Classifying Orthopaedic Patients based on Biomechanical Features

Olivia Malkowski

06/12/2019

Contents

1. Introduction - Part 1

1.1 Overview/Executive Summary

This dataset, available via Kaggle, includes data on six biomechanical features of orthopaedic patients (i.e. pelvic incidence, pelvic tilt, lumbar lordosis angle, sacral slope, pelvic radius, and degree of spondylolisthesis). The objective of Part 1 of this project is to use these features to classify patients into one of three groups (i.e. 100 Normal, 60 Disk Hernia, and 150 Spondylolisthesis patients). In order to fulfil this task, a number of steps were performed. Notably, once the data were cleaned, data visualisation was used to gain an insight into the predictors of each of the three conditions/classes. Following this, methods including recursive partitioning, k-nearest neighbours, random forests, and multiclass support vector machine were implemented to improve the accuracy of the classifications.

1.2 Loading the Data

Note: this process could take a couple of minutes

```r
### Download file part 1
### https://www.kaggle.com/uciml/biomechanical-features-of-orthopedic-patients/data
filename <- "column_3C_weka.csv"
library(readr)
dat3C <- read_csv(filename)
```

```
## Parsed with column specification:
## cols(
##   pelvic_incidence = col_double(),
##   pelvic_tilt = col_double(),
##   lumbar_lordosis_angle = col_double(),
##   sacral_slope = col_double(),
##   pelvic_radius = col_double(),
##   degree_spondylolisthesis = col_double(),
##   class = col_character()
## )
```

1.3 Tidy data and Summary

Before processing the data, it is important to familiarise ourselves with the `dat3C` dataset. This contains 310 observations of 7 variables.

```
### Summarising the data
table(dat3C$class)
```

```
##
##          Hernia            Normal Spondylolisthesis
##              60               100               150
```

```
summary(dat3C)
```

```
##  pelvic_incidence  pelvic_tilt     lumbar_lordosis_angle  sacral_slope
##  Min.   : 26.15   Min.   :-6.555   Min.   : 14.00        Min.   : 13.37
##  1st Qu.: 46.43   1st Qu.:10.667   1st Qu.: 37.00        1st Qu.: 33.35
##  Median : 58.69   Median :16.358   Median : 49.56        Median : 42.40
##  Mean   : 60.50   Mean   :17.543   Mean   : 51.93        Mean   : 42.95
##  3rd Qu.: 72.88   3rd Qu.:22.120   3rd Qu.: 63.00        3rd Qu.: 52.70
##  Max.   :129.83   Max.   :49.432   Max.   :125.74        Max.   :121.43
##  pelvic_radius    degree_spondylolisthesis    class
##  Min.   : 70.08   Min.   :-11.058          Length:310
##  1st Qu.:110.71   1st Qu.:  1.604          Class :character
##  Median :118.27   Median : 11.768          Mode  :character
##  Mean   :117.92   Mean   : 26.297
##  3rd Qu.:125.47   3rd Qu.: 41.287
##  Max.   :163.07   Max.   :418.543
```

```
### We can look at the first six lines of the `dat3C` with the
### `head` function:
head(dat3C)
```

```
## # A tibble: 6 x 7
##   pelvic_incidence pelvic_tilt lumbar_lordosis~ sacral_slope pelvic_radius
##              <dbl>       <dbl>            <dbl>        <dbl>         <dbl>
## 1             63.0       22.6              39.6         40.5          98.7
## 2             39.1       10.1              25.0         29.0         114.
## 3             68.8       22.2              50.1         46.6         106.
## 4             69.3       24.7              44.3         44.6         102.
## 5             49.7        9.65             28.3         40.1         108.
## 6             40.3       13.9              25.1         26.3         130.
## # ... with 2 more variables: degree_spondylolisthesis <dbl>, class <chr>
```

```
### We can check that the data is in tidy format with the
### `as_tibble` function:
dat3C %>% as_tibble
```

```
## # A tibble: 310 x 7
##    pelvic_incidence pelvic_tilt lumbar_lordosis~ sacral_slope pelvic_radius
##               <dbl>       <dbl>            <dbl>        <dbl>         <dbl>
##  1             63.0       22.6              39.6         40.5          98.7
##  2             39.1       10.1              25.0         29.0         114.
##  3             68.8       22.2              50.1         46.6         106.
```

```
## 4             69.3       24.7              44.3        44.6         102.
## 5             49.7        9.65             28.3        40.1         108.
## 6             40.3       13.9              25.1        26.3         130.
## 7             53.4       15.9              37.2        37.6         121.
## 8             45.4       10.8              29.0        34.6         117.
## 9             43.8       13.5              42.7        30.3         125.
## 10            36.7        5.01             41.9        31.7          84.2
## # ... with 300 more rows, and 2 more variables:
## #   degree_spondylolisthesis <dbl>, class <chr>
```

```
### We then check that there is no missing data:
any(is.na(dat3C))
```

```
## [1] FALSE
```

```
sum(is.na(dat3C))
```

```
## [1] 0
```

2. Methods and Analysis

2.1 Data Cleaning and Exploration

Before commencing the data visualisation process, it is important to get the dataset into the right format in R.

Transform dataset into dataframe for classification purposes

```
class(dat3C)
```

```
## [1] "spec_tbl_df" "tbl_df"      "tbl"         "data.frame"
```

```
dat3C$class <- as.factor(dat3C$class)
dat3C <- as.data.frame(dat3C)
class(dat3C)
```

```
## [1] "data.frame"
```

2.2 Data Visualisation

To gain an insight into the dataset's trends and patterns, we will map the data for each of the three classes, split by the six biomechanical features, using boxplots.

Pelvic incidence

```
options(scipen = 999)
dat3C %>% group_by(class) %>% ggplot(aes(class, pelvic_incidence)) +
    geom_boxplot(aes(class, pelvic_incidence, col = class)) +
    ggtitle("Pelvic Incidence per Class with Individual Data") +
    xlab("Class") + ylab("Pelvic Incidence per Individual") +
    geom_point(alpha = 0.1)
```

From the plot, it would seem that pelvic incidence is a particularly strong predictor of Spondylolisthesis.

Pelvic tilt

```
options(scipen = 999)
dat3C %>% group_by(class) %>% ggplot(aes(class, pelvic_tilt)) +
    geom_boxplot(aes(class, pelvic_tilt, col = class)) + ggtitle("Pelvic Tilt per Class with Individual Da
    xlab("Class") + ylab("Pelvic Tilt per Individual") + geom_point(alpha = 0.1)
```

Pelvic tilt may be an important predictor of both Hernia and Spondylolisthesis.

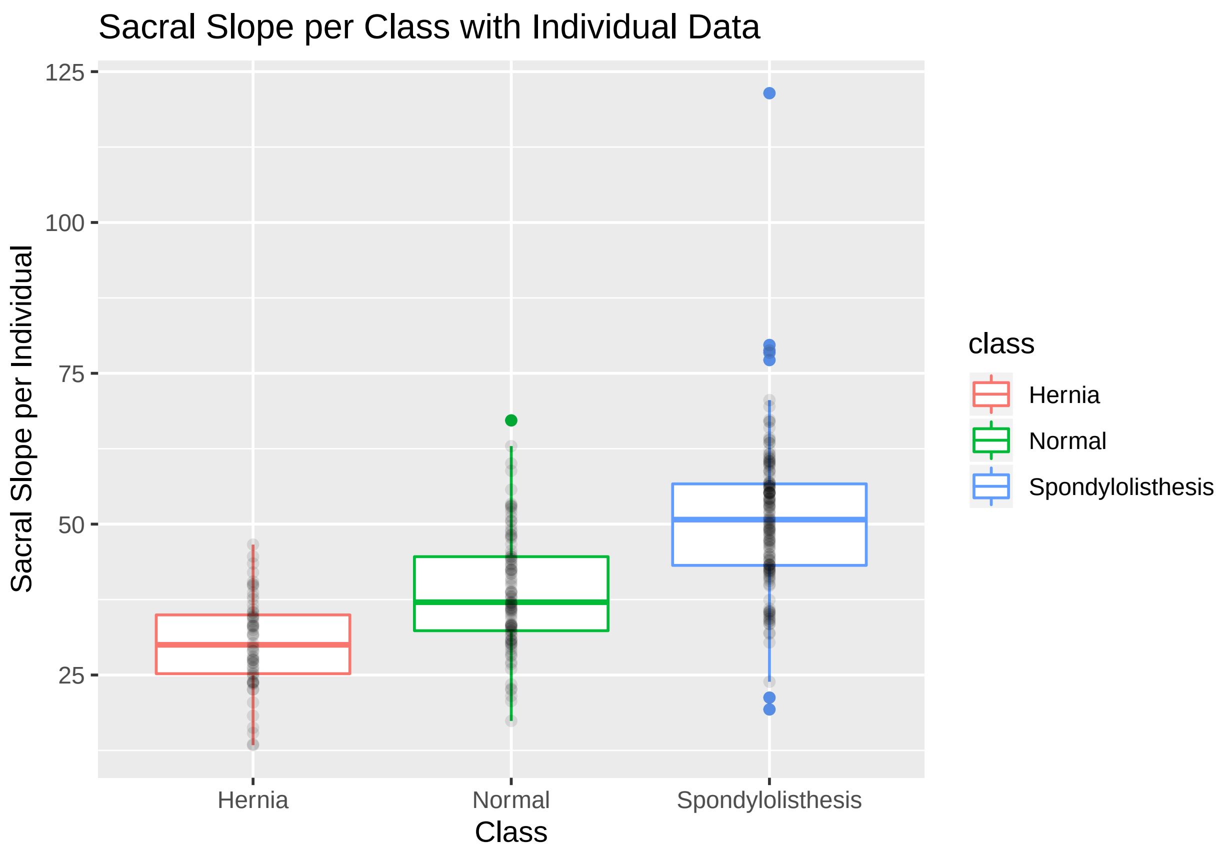
Lumbar lordosis angle

```
options(scipen = 999)
dat3C %>% group_by(class) %>% ggplot(aes(class, lumbar_lordosis_angle)) +
    geom_boxplot(aes(class, lumbar_lordosis_angle, col = class)) +
    ggtitle("Lumbar Lordosis Angle per Class with Individual Data") +
    xlab("Class") + ylab("Lumbar Lordosis Angle per Individual") +
    geom_point(alpha = 0.1)
```

The lumbar lordosis angle is generally higher in patients classified as having Spondylolisthesis.

Sacral slope

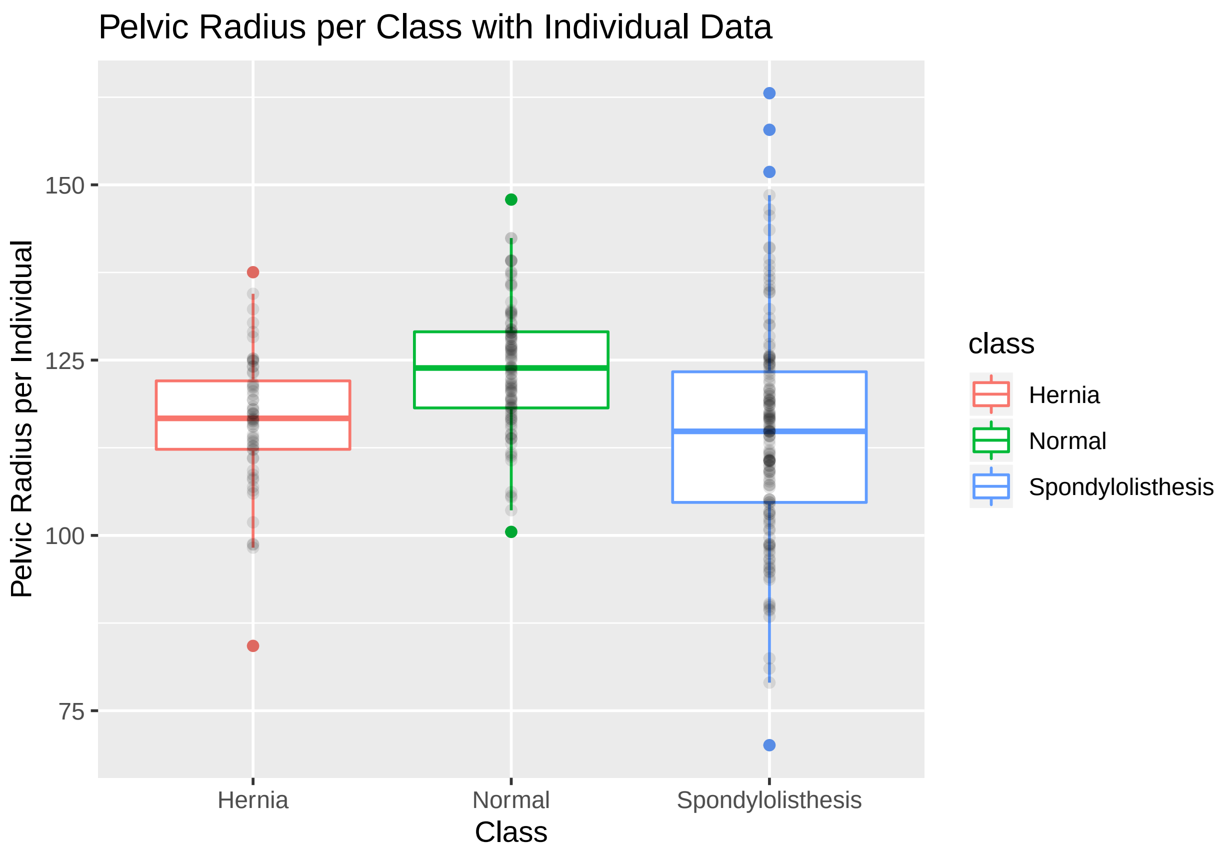
```
options(scipen = 999)
dat3C %>% group_by(class) %>% ggplot(aes(class, sacral_slope)) +
    geom_boxplot(aes(class, sacral_slope, col = class)) + ggtitle("Sacral Slope per Class with Individual I
    xlab("Class") + ylab("Sacral Slope per Individual") + geom_point(alpha = 0.1)
```



The sacral slope is highest in individuals with Spondylolisthesis and lowest in those with Disk Herni, in stratified fashion.

Pelvic radius

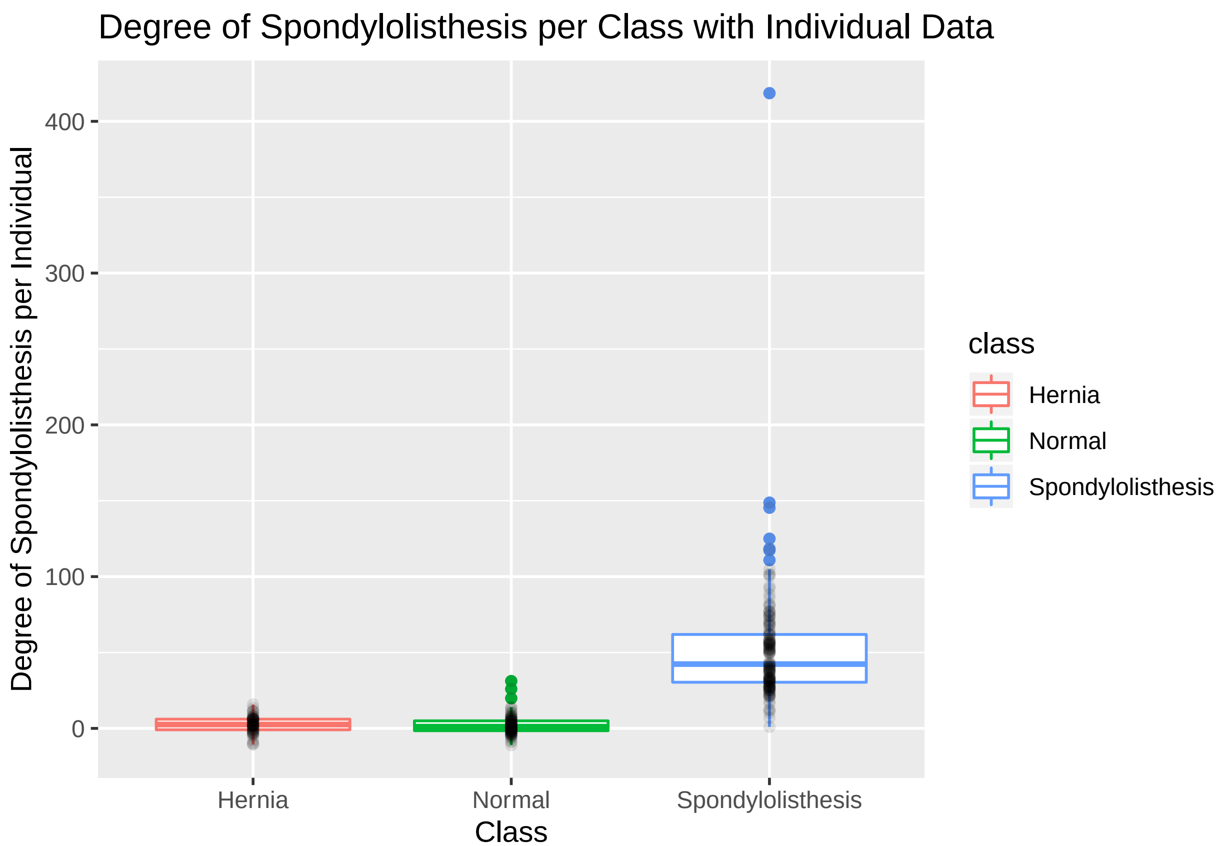
```
options(scipen = 999)
dat3C %>% group_by(class) %>% ggplot(aes(class, pelvic_radius)) +
    geom_boxplot(aes(class, pelvic_radius, col = class)) + ggtitle("Pelvic Radius per Class with Individual
    xlab("Class") + ylab("Pelvic Radius per Individual") + geom_point(alpha = 0.1)
```



The pelvic radius is similar across the three classes.

Degree of spondylolisthesis

```
options(scipen = 999)
dat3C %>% group_by(class) %>% ggplot(aes(class, degree_spondylolisthesis)) +
    geom_boxplot(aes(class, degree_spondylolisthesis, col = class)) +
    ggtitle("Degree of Spondylolisthesis per Class with Individual Data") +
    xlab("Class") + ylab("Degree of Spondylolisthesis per Individual") +
    geom_point(alpha = 0.1)
```



The degree of spondylolisthesis is only really relevant for the Spondylolisthesis patients.

3. Results and Discussion

Before applying machine learning algorithms, the first step is to split our data into train and test sets.

3.1 Partitioning the data into train and test sets

```
set.seed(1, sample.kind = "Rounding")
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding'
## sampler used
```

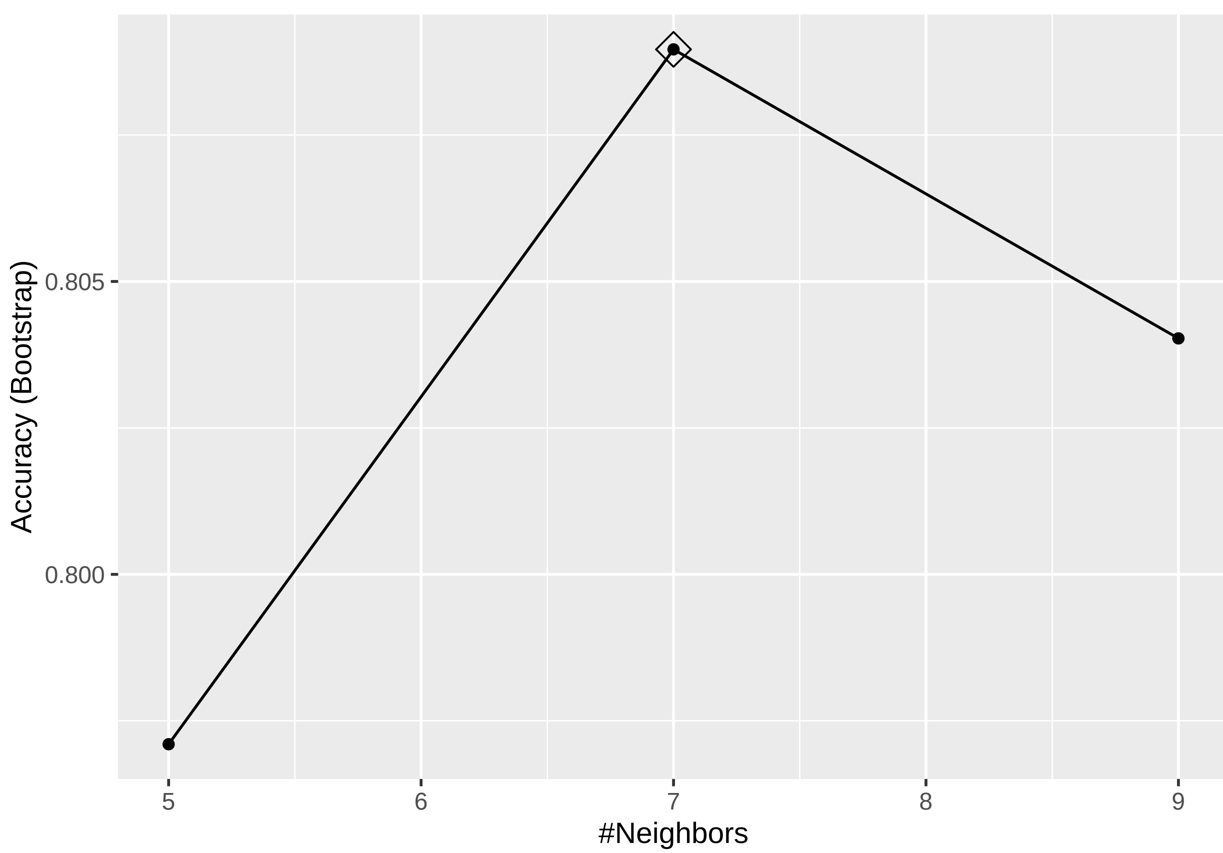
```
test_index1 <- createDataPartition(y = dat3C$class, times = 1,
    p = 0.2, list = FALSE)
train_set1 <- dat3C[-test_index1, ]  ### Create train set
test_set1 <- dat3C[test_index1, ]  ### Create test set
```

3.2 k-nearest neighbors - train and test sets

```
fit_knn <- train(class ~ ., method = "knn", data = train_set1)
fit_knn
```

```
## k-Nearest Neighbors
##
## 248 samples
##   6 predictor
##   3 classes: 'Hernia', 'Normal', 'Spondylolisthesis'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 248, 248, 248, 248, 248, 248, ...
## Resampling results across tuning parameters:
##
##   k  Accuracy   Kappa
##   5  0.7971004  0.6738748
##   7  0.8089624  0.6929989
##   9  0.8040285  0.6854077
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 7.
```

```
ggplot(fit_knn, highlight = TRUE)
```




```
fit_knn$bestTune
```

```
##   k
## 2 7
```

```r
fit_knn$finalModel
```

```
## 7-nearest neighbor model
## Training set outcome distribution:
##
##            Hernia            Normal Spondylolisthesis
##                48                80               120
```

```r
y_hat_knn <- predict(fit_knn, test_set1, type = "raw")
confusionMatrix(y_hat_knn, test_set1$class)$overall[["Accuracy"]]
```

```
## [1] 0.9032258
```

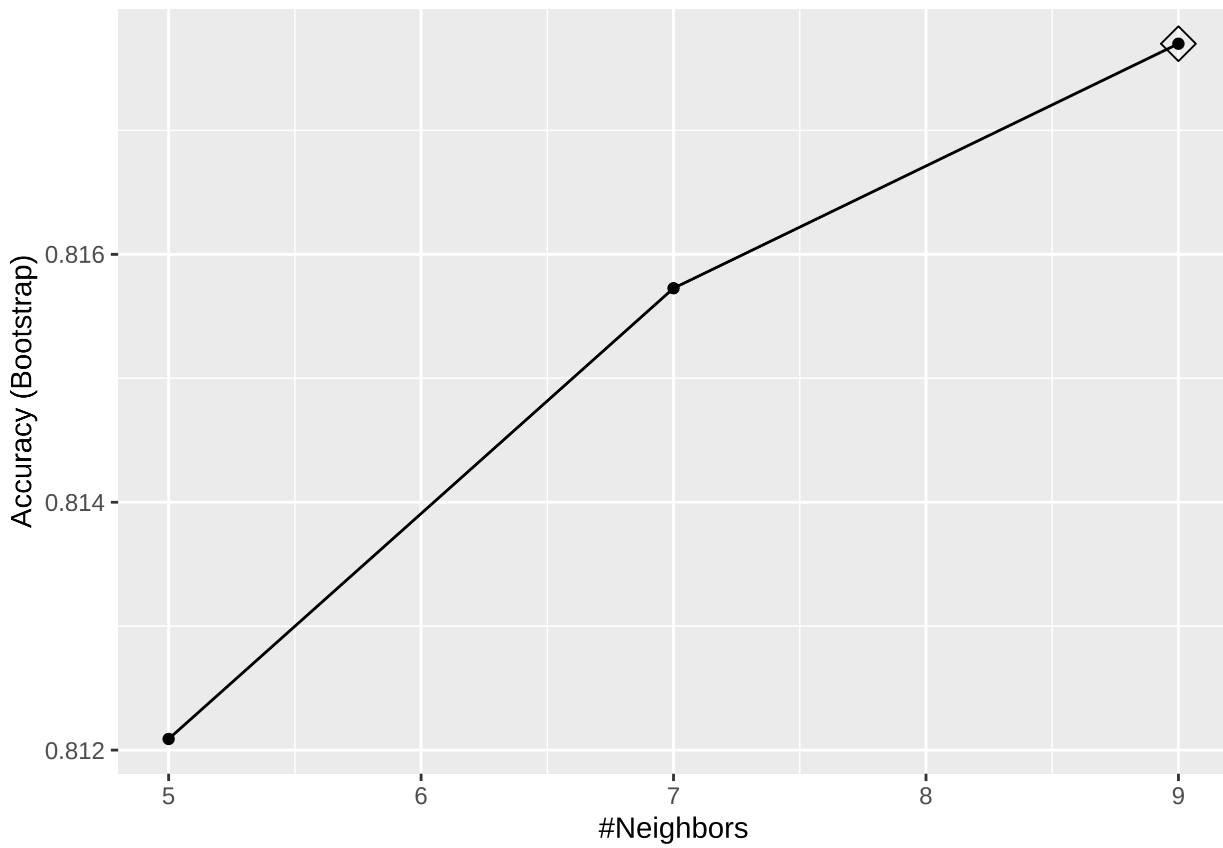
The first step is to try and predict the appropriate class usng k-nearest neighbours. We can see that by using just one neighbour, we have a high accuracy (~0.90). However, we can do better.

3.3 k-nearest neighbors - whole dataset

```r
fit_knn1 <- train(class ~ ., method = "knn", data = dat3C)
fit_knn1
```

```
## k-Nearest Neighbors
##
## 310 samples
##   6 predictor
##   3 classes: 'Hernia', 'Normal', 'Spondylolisthesis'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 310, 310, 310, 310, 310, 310, ...
## Resampling results across tuning parameters:
##
##   k  Accuracy   Kappa
##   5  0.8120900  0.7017458
##   7  0.8157256  0.7077508
##   9  0.8176982  0.7103226
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 9.
```

```r
ggplot(fit_knn1, highlight = TRUE)
```




```
fit_knn1$bestTune
```

```
##   k
## 3 9
```

```
fit_knn1$finalModel
```

```
## 9-nearest neighbor model
## Training set outcome distribution:
##
##            Hernia            Normal Spondylolisthesis
##                60               100               150
```

We were able to accurately put the 100 Normal, 60 Disk Hernia, and 150 Spondylolisthesis patients into the right class.

3.4 rpart - whole dataset

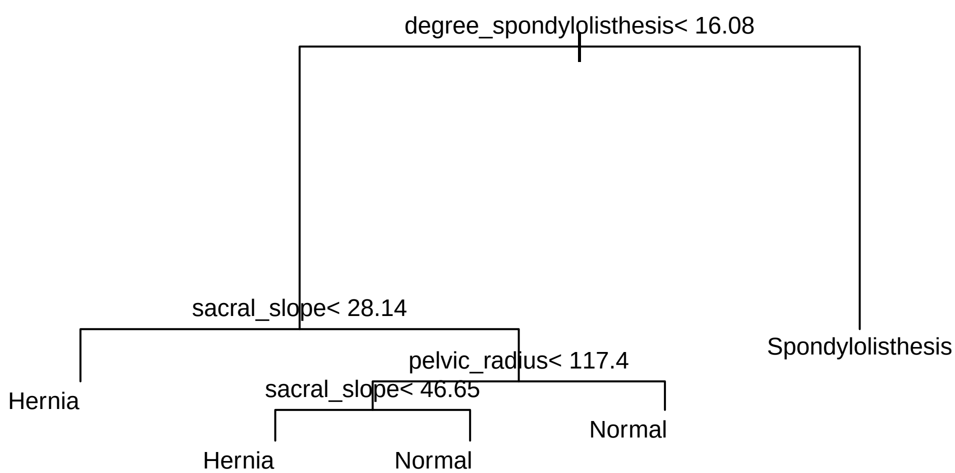
```
install.packages("rpart")
```

```
##
## The downloaded binary packages are in
##  /var/folders/4w/60c04z2j2fn103mghfmd8ntr0000gn/T//RtmpUploQ4/downloaded_packages
```

```r
library(rpart)
fit_2 <- rpart(class ~ ., data = dat3C)
fit_2
```

```
## n= 310
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 310 160 Spondylolisthesis (0.19354839 0.32258065 0.48387097)
##    2) degree_spondylolisthesis< 16.07889 162  65 Normal (0.37037037 0.59876543 0.03086420)
##      4) sacral_slope< 28.13647 35   9 Hernia (0.74285714 0.25714286 0.00000000) *
##      5) sacral_slope>=28.13647 127  39 Normal (0.26771654 0.69291339 0.03937008)
##       10) pelvic_radius< 117.3596 47  23 Hernia (0.51063830 0.40425532 0.08510638)
##         20) sacral_slope< 46.64911 34  10 Hernia (0.70588235 0.26470588 0.02941176) *
##         21) sacral_slope>=46.64911 13   3 Normal (0.00000000 0.76923077 0.23076923) *
##       11) pelvic_radius>=117.3596 80  11 Normal (0.12500000 0.86250000 0.01250000) *
##    3) degree_spondylolisthesis>=16.07889 148   3 Spondylolisthesis (0.00000000 0.02027027 0.97972973) *
```

```r
plot(fit_2, margin = 0.1)
text(fit_2, cex = 0.75)
```



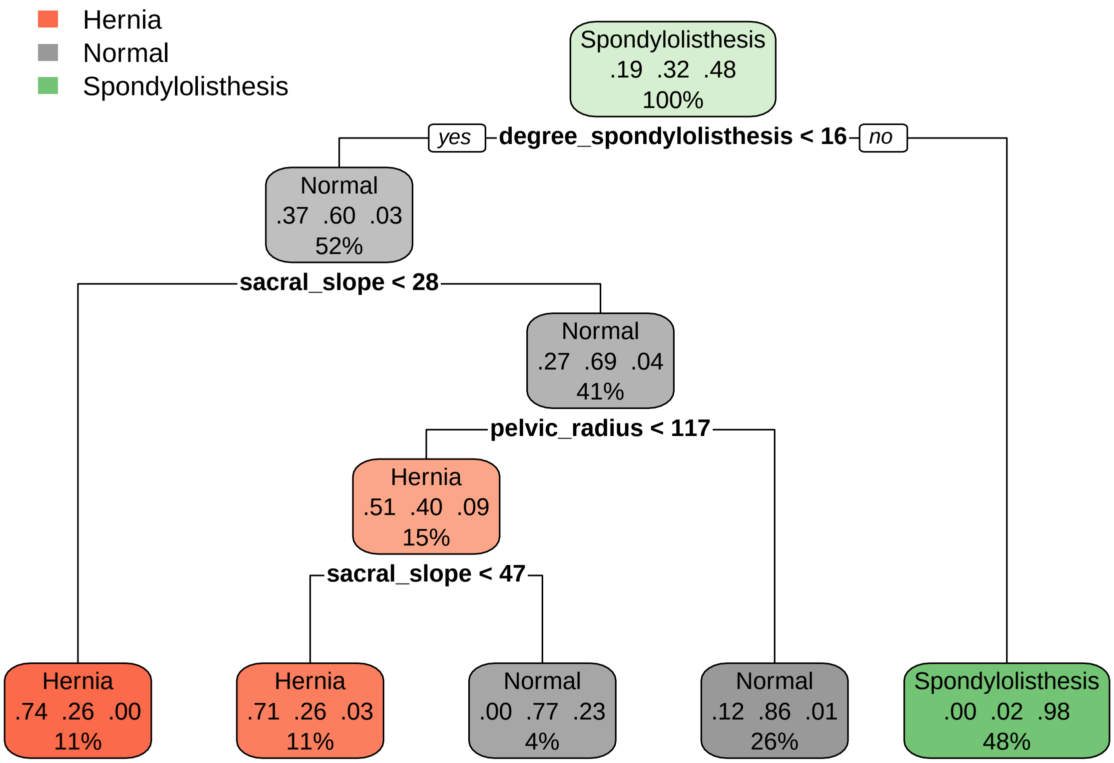

```r
install.packages("rpart.plot")
```

```
##
## The downloaded binary packages are in
##  /var/folders/4w/60c04z2j2fn103mghfmd8ntr0000gn/T//RtmpUploQ4/downloaded_packages
```

```r
library(rpart.plot)
install.packages("RColorBrewer")
```

```
##
## The downloaded binary packages are in
##  /var/folders/4w/60c04z2j2fn103mghfmd8ntr0000gn/T//RtmpUploQ4/downloaded_packages
```

```r
library(RColorBrewer)
rpart.plot(fit_2)
```




```r
printcp(fit_2)
```

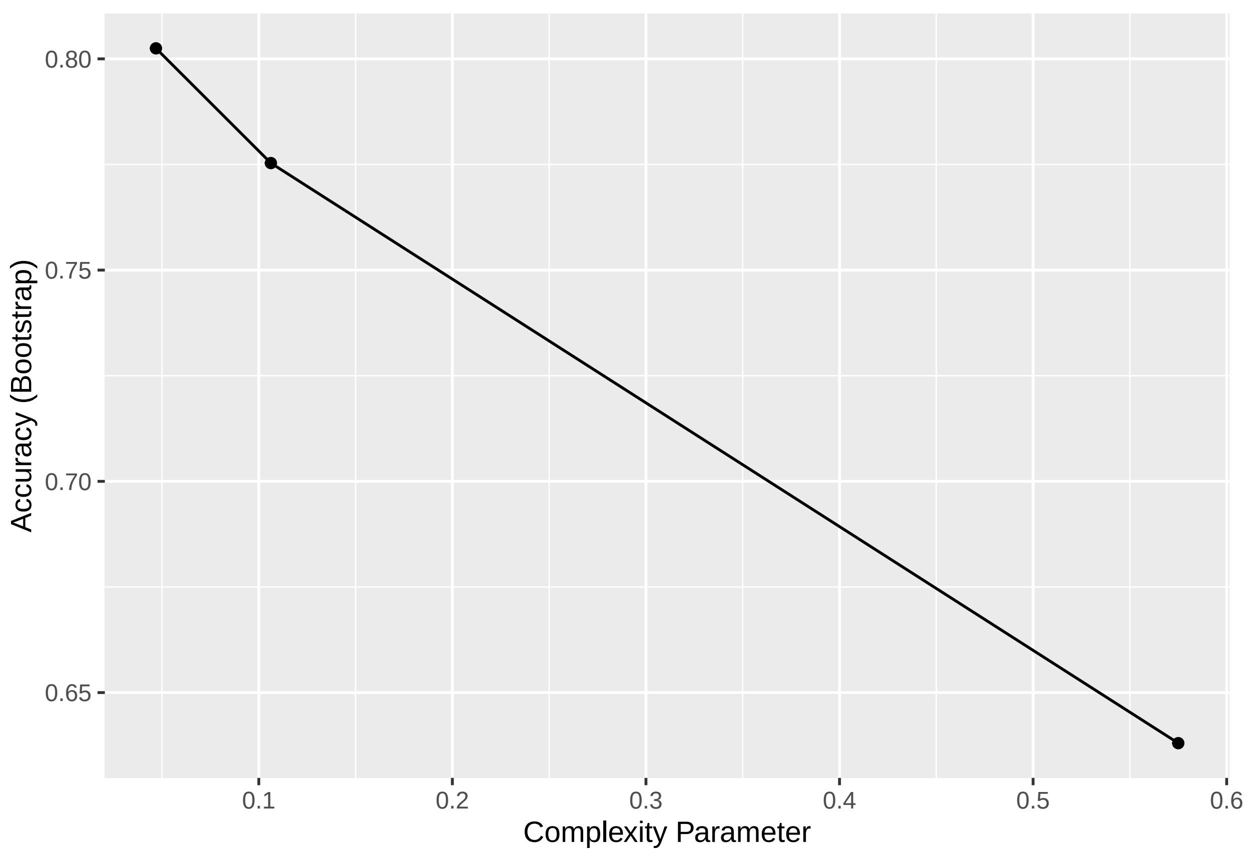
```
##
## Classification tree:
## rpart(formula = class ~ ., data = dat3C)
##
## Variables actually used in tree construction:
## [1] degree_spondylolisthesis pelvic_radius
## [3] sacral_slope
##
## Root node error: 160/310 = 0.51613
##
## n= 310
##
##          CP nsplit rel error xerror     xstd
## 1 0.575000      0   1.00000 1.0000 0.054993
## 2 0.106250      1   0.42500 0.4375 0.046010
## 3 0.046875      2   0.31875 0.4125 0.045047
## 4 0.010000      4   0.22500 0.3375 0.041737
```

```r
train_rpart <- train(class ~ ., method = "rpart", data = dat3C)
train_rpart
```

```
## CART
##
## 310 samples
##   6 predictor
##   3 classes: 'Hernia', 'Normal', 'Spondylolisthesis'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
```

```
## Summary of sample sizes: 310, 310, 310, 310, 310, 310, ...
## Resampling results across tuning parameters:
##
##   cp        Accuracy   Kappa
##   0.046875  0.8024868  0.6825661
##   0.106250  0.7753264  0.6362460
##   0.575000  0.6380329  0.3262683
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.046875.
```

```
ggplot(train_rpart)
```




```
plot(train_rpart$finalModel, margin = 0.1)
text(train_rpart$finalModel, cex = 0.75)
```


```r
pruned_fit <- prune(fit_2, cp = 0.01)
pruned_fit
```

```
## n= 310
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 310 160 Spondylolisthesis (0.19354839 0.32258065 0.48387097)
##    2) degree_spondylolisthesis< 16.07889 162  65 Normal (0.37037037 0.59876543 0.03086420)
##      4) sacral_slope< 28.13647 35   9 Hernia (0.74285714 0.25714286 0.00000000) *
##      5) sacral_slope>=28.13647 127  39 Normal (0.26771654 0.69291339 0.03937008)
##       10) pelvic_radius< 117.3596 47  23 Hernia (0.51063830 0.40425532 0.08510638)
##         20) sacral_slope< 46.64911 34  10 Hernia (0.70588235 0.26470588 0.02941176) *
##         21) sacral_slope>=46.64911 13   3 Normal (0.00000000 0.76923077 0.23076923) *
##       11) pelvic_radius>=117.3596 80  11 Normal (0.12500000 0.86250000 0.01250000) *
##    3) degree_spondylolisthesis>=16.07889 148   3 Spondylolisthesis (0.00000000 0.02027027 0.97972973) *
```

```r
ind <- !(train_rpart$finalModel$frame$var == "<leaf>")
tree_terms <- train_rpart$finalModel$frame$var[ind] %>% unique() %>%
    as.character()
tree_terms
```

```
## [1] "degree_spondylolisthesis" "sacral_slope"
```

The next step is to use the partitioning method using the `rpart` function and package. This method generated two different trees. Degree of spondylolisthesis and sacral slope appear to be the most obvious predictors to clearly distinguish the three classes. We used the complexity parameter (cp) to decide whether or not to partition. We used the `prune` function to select the cp criterion.

3.5 rpart - train and test sets

```r
library(caret)
train_rpart1 <- train(class ~ pelvic_incidence + pelvic_tilt +
    lumbar_lordosis_angle + sacral_slope + pelvic_radius + degree_spondylolisthesis,
    method = "rpart", data = train_set1)
train_rpart1
```

```
## CART
##
## 248 samples
##   6 predictor
##   3 classes: 'Hernia', 'Normal', 'Spondylolisthesis'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 248, 248, 248, 248, 248, 248, ...
## Resampling results across tuning parameters:
##
##   cp         Accuracy   Kappa
##   0.0546875  0.8140727  0.7010568
##   0.0703125  0.8101611  0.6938301
##   0.5703125  0.6444532  0.3460073
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.0546875.
```

```r
confusionMatrix(predict(train_rpart1, test_set1), test_set1$class)$overall["Accuracy"]
```

```
##  Accuracy
## 0.7903226
```

```r
ind <- !(train_rpart1$finalModel$frame$var == "<leaf>")
tree_terms <- train_rpart1$finalModel$frame$var[ind] %>% unique() %>%
    as.character()
tree_terms
```

```
## [1] "degree_spondylolisthesis" "sacral_slope"
## [3] "pelvic_radius"
```

Using the train and test sets, we observe that the resulting accuracy is lower than that of our initial model. Degree of spondylolisthesis, sacral slope, and pelvic tilt, were the major predictors.

3.6 randomForest - whole dataset

```r
library(randomForest)
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

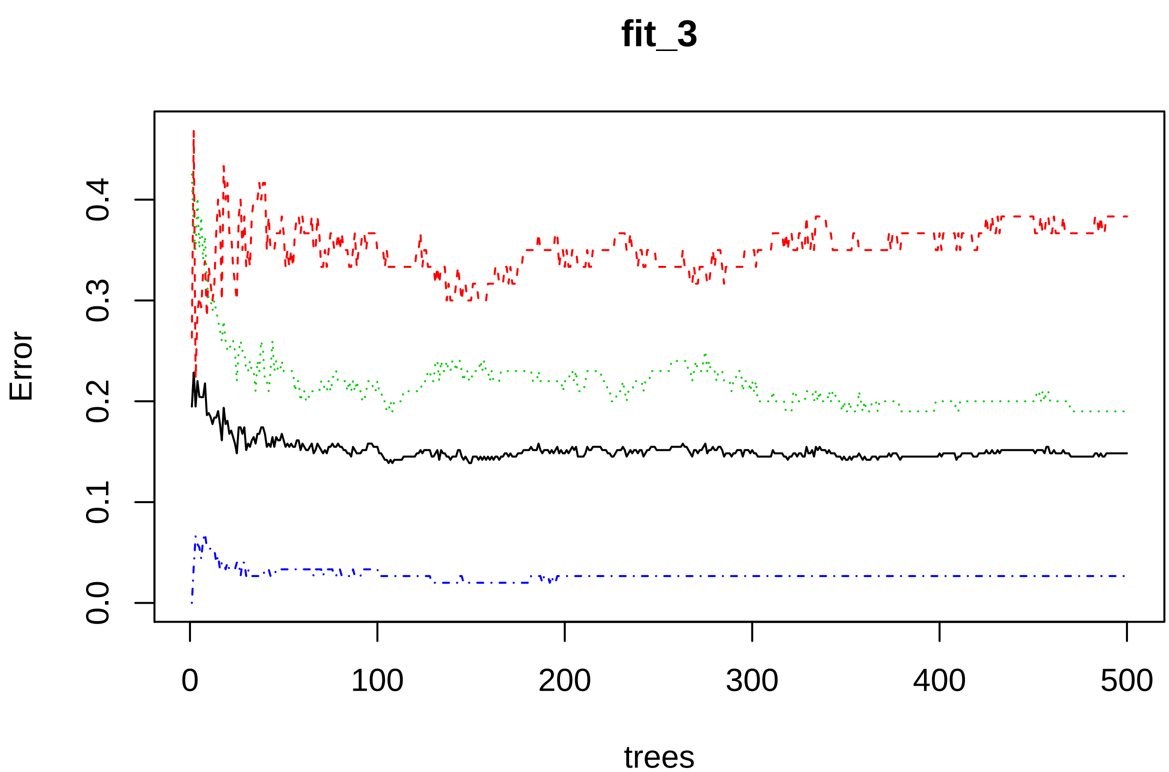
```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```r
fit_3 <- randomForest(class ~ ., data = dat3C)
fit_3
```

```
##
## Call:
##  randomForest(formula = class ~ ., data = dat3C)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 14.84%
## Confusion matrix:
##                   Hernia Normal Spondylolisthesis class.error
## Hernia                37     22                 1  0.38333333
## Normal                14     81                 5  0.19000000
## Spondylolisthesis      0      4               146  0.02666667
```

```r
plot(fit_3)
```

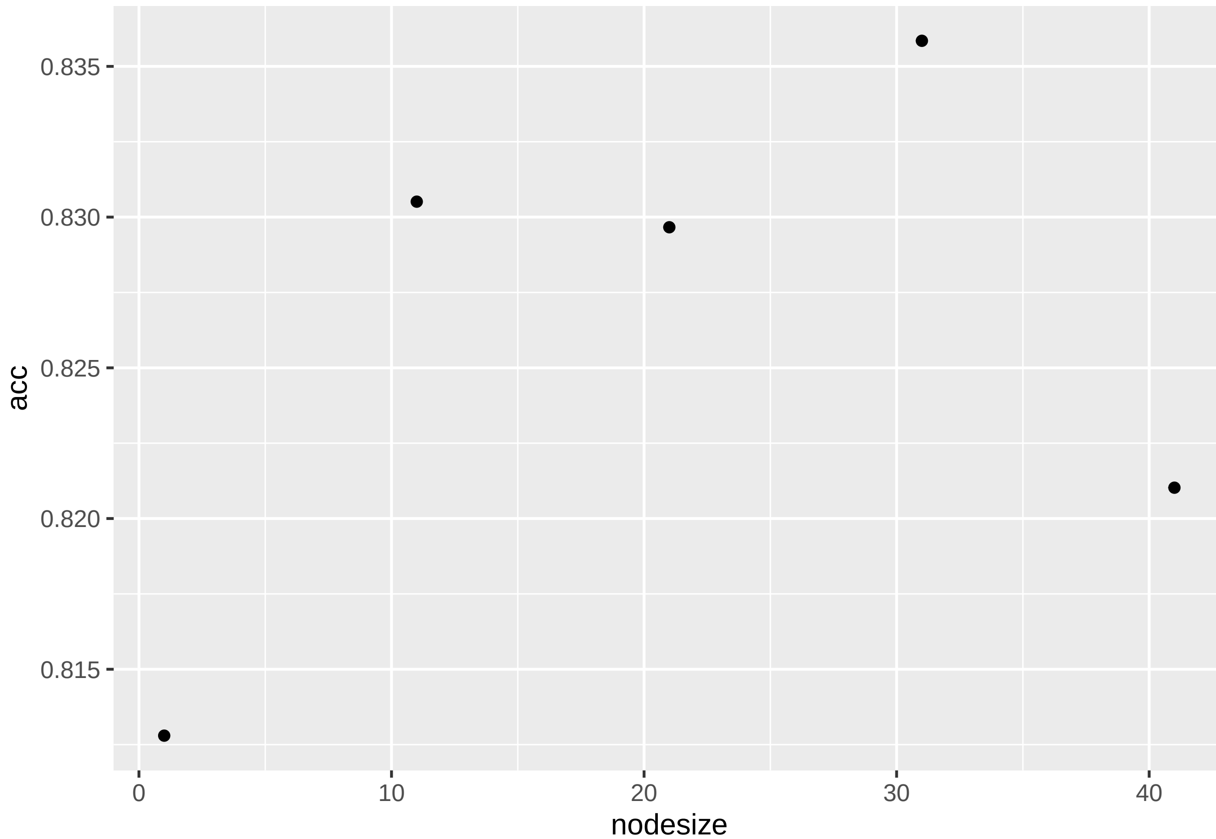


Although the classification trees thus far have been easily interpretable, it is useful to experiment with random forests to address the shortcomings of these previous methods. In particular, this method appears more robust as it averages across multiple decision trees, and improves upon decision trees which tend to over-fit to the training set.

3.7 randomForest - train and test sets

```r
nodesize <- seq(1, 50, 10)
acc <- sapply(nodesize, function(ns) {
    train(class ~ ., method = "rf", data = train_set1, tuneGrid = data.frame(mtry = 2),
```

```r
        nodesize = ns)$results$Accuracy
})
qplot(nodesize, acc)
```




```r
train_rf <- randomForest(class ~ ., data = train_set1, ns = ns[which.max(acc)])
train_rf
```

```
##
## Call:
##  randomForest(formula = class ~ ., data = train_set1, ns = ns[which.max(acc)])
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 17.34%
## Confusion matrix:
##                   Hernia Normal Spondylolisthesis class.error
## Hernia                27     21                 0  0.43750000
## Normal                14     62                 4  0.22500000
## Spondylolisthesis      0      4               116  0.03333333
```

```r
confusionMatrix(predict(train_rf, test_set1), test_set1$class)$overall["Accuracy"]
```

```
##  Accuracy
## 0.8709677
```

Using the train and test sets, we can see that our accuracy is substantially higher in comparison to the partitioning function. We used the `caret` package to optimise over the minimum node size.

3.8 Multiclass support vector machine

```r
install.packages("e1071", repos = "https://cran.r-project.org/package=e1071")
```

```
## Warning: unable to access index for repository https://cran.r-project.org/package=e1071/src/contrib:
##   cannot open URL 'https://cran.r-project.org/package=e1071/src/contrib/PACKAGES'

## Warning: package 'e1071' is not available (for R version 3.6.1)

## Warning: unable to access index for repository https://cran.r-project.org/package=e1071/bin/macosx/el-ca
##   cannot open URL 'https://cran.r-project.org/package=e1071/bin/macosx/el-capitan/contrib/3.6/PACKAGES'
```

```r
library(e1071)
svm1 <- svm(class ~ ., data = train_set1, method = "C-classification",
    kernal = "radial", gamma = 0.1, cost = 10)
summary(svm1)
```

```
##
## Call:
## svm(formula = class ~ ., data = train_set1, method = "C-classification",
##     kernal = "radial", gamma = 0.1, cost = 10)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  10
##
## Number of Support Vectors:  103
##
##  ( 34 26 43 )
##
##
## Number of Classes:  3
##
## Levels:
##  Hernia Normal Spondylolisthesis
```

```r
svm1$SV
```

```
##     pelvic_incidence  pelvic_tilt lumbar_lordosis_angle sacral_slope
## 2        -1.26523479 -0.797845936           -1.48193946 -1.035178915
## 3         0.43517877  0.423675816           -0.13807450  0.247865190
## 5        -0.65669042 -0.838931760           -1.30498400 -0.229355477
## 6        -1.19708998 -0.409921162           -1.47606753 -1.229458366
## 7        -0.44424237 -0.214755891           -0.83079195 -0.410855598
## 8        -0.90489058 -0.728054116           -1.26634871 -0.626239250
## 10       -1.40061641 -1.305254594           -0.57448108 -0.840036981
## 13       -0.70222359  0.197213004           -0.66477784 -1.038453552
## 15       -0.22338625  0.621651670           -0.30378496 -0.735465937
## 17        0.14980469  0.237197052            0.10094948  0.019109353
## 18       -1.70959359 -1.492762186           -1.07746310 -1.098146813
## 19       -1.28573899 -0.457863483           -1.16122395 -1.307757365
```

```
## 20       -1.11258272 -0.577496342            -1.20824449 -1.000226855
## 21       -0.98735052 -0.384194387            -0.79506821 -0.980636820
## 23        0.10632385  0.644250278             0.07134460 -0.331385059
## 24       -0.89495161 -0.495556759            -1.19882691 -0.782086431
## 27       -2.00245183 -0.727668078            -2.07225288 -2.026178392
## 30        0.32365649  0.692296956            -0.18167511 -0.089058556
## 31       -0.59350495 -0.261187679            -0.54338885 -0.567546621
## 37       -1.66305148 -1.106465757            -0.89337081 -1.318794713
## 38       -1.45674820  0.144835171            -1.71320030 -1.962691982
## 39       -0.30659013  1.089719973            -0.26677879 -1.180842718
## 40       -0.50212443  0.101460933            -0.90009844 -0.713873851
## 42       -0.84348025 -0.965232533            -1.26640866 -0.376011249
## 43       -0.42014994  0.123473241            -1.06588425 -0.625291553
## 44        0.28974437  0.836563328            -0.27698999 -0.236873861
## 45       -0.29591471 -0.171192139             0.51481604 -0.253277967
## 49       -1.17954867 -0.002407576            -1.00045414 -1.502468277
## 51       -0.33842444  0.244995034            -1.00045414 -0.609154559
## 53       -0.62831825  1.181420797            -0.88770052 -1.657592650
## 54       -1.77390222 -0.611319066            -1.00045414 -1.819055314
## 58       -0.81985428 -0.266281612            -0.78609439 -0.852505195
## 59       -1.02844903  0.166928843            -0.94686421 -1.432519897
## 60       -0.74827114 -0.308560335            -0.91660289 -0.730574136
## 63       -0.95273063 -0.860919737            -0.03583528 -0.590941615
## 65        0.85293934  0.395312125             1.62338969  0.801171150
## 66        1.29757554  2.339512055             0.50006409 -0.041029508
## 67        0.98683157  0.419989972             0.39288422  0.954029203
## 69        0.62045388  0.094891439            -0.08942522  0.722450154
## 80       -0.76909052 -0.594046898            -0.73250446 -0.550193810
## 86       -0.90049345 -0.813411463            -0.41096484 -0.558761987
## 87       -0.08146908 -0.012300403             0.35035854 -0.094977287
## 89       -0.26304564 -0.120725114            -0.57173465 -0.247942026
## 90        0.56965762  0.592244347            -0.48081047  0.297174580
## 101       1.33484455  1.241862699             0.68652951  0.802110314
## 104      -0.65273501 -1.158971211             0.01775466  0.007663866
## 106       0.21679212  0.964644897            -0.09223767 -0.422743623
## 113      -1.09593995 -2.467329600             0.81624472  0.390810657
## 116       3.91892076 -0.964284140            -0.22961343  5.696528691
## 131      -0.63647816 -0.892357711            -1.09860611 -0.164854986
## 132       0.48937402 -0.424433873             0.28570434  0.931715469
## 155      -1.14354664 -1.226674262            -0.52525119 -0.569167720
## 163       3.25135591  2.054484650            -0.09807894  2.657116176
## 164       3.12449512  1.960634606             1.29319516  2.563363239
## 168       0.63571993 -0.158846323             0.38586460  0.925835659
## 171       0.20546046 -0.284109460             0.33071954  0.467944369
## 174      -0.59317595 -0.897945230             0.19460145 -0.105584046
## 175       0.01141650  0.741767218            -0.72731324 -0.523098685
## 181      -1.33108337 -1.358685616            -1.49828995 -0.712636760
## 210      -0.73966582 -0.159180629            -0.87563583 -0.827875626
## 211      -1.29674037 -0.104237522            -0.94081851 -1.578106665
## 214      -0.73629579 -0.055223477            -0.25019503 -0.898929608
## 216      -1.74009381 -0.466884424            -0.89844431 -1.880632334
## 220       0.17703897  0.836615994            -0.09162432 -0.380639019
## 221      -0.95500220  0.380265726            -1.13581407 -1.493490865
## 223      -0.29171404 -0.491868423             0.53420193 -0.015484265
## 224       0.46754780  0.090091846             1.24852052  0.530936364
## 225       1.63461172  0.465949065             2.03077819  1.746794825
```

```
## 227       0.15691700 -0.195001662           0.56028505  0.341451274
## 228       0.01877562  0.168316542           0.01197919 -0.098057913
## 229      -1.32293736 -0.974613513          -1.41648193 -0.980636820
## 232      -0.41693724 -0.508646582          -0.73250446 -0.163013138
## 235      -1.34090139 -0.865635617          -0.57173465 -1.082536148
## 236       0.15520079  0.197870165          -0.66942644  0.054496140
## 241      -1.11732543 -0.920972045          -0.89156074 -0.757312375
## 242      -0.55295269 -0.450518946          -0.94686421 -0.378598974
## 245       0.10362199  0.937881679          -0.05700241 -0.547664385
## 247      -0.83229922 -0.215822835          -0.67891452 -0.904949741
## 248      -0.65010719 -0.127131915          -1.32199376 -0.736896416
## 254       0.14737435  0.335945352           0.71442384 -0.055565969
## 256       0.36128937 -0.336259486           0.28578093  0.704464455
## 260       0.13748124 -0.107230577          -0.17788815  0.253046530
## 262       0.78605425 -0.309466597           0.05687916  1.226723300
## 263      -1.06762055 -0.364329234          -1.46542373 -1.097399685
## 270      -1.37470068 -0.152772609          -1.53635351 -1.642345308
## 272      -1.06771549 -0.146746485          -0.57173465 -1.255231953
## 273      -1.24800201 -1.104266994          -0.83968433 -0.791098745
## 275      -1.02909262 -0.806318744          -1.27171840 -0.727898467
## 280      -0.70871358  0.001609201          -0.03583528 -0.904949741
## 281      -0.63537215 -0.459340947          -0.97593458 -0.477309490
## 282       0.17416298 -0.352051785          -0.46977960  0.477279270
## 288      -1.60875826 -1.841344317          -1.80049431 -0.716893176
## 291      -1.41568551 -0.414189542          -1.73771438 -1.505127323
## 292      -0.57862443 -0.380981527          -0.89588791 -0.461739757
## 294      -0.70294584 -1.247310306           0.15172950  0.007663866
## 297      -1.23097240 -0.180141980          -0.85710873 -1.439218328
## 298      -0.89297037  0.076098567          -1.01255781 -1.193914637
## 300       1.23888740  1.194884097           0.29913089  0.713793095
## 302       1.58779401  0.811257396           0.87633292  1.436799827
## 303      -0.37757444  0.350378648          -1.24909986 -0.735465937
## 306      -0.76001674 -0.440587926          -0.89327427 -0.649854671
## 308       0.01340728  0.471550689          -0.34824600 -0.324698033
## 309      -0.91139877 -0.935278830          -0.59407488 -0.484335428
##     pelvic_radius degree_spondylolisthesis
## 2    -0.249657107           -0.57198935638
## 3    -0.873785040           -0.78049363393
## 5    -0.711933233           -0.48558905545
## 6     0.930544925           -0.63209961126
## 7     0.207089332           -0.53530170431
## 8    -0.037324376           -0.96455250015
## 10   -2.485471026           -0.67244297796
## 13    0.114721185           -0.48274546930
## 15   -0.071678448           -0.54100988865
## 17   -0.405011722           -0.70559325666
## 18    0.832966920           -0.59623426673
## 19    0.399193106           -0.65274418661
## 20   -0.088050584           -0.72161185658
## 21    1.236901457           -0.52337301375
## 23   -0.841231144           -0.28309629415
## 24    0.015358718           -0.81801895934
## 27    0.550701609           -0.94954139790
## 30   -0.318472410           -0.74122669111
## 31   -0.413660821           -0.40957399756
## 37    1.074108953           -0.52435750246
```

```
## 38    1.465167047          -0.69634499597
## 39    0.410503678          -0.61711394642
## 40   -0.089972913          -0.64590480190
## 42   -0.169903471          -0.63686946882
## 43    0.288874813          -0.55226903301
## 44    0.255428718          -0.71015503413
## 45   -0.277994519          -0.74946406137
## 49    0.242573156          -0.72915860217
## 51   -0.140579904          -0.59789949603
## 53    0.779680423          -0.54168818409
## 54   -0.377242002          -0.39428225445
## 58   -0.112865723          -0.64672906192
## 59    0.524226242          -0.76474896260
## 60    0.465695361          -0.48483164567
## 63    1.255488021           0.06018433318
## 65    0.456471468          -0.42084576002
## 66   -0.204671885          -0.00468784642
## 67    0.056088430           0.01579308529
## 69   -0.263919110          -0.66354074687
## 80   -0.019391589          -0.13105137545
## 86    3.357531021          -0.16626548531
## 87    0.114554626          -0.11967983016
## 89    0.705707309          -0.07087501328
## 90    0.155012393           0.01323710288
## 101  -0.723682029          -0.04254312289
## 104  -0.512096261          -0.03694917575
## 106  -0.088391307          -0.50883670313
## 113  -0.458654456           0.01464574219
## 116  -0.747382694          10.09149146622
## 131  -1.338728207          -0.00008746816
## 132   0.085762320          -0.22810537170
## 155  -1.069189306           0.02292983163
## 163  -2.723911899           1.21769979381
## 164  -0.969145258           1.40200610264
## 168  -3.534951250          -0.37858447288
## 171  -0.451673722          -0.13814019446
## 174  -2.874015168          -0.09604238195
## 175  -1.065059464          -0.12690411193
## 181   2.970448530           0.17610929429
## 210  -1.696745691           0.04053635671
## 211   0.730786641          -0.48383277244
## 214   0.830660958          -0.71302237181
## 216   1.826103274          -0.74121332643
## 220  -0.859508005          -0.60923713510
## 221  -0.296126487          -0.69687664444
## 223  -0.114527106           0.11340638216
## 224  -1.051870012          -0.70098593795
## 225  -1.280265405          -0.61122692980
## 227   1.822398461          -0.52730573930
## 228   0.067647089          -0.56552983008
## 229   0.446939658          -0.58946612825
## 232   0.031087764          -0.55885007741
## 235   1.331770854          -0.33710289298
## 236  -0.348942414          -0.97440018313
## 241  -0.090314096          -0.84551381584
## 242   0.663012603          -0.33078473739
```

```
## 245  -0.242157275          -0.49791787569
## 247   0.118890142          -0.45606770614
## 248   0.271429755          -0.64027394003
## 254   0.131697540          -0.37060037666
## 256   0.434253434          -0.02060387631
## 260  -0.050690082          -0.69877459919
## 262  -0.898966905          -0.64847955566
## 263   0.825130078          -0.67005817653
## 270   0.536697409          -0.49980969466
## 272   0.211864095          -0.48666498675
## 273   0.003345751          -0.64051243405
## 275   0.422033726          -0.64471186115
## 280   1.584488054          -0.42056525620
## 281   0.100879964          -0.60997735309
## 282  -0.176806918          -0.53625725089
## 288   0.193828026          -0.44860331931
## 291   0.615451956          -0.68492864766
## 292  -0.146013506          -0.51169301160
## 294   1.433122376          -0.17812872851
## 297   1.048705449          -0.81755144902
## 298  -0.072388529          -0.60888453624
## 300  -0.523646337          -0.53296316245
## 302  -0.466415561          -0.53342242214
## 303   0.042227234          -0.72745040516
## 306  -0.024056947          -0.79891298598
## 308   0.585348337          -0.75930885543
## 309   0.057238447          -0.68402624719
```

```
predict <- predict(svm1, test_set1)
xtab <- table(test_set1$class, predict)
xtab
```

```
##                     predict
##                      Hernia Normal Spondylolisthesis
##   Hernia                  7      4                 1
##   Normal                  3     17                 0
##   Spondylolisthesis       0      3                27
```

```
(7 + 17 + 27)/nrow(test_set1)
```

```
## [1] 0.8225806
```

Support vector machines are common approaches to classification tasks. The final accuracy was approximately 0.82, using the `e1071` package.

4. Introduction - Part 2

4.1 Overview/Executive Summary

Part 2 of the project entailed using similar methods to classify patients into Normal (100 patients) or Abnormal (210 patients) classes. Here, the Spondylolisthesis and Disk Hernia classes were merged into one.

4.2 Loading the Data

Note: this process could take a couple of minutes

```
### Download file part 2
library(readr)
filename <- "column_2C_weka.csv"
dat <- read_csv(filename)
```

```
## Parsed with column specification:
## cols(
##   pelvic_incidence = col_double(),
##   `pelvic_tilt numeric` = col_double(),
##   lumbar_lordosis_angle = col_double(),
##   sacral_slope = col_double(),
##   pelvic_radius = col_double(),
##   degree_spondylolisthesis = col_double(),
##   class = col_character()
## )
```

4.3 Tidy data and Summary

```
### Summarising `dat` confirms that the dataset is comprised of
### 310 observations of 7 variables:
table(dat$class)
```

```
##
## Abnormal   Normal
##      210      100
```

```
summary(dat)
```

```
##  pelvic_incidence pelvic_tilt numeric lumbar_lordosis_angle
##  Min.   : 26.15   Min.   :-6.555      Min.   : 14.00
##  1st Qu.: 46.43   1st Qu.:10.667      1st Qu.: 37.00
##  Median : 58.69   Median :16.358      Median : 49.56
##  Mean   : 60.50   Mean   :17.543      Mean   : 51.93
##  3rd Qu.: 72.88   3rd Qu.:22.120      3rd Qu.: 63.00
##  Max.   :129.83   Max.   :49.432      Max.   :125.74
##   sacral_slope     pelvic_radius    degree_spondylolisthesis
##  Min.   : 13.37   Min.   : 70.08   Min.   :-11.058
##  1st Qu.: 33.35   1st Qu.:110.71   1st Qu.:  1.604
##  Median : 42.40   Median :118.27   Median : 11.768
##  Mean   : 42.95   Mean   :117.92   Mean   : 26.297
##  3rd Qu.: 52.70   3rd Qu.:125.47   3rd Qu.: 41.287
##  Max.   :121.43   Max.   :163.07   Max.   :418.543
##     class
##  Length:310
##  Class :character
##  Mode  :character
##
##
##
```

```r
### We use the `head` function on the dataset as before:
head(dat)
```

```
## # A tibble: 6 x 7
##   pelvic_incidence `pelvic_tilt nu~ lumbar_lordosis~ sacral_slope
##              <dbl>            <dbl>            <dbl>        <dbl>
## 1             63.0            22.6              39.6         40.5
## 2             39.1            10.1              25.0         29.0
## 3             68.8            22.2              50.1         46.6
## 4             69.3            24.7              44.3         44.6
## 5             49.7             9.65             28.3         40.1
## 6             40.3            13.9              25.1         26.3
## # ... with 3 more variables: pelvic_radius <dbl>,
## #   degree_spondylolisthesis <dbl>, class <chr>
```

```r
### We check the data is in tidy format:
dat %>% as_tibble
```

```
## # A tibble: 310 x 7
##    pelvic_incidence `pelvic_tilt nu~ lumbar_lordosis~ sacral_slope
##               <dbl>            <dbl>            <dbl>        <dbl>
##  1             63.0            22.6              39.6         40.5
##  2             39.1            10.1              25.0         29.0
##  3             68.8            22.2              50.1         46.6
##  4             69.3            24.7              44.3         44.6
##  5             49.7             9.65             28.3         40.1
##  6             40.3            13.9              25.1         26.3
##  7             53.4            15.9              37.2         37.6
##  8             45.4            10.8              29.0         34.6
##  9             43.8            13.5              42.7         30.3
## 10             36.7             5.01             41.9         31.7
## # ... with 300 more rows, and 3 more variables: pelvic_radius <dbl>,
## #   degree_spondylolisthesis <dbl>, class <chr>
```

```r
### We ensure there is no missing data:
any(is.na(dat))
```

```
## [1] FALSE
```

```r
sum(is.na(dat))
```

```
## [1] 0
```

5. Methods and Analysis

5.1 Data Cleaning and Exploration

Transform dataset into dataframe for classification purposes

```r
class(dat)
```

```
## [1] "spec_tbl_df" "tbl_df"      "tbl"         "data.frame"
```

```
dat$class <- as.factor(dat$class)
dat <- as.data.frame(dat)
class(dat)
```

```
## [1] "data.frame"
```

Change column name for simplicity

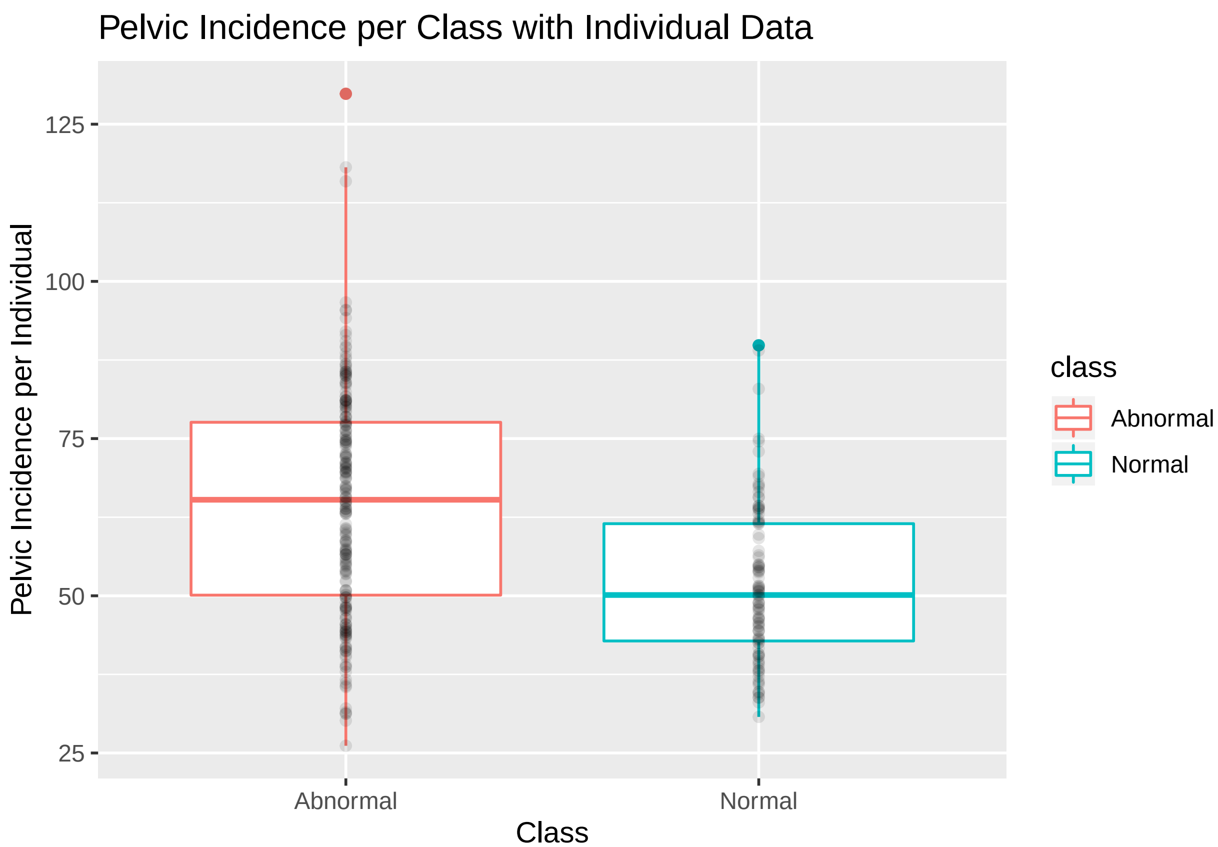
```
colnames(dat)[colnames(dat) == "pelvic_tilt numeric"] <- "pelvic_tilt_numeric"
```

For presentation and simplicity purposes, one of the column labels was fixed.

5.2 Data Visualisation

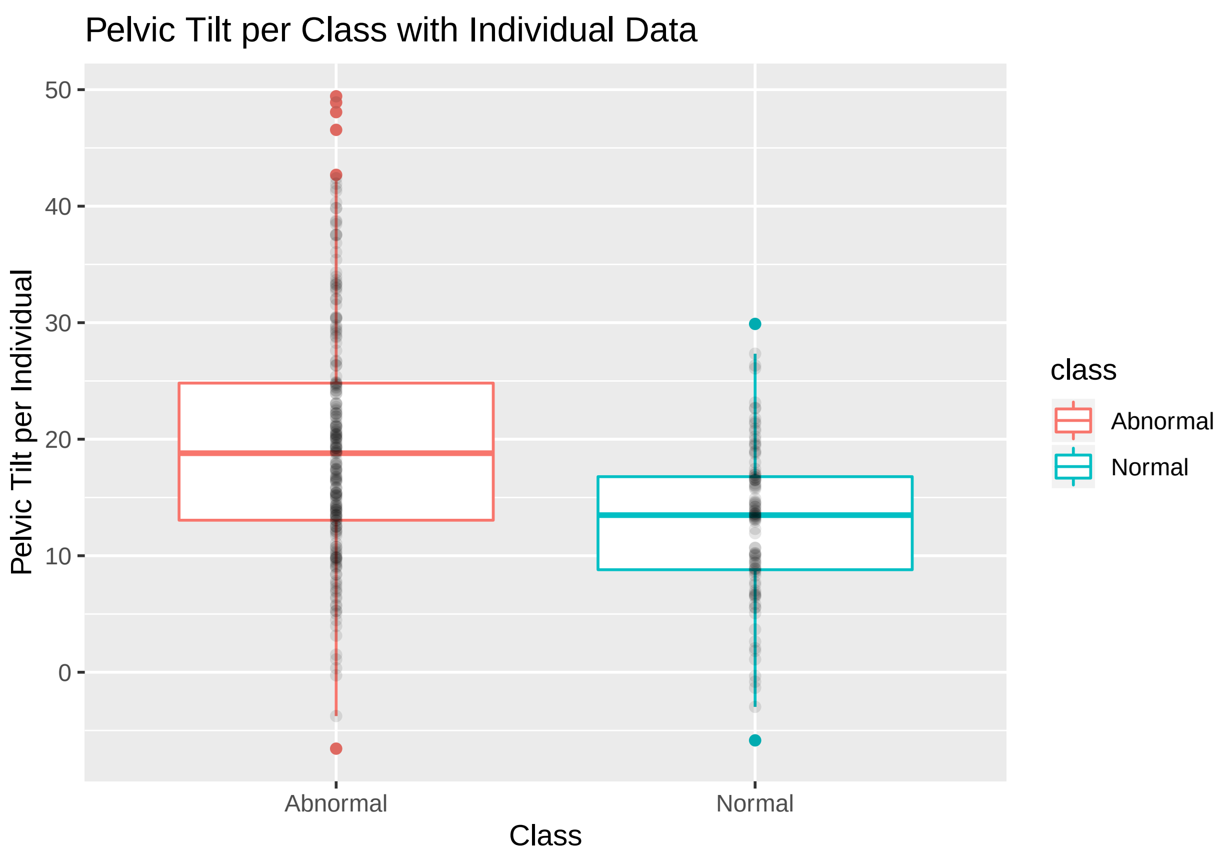
Pelvic incidence

```
options(scipen = 999)
dat %>% group_by(class) %>% ggplot(aes(class, pelvic_incidence)) +
    geom_boxplot(aes(class, pelvic_incidence, col = class)) +
    ggtitle("Pelvic Incidence per Class with Individual Data") +
    xlab("Class") + ylab("Pelvic Incidence per Individual") +
    geom_point(alpha = 0.1)
```



Pelvic tilt

```
options(scipen = 999)
dat %>% group_by(class) %>% ggplot(aes(class, pelvic_tilt_numeric)) +
    geom_boxplot(aes(class, pelvic_tilt_numeric, col = class)) +
    ggtitle("Pelvic Tilt per Class with Individual Data") + xlab("Class") +
    ylab("Pelvic Tilt per Individual") + geom_point(alpha = 0.1)
```

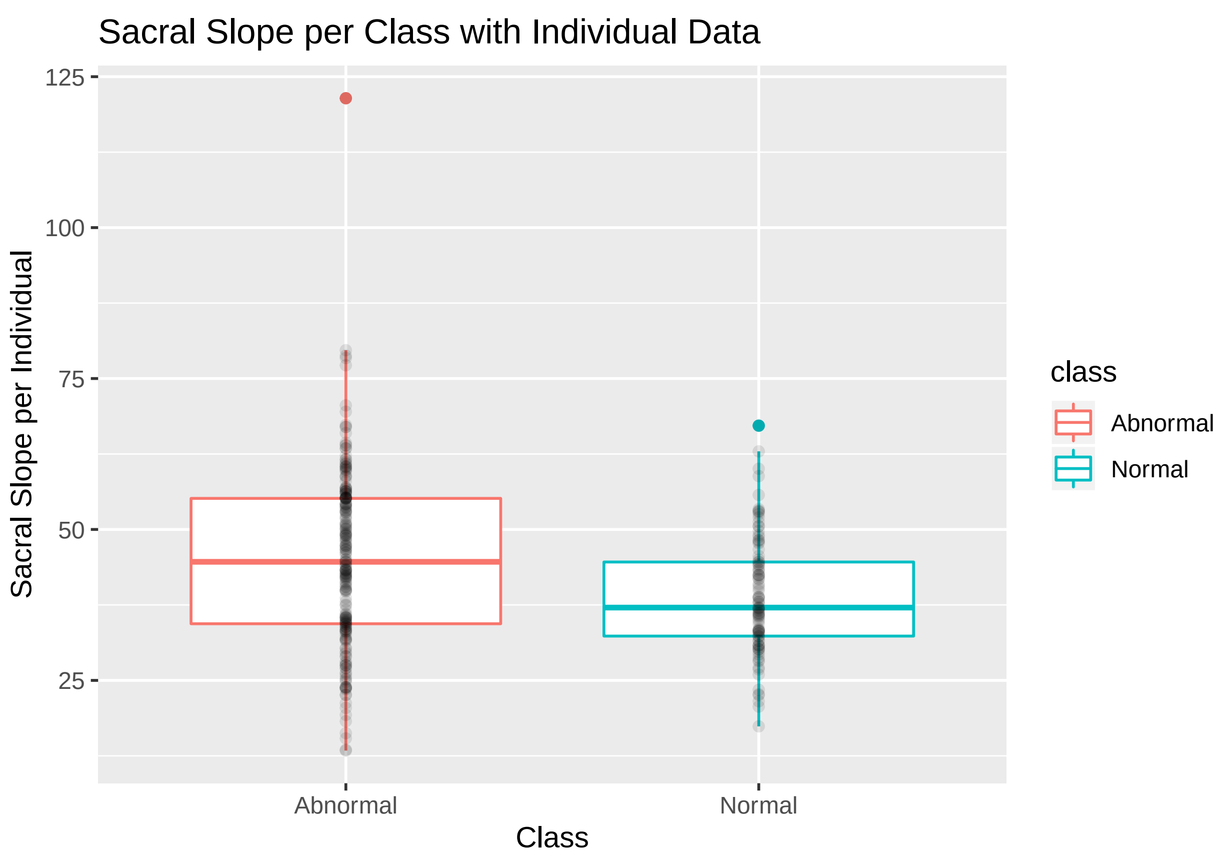


Lumbar lordosis angle

```
options(scipen = 999)
dat %>% group_by(class) %>% ggplot(aes(class, lumbar_lordosis_angle)) +
    geom_boxplot(aes(class, lumbar_lordosis_angle, col = class)) +
    ggtitle("Lumbar Lordosis Angle per Class with Individual Data") +
    xlab("Class") + ylab("Lumbar Lordosis Angle per Individual") +
    geom_point(alpha = 0.1)
```

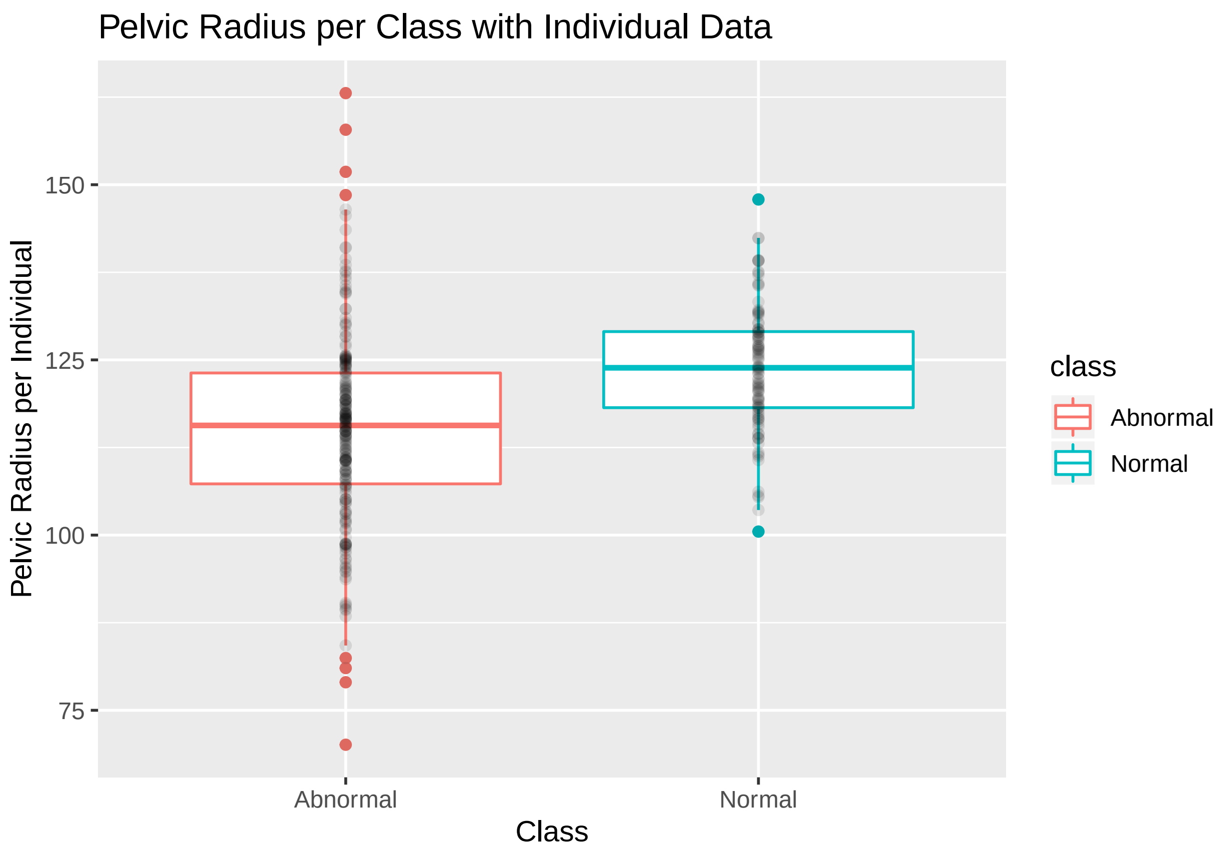
Sacral slope

```
options(scipen = 999)
dat %>% group_by(class) %>% ggplot(aes(class, sacral_slope)) +
    geom_boxplot(aes(class, sacral_slope, col = class)) + ggtitle("Sacral Slope per Class with Individual D
    xlab("Class") + ylab("Sacral Slope per Individual") + geom_point(alpha = 0.1)
```



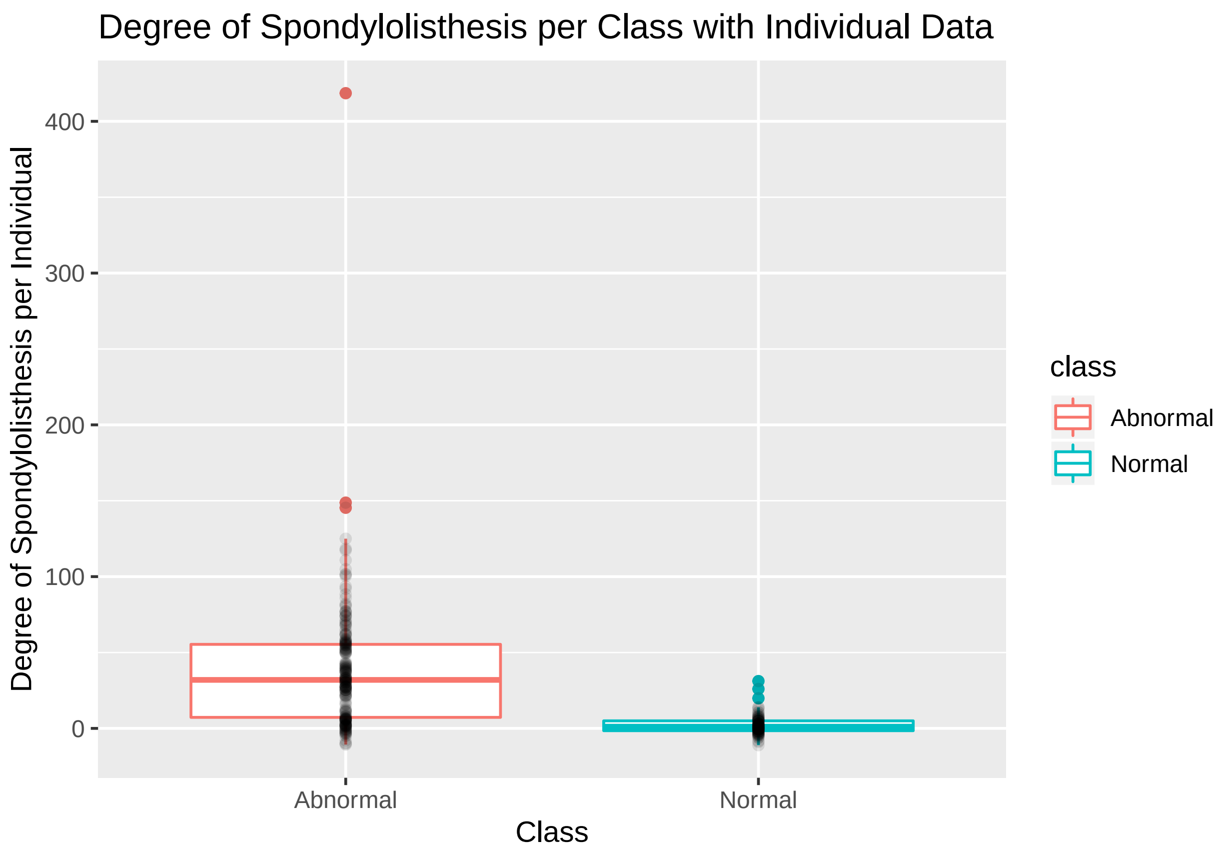
Pelvic radius

```
options(scipen = 999)
dat %>% group_by(class) %>% ggplot(aes(class, pelvic_radius)) +
    geom_boxplot(aes(class, pelvic_radius, col = class)) + ggtitle("Pelvic Radius per Class with Individual
    xlab("Class") + ylab("Pelvic Radius per Individual") + geom_point(alpha = 0.1)
```



Degree of spondylolisthesis

```
options(scipen = 999)
dat %>% group_by(class) %>% ggplot(aes(class, degree_spondylolisthesis)) +
    geom_boxplot(aes(class, degree_spondylolisthesis, col = class)) +
    ggtitle("Degree of Spondylolisthesis per Class with Individual Data") +
    xlab("Class") + ylab("Degree of Spondylolisthesis per Individual") +
    geom_point(alpha = 0.1)
```



We can see that higher rates of all the variables are observed in Abnormal patients, with the exception of pelvic radius, which is smaller in this class.

6. Results and Discussion

6.1 Partitioning the data into train and test sets

```
set.seed(1, sample.kind = "Rounding")
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding'
## sampler used
```

```
test_index <- createDataPartition(y = dat$class, times = 1, p = 0.2,
    list = FALSE)
train_set <- dat[-test_index, ]  ### Create train set
test_set <- dat[test_index, ]  ### Create test set
```

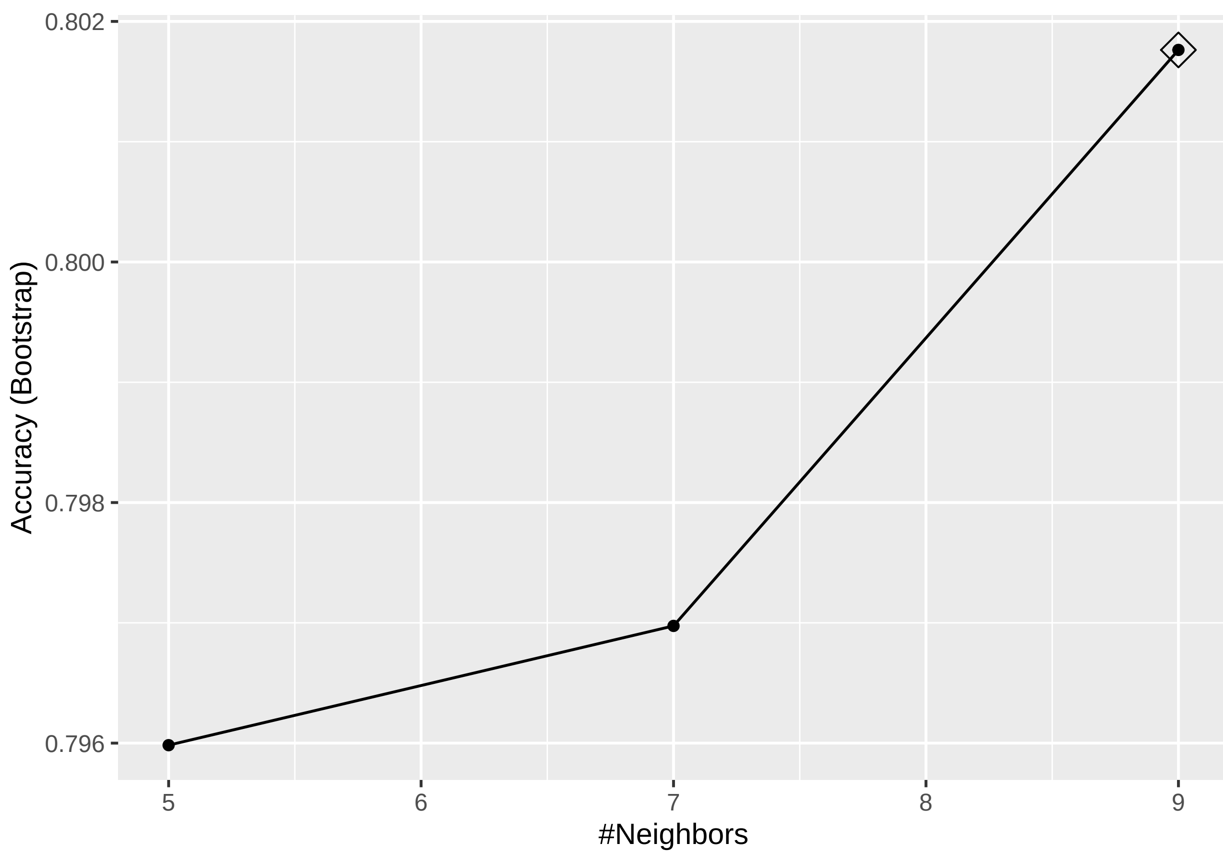
6.2 k-nearest neighbors - train and test sets

```
fit_knn2 <- train(class ~ ., method = "knn", data = train_set)
fit_knn2
```

```
## k-Nearest Neighbors
```

```
##
## 248 samples
##   6 predictor
##   2 classes: 'Abnormal', 'Normal'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 248, 248, 248, 248, 248, 248, ...
## Resampling results across tuning parameters:
##
##   k  Accuracy   Kappa
##   5  0.7959830  0.5305478
##   7  0.7969745  0.5336731
##   9  0.8017630  0.5417197
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 9.
```

```
ggplot(fit_knn2, highlight = TRUE)
```




```
fit_knn2$bestTune
```

```
##   k
## 3 9
```

```
fit_knn2$finalModel
```

```
## 9-nearest neighbor model
```

```
## Training set outcome distribution:
##
## Abnormal   Normal
##      168       80
```

```
y_hat_knn1 <- predict(fit_knn2, test_set, type = "raw")
confusionMatrix(y_hat_knn1, test_set$class)$overall[["Accuracy"]]
```

```
## [1] 0.9193548
```

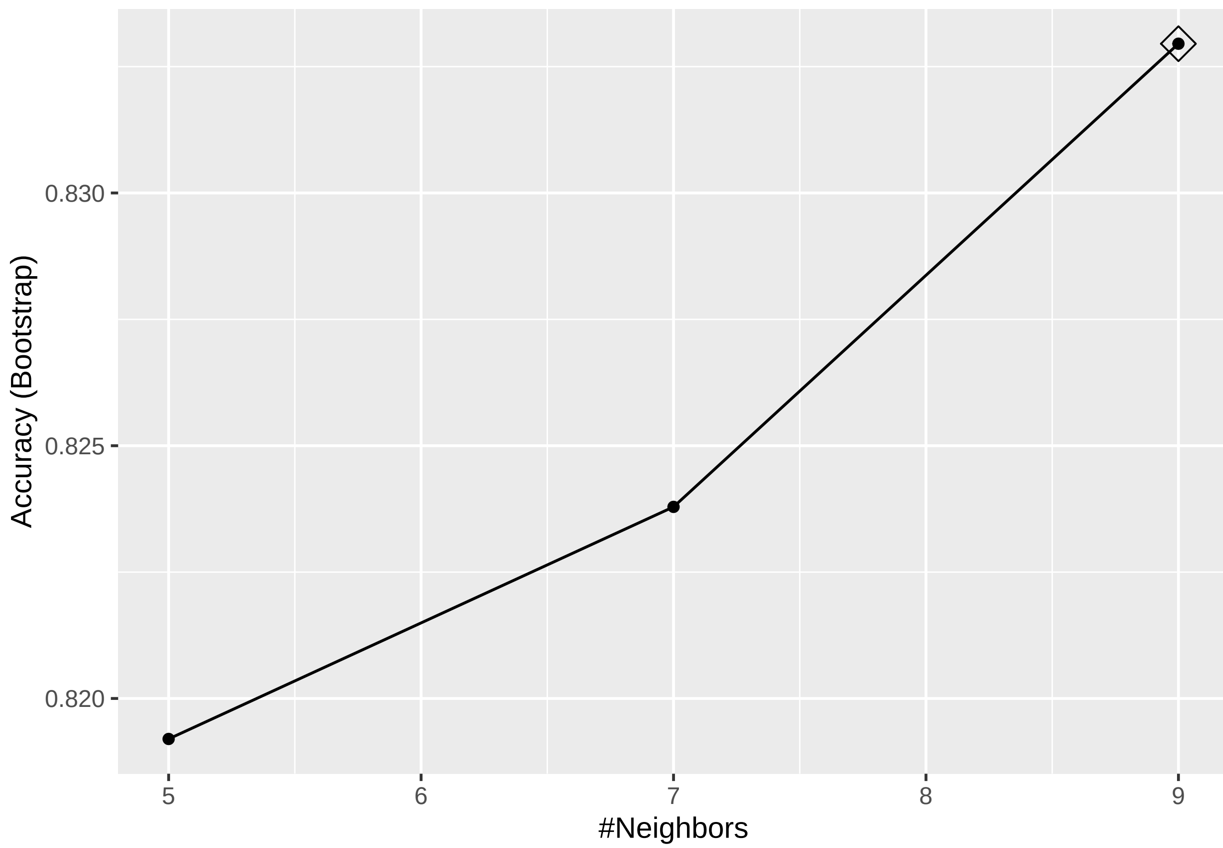
The k-nearest neighbours approach once again provides very good accuracy (~0.92).

6.3 k-nearest neighbors - whole dataset

```
fit_knn3 <- train(class ~ ., method = "knn", data = dat)
fit_knn3
```

```
## k-Nearest Neighbors
##
## 310 samples
##   6 predictor
##   2 classes: 'Abnormal', 'Normal'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 310, 310, 310, 310, 310, 310, ...
## Resampling results across tuning parameters:
##
##   k  Accuracy   Kappa
##   5  0.8191992  0.5919564
##   7  0.8237902  0.6027542
##   9  0.8329526  0.6217068
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 9.
```

```
ggplot(fit_knn3, highlight = TRUE)
```




```
fit_knn3$bestTune
```

```
##   k
## 3 9
```

```
fit_knn3$finalModel
```

```
## 9-nearest neighbor model
## Training set outcome distribution:
##
## Abnormal   Normal
##      210      100
```

This successfully sorts the data into 210 Abnormal and 100 Normal patients.

6.4 rpart - whole dataset

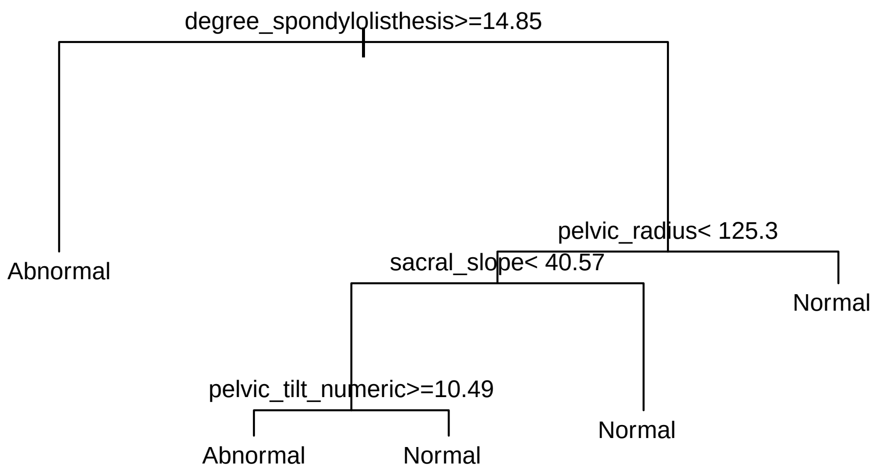
```
install.packages("rpart")
```

```
##
## The downloaded binary packages are in
##  /var/folders/4w/60c04z2j2fn103mghfmd8ntr0000gn/T//RtmpUploQ4/downloaded_packages
```

```r
library(rpart)
fit_4 <- rpart(class ~ ., data = dat)
fit_4
```

```
## n= 310
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 310 100 Abnormal (0.67741935 0.32258065)
##    2) degree_spondylolisthesis>=14.85401 149   3 Abnormal (0.97986577 0.02013423) *
##    3) degree_spondylolisthesis< 14.85401 161  64 Normal (0.39751553 0.60248447)
##      6) pelvic_radius< 125.3019 111  53 Abnormal (0.52252252 0.47747748)
##       12) sacral_slope< 40.566 75  25 Abnormal (0.66666667 0.33333333)
##         24) pelvic_tilt_numeric>=10.48576 59  15 Abnormal (0.74576271 0.25423729) *
##         25) pelvic_tilt_numeric< 10.48576 16   6 Normal (0.37500000 0.62500000) *
##       13) sacral_slope>=40.566 36   8 Normal (0.22222222 0.77777778) *
##      7) pelvic_radius>=125.3019 50   6 Normal (0.12000000 0.88000000) *
```

```r
plot(fit_4, margin = 0.1)
text(fit_4, cex = 0.75)
```



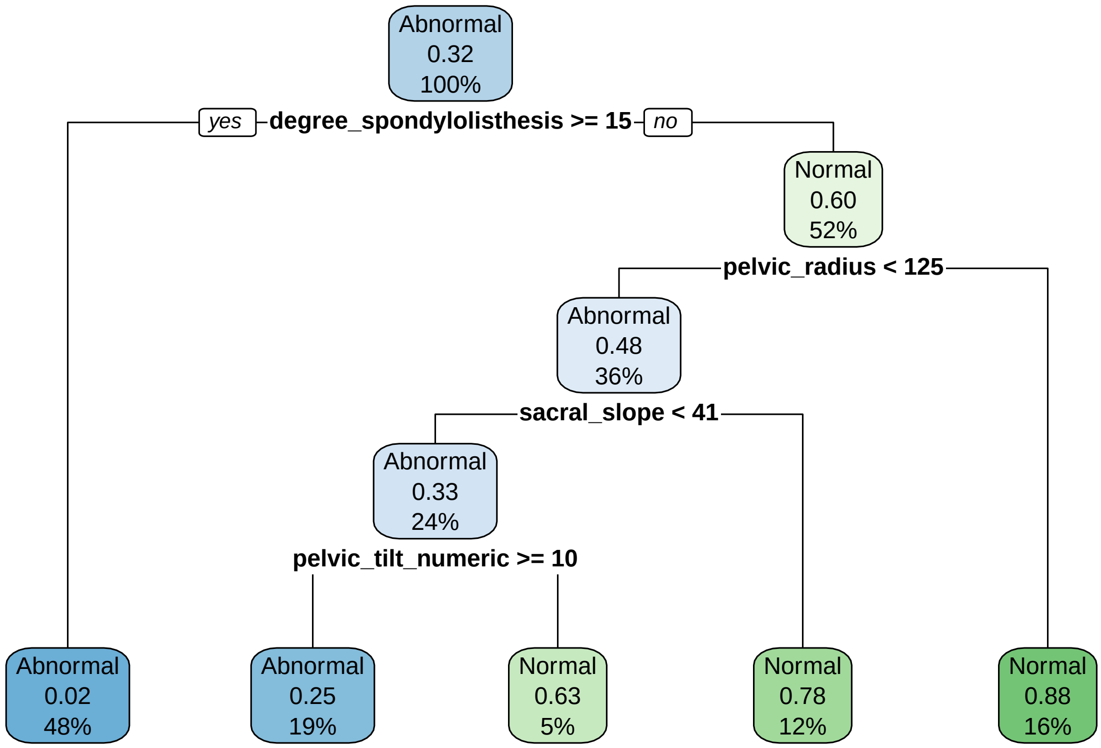

```r
install.packages("rpart.plot")
```

```
##
## The downloaded binary packages are in
##  /var/folders/4w/60c04z2j2fn103mghfmd8ntr0000gn/T//RtmpUploQ4/downloaded_packages
```

```r
library(rpart.plot)
install.packages("RColorBrewer")
```

```
##
## The downloaded binary packages are in
##  /var/folders/4w/60c04z2j2fn103mghfmd8ntr0000gn/T//RtmpUploQ4/downloaded_packages
```

```r
library(RColorBrewer)
rpart.plot(fit_4)
```




```r
printcp(fit_4)
```

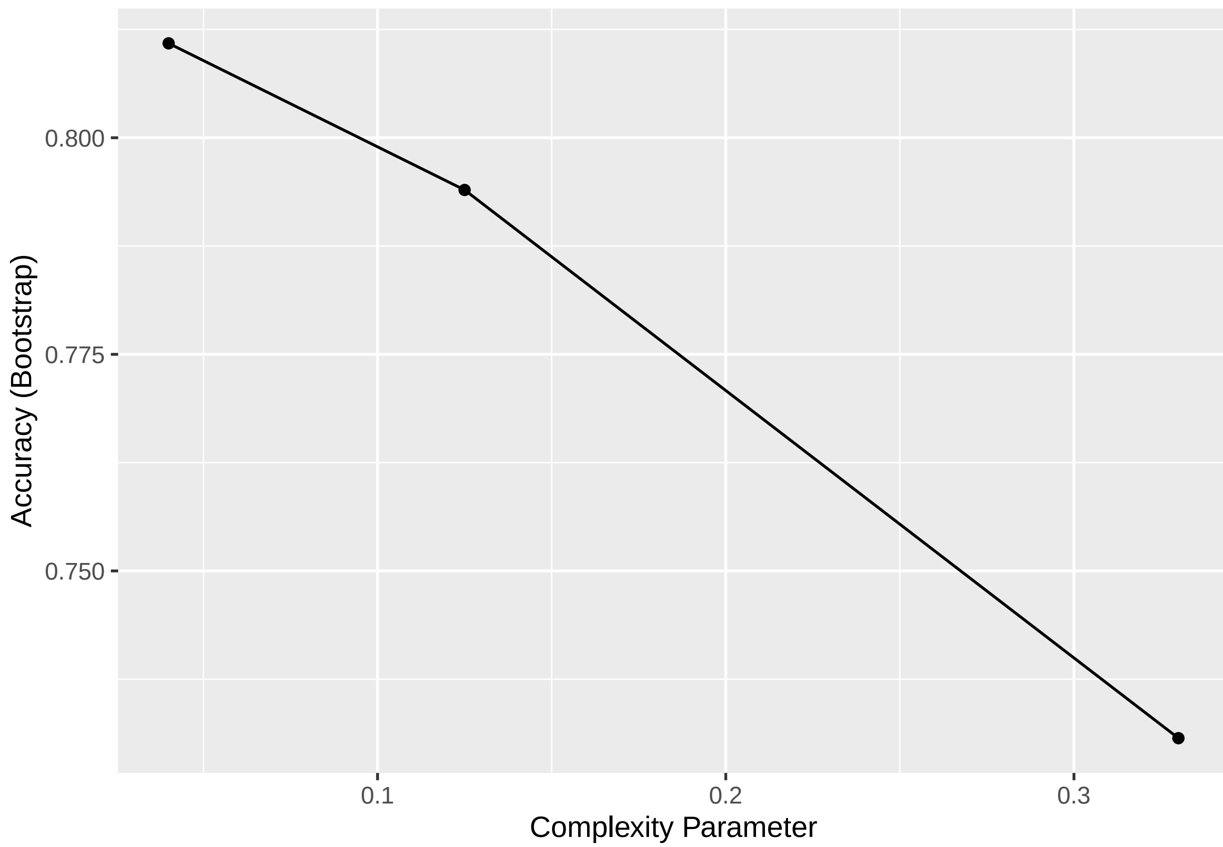
```
##
## Classification tree:
## rpart(formula = class ~ ., data = dat)
##
## Variables actually used in tree construction:
## [1] degree_spondylolisthesis pelvic_radius
## [3] pelvic_tilt_numeric      sacral_slope
##
## Root node error: 100/310 = 0.32258
##
## n= 310
##
##      CP nsplit rel error xerror     xstd
## 1 0.330      0      1.00   1.00 0.082305
## 2 0.125      1      0.67   0.89 0.079654
## 3 0.040      3      0.42   0.63 0.070850
## 4 0.010      4      0.38   0.54 0.066778
```

```r
train_rpart2 <- train(class ~ ., method = "rpart", data = dat)
train_rpart2
```

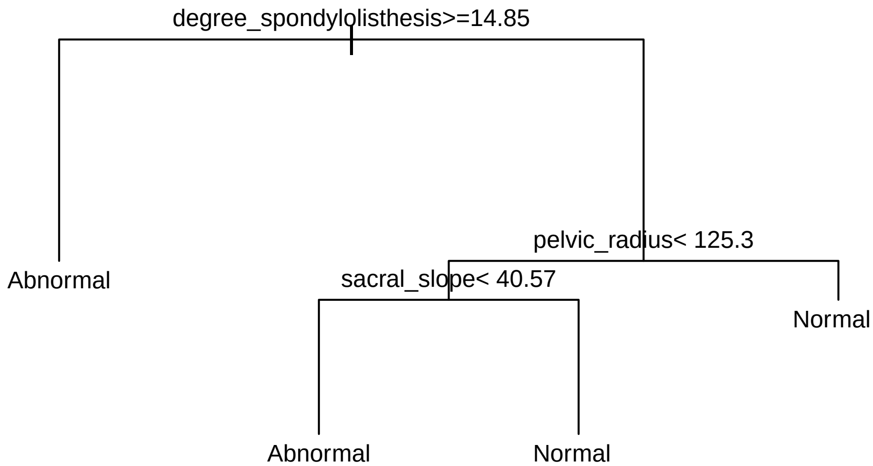
```
## CART
##
## 310 samples
##   6 predictor
##   2 classes: 'Abnormal', 'Normal'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
```

```
## Summary of sample sizes: 310, 310, 310, 310, 310, 310, ...
## Resampling results across tuning parameters:
##
##   cp     Accuracy   Kappa
##   0.040  0.8108970  0.5557917
##   0.125  0.7939720  0.5261025
##   0.330  0.7306978  0.3046176
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.04.
```

```r
ggplot(train_rpart2)
```




```r
plot(train_rpart2$finalModel, margin = 0.1)
text(train_rpart2$finalModel, cex = 0.75)
```




```r
pruned_fit <- prune(fit_4, cp = 0.01)
pruned_fit
```

```
## n= 310
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 310 100 Abnormal (0.67741935 0.32258065)
##    2) degree_spondylolisthesis>=14.85401 149   3 Abnormal (0.97986577 0.02013423) *
##    3) degree_spondylolisthesis< 14.85401 161  64 Normal (0.39751553 0.60248447)
##      6) pelvic_radius< 125.3019 111  53 Abnormal (0.52252252 0.47747748)
##       12) sacral_slope< 40.566 75  25 Abnormal (0.66666667 0.33333333)
##         24) pelvic_tilt_numeric>=10.48576 59  15 Abnormal (0.74576271 0.25423729) *
##         25) pelvic_tilt_numeric< 10.48576 16   6 Normal (0.37500000 0.62500000) *
##       13) sacral_slope>=40.566 36   8 Normal (0.22222222 0.77777778) *
##      7) pelvic_radius>=125.3019 50   6 Normal (0.12000000 0.88000000) *
```

```r
ind <- !(train_rpart2$finalModel$frame$var == "<leaf>")
tree_terms <- train_rpart2$finalModel$frame$var[ind] %>% unique() %>%
    as.character()
tree_terms
```

```
## [1] "degree_spondylolisthesis" "pelvic_radius"
## [3] "sacral_slope"
```

Using the same methods as before, the partitioning function revealed degree of spondylolisthesis, pelvic radius, and sacral slope were important predictors.

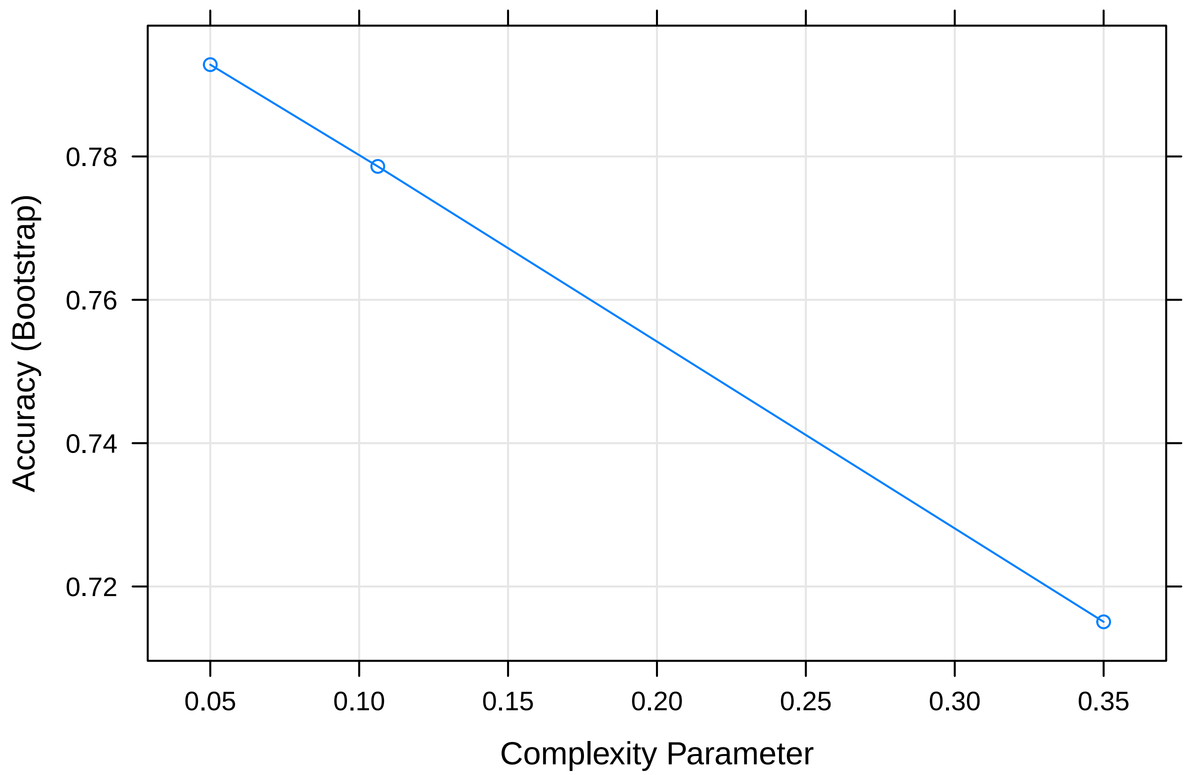
6.5 rpart - train and test sets

```r
train_rpart3 <- train(class ~ ., method = "rpart", data = train_set)
train_rpart3
```

```
## CART
##
## 248 samples
```

```
##   6 predictor
##   2 classes: 'Abnormal', 'Normal'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 248, 248, 248, 248, 248, 248, ...
## Resampling results across tuning parameters:
##
##   cp       Accuracy   Kappa
##   0.05000  0.7928099  0.5258113
##   0.10625  0.7786115  0.4997245
##   0.35000  0.7150704  0.2673376
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.05.
```

```
plot(train_rpart3)
```




```
confusionMatrix(predict(train_rpart3, test_set), test_set$class)$overall["Accuracy"]
```

```
##  Accuracy
## 0.8870968
```

```
ind <- !(train_rpart3$finalModel$frame$var == "<leaf>")
tree_terms <- train_rpart3$finalModel$frame$var[ind] %>% unique() %>%
    as.character()
tree_terms
```

```
## [1] "degree_spondylolisthesis" "pelvic_radius"
## [3] "sacral_slope"
```

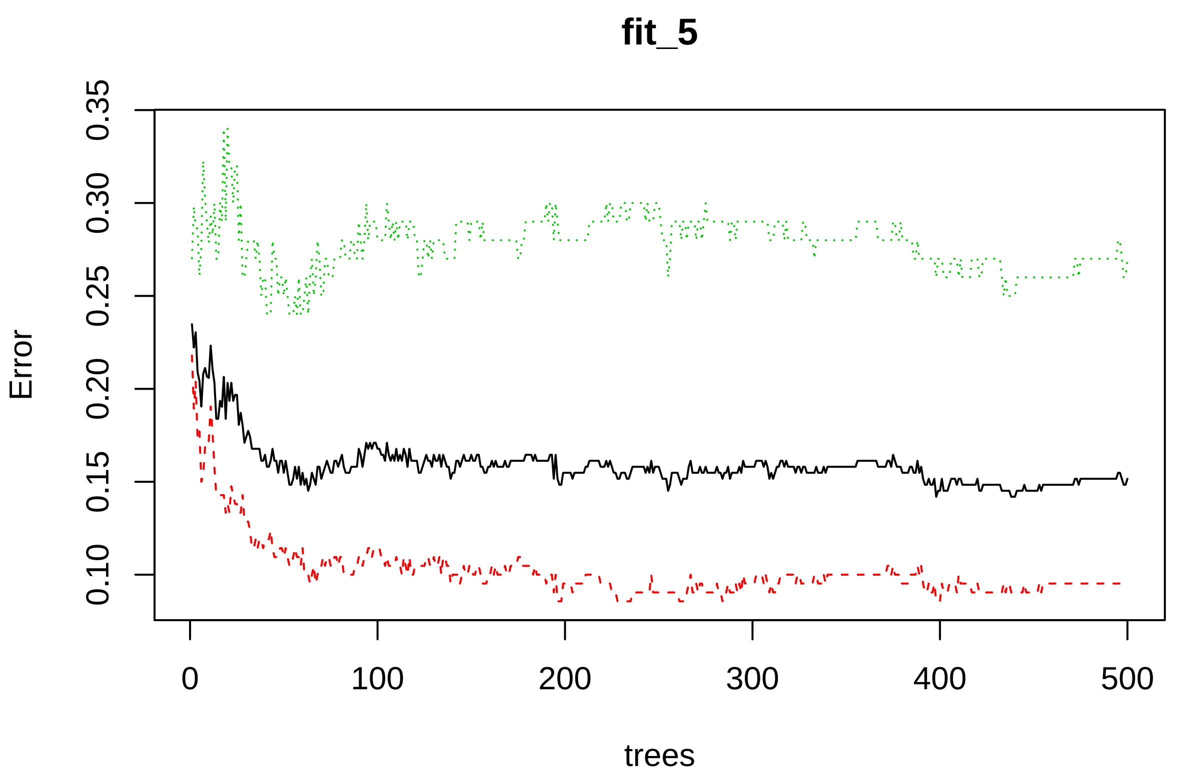
When applied to the train and test sets, this generated an accuracy of approximately 0.89!

6.6 randomForest - whole dataset

```
library(randomForest)
fit_5 <- randomForest(class ~ ., data = dat)
fit_5
```

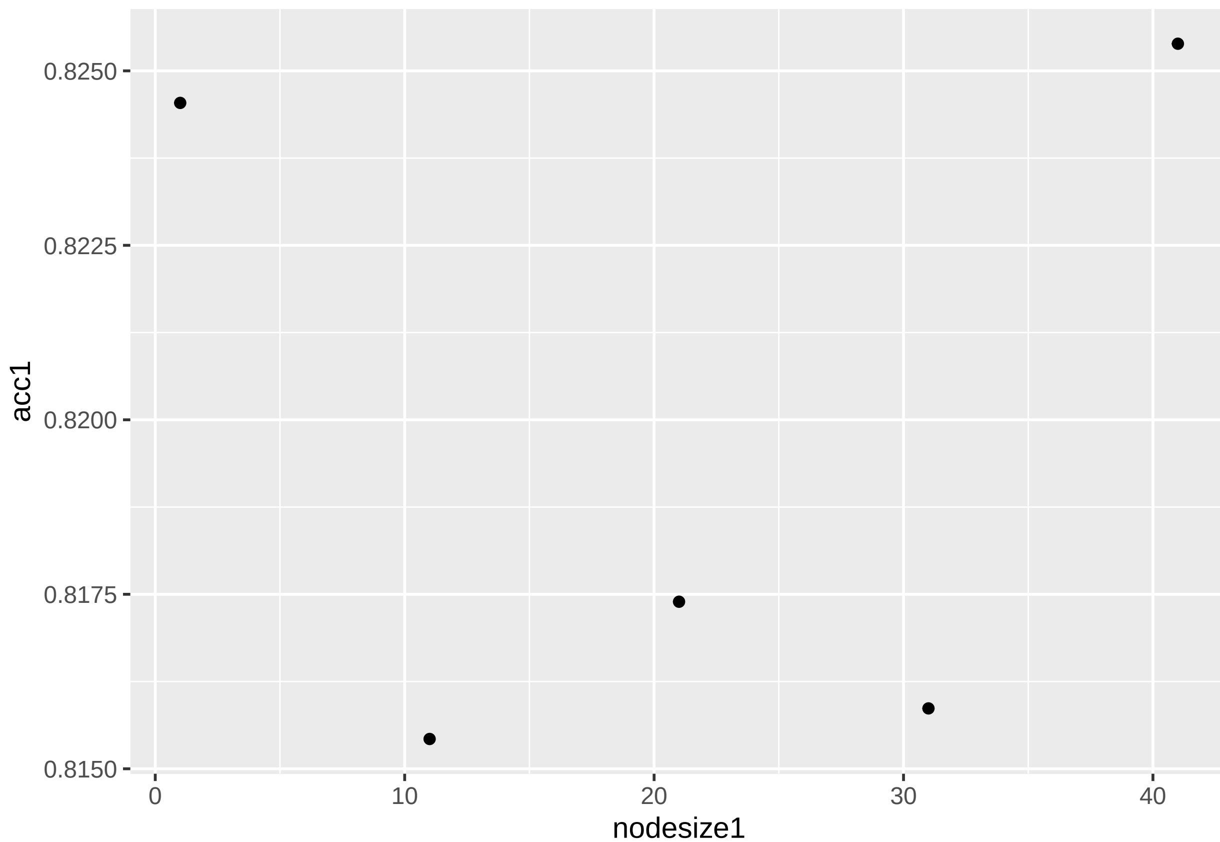
```
##
## Call:
##  randomForest(formula = class ~ ., data = dat)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 15.16%
## Confusion matrix:
##          Abnormal Normal class.error
## Abnormal      190     20   0.0952381
## Normal         27     73   0.2700000
```

```
plot(fit_5)
```



6.7 randomForest - train and test sets

```
nodesize1 <- seq(1, 50, 10)
acc1 <- sapply(nodesize, function(ns) {
    train(class ~ ., method = "rf", data = train_set, tuneGrid = data.frame(mtry = 2),
        nodesize1 = ns)$results$Accuracy
})
qplot(nodesize1, acc1)
```




```
train_rf1 <- randomForest(class ~ ., data = train_set, ns = ns[which.max(acc)])
train_rf1
```

```
##
## Call:
##  randomForest(formula = class ~ ., data = train_set, ns = ns[which.max(acc)])
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 17.34%
## Confusion matrix:
##          Abnormal Normal class.error
## Abnormal      148     20   0.1190476
## Normal         23     57   0.2875000
```

```
confusionMatrix(predict(train_rf1, test_set), test_set$class)$overall["Accuracy"]
```

```
##  Accuracy
## 0.8709677
```

The random forest performed similarly in part 2 as in part 1, generating an accuracy of about 0.87.

6.8 Multiclass support vector machine

```r
install.packages("e1071")
```

```
##
## The downloaded binary packages are in
##  /var/folders/4w/60c04z2j2fn103mghfmd8ntr0000gn/T//RtmpUploQ4/downloaded_packages
```

```r
library(e1071)
svm2 <- svm(class ~ ., data = train_set, method = "C-classification",
    kernal = "radial", gamma = 0.1, cost = 10)
summary(svm2)
```

```
##
## Call:
## svm(formula = class ~ ., data = train_set, method = "C-classification",
##     kernal = "radial", gamma = 0.1, cost = 10)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  10
##
## Number of Support Vectors:  97
##
##  ( 50 47 )
##
##
## Number of Classes:  2
##
## Levels:
##  Abnormal Normal
```

```r
svm2$SV
```

```
##     pelvic_incidence pelvic_tilt_numeric lumbar_lordosis_angle
## 2        -1.19983343         -0.75991508          -1.440358270
## 4         0.53091588          0.72844156          -0.399921998
## 5        -0.58995674         -0.80162413          -1.262312333
## 6        -1.13153943         -0.36610592          -1.434450162
## 7        -0.37704356         -0.16798020          -0.785198102
## 8        -0.83870030         -0.68906460          -1.223438957
## 9        -0.92893280         -0.40569715          -0.487295563
## 11       -0.59031444         -0.45596008          -1.099630071
## 15       -0.15570390          0.68111474          -0.254943470
## 16       -0.89867241         -0.50726374          -0.842740149
## 17        0.21830408          0.29082837           0.152285126
## 18       -1.64516509         -1.46537245          -1.033389353
## 20       -1.04684715         -0.53622303          -1.164976674
## 21       -0.92134077         -0.33998889          -0.749254217
## 22       -0.29196489          0.36220980          -0.513760342
## 27       -1.93866451         -0.68867270          -2.034309466
## 30        0.39253651          0.75283164          -0.132081120
## 31       -0.52663292         -0.21511631          -0.496023898
## 34       -1.22236597         -0.46152235          -0.632384742
```

```
## 37     -1.59852109    -1.07321633    -0.848162606
## 38     -1.39176613     0.19706546    -1.673044248
## 40     -0.43505235     0.15303328    -0.854931689
## 43     -0.35289838     0.17537949    -1.021739150
## 44      0.35855015     0.89928637    -0.227983379
## 45     -0.22839116    -0.12375564     0.568702111
## 46     -0.52130032     0.56139705    -0.254943470
## 49     -1.11395971     0.04758920    -0.955905832
## 51     -0.27099395     0.29874464    -0.955905832
## 55     -1.07879974    -0.01938105    -0.984515731
## 57     -0.95414904    -1.02445157    -1.275896078
## 58     -0.75347782    -0.22028751    -0.740225105
## 59     -0.96252926     0.21949427    -0.901985650
## 60     -0.68173796    -0.26320756    -0.871537852
## 63     -0.88664509    -0.82394564     0.014657439
## 65      0.92297816     0.45134187     1.684107279
## 69      0.68998370     0.14636414    -0.039262743
## 80     -0.70260292    -0.55302464    -0.686304924
## 86     -0.83429353    -0.77571672    -0.362783833
## 89     -0.19545012    -0.07252308    -0.524544378
## 101     1.40593843     1.31073369     0.741473754
## 106     0.28543818     1.02931080    -0.042092528
## 119     0.31565974     0.67791225    -0.320986530
## 132     0.55861686    -0.38083877     0.338178529
## 152    -0.68623724    -1.38120852     0.356755069
## 163     3.32664574     2.13568222    -0.047969798
## 165    -0.34792708    -0.83686276    -0.465205315
## 168     0.70528317    -0.11122255     0.438956017
## 169     2.02388971     0.74575622     2.341724838
## 171     0.27408171    -0.23838579     0.383471133
## 172     1.05198523    -0.35383003     1.507937110
## 211    -1.23140799    -0.05578539    -0.895902695
## 212    -0.29188391     0.14863397    -0.006832157
## 216    -1.67573209    -0.42393325    -0.853267365
## 220     0.24559800     0.89933984    -0.041475402
## 221    -0.88892163     0.43606724    -1.092099911
## 223    -0.22418129    -0.44929623     0.588207465
## 224     0.53674285     0.14149174     1.306928003
## 225     1.70636190     0.52305030     2.094006287
## 227     0.22543197    -0.14792632     0.614451323
## 228     0.08698814     0.22090302     0.062766565
## 229    -1.25766234    -0.93936403    -1.374497366
## 231     0.31999804     0.57391741     0.585250450
## 233    -0.96740751    -0.37693443    -0.613653529
## 235    -1.27566569    -0.82873306    -0.524544378
## 236     0.22371200     0.25090494    -0.622838193
## 239     0.51419819    -0.43037715     0.207160193
## 241    -1.05160024    -0.88490888    -0.846341379
## 242    -0.48599188    -0.40731952    -0.901985650
## 245     0.17202027     1.00214161    -0.006640135
## 247    -0.76595001    -0.16906333    -0.632384742
## 248    -0.58335910    -0.07902707    -1.279426922
## 253    -0.97884912    -2.38241043     0.338178529
## 263    -1.00178655    -0.31982241    -1.423740763
## 264    -1.50134655    -1.41126809    -1.414227376
## 269    -0.16453935    -0.10425583    -0.479135624
```

```
## 270      -1.30953899         -0.10505670          -1.495107649
## 272      -1.00188169         -0.09893917          -0.524544378
## 273      -1.18256292         -1.07098421          -0.794145286
## 280      -0.64209379          0.05166691           0.014657439
## 281      -0.56859179         -0.41627534          -0.931235164
## 282       0.24271571         -0.30735872          -0.421961036
## 283      -0.36270929         -0.41454366          -0.546959152
## 286       0.44550530         -0.09797574          -0.456777394
## 288      -1.54410900         -1.81924218          -1.760876202
## 291      -1.35061354         -0.37043904          -1.697709395
## 292      -0.51171983         -0.33672730          -0.850695213
## 294      -0.63631343         -1.21619733           0.203378075
## 297      -1.16549603         -0.13284124          -0.811677059
## 298      -0.82675399          0.12728620          -0.968084084
## 299       0.37124371          0.34541462          -1.078440988
## 300       1.30977119          1.26304247           0.351687811
## 302       1.65944169          0.87359658           0.932446816
## 303      -0.31022967          0.40572681          -1.206083818
## 306      -0.69350927         -0.39723786          -0.848065468
## 307      -0.34820812          0.32744488          -1.213615517
## 308       0.08160806          0.52873689          -0.299678493
## 309      -0.84522273         -0.89943268          -0.547022278
##     sacral_slope pelvic_radius degree_spondylolisthesis
## 2   -0.979368395  -0.261772653             -0.542312483
## 4    0.154703027  -1.186684348             -0.368879910
## 5   -0.177466241  -0.721885064             -0.454797568
## 6   -1.172702446   0.912905364             -0.603198196
## 7   -0.358083149   0.192835954             -0.505151539
## 8   -0.572418705  -0.050433759             -0.939939923
## 9   -0.888017673   0.520055212             -0.314676373
## 11  -0.423523209  -0.686506976             -0.865584091
## 15  -0.681113875  -0.084627035             -0.510933362
## 16  -0.777534033   0.454612613             -0.520094417
## 17   0.069789514  -0.416400120             -0.677639953
## 18  -1.042029879   0.815784080             -0.566870167
## 20  -0.944586417  -0.100922540             -0.693865202
## 21  -0.925091712   1.217827974             -0.493068961
## 22  -0.627065744   0.535534942             -0.597930207
## 27  -1.965545486   0.534839930             -0.924735169
## 30  -0.037852031  -0.330265861             -0.713733079
## 31  -0.514011685  -0.425008737             -0.377801871
## 34  -1.219917034   0.513505861             -0.590874033
## 37  -1.261604066   1.055797432             -0.494066151
## 38  -1.902368013   1.445025152             -0.668272384
## 40  -0.659626860  -0.102835871             -0.617181482
## 43  -0.571475620   0.274238632             -0.522337756
## 44  -0.184948039   0.240949084             -0.682260580
## 45  -0.201272319  -0.289977430             -0.722076717
## 46  -1.058993754  -0.039205828             -0.727321432
## 49  -1.444383841   0.228153693             -0.701509305
## 51  -0.555417151  -0.153205993             -0.568556879
## 55  -1.352277682  -0.116254174             -0.911067277
## 57  -0.480305754  -0.381969701             -0.511290010
## 58  -0.797583598  -0.125621530             -0.618016375
## 59  -1.374775842   0.508488484             -0.737558802
## 60  -0.676245879   0.450231560             -0.454030387
```

```
## 63  -0.537292834   1.236327542              0.098016619
## 65   0.848045654   0.441050840             -0.389219046
## 69   0.769707729  -0.275967902             -0.635044941
## 80  -0.496743316  -0.032584908             -0.095686143
## 86  -0.505269798   3.328531789             -0.131354536
## 89  -0.195962345   0.689120116             -0.034733470
## 101  0.848980249  -0.733578869             -0.006036082
## 106 -0.369913325  -0.101261668             -0.478345124
## 119 -0.081966203   1.363849738             -0.234080077
## 132  0.977954721   0.072076821             -0.193992193
## 152  0.112427937   0.545734052              0.251782926
## 163  2.694959313  -2.724446529              1.270464710
## 165  0.154045949   0.475416071             -0.006984532
## 168  0.972103524  -3.531689756             -0.346412563
## 169  2.032903902  -2.113367985              0.843020775
## 171  0.516440416  -0.462843715             -0.102866412
## 172  1.583500318  -0.975485848             -0.338058049
## 211 -1.519654158   0.714082063             -0.453018628
## 212 -0.475211565   0.582303103             -0.609173235
## 216 -1.820707683   1.804271992             -0.713719542
## 220 -0.328013609  -0.868769102             -0.580040780
## 221 -1.435450114  -0.308024529             -0.668810891
## 223  0.035364234  -0.127275137              0.151925264
## 224  0.579125881  -1.060230746             -0.672973197
## 225  1.789067752  -1.287557119             -0.582056245
## 227  0.390562859   1.800584520             -0.497052421
## 228 -0.046807595   0.054046380             -0.535769625
## 229 -0.925091712   0.431563644             -0.560014716
## 231 -0.002581505   0.455510115             -0.767934318
## 233 -0.957173655   0.779361626             -0.636065460
## 235 -1.026495179   1.312253329             -0.304395846
## 236  0.105004103  -0.360593248             -0.949914647
## 239  0.956907241   0.638739136             -0.378781698
## 241 -0.702854004  -0.103175458             -0.819365569
## 242 -0.325983492   0.646625245             -0.297996183
## 245 -0.494226199  -0.254307924             -0.467285437
## 247 -0.849772940   0.105049586             -0.424895376
## 248 -0.682537393   0.256875227             -0.611477978
## 253  0.453285287   0.269512888             -0.749116218
## 263 -1.041286387   0.807983918             -0.641646449
## 264 -0.898359875   0.765169823             -0.707737918
## 269 -0.134274043  -0.305982150             -0.530547428
## 270 -1.583580205   0.520901278             -0.469201662
## 272 -1.198350614   0.197588368             -0.455887379
## 273 -0.736475964  -0.009953992             -0.611719549
## 280 -0.849772940   1.563787668             -0.388934923
## 281 -0.424213664   0.087123706             -0.580790548
## 282  0.525729892  -0.189263444             -0.506119413
## 283 -0.164741860  -0.298049489             -0.589986235
## 286  0.633743150   0.127775413             -0.534399626
## 288 -0.662631492   0.179636718             -0.417334694
## 291 -1.447029947   0.599287209             -0.656708758
## 292 -0.408719696  -0.158614163             -0.481238280
## 294  0.058399723   1.413130468             -0.143370822
## 297 -1.381441678   1.030512830             -0.791042472
## 298 -1.137331680  -0.085333792             -0.579683633
```

```
## 299  0.224666847   0.807778507             -0.621812647
## 300  0.761092798  -0.534479458             -0.502782829
## 302  1.480581247  -0.477516554             -0.503248013
## 303 -0.681113875   0.028745504             -0.699779071
## 306 -0.595919209  -0.037228430             -0.772163561
## 307 -0.673583229  -0.264692696             -0.672382229
## 308 -0.272344843   0.569324492             -0.732048514
## 309 -0.431205412   0.043686456             -0.655794716
```

```
predict <- predict(svm2, test_set)
xtab <- table(test_set$class, predict)
xtab
```

```
##           predict
##            Abnormal Normal
##   Abnormal       39      3
##   Normal          3     17
```

```
(39 + 17)/nrow(test_set)
```

```
## [1] 0.9032258
```

The multiclass support vector machine performed much better in part 2, generating an accuracy of 0.90.

7. Conclusion

7.1 Summary

A variety of models were tested on the two datasets to classify patients based on six biomechanical features. This work may have important implications for medical practitioners, improving diagnostic criteria for each of the conditions (i.e. Disk Hernia and Spondylolisthesis), as well as helping to detect abnormalities to begin with (part 2), even if these are not distinguished into more specific conditions.

7.2 Limitations and future work

A broader set of predictors may be needed to improve the accuracy of the diagnosis. Future research could expand on the used methodologies, exploring logistic regression and neural network analyses.