Machine Learning Assignment 2

Madhukar Heerekar

2022-03-17

```
UB <- read.csv("c://users//heere//OneDrive//Desktop//FML ASSIGNMENT 2//universalbank .csv")
summary(UB)
```

```
##        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
```

```
library(caret)
```

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

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

```
UB$Personal.Loan<- as.factor(UB$Personal.Loan)
library(dummies)
```

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

```
dummy_model<-dummyVars(~Education, data=UB)
head(predict(dummy_model,UB))
```

```
##   Education
## 1         1
## 2         1
## 3         1
## 4         2
## 5         2
## 6         2
```

```
UB_dummy <-dummy.data.frame(UB, names =c("Education"), sep="-")
```

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

```
UBank<-subset(UB_dummy,select = -c(1, 5))
head(UBank)
```

```
##   Age Experience Income Family CCAvg Education-1 Education-2 Education-3
## 1  25          1     49      4   1.6           1           0           0
## 2  45         19     34      3   1.5           1           0           0
## 3  39         15     11      1   1.0           1           0           0
## 4  35          9    100      1   2.7           0           1           0
## 5  35          8     45      4   1.0           0           1           0
## 6  37         13     29      4   0.4           0           1           0
##   Mortgage Personal.Loan Securities.Account CD.Account Online CreditCard
## 1        0             0                  1          0      0          0
## 2        0             0                  1          0      0          0
## 3        0             0                  0          0      0          0
## 4        0             0                  0          0      0          0
## 5        0             0                  0          0      0          1
## 6      155             0                  0          0      1          0
```

```
##Splitting the Data into Test and validation
set.seed(15)
Train_Index <-createDataPartition(UBank$Personal.Loan,p=0.6, list = FALSE)
```

```
#use 60% for training and the remaining for validation
Train <-UBank[Train_Index,]
Valid <- UBank[-Train_Index,]
train.norm.df<-Train
valid.norm.df<-Valid
```

```
##Normalising the Data
norm.values<-preProcess(Train[,-10], method = c("range"))
train.norm.df[,-10] <- predict(norm.values,Train[,-10])
valid.norm.df[,-10]<-predict(norm.values,Valid[,-10])
```

```
##Modelling using K=1
library(FNN)
```

```
## Warning: package 'FNN' was built under R version 4.1.3
```

```
nn <- knn(train = train.norm.df[, -10], test = valid.norm.df[, -10],
          cl = train.norm.df[, 10], k = 1, prob=TRUE)
```

```
head(nn)
```

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

```
##value of k that provides the best performance
library(caret)
accuracy.df <-data.frame(k= seq(1,14,1), accuracy = rep(0,14))
for(i in 1:14) {
                  knn <- knn(train.norm.df[, -10], valid.norm.df[, -10], cl = train.norm.df[, 10], k = i
                  accuracy.df[i, 2] <- confusionMatrix(knn, valid.norm.df[, 10])$overall[1]
                }
accuracy.df
```

```
##     k accuracy
## 1   1   0.9515
## 2   2   0.9525
## 3   3   0.9560
## 4   4   0.9505
## 5   5   0.9555
## 6   6   0.9470
## 7   7   0.9495
## 8   8   0.9415
## 9   9   0.9445
## 10 10   0.9425
## 11 11   0.9420
## 12 12   0.9385
## 13 13   0.9380
## 14 14   0.9310
```

```
which.max((accuracy.df$accuracy))
```

```
## [1] 3
```

```
##Test data development
L_Predictors<-UBank[,-10]
L_labels<-UBank[,10]
```

```
Test <- data.frame(40, 10, 84, 2, 2, 0, 1, 0, 0, 0, 0, 1, 1)
colnames(Test) <- colnames(L_Predictors)
Test.norm.df <- Test
head(Test.norm.df)
```

```
##   Age Experience Income Family CCAvg Education-1 Education-2 Education-3
## 1  40         10     84      2     2           0           1           0
##   Mortgage Securities.Account CD.Account Online CreditCard
## 1        0                  0          0      1          1
```

```
##combining Training and Validation set to normalise new set
Traval.norm.df <- UBank
norm.values <- preProcess(UBank[,-10], method = c("range"))
Traval.norm.df[,-10]<-predict(norm.values, UBank[,-10])
Test.norm.df<-predict(norm.values, Test)
```

```
##Predicting using k=1
nn <- knn(train = Traval.norm.df[, -10], test = Test.norm.df,
          cl = Traval.norm.df[, 10], k = 1, prob=TRUE)
```

```
##View predicted class
head(nn)
```

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

If a Customer is classified as zero, customer will not accept the loan

```
##Predicting using k=3
nn <- knn(train = Traval.norm.df[, -10], test = Test.norm.df,
cl = Traval.norm.df[, 10], k = 3, prob=TRUE)
```

```
##View predicted class
head(nn)
```

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

```
##Customer classified as zero, customer will not accept the loan
```

```
##Show the confusion matrix for the validation data that results from using the best k.
knn.valid <- knn(train.norm.df[, -10],valid.norm.df[, -10],cl=train.norm.df[, 10],k=3,prob = 0.5)
confusionMatrix(knn.valid, valid.norm.df[, 10])
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction    0    1
```

```
##          0 1794   74
##          1   14  118
##
##                Accuracy : 0.956
##                  95% CI : (0.9461, 0.9646)
##     No Information Rate : 0.904
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7053
##
##  Mcnemar's Test P-Value : 3.187e-10
##
##             Sensitivity : 0.9923
##             Specificity : 0.6146
##          Pos Pred Value : 0.9604
##          Neg Pred Value : 0.8939
##              Prevalence : 0.9040
##          Detection Rate : 0.8970
##    Detection Prevalence : 0.9340
##       Balanced Accuracy : 0.8034
##
##        'Positive' Class : 0
##
```

```
##Error types
##True Negative - 1794
##False Negative - 14
##True Positive - 118
##False Positive - 74
##Sensitivity(TPR) - TP/(TP+FN) = 118/(118+14)=0.8939
#specificity(TNR)- TN/(TN+FP) = 1794/(1794+74)=0.9603
```

```
#modelling with diff partitioning -  training, validation, and test sets (50% : 30% : 20%)
#split the data
set.seed(15)
Train_Index_2 <-createDataPartition(UBank$Personal.Loan,p=0.5, list = FALSE)
```

```
#use 50% for training and the rest for validation and test
Train_2 <-UBank[Train_Index_2,]
ValTest <- UBank[-Train_Index_2,]
Valid_Index  <- createDataPartition(ValTest$Personal.Loan,p=0.6, list = FALSE)
Valid_2 <- ValTest[Valid_Index,]
Test_2 <- ValTest[-Valid_Index,]
```

```
#copy original data
train_2.norm.df<-Train_2
valid_2.norm.df<-Valid_2
test_2.norm.df <-Test_2
```

```
#normalize data
norm.values_2<-preProcess(Train_2[,-10], method = c("center", "scale"))
train_2.norm.df[,-10] <- predict(norm.values_2,Train_2[,-10])
```

```
valid_2.norm.df[,-10]<-predict(norm.values_2,Valid_2[,-10])
test_2.norm.df[,-10]<-predict(norm.values_2,Test_2[,-10])
```

```
#Modelling using k=3 for testset
library(FNN)
nn_2 <- knn(train = train_2.norm.df[, -10], test = test_2.norm.df[, -10],
          cl = train_2.norm.df[, 10], k = 3, prob=TRUE)
```

```
#view predicted class
head(nn_2)
```

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

```
#Modelling using k=3 for validation set
nn_2_valid<- knn(train = train_2.norm.df[, -10], test = valid_2.norm.df[, -10],
          cl = train_2.norm.df[, 10], k = 3, prob=TRUE)
```

```
#view predicted class
head(nn_2_valid)
```

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

```
#compare confusion matrix for test set with validation set
confusionMatrix(nn_2, test_2.norm.df[, 10])
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 898  38
##          1   6  58
##
##                Accuracy : 0.956
##                  95% CI : (0.9414, 0.9679)
##     No Information Rate : 0.904
##     P-Value [Acc > NIR] : 5.202e-10
##
##                   Kappa : 0.7021
##
##  Mcnemar's Test P-Value : 2.962e-06
##
##             Sensitivity : 0.9934
##             Specificity : 0.6042
##          Pos Pred Value : 0.9594
##          Neg Pred Value : 0.9063
##              Prevalence : 0.9040
##          Detection Rate : 0.8980
##    Detection Prevalence : 0.9360
##       Balanced Accuracy : 0.7988
```

```
##
##        'Positive' Class : 0
##
```

```
#Accuracy for Test is 0.956
confusionMatrix(nn_2_valid, valid_2.norm.df[, 10])
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1345   58
##          1   11   86
##
##                Accuracy : 0.954
##                  95% CI : (0.9421, 0.964)
##     No Information Rate : 0.904
##     P-Value [Acc > NIR] : 3.461e-13
##
##                   Kappa : 0.6897
##
##  Mcnemar's Test P-Value : 3.064e-08
##
##             Sensitivity : 0.9919
##             Specificity : 0.5972
##          Pos Pred Value : 0.9587
##          Neg Pred Value : 0.8866
##              Prevalence : 0.9040
##          Detection Rate : 0.8967
##    Detection Prevalence : 0.9353
##       Balanced Accuracy : 0.7946
##
##        'Positive' Class : 0
##
```

#Accuracy for validation is 0.954 & test set is 0.956.From the above, comparing confusion matrix of the test set with that of the training and validation sets we can determine that a slightly higher training set means that there is no over fitting of data and found the #better value of k.