Lab kNN Khanh Quach_Lab1

kNN Lab loan

```
#run all of the required library
library(ggplot2)
library(caret)
```

```
Loading required package: lattice
```

```
Warning: package 'lattice' was built under R version 4.3.1
```

```
# 1.0 Load Data
bank <- read.csv("UniversalBank.csv")
names(bank)
```

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

```
head(bank,10)
```

```
   ID Age Experience Income ZIP.Code Family CCAvg Education Mortgage
1   1  25          1     49    91107      4   1.6         1        0
2   2  45         19     34    90089      3   1.5         1        0
3   3  39         15     11    94720      1   1.0         1        0
4   4  35          9    100    94112      1   2.7         2        0
5   5  35          8     45    91330      4   1.0         2        0
```

```
6   6  37         13     29    92121      4   0.4         2      155
7   7  53         27     72    91711      2   1.5         2        0
8   8  50         24     22    93943      1   0.3         3        0
9   9  35         10     81    90089      3   0.6         2      104
10 10  34          9    180    93023      1   8.9         3        0
   Personal.Loan Securities.Account CD.Account Online CreditCard
1              0                  1          0      0          0
2              0                  1          0      0          0
3              0                  0          0      0          0
4              0                  0          0      0          0
5              0                  0          0      0          1
6              0                  0          0      1          0
7              0                  0          0      1          0
8              0                  0          0      0          1
9              0                  0          0      1          0
10             1                  0          0      0          0
```

```
str(bank)
```

```
'data.frame':   5000 obs. of  14 variables:
 $ ID                : int  1 2 3 4 5 6 7 8 9 10 ...
 $ Age               : int  25 45 39 35 35 37 53 50 35 34 ...
 $ Experience        : int  1 19 15 9 8 13 27 24 10 9 ...
 $ Income            : int  49 34 11 100 45 29 72 22 81 180 ...
 $ ZIP.Code          : int  91107 90089 94720 94112 91330 92121 91711 93943 90089 93023 ...
 $ Family            : int  4 3 1 1 4 4 2 1 3 1 ...
 $ CCAvg             : num  1.6 1.5 1 2.7 1 0.4 1.5 0.3 0.6 8.9 ...
 $ Education         : int  1 1 1 2 2 2 2 3 2 3 ...
 $ Mortgage          : int  0 0 0 0 0 155 0 0 104 0 ...
 $ Personal.Loan     : int  0 0 0 0 0 0 0 0 0 1 ...
 $ Securities.Account: int  1 1 0 0 0 0 0 0 0 0 ...
 $ CD.Account        : int  0 0 0 0 0 0 0 0 0 0 ...
 $ Online            : int  0 0 0 0 0 1 1 0 1 0 ...
 $ CreditCard        : int  0 0 0 0 1 0 0 1 0 0 ...
```

```
# Read file
str(bank)
```

```
'data.frame':   5000 obs. of  14 variables:
 $ ID                : int  1 2 3 4 5 6 7 8 9 10 ...
```

```
 $ Age               : int  25 45 39 35 35 37 53 50 35 34 ...
 $ Experience        : int  1 19 15 9 8 13 27 24 10 9 ...
 $ Income            : int  49 34 11 100 45 29 72 22 81 180 ...
 $ ZIP.Code          : int  91107 90089 94720 94112 91330 92121 91711 93943 90089 93023 ...
 $ Family            : int  4 3 1 1 4 4 2 1 3 1 ...
 $ CCAvg             : num  1.6 1.5 1 2.7 1 0.4 1.5 0.3 0.6 8.9 ...
 $ Education         : int  1 1 1 2 2 2 2 3 2 3 ...
 $ Mortgage          : int  0 0 0 0 0 155 0 0 104 0 ...
 $ Personal.Loan     : int  0 0 0 0 0 0 0 0 0 1 ...
 $ Securities.Account: int  1 1 0 0 0 0 0 0 0 0 ...
 $ CD.Account        : int  0 0 0 0 0 0 0 0 0 0 ...
 $ Online            : int  0 0 0 0 0 1 1 0 1 0 ...
 $ CreditCard        : int  0 0 0 0 1 0 0 1 0 0 ...
```

```r
# 1.1 Clean up
# Drop ID and zip code columns.( Except ID and ZIP code- mentioned in the doc)

bank <- bank[ , -c(1, 5)]
names(bank)
```

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

```r
# Reorder variables. Put the response last.

bank <- bank[ , c(1:7, 9:12, 8)]

head(bank,10)
```

```
   Age Experience Income Family CCAvg Education Mortgage Securities.Account
1   25          1     49      4   1.6         1        0                   1
2   45         19     34      3   1.5         1        0                   1
3   39         15     11      1   1.0         1        0                   0
4   35          9    100      1   2.7         2        0                   0
5   35          8     45      4   1.0         2        0                   0
6   37         13     29      4   0.4         2      155                   0
7   53         27     72      2   1.5         2        0                   0
8   50         24     22      1   0.3         3        0                   0
```

```
9   35          10     81       3   0.6          2      104                    0
10  34           9    180       1   8.9          3        0                    0
   CD.Account Online CreditCard Personal.Loan
1           0      0          0             0
2           0      0          0             0
3           0      0          0             0
4           0      0          0             0
5           0      0          1             0
6           0      1          0             0
7           0      1          0             0
8           0      0          1             0
9           0      1          0             0
10          0      0          0             1
```

```r
# Set categorical variables as factor.

bank$Education <- as.factor(bank$Education)
bank$Securities.Account <- as.factor(bank$Securities.Account)
bank$CD.Account <- as.factor(bank$CD.Account)
bank$Online <- as.factor(bank$Online)
bank$CreditCard <- as.factor(bank$CreditCard)
```

```r
# Rename outcome variable values (optional).
# Note: We can do the problem in "0" and "1" or name them.

bank$Personal.Loan <- factor(bank$Personal.Loan,
                             levels = c("0", "1"),
                             labels = c("No", "Yes"))

table(bank$Personal.Loan)
```

```
  No  Yes
4520  480
```

```r
# 1.2. Set training and validation sets
set.seed(666)

train_index <- sample(1:nrow(bank), 0.6 * nrow(bank))
valid_index <- setdiff(1:nrow(bank), train_index)
```

```r
train <- bank[train_index, ]
valid <- bank[valid_index, ]



nrow(train)
```

```
[1] 3000
```

```r
nrow(valid)
```

```
[1] 2000
```

```r
# 4. Define new customer
new_cust <- data.frame(Age = 40,
                       Experience = 10,
                       Income = 84,
                       Family = 2,
                       CCAvg = 2,
                       Education = 2,
                       Mortgage = 0,
                       Securities.Account = 0,
                       CD.Account = 0,
                       Online = 1,
                       CreditCard = 1)
```

```r
# Set categorical variables as factor.

new_cust$Education <- as.factor(new_cust$Education)
new_cust$Securities.Account <- as.factor(new_cust$Securities.Account)
new_cust$CD.Account <- as.factor(new_cust$CD.Account)
new_cust$Online <- as.factor(new_cust$Online)
new_cust$CreditCard <- as.factor(new_cust$CreditCard)

new_cust
```

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

```r
# 5.0 prepare for kNN.

# Normalisation, only for numerical variables

train_norm <- train
valid_norm <- valid

norm_values <- preProcess(train[, -c(6, 8:12)],
                          method = c("center",
                                     "scale"))

# Then normalise the training and validation sets.
# need to fix
train_norm[, -c(6, 8:12)] <- predict(norm_values,
train[, -c(6, 8:12)])
valid_norm[, -c(6, 8:12)] <-predict(norm_values,
valid[, -c(6, 8:12)])
newcust_norm <- predict(norm_values, new_cust)
newcust_norm
```

```
      Age Experience   Income     Family      CCAvg Education   Mortgage
1 -0.475848 -0.8943854 0.231983 -0.3581222 0.04725121         2 -0.5495866
  Securities.Account CD.Account Online CreditCard
1                  0          0      1          1
```

```r
# 7.0 Train kNN for predictions

# 7.1 k = 3
knn_model_k3 <- caret::knn3(Personal.Loan ~ .,
                            data = train_norm, k = 3)
knn_model_k3
```

```
3-nearest neighbor model
Training set outcome distribution:

  No  Yes
2692  308
```

```r
# Predict training set with k = 3

knn_pred_k3_train <- predict(knn_model_k3,
                             newdata = train_norm[, -c(12)],
                             type = "class")
head(knn_pred_k3_train)
```

```
[1] No No No No No No
Levels: No Yes
```

```r
# Evaluate the confusion matrix with k = 3
confusionMatrix(knn_pred_k3_train, as.factor(train_norm[, 12]),
                positive = "Yes")
```

```
Confusion Matrix and Statistics

          Reference
Prediction   No  Yes
       No  2688   70
       Yes    4  238

               Accuracy : 0.9753
                 95% CI : (0.9691, 0.9806)
    No Information Rate : 0.8973
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.8521

 Mcnemar's Test P-Value : 4.153e-14

            Sensitivity : 0.77273
            Specificity : 0.99851
         Pos Pred Value : 0.98347
         Neg Pred Value : 0.97462
             Prevalence : 0.10267
         Detection Rate : 0.07933
   Detection Prevalence : 0.08067
      Balanced Accuracy : 0.88562

       'Positive' Class : Yes
```

```r
# 7.2 k = 5

# train k = 5
knn_model_k5 <- caret::knn3(Personal.Loan ~ .,
                            data = train_norm, k = 5)
knn_model_k5
```

```
5-nearest neighbor model
Training set outcome distribution:

  No  Yes
2692  308
```

```r
# Predict training set with k = 5
knn_pred_k5_train <- predict(knn_model_k5,
                             newdata = train_norm[, -c(12)],
                             type = "class")
head(knn_pred_k5_train)
```

```
[1] No No No No No No
Levels: No Yes
```

```r
# Evaluate the confusion matrix with k = 5
confusionMatrix(knn_pred_k5_train, as.factor(train_norm[, 12]),
                positive = "Yes")
```

```
Confusion Matrix and Statistics

          Reference
Prediction   No  Yes
       No  2687   95
       Yes    5  213

               Accuracy : 0.9667
                 95% CI : (0.9596, 0.9728)
    No Information Rate : 0.8973
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7922
```

```
 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.69156
            Specificity : 0.99814
         Pos Pred Value : 0.97706
         Neg Pred Value : 0.96585
             Prevalence : 0.10267
         Detection Rate : 0.07100
   Detection Prevalence : 0.07267
      Balanced Accuracy : 0.84485

       'Positive' Class : Yes
```

```r
# 7.3 k = 7

# train k = 7
knn_model_k7 <- caret::knn3(Personal.Loan ~ .,
                            data = train_norm, k = 7)
knn_model_k7
```

```
7-nearest neighbor model
Training set outcome distribution:

  No  Yes
2692  308
```

```r
# Predict training set with k = 7
knn_pred_k7_train <- predict(knn_model_k7,
                             newdata = train_norm[, -c(12)],
                             type = "class")
head(knn_pred_k7_train)
```

```
[1] No No No No No No
Levels: No Yes
```

```r
# Evaluate confusion matrix with k = 7
confusionMatrix(knn_pred_k7_train, as.factor(train_norm[, 12]),
```

```
                positive = "Yes")
```

```
Confusion Matrix and Statistics

          Reference
Prediction   No  Yes
       No  2691  123
       Yes    1  185

               Accuracy : 0.9587
                 95% CI : (0.9509, 0.9655)
    No Information Rate : 0.8973
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.728

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.60065
            Specificity : 0.99963
         Pos Pred Value : 0.99462
         Neg Pred Value : 0.95629
             Prevalence : 0.10267
         Detection Rate : 0.06167
   Detection Prevalence : 0.06200
      Balanced Accuracy : 0.80014

       'Positive' Class : Yes
```

Predict the validation set

```
# 7.4 predict validation set

# use k = 3
# Predict training set with k = 3
knn_pred_k3_valid <- predict(knn_model_k3,
                             newdata = valid_norm[, -c(12)],
                             type = "class")
```

```r
head(knn_pred_k3_valid)
```

```
[1] No No No No No No
Levels: No Yes
```

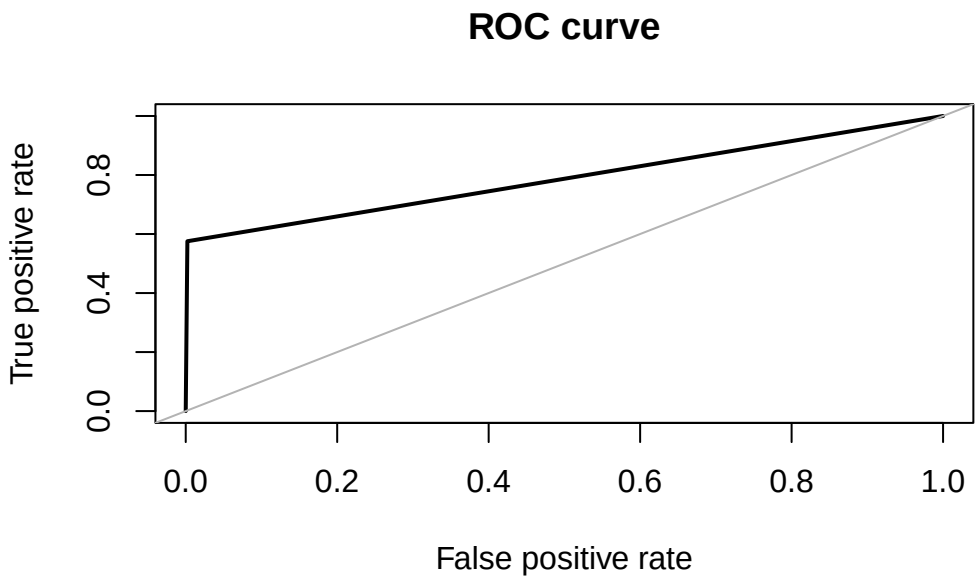
```r
# Evaluate confusion matrix with k = 3
confusionMatrix(knn_pred_k3_valid, as.factor(valid_norm[, 12]),
                positive = "Yes")
```

```
Confusion Matrix and Statistics

          Reference
Prediction   No  Yes
       No  1824   73
       Yes    4   99

               Accuracy : 0.9615
                 95% CI : (0.9521, 0.9695)
    No Information Rate : 0.914
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7007

 Mcnemar's Test P-Value : 9.239e-15

            Sensitivity : 0.5756
            Specificity : 0.9978
         Pos Pred Value : 0.9612
         Neg Pred Value : 0.9615
             Prevalence : 0.0860
         Detection Rate : 0.0495
   Detection Prevalence : 0.0515
      Balanced Accuracy : 0.7867

       'Positive' Class : Yes
```

```r
library(ROSE)
```

```
Loaded ROSE 0.0-4
```

```
# Graph the ROC and show the AUC - number show @ the console for k = 3
ROSE::roc.curve(valid_norm$Personal.Loan,
                knn_pred_k3_valid)
```




```
Area under the curve (AUC): 0.787
```

```
# Area under the curve (AUC): 0.787
```

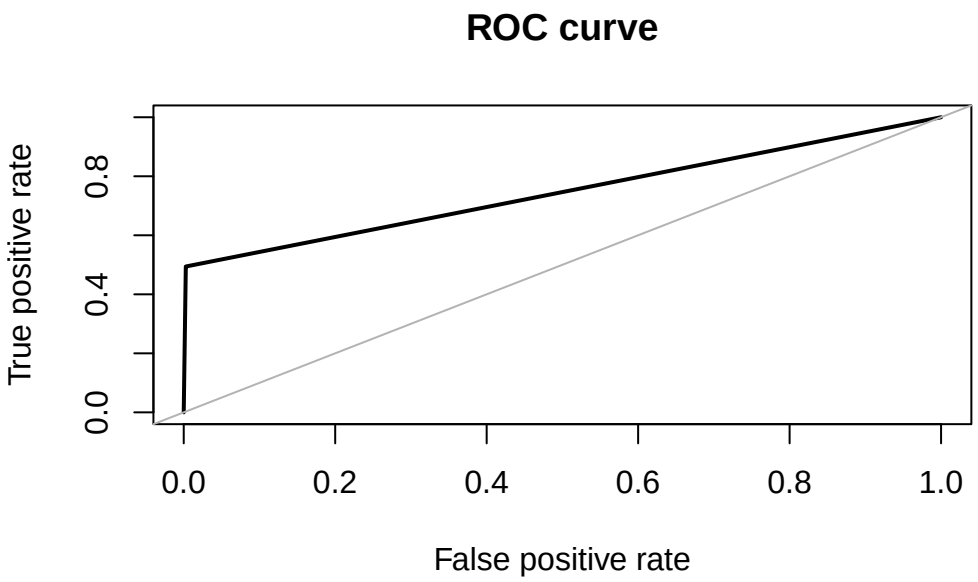
```
# use k = 5
# Predict training set with k = 5
knn_pred_k5_valid <- predict(knn_model_k5,
                             newdata = valid_norm[, -c(12)],
                             type = "class")
head(knn_pred_k5_valid)
```

```
[1] No No No No No No
Levels: No Yes
```

```r
# Evaluate confusion matrix with k = 5
confusionMatrix(knn_pred_k5_valid, as.factor(valid_norm[, 12]),
                positive = "Yes")
```

```
Confusion Matrix and Statistics

          Reference
Prediction   No  Yes
       No  1823   87
       Yes    5   85

               Accuracy : 0.954
                 95% CI : (0.9439, 0.9628)
    No Information Rate : 0.914
    P-Value [Acc > NIR] : 2.771e-12

                  Kappa : 0.6268

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.4942
            Specificity : 0.9973
         Pos Pred Value : 0.9444
         Neg Pred Value : 0.9545
             Prevalence : 0.0860
         Detection Rate : 0.0425
   Detection Prevalence : 0.0450
      Balanced Accuracy : 0.7457

       'Positive' Class : Yes
```

```r
# Graph the ROC and show the AUC - number show @ the console for k = 5
ROSE::roc.curve(valid_norm$Personal.Loan,
                knn_pred_k5_valid)
```




```
Area under the curve (AUC): 0.746
```

```
# Area under the curve (AUC): 0.746
```

```
# use k = 7
# Predict training set for k = 7
knn_pred_k7_valid <- predict(knn_model_k7,
                             newdata = valid_norm[, -c(12)],
                             type = "class")
head(knn_pred_k7_valid)
```

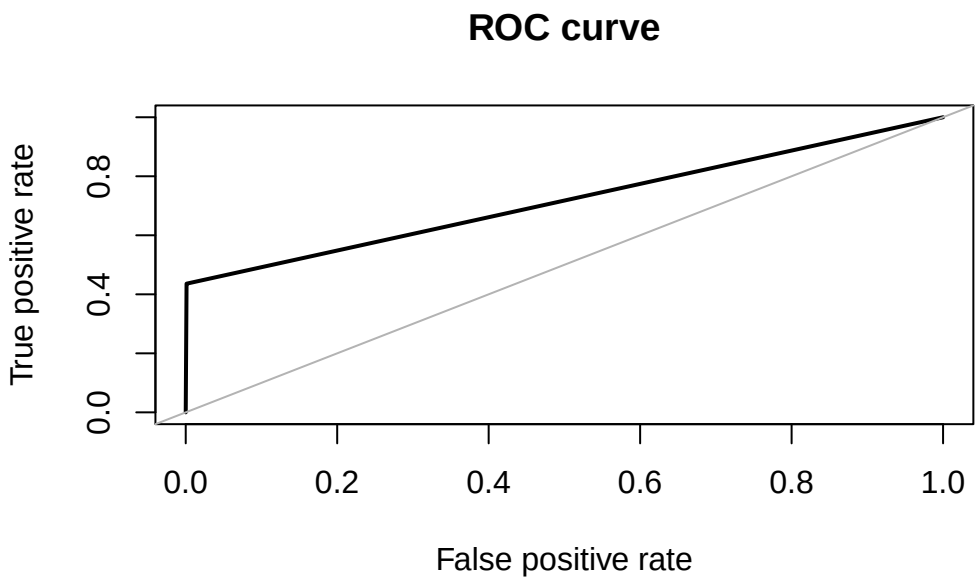
```
[1] No No No No No No
Levels: No Yes
```

```
# Evaluate confusion matrix with k = 7
confusionMatrix(knn_pred_k7_valid, as.factor(valid_norm[, 12]),
                positive = "Yes")
```

```
Confusion Matrix and Statistics
```

```
          Reference
Prediction   No  Yes
       No  1826   97
       Yes    2   75

               Accuracy : 0.9505
                 95% CI : (0.9401, 0.9596)
    No Information Rate : 0.914
    P-Value [Acc > NIR] : 2.389e-10

                  Kappa : 0.5801

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.4360
            Specificity : 0.9989
         Pos Pred Value : 0.9740
         Neg Pred Value : 0.9496
             Prevalence : 0.0860
         Detection Rate : 0.0375
   Detection Prevalence : 0.0385
      Balanced Accuracy : 0.7175

       'Positive' Class : Yes
```

```r
# Graph the ROC and show the AUC - number show @ the console for k = 7
ROSE::roc.curve(valid_norm$Personal.Loan,
                knn_pred_k7_valid)
```




```
Area under the curve (AUC): 0.717
```

```
# Area under the curve (AUC): 0.717
```

```
# Note on the sensitivity, specificity, precision, and AUC ROC

#knn_pred_k3_train
#Sensitivity : 0.77273
#Specificity : 0.99851

#knn_pred_k5_train
#Sensitivity : 0.69156
#Specificity : 0.99814

#knn_pred_k7_train
#Sensitivity : 0.60065
#Specificity : 0.99963

#knn_pred_k3_valid
#Sensitivity : 0.5756
#Specificity : 0.9978
```

```
#knn_pred_k5_valid
#Sensitivity : 0.4942
#Specificity : 0.9973

#knn_pred_k7_valid
#Sensitivity : 0.4360
#Specificity : 0.9989

# Sensitivity (tpr) decrease when k increase -
# higher sensitivity is better -> k=3 is the best option.
```

k = 3 train: 0.9753 and validation = 0.9615, AUC: 0.787 k = 5 train: 0.9667 and validation = 0.954, AUC: 0.746 k = 7 train: 0.9587 and validation = 0.9505, AUC: 0.717

ROC AUC score shows how well the classifier distinguishes positive and negative classes. It can take values from 0 to 1. A higher ROC AUC indicates better performance. Therefore based on all of the above we can conclude that k=3 will be the highest accuracy model.

k=3 provides the highest accuracy on the validation set, but it's also the most complex model lowest 'k'.

```
# 8. use kNN for new customer, k = ???

# k=3 has the best of both

# Use k = 3 for kNN new customer, k = 3

# Using k=3  to predict the new customer
knn_pred_new_cust <- predict(knn_model_k3,newdata = newcust_norm, type = "class")
knn_pred_new_cust
```

```
[1] No
Levels: No Yes
```

```
# 2688 cases were correctly predicted as "No" and
# 238 cases were correctly predicted as "Yes".

# The accuracies of the predictions on the training and validation
# sets are  both high (0.9753 VS 0.9615) for k = 3, which do not suggest overfitting
```

```
# 9. Answers

# The result for the new customer in regards to Personal.Loan is "No".
# We can assume that the new customer is not likely to accept the loan offer.
```