Assignment_2_

MadhusudhanMasineni

2/20/2022

assignment:: k-NN for classification

This assignment describes the steps for K-NN classification in R.

We used **Universal bank** customers data includes demographic information, the customer's relationship with the bank (mortgage, securities account, etc.), and the customer response to the last personal loan campaign (Personal Loan):

Load the dataset and packages into R.

```
library("readr")
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("ggplot2")
library("caret")
```

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

```
library("tidyverse")
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v tibble  3.1.6     v stringr 1.4.0
## v tidyr   1.2.0     v forcats 0.5.1
## v purrr   0.3.4

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x purrr::lift()   masks caret::lift()
```

```
library("dummies")
```

```
## dummies-1.5.6 provided by Decision Patterns
```

```
library('FNN')
library("dplyr")


rm(list=ls())
uniBank <- read.csv("C:\\Users\\madhu\\OneDrive\\Desktop\\MS\\1stSem\\2.Fundamentals of ML\\Modules\\Mod

uniBank$Education = as.factor(uniBank$Education) # store categorized data levels
uniBank_dummy = dummy.data.frame(select(uniBank,c(-ID,-ZIP.Code))) #remove zip, id
```

```
## Warning in model.matrix.default(~x - 1, model.frame(~x - 1), contrasts = FALSE):
## non-list contrasts argument ignored
```

```
uniBank_dummy$Personal.Loan = as.factor(uniBank_dummy$Personal.Loan) #accept = 1, not accept = 0
uniBank_dummy$CCAvg = as.integer(uniBank_dummy$CCAvg) #uniform all Datatypes

set.seed(1234)
train_index_1 = createDataPartition(uniBank_dummy$Personal.Loan, p = .6, list = FALSE)
test_index_1 = setdiff(row.names(uniBank_dummy), train_index_1)
train_data_1 = uniBank_dummy[train_index_1,] # train
validation_data_1 = uniBank_dummy[-train_index_1,] # test

summary(train_data_1$Personal.Loan)
```

```
##    0    1
## 2712  288
```

```
summary(validation_data_1$Personal.Loan)
```

```
##    0    1
## 1808  192
```

```
new_DF = data.frame(Age = as.integer(40), Experience = as.integer(10), Income = as.integer(84), Family =

# preProcess for normalization :: change the values of numeric columns in the dataset to a common scale

normalize_values <- preProcess(train_data_1[,c(-10)], method=c("center", "scale"))
train_data_1[,c(-10)] <- predict(normalize_values, train_data_1[,c(-10)]) # Replace first two columns w

validation_data_1[,c(-10)] <- predict(normalize_values, validation_data_1[,c(-10)])
new_DF <- predict(normalize_values, new_DF)


## summary
summary(train_data_1)
```

```
##       Age             Experience            Income            Family
##  Min.   :-1.94349   Min.   :-2.013383   Min.   :-1.4282   Min.   :-1.2154
##  1st Qu.:-0.89541   1st Qu.:-0.877146   1st Qu.:-0.7560   1st Qu.:-1.2154
##  Median :-0.02201   Median :-0.003117   Median :-0.2140   Median :-0.3429
##  Mean   : 0.00000   Mean   : 0.000000   Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.85139   3rd Qu.: 0.870911   3rd Qu.: 0.5449   3rd Qu.: 0.5296
##  Max.   : 1.89947   Max.   : 2.007149   Max.   : 3.2553   Max.   : 1.4021
##      CCAvg           Education1        Education2        Education3
##  Min.   :-0.8608   Min.   :-0.8543   Min.   :-0.6287   Min.   :-0.6462
##  1st Qu.:-0.8608   1st Qu.:-0.8543   1st Qu.:-0.6287   1st Qu.:-0.6462
##  Median :-0.2871   Median :-0.8543   Median :-0.6287   Median :-0.6462
##  Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.2867   3rd Qu.: 1.1701   3rd Qu.: 1.5901   3rd Qu.: 1.5469
##  Max.   : 4.8768   Max.   : 1.1701   Max.   : 1.5901   Max.   : 1.5469
##     Mortgage       Personal.Loan Securities.Account   CD.Account
##  Min.   :-0.5612   0:2712        Min.   :-0.3314    Min.   :-0.2504
##  1st Qu.:-0.5612   1: 288        1st Qu.:-0.3314    1st Qu.:-0.2504
##  Median :-0.5612                 Median :-0.3314    Median :-0.2504
##  Mean   : 0.0000                 Mean   : 0.0000    Mean   : 0.0000
##  3rd Qu.: 0.4401                 3rd Qu.:-0.3314    3rd Qu.:-0.2504
##  Max.   : 5.4463                 Max.   : 3.0163    Max.   : 3.9930
##      Online          CreditCard
##  Min.   :-1.2119   Min.   :-0.639
##  1st Qu.:-1.2119   1st Qu.:-0.639
##  Median : 0.8249   Median :-0.639
##  Mean   : 0.0000   Mean   : 0.000
##  3rd Qu.: 0.8249   3rd Qu.: 1.564
##  Max.   : 0.8249   Max.   : 1.564
```

```
summary(validation_data_1)
```

```
##       Age             Experience           Income              Family
##  Min.   :-1.94349   Min.   :-2.01338   Min.   :-1.428210   Min.   :-1.215375
##  1st Qu.:-0.89541   1st Qu.:-0.87715   1st Qu.:-0.756047   1st Qu.:-1.215375
##  Median : 0.06533   Median : 0.08429   Median :-0.213979   Median :-0.342888
##  Mean   : 0.01887   Mean   : 0.01506   Mean   :-0.005121   Mean   : 0.007416
##  3rd Qu.: 0.85139   3rd Qu.: 0.87091   3rd Qu.: 0.523232   3rd Qu.: 0.529600
##  Max.   : 1.89947   Max.   : 2.00715   Max.   : 3.125156   Max.   : 1.402087
##      CCAvg            Education1         Education2         Education3
##  Min.   :-0.86083   Min.   :-0.85432   Min.   :-0.62866   Min.   :-0.64624
##  1st Qu.:-0.86083   1st Qu.:-0.85432   1st Qu.:-0.62866   1st Qu.:-0.64624
##  Median :-0.28707   Median :-0.85432   Median :-0.62866   Median :-0.64624
##  Mean   : 0.01932   Mean   :-0.01417   Mean   :-0.01516   Mean   : 0.03034
##  3rd Qu.: 0.28669   3rd Qu.: 1.17013   3rd Qu.: 1.59015   3rd Qu.: 1.54689
##  Max.   : 4.87676   Max.   : 1.17013   Max.   : 1.59015   Max.   : 1.54689
##     Mortgage        Personal.Loan Securities.Account   CD.Account
##  Min.   :-0.56117   0:1808        Min.   :-0.33142   Min.   :-0.25036
##  1st Qu.:-0.56117   1: 192        1st Qu.:-0.33142   1st Qu.:-0.25036
##  Median :-0.56117                 Median :-0.33142   Median :-0.25036
##  Mean   :-0.01642                 Mean   : 0.04519   Mean   : 0.01485
##  3rd Qu.: 0.40327                 3rd Qu.:-0.33142   3rd Qu.:-0.25036
##  Max.   : 5.67207                 Max.   : 3.01629   Max.   : 3.99297
##      Online            CreditCard
##  Min.   :-1.211877   Min.   :-0.63899
##  1st Qu.:-1.211877   1st Qu.:-0.63899
##  Median : 0.824891   Median :-0.63899
##  Mean   : 0.009166   Mean   : 0.02203
##  3rd Qu.: 0.824891   3rd Qu.: 1.56444
##  Max.   : 0.824891   Max.   : 1.56444
```

```
## knn

knn_1 <- knn(train = train_data_1[,c(-10)], test = new_DF,
                cl = train_data_1[,10],
                k = 5, prob=TRUE) # suggested cutoff .5
knn_attributes <- attributes(knn_1)
knn_attributes[1]
```

```
## $levels
## [1] "0"
```

#here levels 0 # all 5 nearest neighbors will classified as a 0, in turn the customer will be classified as a 0.

```
knn_attributes[3]
```

```
## $prob
## [1] 1
```

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

```
accuracy_DF <- data.frame(k = seq(1, 14, 1), accuracy = rep(0, 14))

for(i in 1:14) {
  knn_2 <- knn(train = train_data_1[,-10],test = validation_data_1[,-10], cl = train_data_1[,10], k=i,
  accuracy_DF[i, 2] <- confusionMatrix(knn_2, validation_data_1[,10])$overall[1]
}
accuracy_DF
```

```
##     k accuracy
## 1   1   0.9590
## 2   2   0.9515
## 3   3   0.9590
## 4   4   0.9480
## 5   5   0.9560
## 6   6   0.9530
## 7   7   0.9555
## 8   8   0.9495
## 9   9   0.9520
## 10 10   0.9470
## 11 11   0.9500
## 12 12   0.9470
## 13 13   0.9485
## 14 14   0.9460
```

best choice of k which also balances the model from overfitting is k = 3

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

confusion matrix

```
knn_3 <- knn(train = train_data_1[,-10],test = validation_data_1[,-10], cl = train_data_1[,10], k=3, pr
confusionMatrix(knn_3, validation_data_1[,10])
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1802   76
##          1    6  116
##
##                Accuracy : 0.959
##                  95% CI : (0.9494, 0.9673)
##     No Information Rate : 0.904
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7178
##
##  Mcnemar's Test P-Value : 2.541e-14
##
##             Sensitivity : 0.9967
##             Specificity : 0.6042
##          Pos Pred Value : 0.9595
##          Neg Pred Value : 0.9508
##              Prevalence : 0.9040
##          Detection Rate : 0.9010
##    Detection Prevalence : 0.9390
##       Balanced Accuracy : 0.8004
##
##        '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 Credit Card = 1. Classify the customer using the best k

```
customer_DF= data.frame(Age = 40, Experience = 10, Income = 84, Family = 2, CCAvg = 2, Education_1 = 0,
knn_4 <- knn(train = train_data_1[,-10],test = customer_DF, cl = train_data_1[,10], k=3, prob=TRUE)
knn_4
```

```
## [1] 1
## attr(,"prob")
## [1] 1
## attr(,"nn.index")
##      [,1] [,2] [,3]
## [1,]  563  414 2582
## attr(,"nn.dist")
##          [,1]     [,2]     [,3]
## [1,] 90.54673 90.57235 90.59887
## Levels: 1
```

customer classified as 1 with 100% probability, for k=3

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.

```
uniBank_dummy = dummy.data.frame(select(uniBank,c(-ID,-ZIP.Code)))
```

```
## Warning in model.matrix.default(~x - 1, model.frame(~x - 1), contrasts = FALSE):
## non-list contrasts argument ignored
```

```
uniBank_dummy$Personal.Loan = as.factor(uniBank_dummy$Personal.Loan)
uniBank_dummy$CCAvg = as.integer(uniBank_dummy$CCAvg)

set.seed(1234)
train_index_1 = createDataPartition(uniBank_dummy$Personal.Loan, p = .5, list = FALSE)
validation_index_1 =  sample(setdiff(rownames(uniBank_dummy),train_index_1), 0.3*dim(uniBank_dummy)[1])
test_index_1 = setdiff(row.names(uniBank_dummy), union(train_index_1,validation_index_1))

train_DF = uniBank_dummy[train_index_1,] # train
validation_DF = uniBank_dummy[validation_index_1,] # validation
```

```
test_DF = uniBank_dummy[test_index_1,] #test

summary(train_DF)
```

```
##       Age          Experience       Income           Family     
##  Min.   :23.00   Min.   :-3.00   Min.   :  8.00   Min.   :1.000  
##  1st Qu.:35.00   1st Qu.:10.00   1st Qu.: 39.00   1st Qu.:1.000  
##  Median :45.00   Median :20.00   Median : 64.50   Median :2.000  
##  Mean   :45.06   Mean   :19.85   Mean   : 74.11   Mean   :2.404  
##  3rd Qu.:55.00   3rd Qu.:29.00   3rd Qu.: 99.00   3rd Qu.:3.000  
##  Max.   :67.00   Max.   :43.00   Max.   :224.00   Max.   :4.000  
##      CCAvg          Education1       Education2       Education3  
##  Min.   : 0.000   Min.   :0.0000   Min.   :0.0000   Min.   :0.000  
##  1st Qu.: 0.000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.000  
##  Median : 1.000   Median :0.0000   Median :0.0000   Median :0.000  
##  Mean   : 1.479   Mean   :0.4224   Mean   :0.2816   Mean   :0.296  
##  3rd Qu.: 2.000   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.000  
##  Max.   :10.000   Max.   :1.0000   Max.   :1.0000   Max.   :1.000  
##     Mortgage      Personal.Loan Securities.Account   CD.Account   
##  Min.   :  0.00   0:2260        Min.   :0.0000     Min.   :0.000  
##  1st Qu.:  0.00   1: 240        1st Qu.:0.0000     1st Qu.:0.000  
##  Median :  0.00                 Median :0.0000     Median :0.000  
##  Mean   : 56.44                 Mean   :0.1044     Mean   :0.064  
##  3rd Qu.:102.00                 3rd Qu.:0.0000     3rd Qu.:0.000  
##  Max.   :612.00                 Max.   :1.0000     Max.   :1.000  
##      Online          CreditCard    
##  Min.   :0.0000   Min.   :0.0000  
##  1st Qu.:0.0000   1st Qu.:0.0000  
##  Median :1.0000   Median :0.0000  
##  Mean   :0.5988   Mean   :0.2916  
##  3rd Qu.:1.0000   3rd Qu.:1.0000  
##  Max.   :1.0000   Max.   :1.0000
```

```
summary(validation_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.: 38.00   1st Qu.:1.000  
##  Median :46.00   Median :21.00   Median : 63.00   Median :2.000  
##  Mean   :45.63   Mean   :20.35   Mean   : 73.24   Mean   :2.377  
##  3rd Qu.:56.00   3rd Qu.:30.00   3rd Qu.: 95.00   3rd Qu.:3.000  
##  Max.   :67.00   Max.   :42.00   Max.   :218.00   Max.   :4.000  
##      CCAvg         Education1       Education2       Education3  
##  Min.   :0.000   Min.   :0.0000   Min.   :0.0000   Min.   :0.000  
##  1st Qu.:0.000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.000  
##  Median :1.000   Median :0.0000   Median :0.0000   Median :0.000  
##  Mean   :1.543   Mean   :0.4247   Mean   :0.2733   Mean   :0.302  
##  3rd Qu.:2.000   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.000  
##  Max.   :9.000   Max.   :1.0000   Max.   :1.0000   Max.   :1.000  
##     Mortgage      Personal.Loan Securities.Account   CD.Account     
##  Min.   :  0.00   0:1371        Min.   :0.0000     Min.   :0.00000  
##  1st Qu.:  0.00   1: 129        1st Qu.:0.0000     1st Qu.:0.00000  
##  Median :  0.00                  Median :0.0000     Median :0.00000
##  Mean   : 58.18                  Mean   :0.1067     Mean   :0.05733
##  3rd Qu.:104.00                  3rd Qu.:0.0000     3rd Qu.:0.00000
##  Max.   :635.00                  Max.   :1.0000     Max.   :1.00000
##      Online         CreditCard
##  Min.   :0.0000   Min.   :0.000
##  1st Qu.:0.0000   1st Qu.:0.000
##  Median :1.0000   Median :0.000
##  Mean   :0.5973   Mean   :0.308
##  3rd Qu.:1.0000   3rd Qu.:1.000
##  Max.   :1.0000   Max.   :1.000
```

```r
summary(test_DF)
```

```
##       Age          Experience        Income          Family
##  Min.   :23.00   Min.   :-2.00   Min.   :  8.00   Min.   :1.000
##  1st Qu.:35.00   1st Qu.:10.00   1st Qu.: 38.00   1st Qu.:1.000
##  Median :46.00   Median :21.00   Median : 62.00   Median :2.000
##  Mean   :45.61   Mean   :20.37   Mean   : 73.75   Mean   :2.408
##  3rd Qu.:55.00   3rd Qu.:30.00   3rd Qu.: 99.25   3rd Qu.:3.000
##  Max.   :67.00   Max.   :43.00   Max.   :204.00   Max.   :4.000
##      CCAvg          Education1      Education2      Education3
##  Min.   : 0.000   Min.   :0.000   Min.   :0.000   Min.   :0.000
##  1st Qu.: 0.000   1st Qu.:0.000   1st Qu.:0.000   1st Qu.:0.000
##  Median : 1.000   Median :0.000   Median :0.000   Median :0.000
##  Mean   : 1.558   Mean   :0.403   Mean   :0.289   Mean   :0.308
##  3rd Qu.: 2.000   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   0:889          Min.   :0.000      Min.   :0.000
##  1st Qu.:  0.00   1:111          1st Qu.:0.000      1st Qu.:0.000
##  Median :  0.00                  Median :0.000      Median :0.000
##  Mean   : 54.12                  Mean   :0.101      Mean   :0.056
##  3rd Qu.: 94.00                  3rd Qu.:0.000      3rd Qu.:0.000
##  Max.   :617.00                  Max.   :1.000      Max.   :1.000
##      Online        CreditCard
##  Min.   :0.000   Min.   :0.000
##  1st Qu.:0.000   1st Qu.:0.000
##  Median :1.000   Median :0.000
##  Mean   :0.591   Mean   :0.279
##  3rd Qu.:1.000   3rd Qu.:1.000
##  Max.   :1.000   Max.   :1.000
```

```r
norm_values <- preProcess(train_DF[,c(-10)],method = c("center","scale"))
train_DF[,c(-10)] <- predict(norm_values, train_DF[,c(-10)])
validation_DF[,c(-10)] <- predict(norm_values, validation_DF[,c(-10)])
test_DF[,c(-10)] <- predict(norm_values, test_DF[,c(-10)])

test_knn <- knn(train = train_DF[,c(-10)], test = test_DF[,c(-10)],
                cl=train_DF[,10], k=3, prob=TRUE)

confusionMatrix(test_knn, test_DF[,10])
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 883  43
##          1   6  68
##
##                Accuracy : 0.951
##                  95% CI : (0.9357, 0.9635)
##     No Information Rate : 0.889
##     P-Value [Acc > NIR] : 3.545e-12
##
##                   Kappa : 0.7093
##
##  Mcnemar’s Test P-Value : 2.706e-07
##
##             Sensitivity : 0.9933
##             Specificity : 0.6126
##          Pos Pred Value : 0.9536
##          Neg Pred Value : 0.9189
##              Prevalence : 0.8890
##          Detection Rate : 0.8830
##    Detection Prevalence : 0.9260
##       Balanced Accuracy : 0.8029
##
##        ’Positive’ Class : 0
##
```

```
validation_knn <- knn(train = train_DF[,-c(10)],test = validation_DF[,-c(10)], cl = train_DF[,10], k=3,

confusionMatrix(validation_knn, validation_DF[,10])
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1366   48
##          1    5   81
##
##                Accuracy : 0.9647
##                  95% CI : (0.954, 0.9734)
##     No Information Rate : 0.914
##     P-Value [Acc > NIR] : 3.361e-15
##
##                   Kappa : 0.7353
##
##  Mcnemar’s Test P-Value : 7.968e-09
##
##             Sensitivity : 0.9964
##             Specificity : 0.6279
##          Pos Pred Value : 0.9661
##          Neg Pred Value : 0.9419
##              Prevalence : 0.9140
##          Detection Rate : 0.9107
##    Detection Prevalence : 0.9427
##       Balanced Accuracy : 0.8121
##
##        'Positive' Class : 0
##
```

```
train_knn <- knn(train = train_DF[,-c(10)],test = train_DF[,-c(10)], cl = train_DF[,10], k=3, prob=TRUE

confusionMatrix(train_knn, train_DF[,10])
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 2250   55
##          1   10  185
##
##                Accuracy : 0.974
##                  95% CI : (0.967, 0.9799)
##     No Information Rate : 0.904
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8365
##
##  Mcnemar's Test P-Value : 4.828e-08
##
##             Sensitivity : 0.9956
##             Specificity : 0.7708
##          Pos Pred Value : 0.9761
##          Neg Pred Value : 0.9487
##              Prevalence : 0.9040
##          Detection Rate : 0.9000
##    Detection Prevalence : 0.9220
##       Balanced Accuracy : 0.8832
##
##        'Positive' Class : 0
##
```

Test Accuracy : 0.965 Valid Accuracy: 0.956 Train Accuracy: 0.9748

The model is being fit on the training data, we say that the classifications are most accurate on the training data set and least accurate on the test datasets.