Practice-3

Harsh

25/05/2020

1. Download the data set for the tutorial.
2. Follow this tutorial on applying kNN to prostate cancer detection and implement all of the steps in an R Notebook. Make sure to explain each step and what it does.

```
library(data.table)

cancer_data <- read.csv("C:\\Users\\harsh\\Documents\\GitHub\\DA5030-Intro-to-ML-and-DM\\Introduction-t

head(cancer_data)
```

```
##   id diagnosis_result radius texture perimeter area smoothness compactness
## 1  1                M     23      12       151  954      0.143       0.278
## 2  2                B      9      13       133 1326      0.143       0.079
## 3  3                M     21      27       130 1203      0.125       0.160
## 4  4                M     14      16        78  386      0.070       0.284
## 5  5                M      9      19       135 1297      0.141       0.133
## 6  6                B     25      25        83  477      0.128       0.170
##   symmetry fractal_dimension
## 1    0.242             0.079
## 2    0.181             0.057
## 3    0.207             0.060
## 4    0.260             0.097
## 5    0.181             0.059
## 6    0.209             0.076
```

```
#Creating a copy of cancer_data
cancer_data_1 <- cancer_data

#Deleting the first column which is id
cancer_data_1 <- cancer_data_1[-1]

#Checking the total number of patients
table(cancer_data$diagnosis_result)
```

```
##
##  B  M
## 38 62
```

```r
setDT(cancer_data_1)

#Creating a Normalize function using min/max method. This helps in
normalize <- function(x){
  return((x - min(x)) / (max(x) - min(x)))
}

#Normalizing the values by calling the function with the help of lapply
cancer_data_1 <- as.data.frame(lapply(cancer_data[,3:9],normalize))

summary(cancer_data_1)
```

```
##      radius          texture         perimeter           area
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.1875   1st Qu.:0.1875   1st Qu.:0.2542   1st Qu.:0.1639
##  Median :0.5000   Median :0.4062   Median :0.3500   Median :0.2637
##  Mean   :0.4906   Mean   :0.4519   Mean   :0.3732   Mean   :0.2989
##  3rd Qu.:0.7500   3rd Qu.:0.7031   3rd Qu.:0.5188   3rd Qu.:0.4266
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##    smoothness      compactness        symmetry
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.3219   1st Qu.:0.1384   1st Qu.:0.2189
##  Median :0.4384   Median :0.2622   Median :0.3254
##  Mean   :0.4484   Mean   :0.2889   Mean   :0.3442
##  3rd Qu.:0.5753   3rd Qu.:0.3876   3rd Qu.:0.4379
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
```

```r
#Dividing the data in 65:35 ratio
training_data <- cancer_data_1[1:65,]
testing_data <- cancer_data_1[66:100,]

#Selecting the labels and storing in a new variable
training_data_labels <- cancer_data[1:65,2]
testing_data_labels <- cancer_data[66:100,2]
```

```r
#install.packages("class")
#install.packages("gmodels")

library(class)
```

```
## Warning: package 'class' was built under R version 3.6.3
```

```r
library(gmodels)
```

```
## Warning: package 'gmodels' was built under R version 3.6.3
```

```r
#Using the knn function from the class library. We select the efficient value which is k = 7. We get an
cancer_pred_1 <- knn(train = training_data, test = testing_data, cl = training_data_labels, k = 10)
cancer_pred <- knn(train = training_data, test = testing_data, cl = training_data_labels, k = 7)

CrossTable(x = testing_data_labels, y = cancer_pred_1, prop.chisq = FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  35
##
##
##                      | cancer_pred_1
## testing_data_labels |         B |         M | Row Total |
## --------------------|-----------|-----------|-----------|
##                   B |         7 |        12 |        19 |
##                     |     0.368 |     0.632 |     0.543 |
##                     |     0.875 |     0.444 |           |
##                     |     0.200 |     0.343 |           |
## --------------------|-----------|-----------|-----------|
##                   M |         1 |        15 |        16 |
##                     |     0.062 |     0.938 |     0.457 |
##                     |     0.125 |     0.556 |           |
##                     |     0.029 |     0.429 |           |
## --------------------|-----------|-----------|-----------|
##        Column Total |         8 |        27 |        35 |
##                     |     0.229 |     0.771 |           |
## --------------------|-----------|-----------|-----------|
##
##
```

```
CrossTable(x = testing_data_labels, y = cancer_pred, prop.chisq = FALSE)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  35
##
##
##                      | cancer_pred
## testing_data_labels |         B |         M | Row Total |
## --------------------|-----------|-----------|-----------|
##                   B |         9 |        10 |        19 |
##                     |     0.474 |     0.526 |     0.543 |
```

```
##                      |     1.000 |     0.385 |           |
##                      |     0.257 |     0.286 |           |
## --------------------|-----------|-----------|-----------|
##                   M |         0 |        16 |        16 |
##                      |     0.000 |     1.000 |     0.457 |
##                      |     0.000 |     0.615 |           |
##                      |     0.000 |     0.457 |           |
## --------------------|-----------|-----------|-----------|
##        Column Total |         9 |        26 |        35 |
##                      |     0.257 |     0.743 |           |
## --------------------|-----------|-----------|-----------|
##
##
```

We get an accuracy of around 60% for k = 10 which is quite low. We try to improve the accuracy by testing with different k values, the best one achieved is for k = 7 i.e 71%.

3. Once you've complete the tutorial, try another kNN implementation from another package, such as the caret package. Compare the accuracy of the two implementations.
4. Try the confusionMatrix function from the caret package to determine the accuracy of both algorithms.

```
#install.packages("caret")
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.6.3
```

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

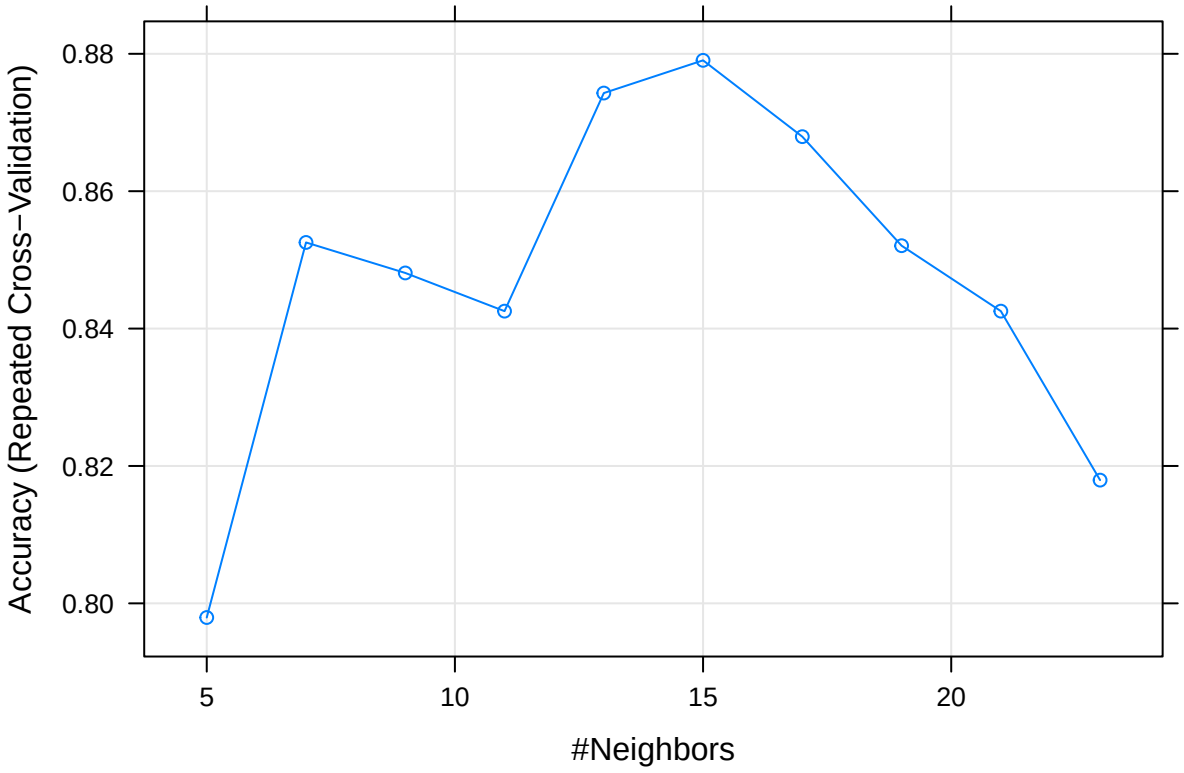
```
## Loading required package: ggplot2
```

```
## Warning: package 'ggplot2' was built under R version 3.6.3
```

```
cancer_data$diagnosis_result <- as.factor(cancer_data$diagnosis_result)

set.seed(123)

#Creating new data variables for testing caret package
train.cancer <- training_data
train.cancer$diagnosis_result <- as.factor(training_data_labels)
test.cancer <- testing_data
test.cancer$diagnosis_result <- as.factor(testing_data_labels)

#Assigning trainControl values to a variable
trctrl <- trainControl(method = "repeatedcv", number = 10, repeats = 3)

#Using train function from the caret package to train the model
knn_fit <- train(diagnosis_result ~., data = train.cancer, method = "knn",
                 trControl=trctrl,
                 preProcess = c("center", "scale"),
                 tuneLength = 10)

#Calling the trained function to get the accuracy of different k values
knn_fit
```

```
## k-Nearest Neighbors
##
## 65 samples
##  7 predictor
##  2 classes: 'B', 'M'
##
## Pre-processing: centered (7), scaled (7)
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 58, 58, 58, 58, 58, 59, ...
## Resampling results across tuning parameters:
##
##   k   Accuracy   Kappa
##    5  0.7979365  0.4548640
##    7  0.8525397  0.5658845
##    9  0.8480952  0.5424646
##   11  0.8425397  0.5310177
##   13  0.8742857  0.6130812
##   15  0.8790476  0.6220448
##   17  0.8679365  0.5887115
##   19  0.8520635  0.5357703
##   21  0.8425397  0.5130812
##   23  0.8179365  0.4368908
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 15.
```

```
#Plotting accuracy vs k values
plot(knn_fit)
```




```
#Testing the model with 35% test data
test_pred <- predict(knn_fit, newdata = test.cancer)
test_pred
```

```
##  [1] M M B M B M M M M B M M M M B M M M M B M M M M M M M M M M B B B M
## Levels: B M
```

```
# We get an accuracy of 68.57% for Caret Package
confusionMatrix(test_pred, test.cancer$diagnosis_result)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  B  M
##          B  8  0
##          M 11 16
##
##                Accuracy : 0.6857
##                  95% CI : (0.5071, 0.8315)
##     No Information Rate : 0.5429
##     P-Value [Acc > NIR] : 0.061949
##
##                   Kappa : 0.3994
##
```

```
##  Mcnemar's Test P-Value : 0.002569
##
##             Sensitivity : 0.4211
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.5926
##              Prevalence : 0.5429
##          Detection Rate : 0.2286
##    Detection Prevalence : 0.2286
##       Balanced Accuracy : 0.7105
##
##        'Positive' Class : B
##
```

```
# We get an accuracy of 71.43% for Class Package
confusionMatrix(cancer_pred, as.factor(testing_data_labels))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  B  M
##          B  9  0
##          M 10 16
##
##                Accuracy : 0.7143
##                  95% CI : (0.537, 0.8536)
##     No Information Rate : 0.5429
##     P-Value [Acc > NIR] : 0.029336
##
##                   Kappa : 0.4514
##
##  Mcnemar's Test P-Value : 0.004427
##
##             Sensitivity : 0.4737
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.6154
##              Prevalence : 0.5429
##          Detection Rate : 0.2571
##    Detection Prevalence : 0.2571
##       Balanced Accuracy : 0.7368
##
##        'Positive' Class : B
##
```

Results of training model are very high for k = 15 which is 88% accuracy but when we test the model with testing data, observed accuracy is very low which is 68% for caret package compared to class package which is 71%.