=train.norm.t,
                         test = new_customer,
                         cl=train.df.t$Personal.Loan,k=2)
knn_pred_3
```

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

Building confusion matrix

```r
knn_pred_4 <- class::knn(train=train.norm.t,
                         test = valid.norm.v,
                         cl=train.df.t$Personal.Loan,k=2)
confusion_matrix_2 <- confusionMatrix(knn_pred_4,as.factor(valid.df.v$Personal.Loan),positive="1")
confusion_matrix_2
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1338   53
##          1   15   94
##
##                Accuracy : 0.9547
##                  95% CI : (0.9429, 0.9646)
##     No Information Rate : 0.902
##     P-Value [Acc > NIR] : 2.598e-14
##
##                   Kappa : 0.7102
##
##  Mcnemar's Test P-Value : 7.226e-06
##
##             Sensitivity : 0.63946
##             Specificity : 0.98891
##          Pos Pred Value : 0.86239
##          Neg Pred Value : 0.96190
##              Prevalence : 0.09800
##          Detection Rate : 0.06267
##    Detection Prevalence : 0.07267
##       Balanced Accuracy : 0.81418
##
##        'Positive' Class : 1
##
```

```r
knn_pred_5 <- class::knn(train=train.norm.t,
                         test = test.norm.t,
                         cl=train.df.t$Personal.Loan,k=2)
confusion_matrix_3 <- confusionMatrix(knn_pred_5,as.factor(test.df.t$Personal.Loan),positive="1")
confusion_matrix_3
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 903  30
##          1  13  54
##
##                Accuracy : 0.957
##                  95% CI : (0.9425, 0.9687)
##     No Information Rate : 0.916
##     P-Value [Acc > NIR] : 2.411e-07
##
```

```
##                   Kappa : 0.6923
##
##  Mcnemar's Test P-Value : 0.01469
##
##             Sensitivity : 0.6429
##             Specificity : 0.9858
##          Pos Pred Value : 0.8060
##          Neg Pred Value : 0.9678
##              Prevalence : 0.0840
##          Detection Rate : 0.0540
##    Detection Prevalence : 0.0670
##       Balanced Accuracy : 0.8143
##
##        'Positive' Class : 1
##
```

```
knn_pred_6 <- class::knn(train=train.norm.t,
                         test = train.norm.t,
                         cl=train.df.t$Personal.Loan,k=2)
confusion_matrix_4 <- confusionMatrix(knn_pred_6,as.factor(train.df.t$Personal.Loan),positive="1")
confusion_matrix_4
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 2238   30
##          1   13  219
##
##                Accuracy : 0.9828
##                  95% CI : (0.9769, 0.9875)
##     No Information Rate : 0.9004
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.9011
##
##  Mcnemar's Test P-Value : 0.01469
##
##             Sensitivity : 0.8795
##             Specificity : 0.9942
##          Pos Pred Value : 0.9440
##          Neg Pred Value : 0.9868
##              Prevalence : 0.0996
##          Detection Rate : 0.0876
##    Detection Prevalence : 0.0928
##       Balanced Accuracy : 0.9369
##
##        'Positive' Class : 1
##
```