KNN on Bank Customer Data

import the required libraries for tidying data.

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(caTools)
library(class)
```

Read the data file in for Bank Customer Data and look at the first few rows with the head function.

```
bank.df <-read.csv("bank.customer.data.csv")
head(bank.df)
```

```
##   Age   Occupation Marital Education Ever.Defaulted.on.loan balance housing
## 1  58   management married  tertiary                    no    2143     yes
## 2  44   technician  single secondary                    no      29     yes
## 3  33 entrepreneur married secondary                    no       2     yes
## 4  47  blue-collar married   unknown                    no    1506     yes
## 5  33      unknown  single   unknown                    no       1      no
## 6  35   management married  tertiary                    no     231     yes
##   loan contact duration.with.bank.in.days. campaign Churn
## 1   no unknown                         261        1    no
## 2   no unknown                         151        1    no
## 3  yes unknown                          76        1    no
## 4   no unknown                          92        1    no
## 5   no unknown                         198        1    no
## 6   no unknown                         139        1    no
```

look at the data types and some summary statistics.

```
summary(bank.df)
```

```
##       Age         Occupation          Marital           Education
##  Min.   :18.00   Length:45211       Length:45211       Length:45211
##  1st Qu.:33.00   Class :character   Class :character   Class :character
##  Median :39.00   Mode  :character   Mode  :character   Mode  :character
##  Mean   :40.94
##  3rd Qu.:48.00
##  Max.   :95.00
##  Ever.Defaulted.on.loan    balance         housing              loan
```

```
##  Length:45211           Min.   : -8019   Length:45211       Length:45211
##  Class :character       1st Qu.:    72   Class :character   Class :character
##  Mode  :character       Median :   448   Mode  :character   Mode  :character
##                         Mean   :  1362
##                         3rd Qu.:  1428
##                         Max.   :102127
##    contact          duration.with.bank.in.days.    campaign
##  Length:45211       Min.   :   0.0              Min.   : 1.000
##  Class :character   1st Qu.: 103.0              1st Qu.: 1.000
##  Mode  :character   Median : 180.0              Median : 2.000
##                     Mean   : 258.2              Mean   : 2.764
##                     3rd Qu.: 319.0              3rd Qu.: 3.000
##                     Max.   :4918.0              Max.   :63.000
##     Churn
##  Length:45211
##  Class :character
##  Mode  :character
##
##
##
```

```
dim(bank.df)
```

```
## [1] 45211    12
```

There are 12 columns (variables) with the churn column as the outcome variable, as we are predicting customer churn. Check for any missing values to handle them.

Use the sapply() function to check each column and sum the missing values in each column.

```
sapply(bank.df, function(x) sum(is.na(x)))
```

```
##                          Age                   Occupation
##                            0                            0
##                      Marital                    Education
##                            0                            0
##      Ever.Defaulted.on.loan                      balance
##                            0                            0
##                      housing                         loan
##                            0                            0
##                      contact duration.with.bank.in.days.
##                            0                            0
##                     campaign                        Churn
##                            0                            0
```

The data is clean and ready. However, we are only going to train the model on numeric values for: "Age", "Balance", "duration with bank", and "campaign"

```
# select only certain columns
bank.df <- bank.df %>% select(Age,
                              balance,
                              duration.with.bank.in.days.,
                              campaign,
                              Churn)

#normalize/scale the data set using z-score
bank.df[, c("Age", "balance", "duration.with.bank.in.days.", "campaign")] <- scale(bank.df[ , c("Age",
```

```r
# see the scaled data
head(bank.df)
```

```
##          Age     balance duration.with.bank.in.days.   campaign Churn
## 1  1.6069472  0.25641642                  0.01101598 -0.5693443    no
## 2  0.2885261 -0.43788985                 -0.41612236 -0.5693443    no
## 3 -0.7473762 -0.44675753                 -0.70735304 -0.5693443    no
## 4  0.5710449  0.04720492                 -0.64522382 -0.5693443    no
## 5 -0.7473762 -0.44708596                 -0.23361780 -0.5693443    no
## 6 -0.5590303 -0.37154649                 -0.46271926 -0.5693443    no
```

After looking at the data and seeing it is already a clean dataset, it is time to create the K-NN Model and begin training it.

First, partitioning the data into a training and test set with a 70/30 split.

```r
library(caret)
```

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

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```r
set.seed(552)

# use the libary caret to access the "createDataPartition" function and split the data into 1 partition
split_index <- createDataPartition(bank.df$Churn, times=1, p =.70, list=FALSE)

bank_train <- bank.df[split_index, ]

#create testing set
bank_test  <- bank.df[-split_index, ]

#view number of rows in each set
nrow(bank_train)
```

```
## [1] 31649
```

```r
nrow(bank_test)
```

```
## [1] 13562
```

Now that the data is split, we can create the KNN model using the normalized data for the train and test sets.

```r
# KNN_Train

accuracy.df <- data.frame(k = seq(1, 10, 1), accuracy = rep(0, 10))
for(i in 1:10) {
  knn.pred <-  knn(train = bank_train[, -5],
                   test = bank_test[, -5],
                   cl = bank_train[, 5],
                   k=i,
                   prob = TRUE)
  accuracy.df[i, 2] <- confusionMatrix(knn.pred, as.factor(bank_test[, 5]))$overall[1]
```

```
}
accuracy.df
```

```
##     k  accuracy
## 1   1 0.8420587
## 2   2 0.8377820
## 3   3 0.8682348
## 4   4 0.8683085
## 5   5 0.8755346
## 6   6 0.8764194
## 7   7 0.8804749
## 8   8 0.8799587
## 9   9 0.8829081
## 10 10 0.8826132
```