Assignment 3

R Markdown

```
prc <- read.csv("prostate_cancer.csv", stringsAsFactors = FALSE) #This command imports
the required data set and saves it to the prc data frame.

#stringsAsFactors command helps to convert every character vector to a factor whereve
r it makes sense.

str(prc) #We use this command to see whether the data is structured or not.
```

```
## 'data.frame':    100 obs. of  10 variables:
##  $ id               : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ diagnosis_result : chr  "M" "B" "M" "M" ...
##  $ radius           : int  23 9 21 14 9 25 16 15 19 25 ...
##  $ texture          : int  12 13 27 16 19 25 26 18 24 11 ...
##  $ perimeter        : int  151 133 130 78 135 83 120 90 88 84 ...
##  $ area             : int  954 1326 1203 386 1297 477 1040 578 520 476 ...
##  $ smoothness       : num  0.143 0.143 0.125 0.07 0.141 0.128 0.095 0.119 0.127 0.1
19 ...
##  $ compactness      : num  0.278 0.079 0.16 0.284 0.133 0.17 0.109 0.165 0.193 0.2
4 ...
##  $ symmetry         : num  0.242 0.181 0.207 0.26 0.181 0.209 0.179 0.22 0.235 0.20
3 ...
##  $ fractal_dimension: num  0.079 0.057 0.06 0.097 0.059 0.076 0.057 0.075 0.074 0.0
82 ...
```

```
View(prc)

prc <- prc[-1] # #removes the first variable(id) from the data set.

table(prc$diagnosis_result) # it helps us to get the numbers of patients
```

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

```
prc$diagnosis <- factor(prc$diagnosis_result, levels = c("B", "M"), labels = c("Benig
n", "Malignant"))

View(prc$diagnosis)

round(prop.table(table(prc$diagnosis)) * 100, digits = 1)  # it gives the result in th
e percentage form rounded of to 1 decimal place( and so it's digits = 1)
```

```
## 
##    Benign Malignant 
##        38        62
```

```
normalize <- function(x) {
  return ((x - min(x)) / (max(x) - min(x))) }  # Normalization function

prc_n <- as.data.frame(lapply(prc[2:9], normalize)) # Normalizing values in data set

View(prc_n)

summary(prc_n$radius) # summary function to get the mean, median, mode and other stati
stics
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##  0.0000  0.1875  0.5000  0.4906  0.7500  1.0000
```

```
prc_train <- prc_n[1:65,]    # creating training data set
prc_test <- prc_n[66:100,]   # creating test data set

prc_train_labels <- prc[1:65, 1]
prc_test_labels <- prc[66:100, 1] #This code takes the diagnosis factor in column 1 o
f the prc data frame and on turn creates prc_train_labels and prc_test_labels data fra
me.


#install.packages('class')
library(class)

View(prc_test_labels)
prc_test_pred <- knn(train = prc_train, test = na.omit(prc_test),cl = prc_train_label
s, k=10) # Knn function to classify dataset

View(prc_test_pred)

#install.packages('gmodels')
library(gmodels)
CrossTable(x= na.omit(prc_test_labels), y = prc_test_pred,prop.chisq= FALSE) # check t
he accuracy of the predicted values in prc_test_pred as to whether they match up with
the known values in prc_test_label using CrossTable function.
```

```
## 
## 
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
## 
## 
## Total Observations in Table:  35 
## 
## 
##                          | prc_test_pred 
## na.omit(prc_test_labels) |         B |         M | Row Total | 
## -------------------------|-----------|-----------|-----------|
##                        B |         8 |        11 |        19 | 
##                          |     0.421 |     0.579 |     0.543 | 
##                          |     0.889 |     0.423 |           | 
##                          |     0.229 |     0.314 |           | 
## -------------------------|-----------|-----------|-----------|
##                        M |         1 |        15 |        16 | 
##                          |     0.062 |     0.938 |     0.457 | 
##                          |     0.111 |     0.577 |           | 
##                          |     0.029 |     0.429 |           | 
## -------------------------|-----------|-----------|-----------|
##             Column Total |         9 |        26 |        35 | 
##                          |     0.257 |     0.743 |           | 
## -------------------------|-----------|-----------|-----------|
## 
## 
```

There were 8 cases of Benign which are accurately predicted as Benign which constitutes 22.9%. But there were 27 cases of Malignant of which 16 were correctly predicted which constitutes 45.7% as Malignant but 11 cases were predicted incorrectly which constitutes to 31.4%. Hence the total accuracy of the model is 68.6% which shows that there are chances to improve the model performance.

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

library(caret)
```

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

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

```
setwd("C:/Users/Meghana Nadig/Desktop/Knn/") #Using this command, we've imported the
'Prostate_Cancer.csv' data file. This command is used to point to the folder containin
g the required file. Do keep in mind, that it's a common mistake to use "\" instead o
f "/" after the setwd command.

p <- read.csv("C:/Users/Meghana Nadig/Desktop/Knn/prostate_cancer.csv", stringsAsFacto
rs = FALSE)

str(p)
```

```
## 'data.frame':    100 obs. of  10 variables:
##  $ id               : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ diagnosis_result : chr  "M" "B" "M" "M" ...
##  $ radius           : int  23 9 21 14 9 25 16 15 19 25 ...
##  $ texture          : int  12 13 27 16 19 25 26 18 24 11 ...
##  $ perimeter        : int  151 133 130 78 135 83 120 90 88 84 ...
##  $ area             : int  954 1326 1203 386 1297 477 1040 578 520 476 ...
##  $ smoothness       : num  0.143 0.143 0.125 0.07 0.141 0.128 0.095 0.119 0.127 0.1
19 ...
##  $ compactness      : num  0.278 0.079 0.16 0.284 0.133 0.17 0.109 0.165 0.193 0.2
4 ...
##  $ symmetry         : num  0.242 0.181 0.207 0.26 0.181 0.209 0.179 0.22 0.235 0.20
3 ...
##  $ fractal_dimension: num  0.079 0.057 0.06 0.097 0.059 0.076 0.057 0.075 0.074 0.0
82 ...
```

```
p <- p[-1]

set.seed(1)

TrainingSet <- createDataPartition(p$diagnosis_result,p =.66, list=FALSE)

TrainingData <- p[TrainingSet,]
Test <- p[-TrainingSet,]
View(TrainingData)
typeof(TrainingSet)
```

```
## [1] "integer"
```

```
TrainingLabel <- TrainingData[1]
TestLabel <- Test[1]
typeof(TrainingLabel)
```

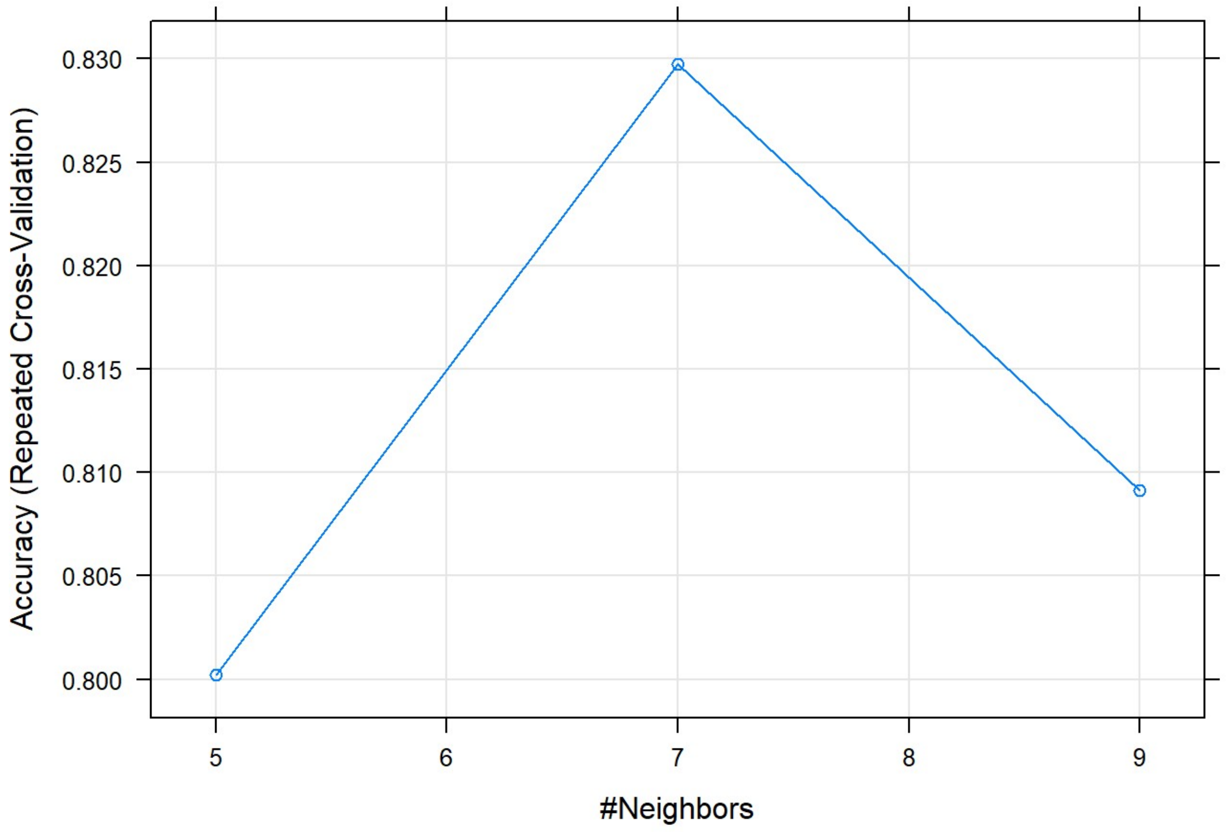
```
## [1] "list"
```

```r
View(TrainingLabel)

#preprocessing
preProcValues <- preProcess(x = TrainingData,method = c("center", "scale"))

set.seed(400)
ctrl <- trainControl(method="repeatedcv",repeats = 3)
knnFit <- train(diagnosis_result ~ ., data = TrainingData, method = "knn", trControl
= ctrl, preProcess = c("center","scale"))

plot(knnFit)
```



```r
knnPredict <- predict(knnFit, Test )
View(Test)
View(knnPredict)
```

```r
#Get the confusion matrix to see accuracy value and other parameter values
confusionMatrix(knnPredict, Test$diagnosis_result)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  B  M
##          B  9  2
##          M  3 19
##
##                Accuracy : 0.8485
##                  95% CI : (0.681, 0.9489)
##     No Information Rate : 0.6364
##     P-Value [Acc > NIR] : 0.006669
##
##                   Kappa : 0.6667
##  Mcnemar's Test P-Value : 1.000000
##
##             Sensitivity : 0.7500
##             Specificity : 0.9048
##          Pos Pred Value : 0.8182
##          Neg Pred Value : 0.8636
##              Prevalence : 0.3636
##          Detection Rate : 0.2727
##    Detection Prevalence : 0.3333
##       Balanced Accuracy : 0.8274
##
##        'Positive' Class : B
##
```

```
#install.packages('e1071', dependencies=TRUE)
```