ASSIGNMENT_2_64060

Rohith Desamseety

2022-10-06

```
#importing the necessary packages
library('caret')
```

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

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

```
library('ISLR')
library('dplyr')
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library('class')

#Importing the dataset
rundata <- read.csv("~/Downloads/UniversalBank.csv", header = TRUE,
                         sep =",", stringsAsFactors = FALSE)
#Question_1
#performing a k-NN classification with all predictors deleted, deleting ID and ZIP Co
de from every single column
rundata$ID <- NULL
rundata$ZIP.Code <- NULL
summary(rundata)
```

```
##       Age           Experience       Income           Family
##  Min.   :23.00   Min.   :-3.0   Min.   :  8.00   Min.   :1.000
##  1st Qu.:35.00   1st Qu.:10.0   1st Qu.: 39.00   1st Qu.:1.000
##  Median :45.00   Median :20.0   Median : 64.00   Median :2.000
##  Mean   :45.34   Mean   :20.1   Mean   : 73.77   Mean   :2.396
##  3rd Qu.:55.00   3rd Qu.:30.0   3rd Qu.: 98.00   3rd Qu.:3.000
##  Max.   :67.00   Max.   :43.0   Max.   :224.00   Max.   :4.000
##      CCAvg          Education        Mortgage     Personal.Loan
##  Min.   : 0.000   Min.   :1.000   Min.   :  0.0   Min.   :0.000
##  1st Qu.: 0.700   1st Qu.:1.000   1st Qu.:  0.0   1st Qu.:0.000
##  Median : 1.500   Median :2.000   Median :  0.0   Median :0.000
##  Mean   : 1.938   Mean   :1.881   Mean   : 56.5   Mean   :0.096
##  3rd Qu.: 2.500   3rd Qu.:3.000   3rd Qu.:101.0   3rd Qu.:0.000
##  Max.   :10.000   Max.   :3.000   Max.   :635.0   Max.   :1.000
##  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
```

```
#translating the categorical variable "personal loan" into a variable that distinguis
hes between "yes" and "no" responses.

rundata$Personal.Loan=  as.factor(rundata$Personal.Loan)


#Apply preProcess() from the caret package to divide the data into training and valid
ation in order to standardize it.
M_norm <- preProcess(rundata[, -8],method = c("center", "scale"))
rundata_norm <- predict(M_norm,rundata)
summary(rundata_norm)
```

```
##       Age              Experience           Income            Family
##  Min.   :-1.94871   Min.   :-2.014710   Min.   :-1.4288   Min.   :-1.2167
##  1st Qu.:-0.90188   1st Qu.:-0.881116   1st Qu.:-0.7554   1st Qu.:-1.2167
##  Median :-0.02952   Median :-0.009121   Median :-0.2123   Median :-0.3454
##  Mean   : 0.00000   Mean   : 0.000000   Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.84284   3rd Qu.: 0.862874   3rd Qu.: 0.5263   3rd Qu.: 0.5259
##  Max.   : 1.88967   Max.   : 1.996468   Max.   : 3.2634   Max.   : 1.3973
##      CCAvg            Education          Mortgage       Personal.Loan
##  Min.   :-1.1089   Min.   :-1.0490   Min.   :-0.5555   0:4520
##  1st Qu.:-0.7083   1st Qu.:-1.0490   1st Qu.:-0.5555   1: 480
##  Median :-0.2506   Median : 0.1417   Median :-0.5555
##  Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.3216   3rd Qu.: 1.3324   3rd Qu.: 0.4375
##  Max.   : 4.6131   Max.   : 1.3324   Max.   : 5.6875
##  Securities.Account   CD.Account          Online          CreditCard
##  Min.   :-0.3414    Min.   :-0.2535   Min.   :-1.2165   Min.   :-0.6452
##  1st Qu.:-0.3414    1st Qu.:-0.2535   1st Qu.:-1.2165   1st Qu.:-0.6452
##  Median :-0.3414    Median :-0.2535   Median : 0.8219   Median :-0.6452
##  Mean   : 0.0000    Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.:-0.3414    3rd Qu.:-0.2535   3rd Qu.: 0.8219   3rd Qu.: 1.5495
##  Max.   : 2.9286    Max.   : 3.9438   Max.   : 0.8219   Max.   : 1.5495
```

```
#separating the data into test and training sets
T_index <- createDataPartition(rundata$Personal.Loan, p = 0.6, list = FALSE)
t.df = rundata_norm[T_index,]
validate.df = rundata_norm[-T_index,]

print(head(t.df))
```

```
##            Age  Experience     Income     Family      CCAvg  Education
## 2  -0.02952064 -0.09632058 -0.8640230  0.5259383 -0.2505855 -1.0489730
## 5  -0.90188002 -1.05551525 -0.6250678  1.3972742 -0.5366825  0.1416887
## 9  -0.90188002 -0.88111622  0.1569675  0.5259383 -0.7655601  0.1416887
## 10 -0.98911595 -0.96831574  2.3075645 -1.2167334  3.9836502  1.3323505
## 11  1.71519811  1.64766972  0.6783244  1.3972742  0.2643891  1.3323505
## 12 -1.42529564 -1.31711380 -0.6250678  0.5259383 -1.0516571  0.1416887
##      Mortgage Personal.Loan Securities.Account CD.Account     Online CreditCard
## 2  -0.5554684             0          2.9286223 -0.2535149 -1.2164961 -0.6452498
## 5  -0.5554684             0         -0.3413892 -0.2535149 -1.2164961  1.5494774
## 9   0.4670084             0         -0.3413892 -0.2535149  0.8218687 -0.6452498
## 10 -0.5554684             1         -0.3413892 -0.2535149 -1.2164961 -0.6452498
## 11 -0.5554684             0         -0.3413892 -0.2535149 -1.2164961 -0.6452498
## 12 -0.5554684             0         -0.3413892 -0.2535149  0.8218687 -0.6452498
```

```
#predictions of data
library(caret)
library(FNN)
```

```
##
## Attaching package: 'FNN'
```

```
## The following objects are masked from 'package:class':
##
##     knn, knn.cv
```

```
my.predict = data.frame(Age = 40, Experience = 10, Income = 84, Family = 2,
                        CCAvg = 2, Education = 1, Mortgage = 0, Securities.Account =
                          0, CD.Account = 0, Online = 1, CreditCard = 1)
print(my.predict)
```

```
##   Age Experience Income Family CCAvg Education Mortgage Securities.Account
## 1  40         10     84      2     2         1        0                   0
##   CD.Account Online CreditCard
## 1          0      1          1
```

```
my.predict_Norm <- predict(M_norm,my.predict)

predictions <- knn(train= as.data.frame(t.df[,1:7,9:12]),
                   test = as.data.frame(my.predict_Norm[,1:7,9:12]),
                   cl= t.df$Personal.Loan,
                   k=1)
```

```
## Warning in drop && !has.j: 'length(x) = 4 > 1' in coercion to 'logical(1)'
```

```
## Warning in drop && length(y) == 1L: 'length(x) = 4 > 1' in coercion to
## 'logical(1)'
```

```
## Warning in drop && !mdrop: 'length(x) = 4 > 1' in coercion to 'logical(1)'
```

```
## Warning in drop && !has.j: 'length(x) = 4 > 1' in coercion to 'logical(1)'
```

```
## Warning in drop && length(y) == 1L: 'length(x) = 4 > 1' in coercion to
## 'logical(1)'
```

```
## Warning in drop && !mdrop: 'length(x) = 4 > 1' in coercion to 'logical(1)'
```

```
print(predictions)
```

```
## [1] 0
## attr(,"nn.index")
##      [,1]
## [1,]  407
## attr(,"nn.dist")
##           [,1]
## [1,] 0.2986486
## Levels: 0
```

```
#Question_2
#finding the K value that strikes a compromise between over- and underfitting.
set.seed(123)
UniBank <- trainControl(method= "repeatedcv", number = 3, repeats = 2)
searchGrid = expand.grid(k=1:10)

knn.model = train(Personal.Loan~., data = t.df, method = 'knn', tuneGrid = searchGri
d,trControl = UniBank)

knn.model
```

```
## k-Nearest Neighbors
##
## 3000 samples
##   11 predictor
##    2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (3 fold, repeated 2 times)
## Summary of sample sizes: 2000, 2000, 2000, 2000, 2000, 2000, ...
## Resampling results across tuning parameters:
##
##   k   Accuracy   Kappa
##    1  0.9563333  0.7261500
##    2  0.9515000  0.6898802
##    3  0.9576667  0.7137311
##    4  0.9541667  0.6886875
##    5  0.9520000  0.6657176
##    6  0.9503333  0.6538505
##    7  0.9508333  0.6506273
##    8  0.9488333  0.6288211
##    9  0.9466667  0.6086325
##   10  0.9451667  0.5941739
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 3.
```

```
#The best value of k is 3, which finds a balance between underfitting and overfitting
of the data.
#Question 3
#confusion Matrix is below
pre_bank <- predict(knn.model,validate.df)

confusionMatrix(pre_bank,validate.df$Personal.Loan)
```

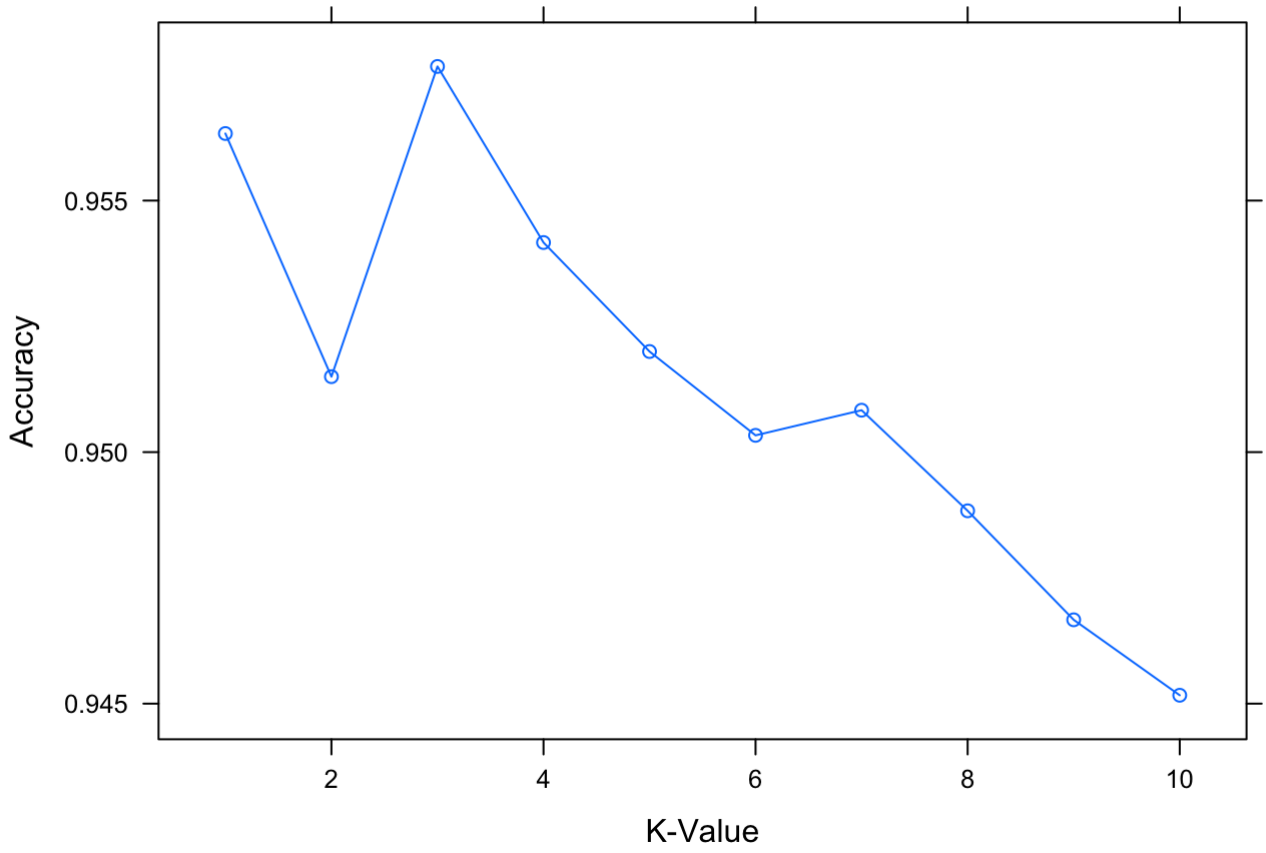
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1797   84
##          1   11  108
##
##                Accuracy : 0.9525
##                  95% CI : (0.9422, 0.9614)
##     No Information Rate : 0.904
##     P-Value [Acc > NIR] : 4.861e-16
##
##                   Kappa : 0.6703
##
##  Mcnemar's Test P-Value : 1.501e-13
##
##             Sensitivity : 0.9939
##             Specificity : 0.5625
##          Pos Pred Value : 0.9553
##          Neg Pred Value : 0.9076
##              Prevalence : 0.9040
##          Detection Rate : 0.8985
##    Detection Prevalence : 0.9405
##       Balanced Accuracy : 0.7782
##
##        'Positive' Class : 0
##
```

```
#The matrix's accuracy rate is 95.1%.
```

```
#Question 4
#Levels
#utilizing the highest K to categorize the customer.
my.predict_Norm = data.frame(Age = 40, Experience = 10, Income = 84, Family = 2,
                                    CCAvg = 2, Education = 1, Mortgage = 0,
                                    Securities.Account =0, CD.Account = 0, Online = 1,
                                    CreditCard = 1)
my.predict_Norm = predict(M_norm, my.predict)
predict(knn.model, my.predict_Norm)
```

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

```
#There is also a plot that displays the optimal K (3) value, which is the one with th
e greatest accuracy.
plot(knn.model, type = "b", xlab = "K-Value", ylab = "Accuracy")
```




```
#Question 5
#constructing training, test, and validation sets using the data that was collected.
train_size = 0.5 #training(50%)
T_index = createDataPartition(rundata$Personal.Loan, p = 0.5, list = FALSE)
t.df = rundata_norm[T_index,]


test_size = 0.2 #Test Data(20%)
Test_index = createDataPartition(rundata$Personal.Loan, p = 0.2, list = FALSE)
Test.df = rundata_norm[Test_index,]


valid_size = 0.3 #validation(30%)
Validation_index = createDataPartition(rundata$Personal.Loan, p = 0.3, list = FALSE)
validate.df = rundata_norm[Validation_index,]



Testingsknn <- knn(train = t.df[,-8], test = Test.df[,-8], cl = t.df[,8], k =3)
validateknn <- knn(train = t.df[,-8], test = validate.df[,-8], cl = t.df[,8], k =3)
Trainsknn <- knn(train = t.df[,-8], test = t.df[,-8], cl = t.df[,8], k =3)

confusionMatrix(Testingsknn, Test.df[,8])
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 899  30
##          1   5  66
##
##                Accuracy : 0.965
##                  95% CI : (0.9517, 0.9755)
##     No Information Rate : 0.904
##     P-Value [Acc > NIR] : 9.645e-14
##
##                   Kappa : 0.7718
##
##  Mcnemar's Test P-Value : 4.976e-05
##
##             Sensitivity : 0.9945
##             Specificity : 0.6875
##          Pos Pred Value : 0.9677
##          Neg Pred Value : 0.9296
##              Prevalence : 0.9040
##          Detection Rate : 0.8990
##    Detection Prevalence : 0.9290
##       Balanced Accuracy : 0.8410
##
##        'Positive' Class : 0
##
```

```
confusionMatrix(validateknn, validate.df[,8])
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1347   39
##          1    9  105
##
##                Accuracy : 0.968
##                  95% CI : (0.9578, 0.9763)
##     No Information Rate : 0.904
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7967
##
##  Mcnemar's Test P-Value : 2.842e-05
##
##             Sensitivity : 0.9934
##             Specificity : 0.7292
##          Pos Pred Value : 0.9719
##          Neg Pred Value : 0.9211
##              Prevalence : 0.9040
##          Detection Rate : 0.8980
##    Detection Prevalence : 0.9240
##       Balanced Accuracy : 0.8613
##
##        'Positive' Class : 0
##
```

```
confusionMatrix(Trainsknn, t.df[,8])
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 2255   63
##          1    5  177
##
##                Accuracy : 0.9728
##                  95% CI : (0.9656, 0.9788)
##     No Information Rate : 0.904
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8243
##
##  Mcnemar's Test P-Value : 4.77e-12
##
##             Sensitivity : 0.9978
##             Specificity : 0.7375
##          Pos Pred Value : 0.9728
##          Neg Pred Value : 0.9725
##              Prevalence : 0.9040
##          Detection Rate : 0.9020
##    Detection Prevalence : 0.9272
##       Balanced Accuracy : 0.8676
##
##        'Positive' Class : 0
##
```

#Final Conclusion: The training data had improved accuracy and sensitivity.
#The above matrices were used to determine the values for the Test, Training, and Val idation sets, which are 96.3%, 97.32%, and 96.73%, respectively.
#If the Training data were more accurate than the other sets, it might be considered that overfitting would take place. When comparing the accuracy of the Training, Tes t, and Validation sets to the testing data and the validation data, we can say that w e have found the highest value of k.