

Assignment-2

Pratheek Sreerangam

2022-10-05

Required libraries.

```
library(caret)
```

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

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

```
library(class)
```

```
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(ggplot2)
```

```
library(fastDummies)
```

```
library(FNN)
```

```
##
```

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

```
## The following objects are masked from 'package:class':
```

```
##
```

```
## knn, knn.cv
```

Added client information and converted the categorical information to elements.

```
getwd()
```

```
## [1] "C:/Users/prath/OneDrive/Documents/ML"
```

```
setwd("C:/Users/prath/OneDrive/Documents/ML")
BankInfo <- read.csv("UniversalBank.csv")
BankInfo$Personal.Loan<-factor(BankInfo$Personal.Loan,levels=c('0','1'),labels=c('No','Yes'))
summary(BankInfo)
```

```
##      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      Personal.Loan
## Min.   :1.000    Min.   : 0.000    Min.   :1.000    Min.   : 0.0    No :4520
## 1st Qu.:1.000    1st Qu.: 0.700    1st Qu.:1.000    1st Qu.: 0.0    Yes: 480
## 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
## 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
```

Data Selection

##used relevant details to divide the collection into training (60%) and validation (40%)..

```
dummy_BankInfo <- dummy_columns(BankInfo, select_columns = 'Education')
m_BankInfo <- select(dummy_BankInfo, Age, Experience, Income, Family, CCAvg, Education_1, Education_2, Education_3)
m_BankInfo <- m_BankInfo %>% relocate(Personal.Loan, .after=last_col())#Personal loan should be placed t
set.seed(1)
Train_Index <- sample(row.names(m_BankInfo), .6*dim(m_BankInfo)[1])
Val_Index <- setdiff(row.names(m_BankInfo), Train_Index)
Train_Data <- m_BankInfo[Train_Index,]
Validation_Data <- m_BankInfo[Val_Index,]
summary(Train_Data)
```

```
##      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.: 39.00    1st Qu.:1.000
## Median :45.00    Median :20.00    Median : 63.00    Median :2.000
## Mean   :45.43    Mean   :20.19    Mean   : 73.08    Mean   :2.388
```

```
## 3rd Qu.:55.00 3rd Qu.:30.00 3rd Qu.: 98.00 3rd Qu.:3.000
## Max. :67.00 Max. :43.00 Max. :224.00 Max. :4.000
## CCAvg Education_1 Education_2 Education_3
## Min. : 0.000 Min. :0.0000 Min. :0.000 Min. :0.0000
## 1st Qu.: 0.700 1st Qu.:0.0000 1st Qu.:0.000 1st Qu.:0.0000
## Median : 1.500 Median :0.0000 Median :0.000 Median :0.0000
## Mean : 1.915 Mean :0.4173 Mean :0.285 Mean :0.2977
## 3rd Qu.: 2.500 3rd Qu.:1.0000 3rd Qu.:1.000 3rd Qu.:1.0000
## Max. :10.000 Max. :1.0000 Max. :1.000 Max. :1.0000
## Mortgage Securities.Account CD.Account Online
## Min. : 0.00 Min. :0.0000 Min. :0.00000 Min. :0.0000
## 1st Qu.: 0.00 1st Qu.:0.0000 1st Qu.:0.00000 1st Qu.:0.0000
## Median : 0.00 Median :0.0000 Median :0.00000 Median :1.0000
## Mean : 57.34 Mean :0.1003 Mean :0.05367 Mean :0.5847
## 3rd Qu.:102.00 3rd Qu.:0.0000 3rd Qu.:0.00000 3rd Qu.:1.0000
## Max. :635.00 Max. :1.0000 Max. :1.00000 Max. :1.0000
## CreditCard Personal.Loan
## Min. :0.0000 No :2725
## 1st Qu.:0.0000 Yes: 275
## Median :0.0000
## Mean :0.2927
## 3rd Qu.:1.0000
## Max. :1.0000
```

Normalizing the data in numerical form.

```
columnsare <- c(1,2,3,4,5,9)
train.norm.df <- Train_Data
valid.norm.df <- Validation_Data
norm.values <- preprocess(Train_Data[,columnsare],method=c("center","scale"))
#updating the dataframes with the normalized data
train.norm.df[, columnsare] <-predict(norm.values,Train_Data[,columnsare])
valid.norm.df[, columnsare] <-predict(norm.values,Validation_Data[,columnsare])
summary(train.norm.df)
```

```
## Age Experience Income Family
## Min. :-1.97257 Min. :-2.03718 Min. :-1.4240 Min. :-1.2058
## 1st Qu.: -0.82922 1st Qu.: -0.89531 1st Qu.: -0.7457 1st Qu.: -1.2058
## Median : -0.03767 Median : -0.01695 Median : -0.2206 Median : -0.3368
## Mean : 0.00000 Mean : 0.00000 Mean : 0.0000 Mean : 0.0000
## 3rd Qu.: 0.84183 3rd Qu.: 0.86141 3rd Qu.: 0.5452 3rd Qu.: 0.5321
## Max. : 1.89723 Max. : 2.00328 Max. : 3.3022 Max. : 1.4010
## CCAvg Education_1 Education_2 Education_3
## Min. :-1.1059 Min. :0.0000 Min. :0.000 Min. :0.0000
## 1st Qu.: -0.7016 1st Qu.:0.0000 1st Qu.:0.000 1st Qu.:0.0000
## Median : -0.2396 Median :0.0000 Median :0.000 Median :0.0000
## Mean : 0.0000 Mean :0.4173 Mean :0.285 Mean :0.2977
## 3rd Qu.: 0.3380 3rd Qu.:1.0000 3rd Qu.:1.000 3rd Qu.:1.0000
## Max. : 4.6700 Max. :1.0000 Max. :1.000 Max. :1.0000
## Mortgage Securities.Account CD.Account Online
## Min. :-0.5679 Min. :0.0000 Min. :0.00000 Min. :0.0000
## 1st Qu.: -0.5679 1st Qu.:0.0000 1st Qu.:0.00000 1st Qu.:0.0000
```

```
## Median :-0.5679   Median :0.0000   Median :0.00000   Median :1.0000
## Mean  : 0.0000   Mean  :0.1003   Mean  :0.05367   Mean  :0.5847
## 3rd Qu.: 0.4423   3rd Qu.:0.0000   3rd Qu.:0.00000   3rd Qu.:1.0000
## Max.   : 5.7216   Max.   :1.0000   Max.   :1.00000   Max.   :1.0000
##   CreditCard   Personal.Loan
## Min.    :0.0000   No :2725
## 1st Qu.:0.0000   Yes: 275
## Median :0.0000
## Mean    :0.2927
## 3rd Qu.:1.0000
## Max.    :1.0000
```

constructing the K-NN model

```
train.knn.predictors <- train.norm.df[, 1:13]
train.knn.success <- train.norm.df[,14]
valid.knn.predictors <- valid.norm.df[, 1:13]
valid.knn.success <- valid.norm.df[,14]
knn.results <- knn (train=train.knn.predictors, test=valid.knn.predictors, cl=train.knn.success, k=1, p=1)
confusionMatrix(knn.results,valid.knn.success, positive="Yes")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No  Yes
##           No 1776  59
##           Yes  19 146
##
##           Accuracy : 0.961
##           95% CI : (0.9516, 0.9691)
##           No Information Rate : 0.8975
##           P-Value [Acc > NIR] : < 2.2e-16
##
##           Kappa : 0.768
##
## Mcnemar's Test P-Value : 1.006e-05
##
##           Sensitivity : 0.7122
##           Specificity : 0.9894
##           Pos Pred Value : 0.8848
##           Neg Pred Value : 0.9678
##           Prevalence : 0.1025
##           Detection Rate : 0.0730
##           Detection Prevalence : 0.0825
##           Balanced Accuracy : 0.8508
##
##           'Positive' Class : Yes
##
```

As you can see, the model is a 95.4 crat.

##A sample consumer with the following characteristics: 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.

Using our model

```
traceback()
```

```
## No traceback available
```

```
customertest = data.frame(Age = as.integer(40), Experience = as.integer(10), Income = as.integer(84), F
#load the data into a customertest dataframe.
customer.norm.df <- customertest
customer.norm.df[, columnsare]<-predict(norm.values,customertest[,columnsare])
#normalize of the quantitative values
```

Testing the KNN

```
set.seed(400)
customer.knn <- knn(train=train.knn.predictors, test=customer.norm.df,cl=train.knn.success,k=1, prob=TR
head(customer.knn)
```

```
## [1] No
## Levels: No
```

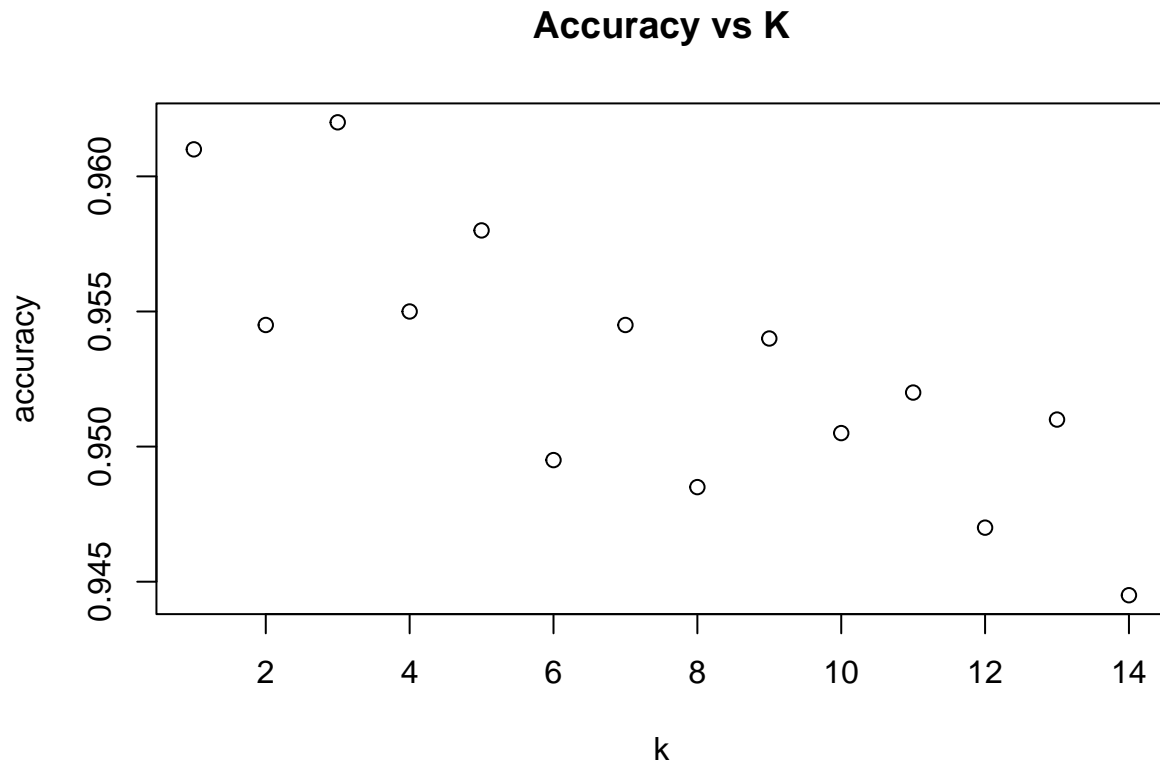
TO find the best k value.

```
accuracy.df <- data.frame(k = seq(1,14,1), accuracy = rep(0 , 14))
#Now we will make a table with all of the k and their accuracies from 1 to 14.
for(i in 1:14){
  knn.pred <- knn(train.knn.predictors,valid.knn.predictors, cl=train.knn.success,k=i)
  accuracy.df[i,2] <- confusionMatrix(knn.pred, valid.knn.success)$overall[1]
}
accuracy.df
```

```
##      k accuracy
## 1    1  0.9610
## 2    2  0.9545
## 3    3  0.9620
## 4    4  0.9550
## 5    5  0.9580
## 6    6  0.9495
## 7    7  0.9545
## 8    8  0.9485
```

```
## 9 9 0.9540
## 10 10 0.9505
## 11 11 0.9520
## 12 12 0.9470
## 13 13 0.9510
## 14 14 0.9445
```

```
plot(x=accuracy.df$k, y=accuracy.df$accuracy, main="Accuracy vs K", xlab="k", ylab="accuracy")
```



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

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

Customer the KNN.

```
customer.knn3 <- knn(train=train.knn.predictors, test=customer.norm.df, cl=train.knn.success, k=3, prob=T)
head(customer.knn3)
```

```
## [1] No
## Levels: No
```

```
## Further study for k=3
```

matrix of the validation data for k=3.

```
knn.k3 <- knn(train = train.knn.predictors, test=valid.knn.predictors, cl=train.knn.success, k=3, prob=TRUE)
confusionMatrix(knn.k3, valid.knn.success,)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No  Yes
##           No 1792  73
##           Yes   3 132
##
##           Accuracy : 0.962
##           95% CI : (0.9527, 0.9699)
##           No Information Rate : 0.8975
##           P-Value [Acc > NIR] : < 2.2e-16
##
##           Kappa : 0.7567
##
##           Mcnemar's Test P-Value : 2.476e-15
##
##           Sensitivity : 0.9983
##           Specificity : 0.6439
##           Pos Pred Value : 0.9609
##           Neg Pred Value : 0.9778
##           Prevalence : 0.8975
##           Detection Rate : 0.8960
##           Detection Prevalence : 0.9325
##           Balanced Accuracy : 0.8211
##
##           'Positive' Class : No
##
```

Repartition of test set

```
set.seed(500)
Train_Index <- sample(row.names(m_BankInfo), .5*dim(m_BankInfo)[1])
#create train index
Val_Index <- sample(setdiff(row.names(m_BankInfo), Train_Index), .3*dim(m_BankInfo)[1])
#create validation index
Test_Index =setdiff(row.names(m_BankInfo), union(Train_Index, Val_Index))
#create test index
#load the data
Train_Data <- m_BankInfo[Train_Index,]
Validation_Data <- m_BankInfo[Val_Index,]
Test_Data <- m_BankInfo[Test_Index,]
#normalize the quantitative data
norm.values3 <- preProcess(m_BankInfo[, columnsare], method=c("center", "scale"))
train.norm.df3 = Train_Data
val.norm.df3 = Validation_Data
```

```

test.norm.df3 = Test_Data
train.norm.df3[, columnsare] <- predict(norm.values3, Train_Data[, columnsare])
val.norm.df3[, columnsare] <- predict(norm.values3, Validation_Data[, columnsare])
test.norm.df3[, columnsare] <- predict(norm.values3, Test_Data[, columnsare])
#run knn for all 3
knn.train <- knn(train=train.norm.df3[,-14],test=train.norm.df3[,-14],cl=train.norm.df3[,14], k=3, prob=TRUE)
knn.val<- knn(train=train.norm.df3[,-14],test=val.norm.df3[,-14],cl=train.norm.df3[,14],k=3, prob=TRUE)
knn.test<- knn(train=train.norm.df3[,-14],test=test.norm.df3[,-14],cl=train.norm.df3[,14],k=3, prob=TRUE)
#display the confusion matrices
confusionMatrix(knn.train,train.norm.df3[,14], positive="Yes")

```

```

## Confusion Matrix and Statistics
##
##              Reference
## Prediction   No  Yes
##           No 2274  50
##           Yes    2 174
##
##              Accuracy : 0.9792
##              95% CI : (0.9728, 0.9844)
##      No Information Rate : 0.9104
##      P-Value [Acc > NIR] : < 2.2e-16
##
##              Kappa : 0.8589
##
##  Mcnemar's Test P-Value : 7.138e-11
##
##              Sensitivity : 0.7768
##              Specificity : 0.9991
##      Pos Pred Value : 0.9886
##      Neg Pred Value : 0.9785
##      Prevalence : 0.0896
##      Detection Rate : 0.0696
##      Detection Prevalence : 0.0704
##      Balanced Accuracy : 0.8880
##
##      'Positive' Class : Yes
##

```

```

confusionMatrix(knn.val,val.norm.df3[,14], positive="Yes")

```

```

## Confusion Matrix and Statistics
##
##              Reference
## Prediction   No  Yes
##           No 1335  65
##           Yes    5  95
##
##              Accuracy : 0.9533
##              95% CI : (0.9414, 0.9634)
##      No Information Rate : 0.8933
##      P-Value [Acc > NIR] : < 2.2e-16

```



```
##
##           Kappa : 0.7067
##
## Mcnemar's Test P-Value : 1.766e-12
##
##           Sensitivity : 0.59375
##           Specificity : 0.99627
##           Pos Pred Value : 0.95000
##           Neg Pred Value : 0.95357
##           Prevalence : 0.10667
##           Detection Rate : 0.06333
##           Detection Prevalence : 0.06667
##           Balanced Accuracy : 0.79501
##
##           'Positive' Class : Yes
##
```

```
confusionMatrix(knn.test,test.norm.df3[,14], positive="Yes")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##           No  904  42
##           Yes   0  54
##
##           Accuracy : 0.958
##           95% CI : (0.9436, 0.9696)
##           No Information Rate : 0.904
##           P-Value [Acc > NIR] : 9.200e-11
##
##           Kappa : 0.6992
##
## Mcnemar's Test P-Value : 2.509e-10
##
##           Sensitivity : 0.5625
##           Specificity : 1.0000
##           Pos Pred Value : 1.0000
##           Neg Pred Value : 0.9556
##           Prevalence : 0.0960
##           Detection Rate : 0.0540
##           Detection Prevalence : 0.0540
##           Balanced Accuracy : 0.7812
##
##           'Positive' Class : Yes
##
```

```
traceback()
```