FML ASSG 2- 811250075

THALLURU PUSHPITHA

2023-10-01

#summary

The assignment's objective is to predict, using KNN(k-Nearest Neighbors)Classification, if the loan offer will be accepted by consumers of Universal Bank. The data set contains demographic information about the customers as well as other silent-related information. Following the reading of the data set and the installation of the required libraries, extra columns are removed, category categories are changed to dummy variables, and the data is eventually normalized. Following that, the data set was divided into two sets: training and validation, each of which contained 60% and 40% of the entire data. A new consumer was categorized as either accepting or rejecting a loan offer using k-NN with k=1.By assessing accuracy on the validation set, the optimal k value—which strikes a balance between over fitting and under fitting—was found, with k=1 being the result.

```
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')
library('gmodels')
library('FNN')
```

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

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

```
library("ggplot2")
```

Import dataset UniversalBank.csv

```
UniversalBank <- read.csv("C:\\Users\\pushp\\OneDrive\\Desktop\\FML ASSIG\\FML\\UniversalBank.csv")
```

#Displaying column names

```
colnames(UniversalBank)
```

```
##  [1] "ID"                "Age"               "Experience"
##  [4] "Income"            "ZIP.Code"          "Family"
##  [7] "CCAvg"             "Education"         "Mortgage"
## [10] "Personal.Loan"     "Securities.Account" "CD.Account"
## [13] "Online"            "CreditCard"
```

Summary of UniversalBank dataset

```
summary(UniversalBank)
```

```
##        ID              Age           Experience       Income          ZIP.Code
##  Min.   :   1   Min.   :23.00   Min.   :-3.0   Min.   :  8.00   Min.   : 9307
##  1st Qu.:1251   1st Qu.:35.00   1st Qu.:10.0   1st Qu.: 39.00   1st Qu.:91911
##  Median :2500   Median :45.00   Median :20.0   Median : 64.00   Median :93437
##  Mean   :2500   Mean   :45.34   Mean   :20.1   Mean   : 73.77   Mean   :93153
##  3rd Qu.:3750   3rd Qu.:55.00   3rd Qu.:30.0   3rd Qu.: 98.00   3rd Qu.:94608
##  Max.   :5000   Max.   :67.00   Max.   :43.0   Max.   :224.00   Max.   :96651
##      Family          CCAvg          Education        Mortgage
##  Min.   :1.000   Min.   : 0.000   Min.   :1.000   Min.   :  0.0
##  1st Qu.:1.000   1st Qu.: 0.700   1st Qu.:1.000   1st Qu.:  0.0
##  Median :2.000   Median : 1.500   Median :2.000   Median :  0.0
##  Mean   :2.396   Mean   : 1.938   Mean   :1.881   Mean   : 56.5
##  3rd Qu.:3.000   3rd Qu.: 2.500   3rd Qu.:3.000   3rd Qu.:101.0
##  Max.   :4.000   Max.   :10.000   Max.   :3.000   Max.   :635.0
##  Personal.Loan   Securities.Account   CD.Account         Online
##  Min.   :0.000   Min.   :0.0000     Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.000   1st Qu.:0.0000     1st Qu.:0.0000   1st Qu.:0.0000
##  Median :0.000   Median :0.0000     Median :0.0000   Median :1.0000
##  Mean   :0.096   Mean   :0.1044     Mean   :0.0604   Mean   :0.5968
##  3rd Qu.:0.000   3rd Qu.:0.0000     3rd Qu.:0.0000   3rd Qu.:1.0000
##  Max.   :1.000   Max.   :1.0000     Max.   :1.0000   Max.   :1.0000
##    CreditCard
##  Min.   :0.000
```

```
##  1st Qu.:0.000
##  Median :0.000
##  Mean   :0.294
##  3rd Qu.:1.000
##  Max.   :1.000
```

Making columns ID and ZIP.Code as NULL

```
UniversalBank$ID <- NULL
UniversalBank$ZIP.Code <- NULL
summary(UniversalBank)
```

```
##       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
```

Making the Personal Loan column as factor

```
UniversalBank$Personal.Loan =  as.factor(UniversalBank$Personal.Loan)
```

Normalization

```
Normal_Data <- preProcess(UniversalBank,method = "range")
UniversalBank_Norm <- predict(Normal_Data,UniversalBank)
summary(UniversalBank_Norm)
```

```
##       Age             Experience         Income           Family
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.2727   1st Qu.:0.2826   1st Qu.:0.1435   1st Qu.:0.0000
##  Median :0.5000   Median :0.5000   Median :0.2593   Median :0.3333
##  Mean   :0.5077   Mean   :0.5023   Mean   :0.3045   Mean   :0.4655
##  3rd Qu.:0.7273   3rd Qu.:0.7174   3rd Qu.:0.4167   3rd Qu.:0.6667
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##      CCAvg          Education         Mortgage       Personal.Loan
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.00000   0:4520
##  1st Qu.:0.0700   1st Qu.:0.0000   1st Qu.:0.00000   1: 480
##  Median :0.1500   Median :0.5000   Median :0.00000
##  Mean   :0.1938   Mean   :0.4405   Mean   :0.08897
##  3rd Qu.:0.2500   3rd Qu.:1.0000   3rd Qu.:0.15906
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.00000
##  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
```

Partition the data into training 60% and validation 40% sets

```
Train_index <- createDataPartition(UniversalBank$Personal.Loan, p = 0.6,
list = FALSE)
train.df = UniversalBank_Norm[Train_index,]
validation.df = UniversalBank_Norm[-Train_index,]
```

Classifying the customer as per the date provided

```
To_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(To_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
```

```
Prediction <- knn(train = train.df[,1:7],test = To_Predict[,1:7],
                  cl = train.df$Personal.Loan, k = 1)
print(Prediction)
```

```
## [1] 1
## attr(,"nn.index")
```

```
##      [,1]
## [1,]  554
## attr(,"nn.dist")
##         [,1]
## [1,] 92.3269
## Levels: 1
```

Customer is classified as 1.

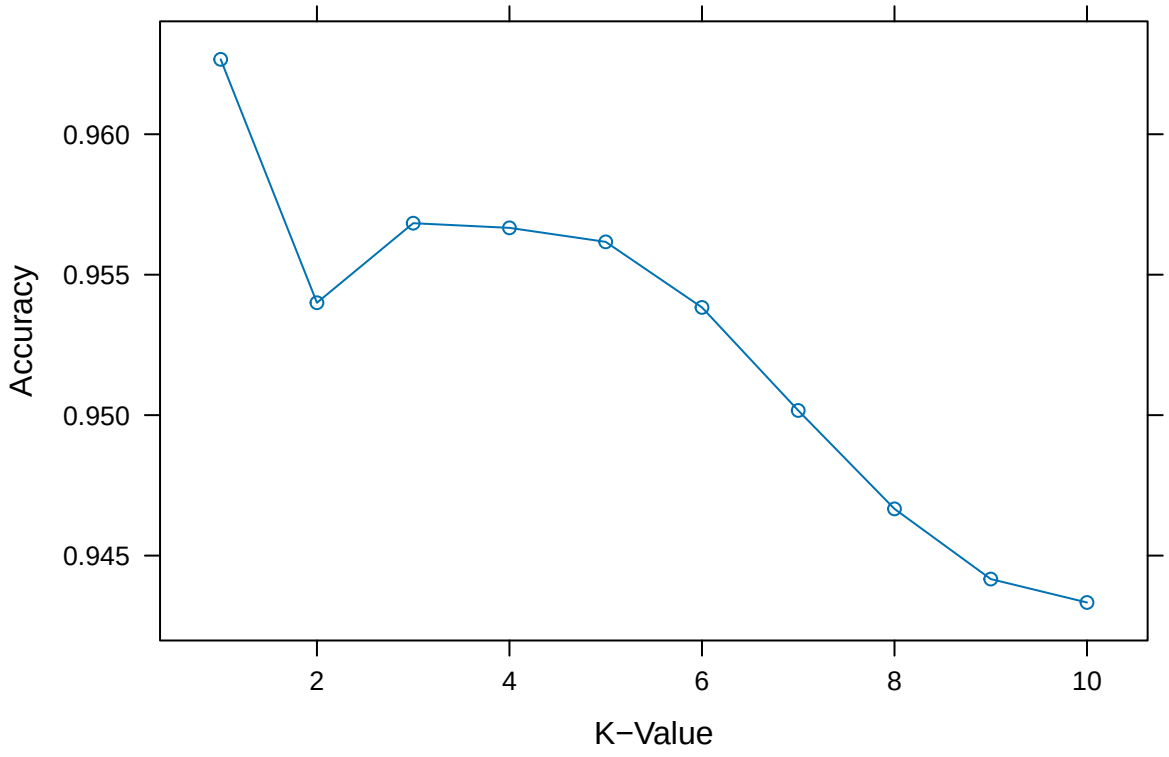
2) What is a choice of k that balances between overfitting and ignoring the predictor information?

```
set.seed(2808)
UniversalBank_control <- trainControl(method= "repeatedcv", number = 5, repeats = 2)
searchGrid =  expand.grid(k=1:10)
knn.model = train(Personal.Loan~., data = train.df, method = 'knn', tuneGrid = searchGrid,  trControl =
knn.model
```

```
## k-Nearest Neighbors
##
## 3000 samples
##   11 predictor
##    2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold, repeated 2 times)
## Summary of sample sizes: 2400, 2400, 2399, 2400, 2401, 2400, ...
## Resampling results across tuning parameters:
##
##   k   Accuracy   Kappa
##    1  0.9626663  0.7597547
##    2  0.9540013  0.6963854
##    3  0.9568336  0.7067528
##    4  0.9566669  0.7017725
##    5  0.9561688  0.6915175
##    6  0.9538344  0.6730514
##    7  0.9501663  0.6393246
##    8  0.9466635  0.6120402
##    9  0.9441649  0.5847765
##   10  0.9433316  0.5760437
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 1.
```

The choice of K that balances between overfitting and ignoring predictor is K=1

```
plot(knn.model, type = "b", xlab = "K-Value", ylab = "Accuracy")
```



#finding the best K

```
best_k <- knn.model$bestTune[[1]]
best_k
```

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

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

```
predictions <- predict(knn.model,validation.df)
confusionMatrix(predictions,validation.df$Personal.Loan)
```

```
## Confusion Matrix and Statistics
##
```

```
##           Reference
## Prediction    0    1
##          0 1794   64
##          1   14  128
##
##                Accuracy : 0.961
##                  95% CI : (0.9516, 0.9691)
##     No Information Rate : 0.904
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7457
##
##  Mcnemar's Test P-Value : 2.887e-08
##
##             Sensitivity : 0.9923
##             Specificity : 0.6667
##          Pos Pred Value : 0.9656
##          Neg Pred Value : 0.9014
##              Prevalence : 0.9040
##          Detection Rate : 0.8970
##    Detection Prevalence : 0.9290
##       Balanced Accuracy : 0.8295
##
##        'Positive' Class : 0
##
```

4) Classify the customer using the best k

```
To_Predict_Normaliz = 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)
To_Predict_Normaliz = predict(Normal_Data, To_Predict)
predict(knn.model, To_Predict_Normaliz)
```

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

5) Repartition the data into 50% for training ,30% for validation, 20% for test

```
train_size = 0.5
Train_index = createDataPartition(UniversalBank$Personal.Loan, p = 0.5,
list = FALSE)
train.df = UniversalBank_Norm[Train_index,]
test_size = 0.2
Test_index = createDataPartition(UniversalBank$Personal.Loan, p = 0.2,
list = FALSE)
Test.df = UniversalBank_Norm[Test_index,]
```

```
valid_size = 0.3
Validation_index = createDataPartition(UniversalBank$Personal.Loan, p = 0.3,
list = FALSE)
validation.df = UniversalBank_Norm[Validation_index,]
Testingknn <- knn(train = train.df[,-8], test = Test.df[,-8], cl = train.df[,8],
k =3)
Validationknn <- knn(train = train.df[,-8],
                     test = validation.df[,-8], cl = train.df[,8], k =3)
Trainingknn <- knn(train = train.df[,-8],
                   test = train.df[,-8], cl = train.df[,8], k =3)
```

Comparing the confusion matrix of the test set with the training and validation sets.

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

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 900  39
##          1   4  57
##
##                Accuracy : 0.957
##                  95% CI : (0.9425, 0.9687)
##     No Information Rate : 0.904
##     P-Value [Acc > NIR] : 2.214e-10
##
##                   Kappa : 0.704
##
##  Mcnemar's Test P-Value : 2.161e-07
##
##             Sensitivity : 0.9956
##             Specificity : 0.5938
##          Pos Pred Value : 0.9585
##          Neg Pred Value : 0.9344
##              Prevalence : 0.9040
##          Detection Rate : 0.9000
##    Detection Prevalence : 0.9390
##       Balanced Accuracy : 0.7947
##
##        'Positive' Class : 0
##
```

```
confusionMatrix(Trainingknn, train.df[,8])
```

```
## Confusion Matrix and Statistics
##
##           Reference
```

```
## Prediction    0    1
##          0 2254   59
##          1    6  181
##
##                Accuracy : 0.974
##                  95% CI : (0.967, 0.9799)
##     No Information Rate : 0.904
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8338
##
##  Mcnemar's Test P-Value : 1.12e-10
##
##             Sensitivity : 0.9973
##             Specificity : 0.7542
##          Pos Pred Value : 0.9745
##          Neg Pred Value : 0.9679
##              Prevalence : 0.9040
##          Detection Rate : 0.9016
##    Detection Prevalence : 0.9252
##       Balanced Accuracy : 0.8758
##
##        'Positive' Class : 0
##
```

```
confusionMatrix(Validationknn, validation.df[,8])
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1351   49
##          1    5   95
##
##                Accuracy : 0.964
##                  95% CI : (0.9533, 0.9728)
##     No Information Rate : 0.904
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7598
##
##  Mcnemar's Test P-Value : 4.87e-09
##
##             Sensitivity : 0.9963
##             Specificity : 0.6597
##          Pos Pred Value : 0.9650
##          Neg Pred Value : 0.9500
##              Prevalence : 0.9040
##          Detection Rate : 0.9007
##    Detection Prevalence : 0.9333
##       Balanced Accuracy : 0.8280
##
##        'Positive' Class : 0
##
```