Orthopedic Condition Prediction

Steven Pramono

12/19/2019

1. Project overview

This project aims to analyze biomechanical features of orthopedic patients and use them to predict patients' conditions. Two datasets were analyzed in this project: "column_2C_weka.csv" and "column_3C_weka.csv". The first classifies diagnostic outcomes into two categories: "Abnormal" and "Normal" while the second further divides abnormality into "Disk Hernia" and "Spondylolisthesis". The datasets are otherwise identical in structure. Both data sets will be referred to in this report as **Data 1** and **Data 2** respectively.

Note: bin_class column is added where "Abnormal" is represented by 0 and "Normal" by 1.

```
if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
if(!require(readr)) install.packages("readr", repos = "http://cran.us.r-project.org")
if(!require(gridExtra)) install.packages("gridExtra", repos = "http://cran.us.r-project.org")
if(!require(corrplot)) install.packages("corrplot", repos = "http://cran.us.r-project.org")
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
if(!require(Matrix)) install.packages("Matrix", repos = "http://cran.us.r-project.org")

col_names=c("pelvic_incidence","pelvic_tilt","lumbar_lordosis_angle","sacral_slope","pelvic_radius","degree_spondylolisthesi
s","class")
dat1<-read_csv("../data/column_2C_weka.csv")%>%setNames(col_names)%>%na.omit%>% mutate(class=factor(class,levels=unique(clas
s)))
dat2<-read_csv("../data/column_3C_weka.csv")%>%setNames(col_names)%>%na.omit%>%mutate(class=factor(class,levels=unique(clas
s)))
```

We start with **Data 1**, here is the first few lines of the data:

```
head(dat1)
```

```
## # 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 <fct>
```

Structure of data analysis is as follows:

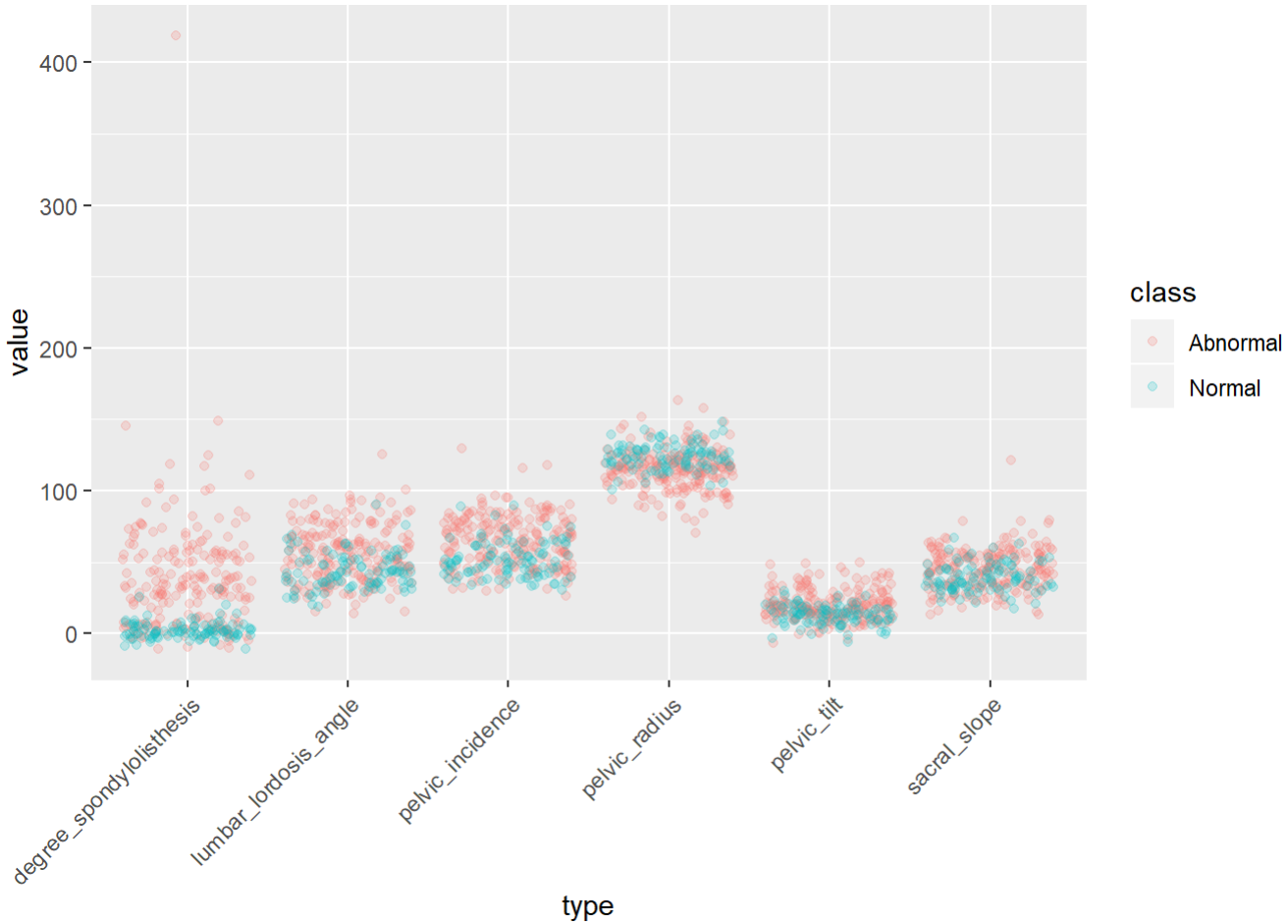
- Datasets exploration
    - Data visualization and structure
- Models fitting
    - Logistic regression
    - K-nearest neighbors
    - random forest
- Analysis and optimization

2. Datasets exploration

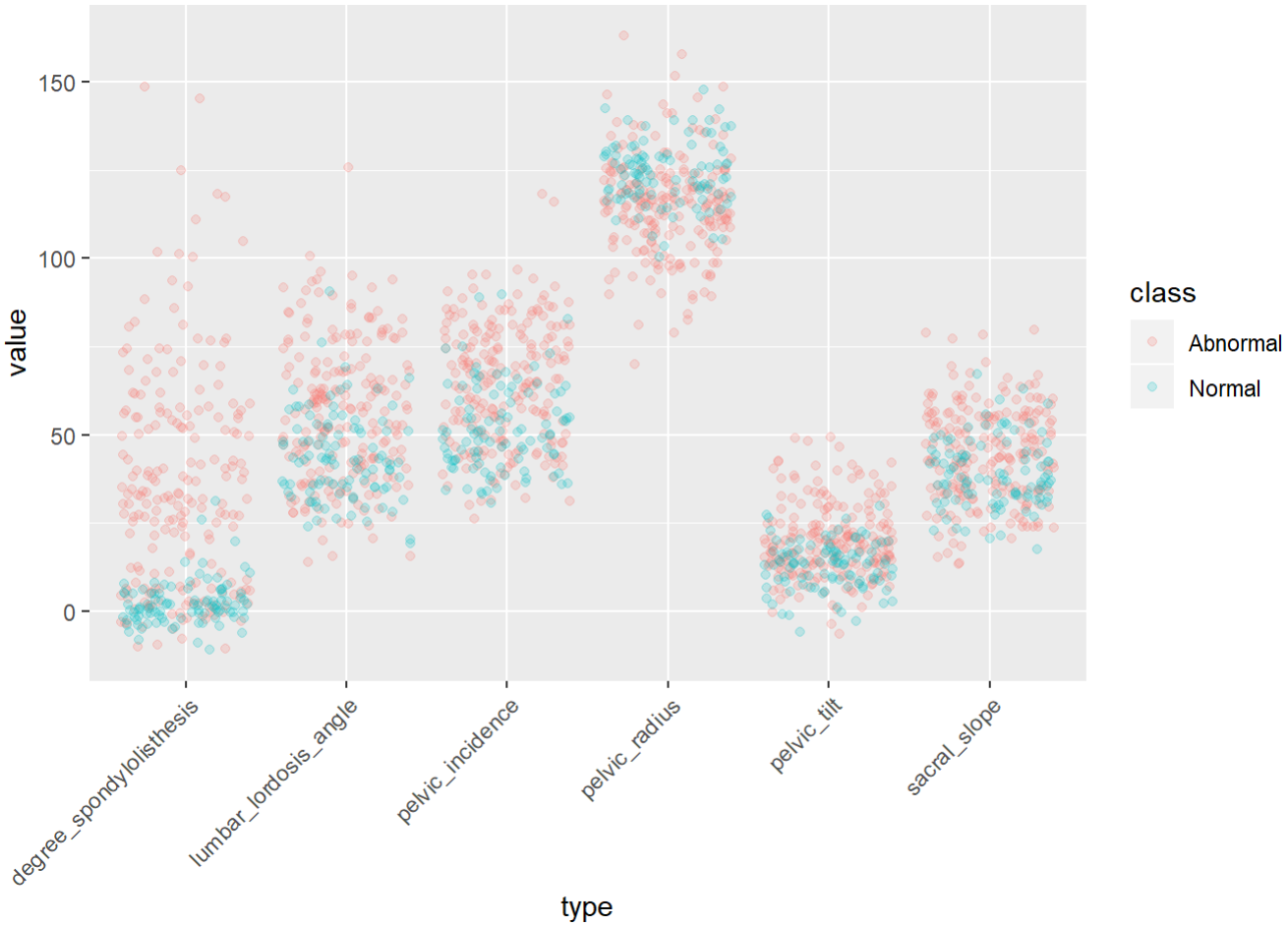
2.1. Visualizing the data

We start by plotting the distribution of each variables in dat1:

```
dat1%>%gather(type,value,-class)%>%
  ggplot(aes(type,value,color=class))+
    geom_jitter(alpha=0.2)+
    theme(axis.text.x=element_text(angle=45,hjust=1))
```

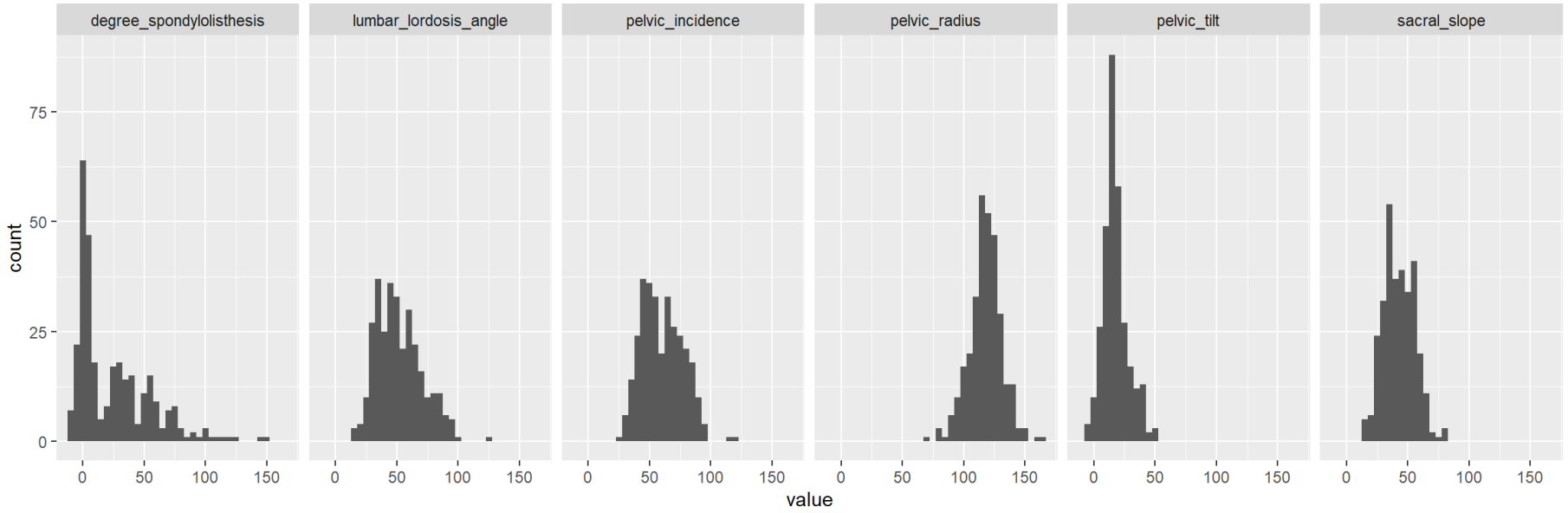


There is a clear outlier at degree_spondylolisthesis >400 possibly due to error. Therefore, we exclude that particular observation from analysis:



Spondylolisthesis degree seems to have the highest predictive power: abnormality is expected from high degree. With histograms, it is easier to analyze the type of distributions each of the predictors has:

```
dat1%>%gather(type,value,-class)%>%
  ggplot(aes(value))+
    geom_histogram(binwidth=5)+
    facet_grid(.~type)
```



We can see that distributions of degree_spondylolisthesis is the only one not well approximated by normal distribution.

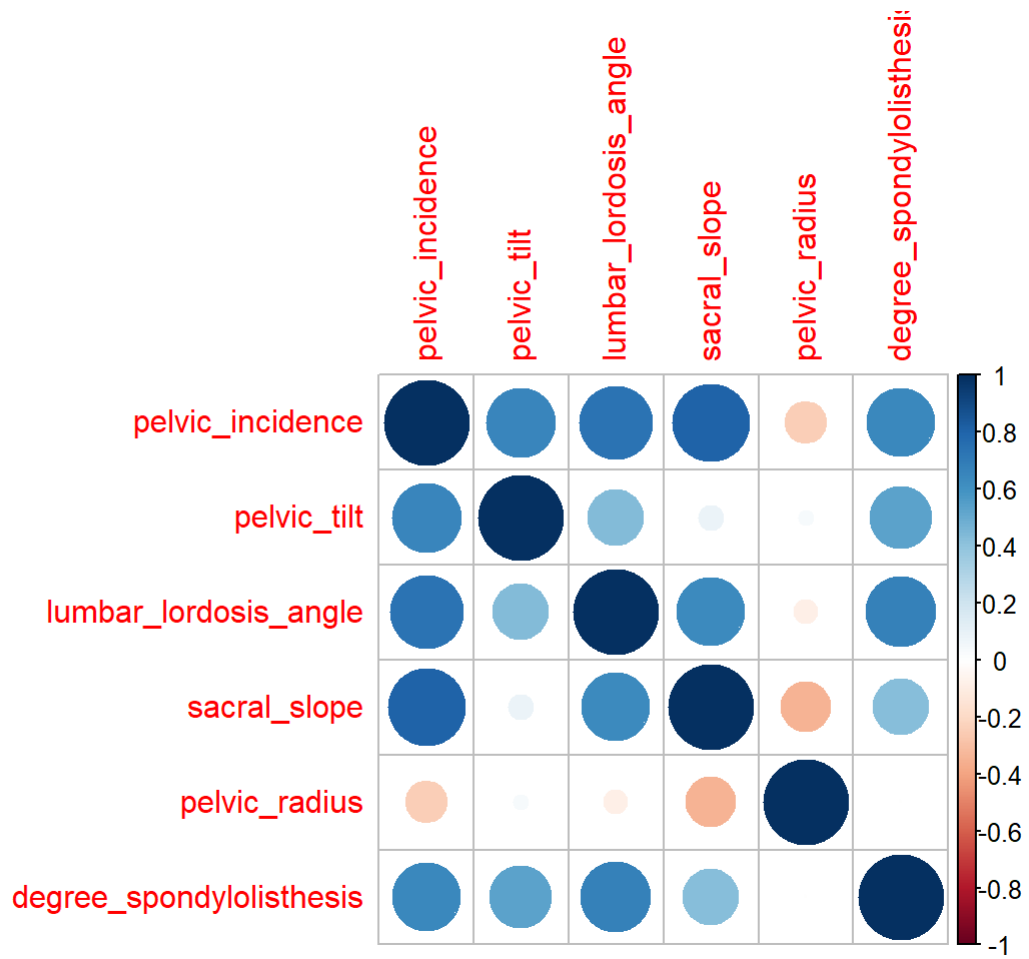
2.2. Data structure

Correlation matrix can be constructed to observe dependencies between predictors:

```
cor1<-cor(dat1[1:(length(dat1)-1)])
cor1
```

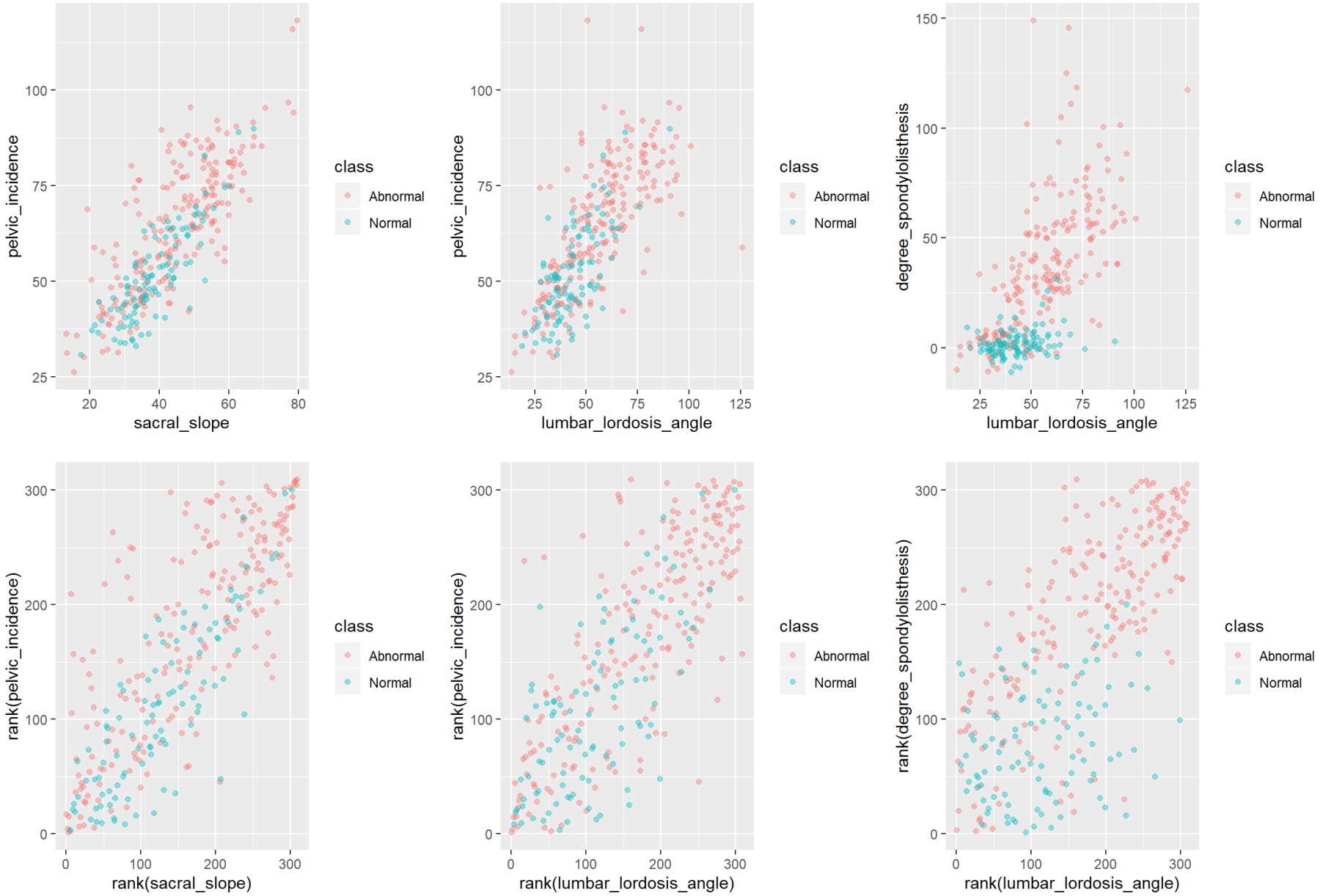
```
##                          pelvic_incidence pelvic_tilt lumbar_lordosis_angle
## pelvic_incidence                1.0000000  0.65955336            0.73951039
## pelvic_tilt                     0.6595534  1.00000000            0.43280896
## lumbar_lordosis_angle           0.7395104  0.43280896            1.00000000
## sacral_slope                    0.8047719  0.08461782            0.63852139
## pelvic_radius                  -0.2441626  0.03046095           -0.08090272
## degree_spondylolisthesis        0.6421566  0.53429988            0.67213848
##                          sacral_slope pelvic_radius degree_spondylolisthesis
## pelvic_incidence           0.80477188 -0.2441626041             0.6421565739
## pelvic_tilt                0.08461782  0.0304609451             0.5342998803
## lumbar_lordosis_angle      0.63852139 -0.0809027207             0.6721384821
## sacral_slope               1.00000000 -0.3477221489             0.4293197149
## pelvic_radius             -0.34772215  1.0000000000            -0.0000226799
## degree_spondylolisthesis   0.42931971 -0.0000226799             1.0000000000
```

```
corrplot(cor1)
```



The high correlations between predictors is obvious:

```
cor_plot_1<-dat1%>%
  ggplot(aes(sacral_slope,pelvic_incidence,color=class))+
    geom_point(alpha=0.4)
cor_plot_2<-dat1%>%
  ggplot(aes(lumbar_lordosis_angle,pelvic_incidence,color=class))+
    geom_point(alpha=0.4)
cor_plot_3<-dat1%>%
  ggplot(aes(lumbar_lordosis_angle,degree_spondylolisthesis,color=class))+
    geom_point(alpha=0.4)
grid.arrange(cor_plot_1,cor_plot_2,cor_plot_3,nrow=1)
```



3. Models fitting

In this section, several fitting models were explored and compared. Here is the list of algorithms discussed:

LIST

Evaluation was done using different metrics such as accuracy, precision, F1 score (collectively referred to as **results**). Tuning parameters are optimized whenever applicable. Result from optimized model in each method applied to a test set is documented in a table.

Default bootstrap samples proportions and numbers of bootstrap sets are used.Test set is comprised of 20% observations, randomly selected using Monte Carlo simulation:

```
test_index1<-createDataPartition(dat1$class,times=1,p=0.2)
test_set1<-dat1[unlist(test_index1),]
train_set1<-dat1[-unlist(test_index1),]
```

```
test_index2<-createDataPartition(dat2$class,times=1,p=0.2)
test_set2<-dat2[unlist(test_index2),]
train_set2<-dat2[-unlist(test_index2),]
```

and prevalence of abnormality in **train_set1** and **train_set2** is approximately 3:

```
train_set1%>%group_by(class)%>%summarize(n=n())
```

```
## # A tibble: 2 x 2
##   class        n
##   <fct>    <int>
## 1 Abnormal   167
## 2 Normal      80
```

```
train_set2%>%group_by(class)%>%summarize(n=n())
```

```
## # A tibble: 3 x 2
##   class                n
##   <fct>            <int>
## 1 Hernia              48
## 2 Spondylolisthesis   120
## 3 Normal              80
```

General function to return accuracies, precision, recall and F1 score are defined with “Abnormal” or “0” as the positive. In this case, it is important to maximize recall to catch as much abnormality as possible.

```
#results function for Data 1 - returns a list of accuracy,recall,precision and F1
results_func<-function(predictions,tests){ #predictions and tests are both vectors
  cf<-confusionMatrix(data=predictions,reference=tests)
  c(cf$overall['Accuracy'],cf$byClass['Sensitivity'],cf$byClass['Precision'],cf$byClass['F1'])
}
#results function for Data 2 - returns a list of (in order): accuracy, recall(Her),precision(Her), F1(Her),recall(Spo),preci
sion(Spo),F1(Spo)
results_func2<-function(predictions,tests){ #predictions and tests are both vectors
  cf<-confusionMatrix(data=predictions,reference=tests)
  c(cf$overall['Accuracy'],cf$byClass[1,1],cf$byClass[1,2],cf$byClass[1,7],cf$byClass[2,1],cf$byClass[2,2],cf$byClass[2,7])
}
```

3.1. Benchmark

Before exploring different machine learning algorithms, we can make use of the high predicting power of one of the predictors, degree_spondylolisthesis to construct a rather simple prediction. For Data 1, we can predict high degree_spondylolisthesis with “Abnormal”. Here we experimented with cutoff values from -12 to 25.

```
cut_dp<-seq(-12,25,0.1)
predict_benchmark<-function(cutoff){ #data is a vector of degree_spondylolisthesis
  mean(ifelse(train_set1$degree_spondylolisthesis>cutoff,"Abnormal","Normal")==train_set1$class)
}
tibble(cutoff=cut_dp, train_accuracy=sapply(cut_dp,predict_benchmark))%>%arrange(desc(train_accuracy))
```

```
## # A tibble: 371 x 2
##    cutoff train_accuracy
##     <dbl>          <dbl>
##  1   5.4           0.810
##  2   6.1           0.810
##  3   6.20          0.810
##  4   6.3           0.810
##  5   5.1           0.806
##  6   5.20          0.806
##  7   5.3           0.806
##  8   5.5           0.806
##  9   5.6           0.806
## 10   5.70          0.806
## # … with 361 more rows
```

For simplicity, only cutoff value of 5.4 is used to evaluate test set corresponding to Data 1. Here is the result:

```
predictions_benchmark1<-factor(ifelse(test_set1$degree_spondylolisthesis>5.4,"Abnormal","Normal"),levels=c("Abnormal","Norma
l"))
results_benchmark1<-results_func(predictions_benchmark1,test_set1$class)
tab_benchmark1<-tibble(data="Data 1",model="benchmark",accuracy=results_benchmark1[1],recall=results_benchmark1[2],precision
=results_benchmark1[3],F1=results_benchmark1[4])
tab_benchmark1
```

```
## # A tibble: 1 x 6
##   data   model     accuracy recall precision    F1
##   <chr>  <chr>        <dbl>  <dbl>     <dbl> <dbl>
## 1 Data 1 benchmark    0.774  0.762     0.889 0.821
```

Benchmarking for Data 2 requires more analysis and deeper domain expertise. Without in-depth knowledge on abnormalities, we can perhaps still construct a benchmark for Data 2 by randomly predicting “Hernia” or “Spondylolisthesis” for each observation falling under “Abnormal” class. However, the accuracy of such benchmark will be noticeably lower than in Data 1’s case.

3.2. Logistic regression

Logistic regression is commonly used when the outcome is binary. To train a model using all predictors, we can use the following code:

```
fit_glm<-train(class~degree_spondylolisthesis+pelvic_incidence+lumbar_lordosis_angle+sacral_slope+pelvic_radius+pelvic_tilt,
               method="glm",data=train_set1,family="binomial")
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
predictions_glm<-predict(fit_glm,newdata=test_set1,type="raw")
results_glm<-results_func(predictions_glm,test_set1$class)
```

We see an error stating probabilities of 0 and 1 are observed in the model fit_glm. Viewing the predictions in descending order of probability_0 helps visualizing the problem.

```
predict(fit_glm,newdata=test_set1,type="prob")%>%arrange(desc(Abnormal))%>%head
```

```
##    Abnormal       Normal
## 1 1.0000000 1.490105e-08
## 2 0.9999937 6.256383e-06
## 3 0.9999869 1.309842e-05
## 4 0.9999832 1.679752e-05
## 5 0.9999782 2.180925e-05
## 6 0.9999600 4.004629e-05
```

Some of the values of probability_0 are 1 or really close to 1 (or close to 0 for probability_1). Earlier we Restablished that degree_spondylolisthesis of >50 seems to result in 100% probability of being abnormal. Therefore, we can exclude the corresponding observations to get rid of the error.

```
fit_glm2<-train(class~degree_spondylolisthesis+pelvic_incidence+lumbar_lordosis_angle+sacral_slope+pelvic_radius+pelvic_til
t,
               method="glm",data=train_set1%>%filter(degree_spondylolisthesis<=50),family="binomial")
predictions_glm2<-predict(fit_glm2,newdata=test_set1,type="raw")
results_glm2<-results_func(predictions_glm2,test_set1$class)
```

Note than the resulting model after exclusion seems to give identical prediction.

```
identical(predictions_glm,predictions_glm2)
```

```
## [1] TRUE
```

Here is the results:

```
## # A tibble: 1 x 6
##   data   model accuracy recall precision    F1
##   <chr>  <chr>    <dbl>  <dbl>     <dbl> <dbl>
## 1 Data 1 glm      0.887  0.929     0.907 0.918
```

3.3. K-nearest neighbors (knn)

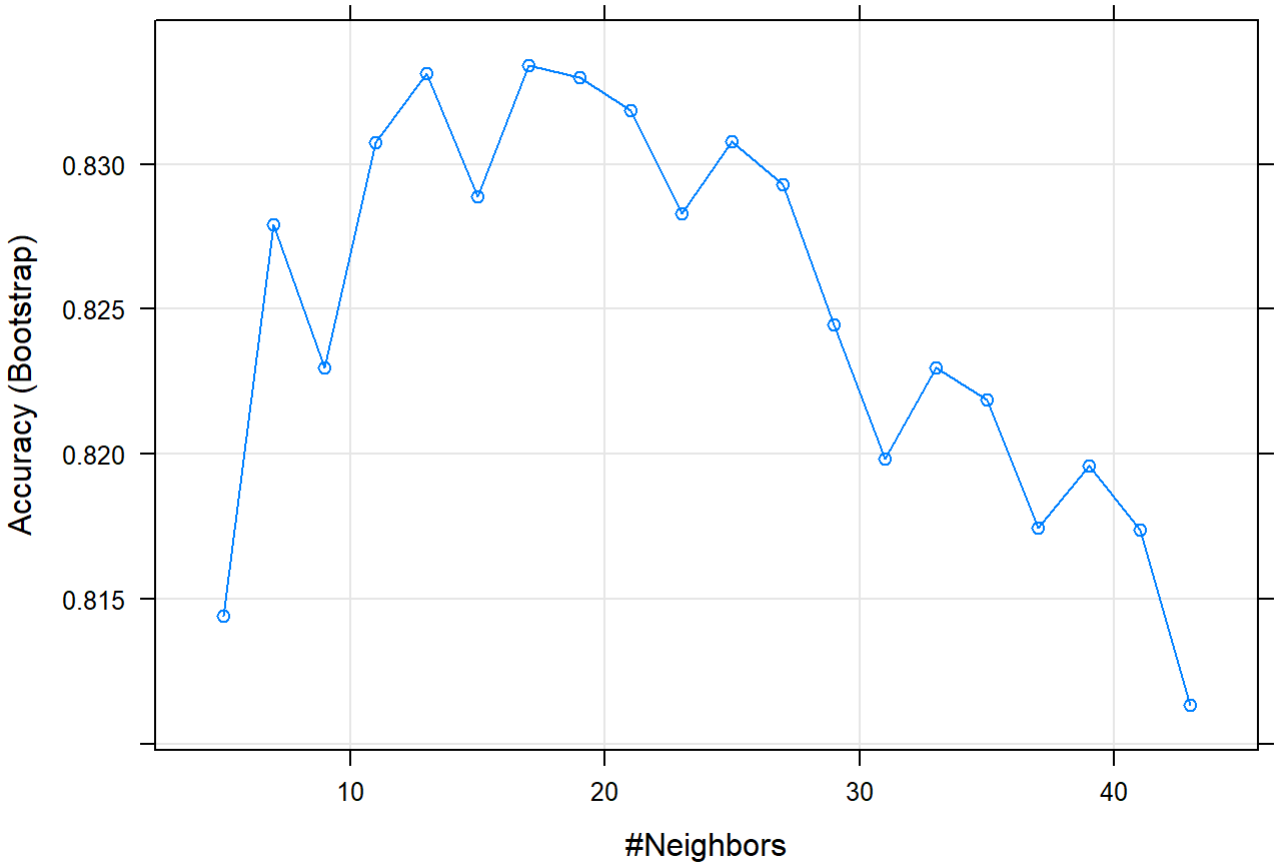
3.3.1 Data 1 - Bootstrap

Here we will train knn models for different values of k (tuneLength = 20) and evaluate their performances on the test set. The plot and training results are summarized below.

```
fit_knn<-train(class~degree_spondylolisthesis+pelvic_incidence+lumbar_lordosis_angle+sacral_slope+pelvic_radius+pelvic_tilt,
               method="knn",data=train_set1,tuneLength=20)
predictions_knn<-predict(fit_knn,newdata=test_set1,type="raw")
results_knn<-results_func(predictions_knn,test_set1$class)
fit_knn$results%>%arrange(desc(Accuracy))%>%head
```

```
##    k  Accuracy     Kappa AccuracySD    KappaSD
## 1 17 0.8334203 0.6262354 0.03859726 0.07886798
## 2 13 0.8331452 0.6217393 0.03105112 0.06975487
## 3 19 0.8330070 0.6251613 0.03857215 0.08221241
## 4 21 0.8318695 0.6276397 0.04131810 0.08092271
## 5 25 0.8307949 0.6255792 0.03627603 0.07373867
## 6 11 0.8307615 0.6176929 0.02898165 0.06411170
```

```
plot(fit_knn)
```



Results for **Data 1** with **knn** and **default bootstrap**:

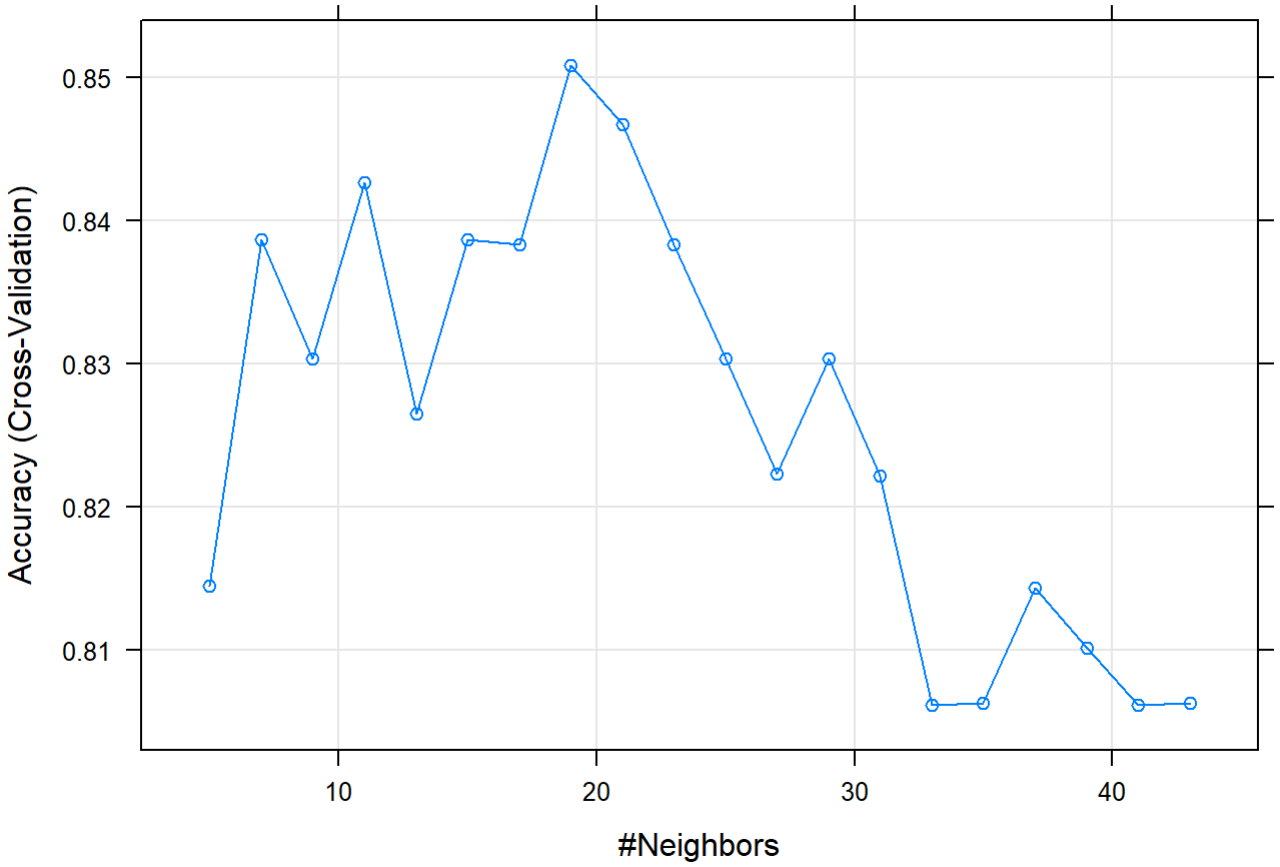
```
## # A tibble: 1 x 6
##   data   model         accuracy recall precision    F1
##   <chr>  <chr>            <dbl>  <dbl>     <dbl> <dbl>
## 1 Data 1 knn_bootstrap    0.839  0.810     0.944 0.872
```

3.3.2. Data 1 - 10-fold cross validation

```
fit_knn_cv1<-train(class~degree_spondylolisthesis+pelvic_incidence+lumbar_lordosis_angle+sacral_slope+pelvic_radius+pelvic_t
ilt,
               method="knn",data=train_set1,tuneLength=20,trControl=(trainControl(method="cv",number=10)))
predictions_knn_cv1<-predict(fit_knn_cv1,newdata=test_set1,type="raw")
results_knn_cv1<-results_func(predictions_knn_cv1,test_set1$class)
fit_knn_cv1$results%>%arrange(desc(Accuracy))%>%head
```

```
##    k  Accuracy     Kappa AccuracySD   KappaSD
## 1 19 0.8508333 0.6637970 0.09177687 0.2097332
## 2 21 0.8466667 0.6539113 0.08120999 0.1871385
## 3 11 0.8426667 0.6534943 0.07792922 0.1622717
## 4  7 0.8386667 0.6334309 0.08168669 0.1858445
## 5 15 0.8386667 0.6397879 0.09724539 0.2188722
## 6 17 0.8383333 0.6372547 0.08384289 0.1963573
```

```
plot(fit_knn_cv1)
```



Results for **Data 1** with **knn** and **10-fold cv**:

```
## # A tibble: 1 x 6
##   data   model  accuracy recall precision    F1
##   <chr>  <chr>     <dbl>  <dbl>     <dbl> <dbl>
## 1 Data 1 knn_cv    0.823  0.786     0.943 0.857
```

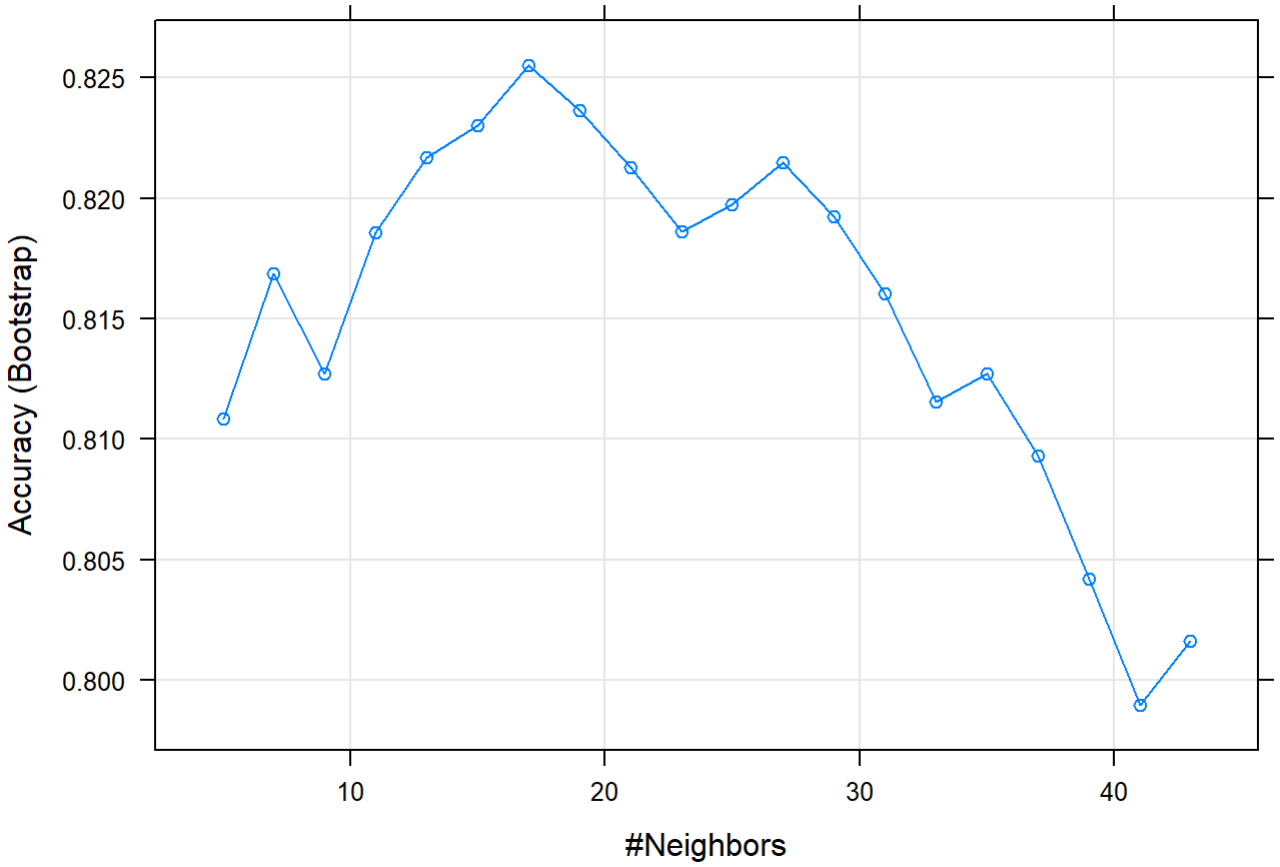
3.3.3. Data 2 - Bootstrap

Knn is supports non-linearity and allows for classification into more than 2 classes. Here is the same procedure done on the second data set with 3 classes of outcome.

```
fit_knn2<-train(class~degree_spondylolisthesis+pelvic_incidence+lumbar_lordosis_angle+sacral_slope+pelvic_radius+pelvic_til
t,
                method="knn",data=train_set2,tuneLength=20)
predictions_knn2<-predict(fit_knn2,newdata=test_set2,type="raw")
results_knn2<-results_func2(predictions_knn2,test_set2$class)
fit_knn2$results%>%arrange(desc(Accuracy))%>%head
```

```
##    k  Accuracy     Kappa AccuracySD    KappaSD
## 1 17 0.8255229 0.7190513 0.03575143 0.05319243
## 2 19 0.8236177 0.7161877 0.03330890 0.04906681
## 3 15 0.8230223 0.7149849 0.03664479 0.05615110
## 4 13 0.8216955 0.7128330 0.03802876 0.05855947
## 5 27 0.8214736 0.7121494 0.03403965 0.05291110
## 6 21 0.8212839 0.7121036 0.03246650 0.04831569
```

```
plot(fit_knn2)
```



Results for **Data 2** with **knn** and **bootstrap**:

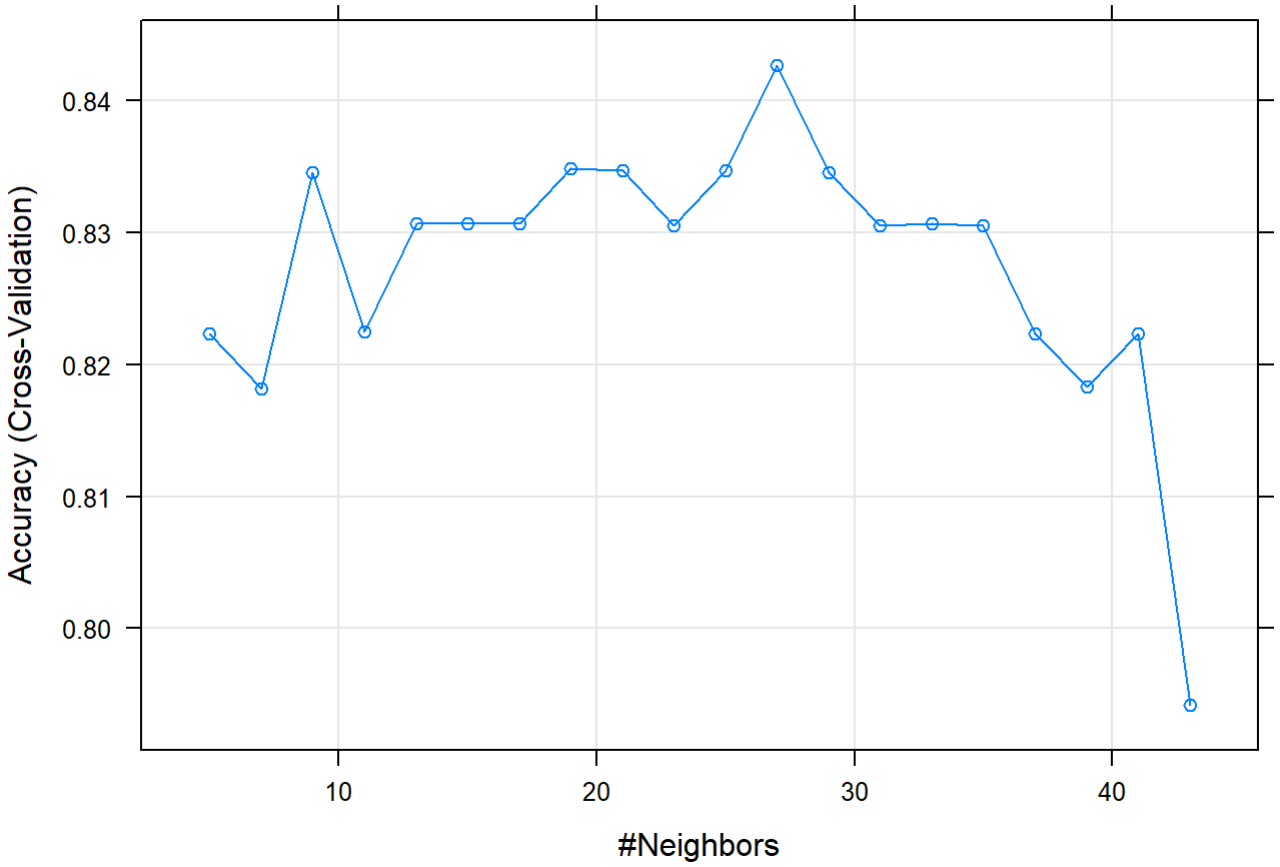
```
## # A tibble: 1 x 9
##   data  model accuracy recall_hernia precision_hernia F1_hernia recall_spond
##   <chr> <chr>    <dbl>         <dbl>            <dbl>     <dbl>        <dbl>
## 1 Data… knn_…    0.855         0.583             0.96     0.667        0.933
## # … with 2 more variables: precision_spond <dbl>, F1_spond <dbl>
```

3.3.4. Data 2 - 10-fold cross validation

```
fit_knn_cv2<-train(class~degree_spondylolisthesis+pelvic_incidence+lumbar_lordosis_angle+sacral_slope+pelvic_radius+pelvic_t
ilt,
               method="knn",data=train_set2,tuneLength=20,trControl=(trainControl(method="cv",number=10)))
predictions_knn_cv2<-predict(fit_knn_cv2,newdata=test_set2,type="raw")
results_knn_cv2<-results_func2(predictions_knn_cv2,test_set2$class)
fit_knn_cv2$results%>%arrange(desc(Accuracy))%>%head
```

```
##    k  Accuracy     Kappa AccuracySD    KappaSD
## 1 27 0.8426667 0.7478750 0.05195083 0.08293530
## 2 19 0.8348333 0.7355140 0.05158267 0.08179379
## 3 21 0.8346667 0.7351148 0.04816253 0.07649915
## 4 25 0.8346667 0.7355095 0.05172219 0.08161677
## 5  9 0.8345000 0.7337946 0.07009715 0.11175690
## 6 29 0.8345000 0.7351113 0.04862333 0.07653480
```

```
plot(fit_knn_cv2)
```




```
## # A tibble: 1 x 9
##   data  model accuracy recall_hernia precision_hernia F1_hernia recall_spond
##   <chr> <chr>    <dbl>         <dbl>            <dbl>     <dbl>        <dbl>
## 1 Data… knn_…    0.823         0.417             0.94       0.5        0.933
## # … with 2 more variables: precision_spond <dbl>, F1_spond <dbl>
```

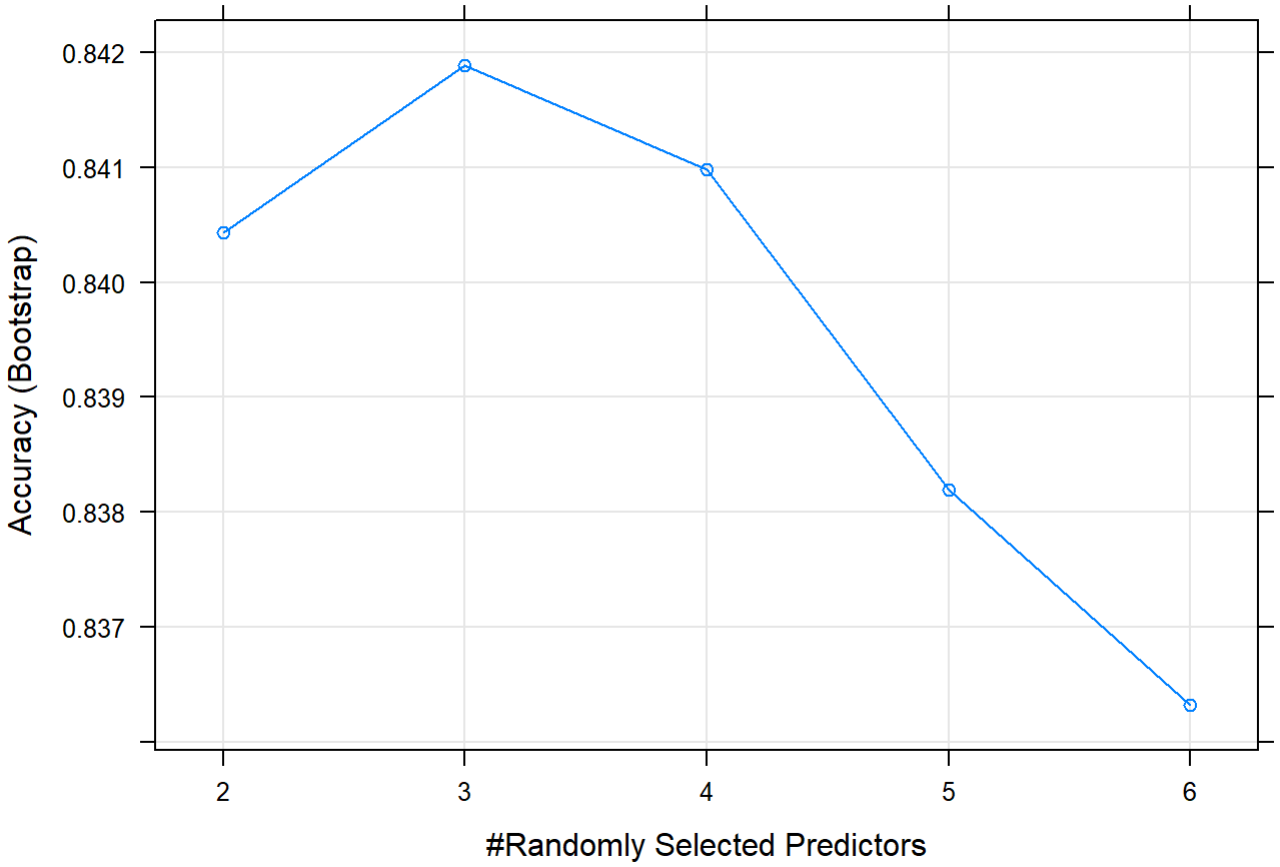
3.4. Random forest

3.4.1. Data 1 - Bootstrap

```
fit_rf<-train(class~degree_spondylolisthesis+pelvic_incidence+lumbar_lordosis_angle+sacral_slope+pelvic_radius+pelvic_tilt,
              method="rf",data=train_set1,tuneLength=5)
predictions_rf<-predict(fit_rf,newdata=test_set1,type="raw")
results_rf<-results_func(predictions_rf,test_set1$class)
fit_rf$results%>%arrange(desc(Accuracy))%>%head
```

```
##   mtry  Accuracy     Kappa AccuracySD    KappaSD
## 1    3 0.8418890 0.6310622 0.03735072 0.07521805
## 2    4 0.8409831 0.6311877 0.03864188 0.07872386
## 3    2 0.8404350 0.6269150 0.03656355 0.07348183
## 4    5 0.8381940 0.6250009 0.03884686 0.07725587
## 5    6 0.8363191 0.6218736 0.04025754 0.08239701
```

```
plot(fit_rf)
```



Results for **Data 1** with **rf** and **default bootstrap**:

```
varImp(fit_rf)
```

```
## rf variable importance
##
##                            Overall
## degree_spondylolisthesis 100.0000
## pelvic_radius             20.7029
## sacral_slope              11.5179
## pelvic_incidence           4.6395
## pelvic_tilt                0.9866
## lumbar_lordosis_angle      0.0000
```

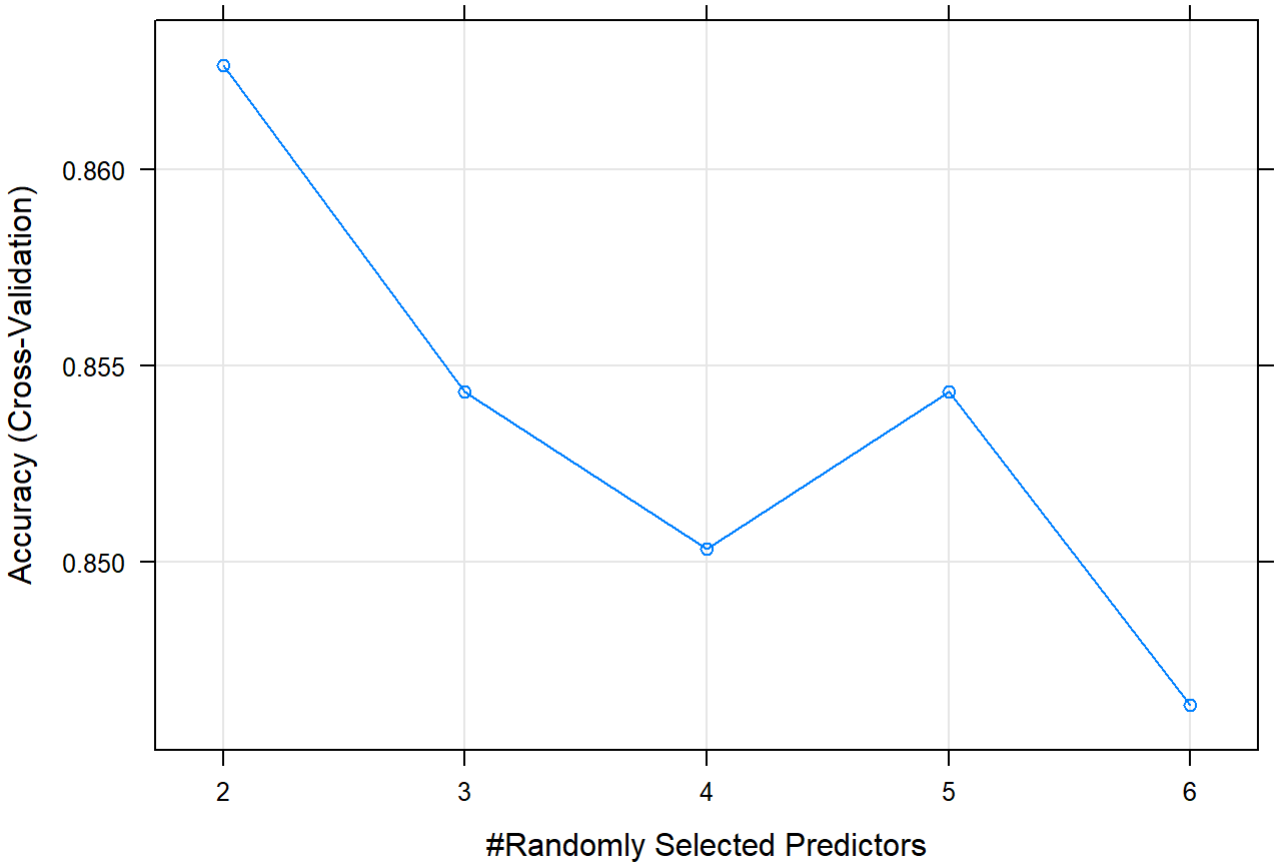
```
## # A tibble: 1 x 6
##   data   model        accuracy recall precision    F1
##   <chr>  <chr>           <dbl>  <dbl>     <dbl> <dbl>
## 1 Data 1 rf_bootstrap    0.823  0.810     0.919 0.861
```

3.4.2. Data 1 - 10-fold cross validation

```
fit_rf_cv1<-train(class~degree_spondylolisthesis+pelvic_incidence+lumbar_lordosis_angle+sacral_slope+pelvic_radius+pelvic_ti
lt,
               method="rf",data=train_set1,tuneLength=5,trControl=trainControl(method="cv",number=10))
predictions_rf_cv1<-predict(fit_rf_cv1,newdata=test_set1,type="raw")
results_rf_cv1<-results_func(predictions_rf_cv1,test_set1$class)
fit_rf$results%>%arrange(desc(Accuracy))%>%head
```

```
##   mtry  Accuracy     Kappa AccuracySD    KappaSD
## 1    3 0.8418890 0.6310622 0.03735072 0.07521805
## 2    4 0.8409831 0.6311877 0.03864188 0.07872386
## 3    2 0.8404350 0.6269150 0.03656355 0.07348183
## 4    5 0.8381940 0.6250009 0.03884686 0.07725587
## 5    6 0.8363191 0.6218736 0.04025754 0.08239701
```

```
plot(fit_rf_cv1)
```



Results for **Data 1** with **rf** and **10-fold cross validation**:

```
varImp(fit_rf_cv1)
```

```
## rf variable importance
##
##                           Overall
## degree_spondylolisthesis 100.000
## pelvic_radius             24.383
## pelvic_incidence           3.414
## sacral_slope               2.588
## pelvic_tilt                1.547
## lumbar_lordosis_angle      0.000
```

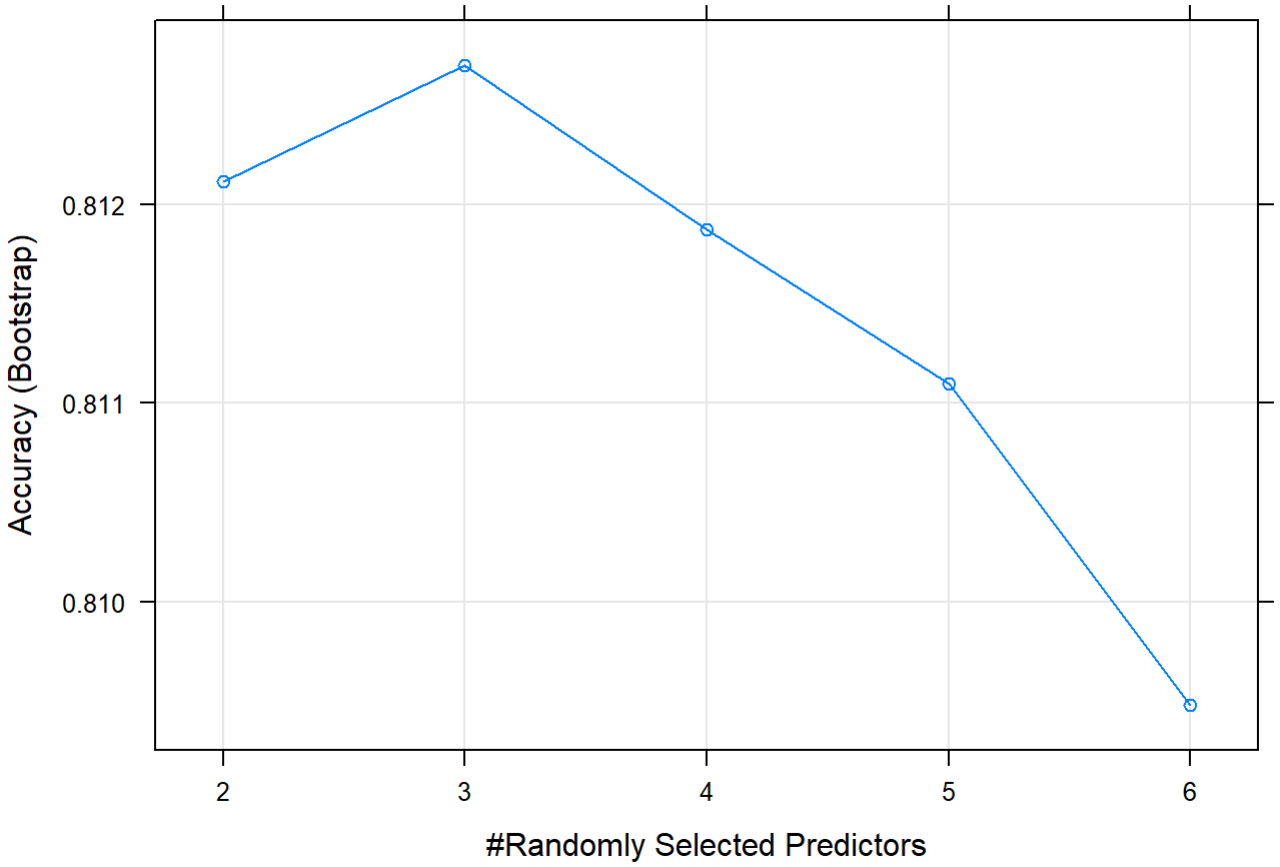
```
## # A tibble: 1 x 6
##   data   model accuracy recall precision    F1
##   <chr>  <chr>    <dbl>  <dbl>     <dbl> <dbl>
## 1 Data 1 rf_cv    0.839  0.810     0.944 0.872
```

3.4.3. Data 2 - Bootstrap

```
fit_rf2<-train(class~degree_spondylolisthesis+pelvic_incidence+lumbar_lordosis_angle+sacral_slope+pelvic_radius+pelvic_tilt,
               method="rf",data=train_set2,tuneLength=5)
predictions_rf2<-predict(fit_rf2,newdata=test_set2,type="raw")
results_rf2<-results_func2(predictions_rf2,test_set2$class)
fit_rf2$results%>%arrange(desc(Accuracy))%>%head
```

```
##   mtry  Accuracy     Kappa AccuracySD    KappaSD
## 1    3 0.8127017 0.6998189 0.03414749 0.05076528
## 2    2 0.8121171 0.6989264 0.03692254 0.05486647
## 3    4 0.8118729 0.6987026 0.03609257 0.05333820
## 4    5 0.8110993 0.6976400 0.03247376 0.04779947
## 5    6 0.8094792 0.6951252 0.03582527 0.05310205
```

```
plot(fit_rf2)
```



Results for **Data 2** with **rf** and **default bootstrap**:

```
varImp(fit_rf2)
```

```
## rf variable importance
##
##                           Overall
## degree_spondylolisthesis  100.00
## sacral_slope               15.65
## pelvic_incidence           12.46
## lumbar_lordosis_angle      10.36
## pelvic_radius              10.31
## pelvic_tilt                 0.00
```

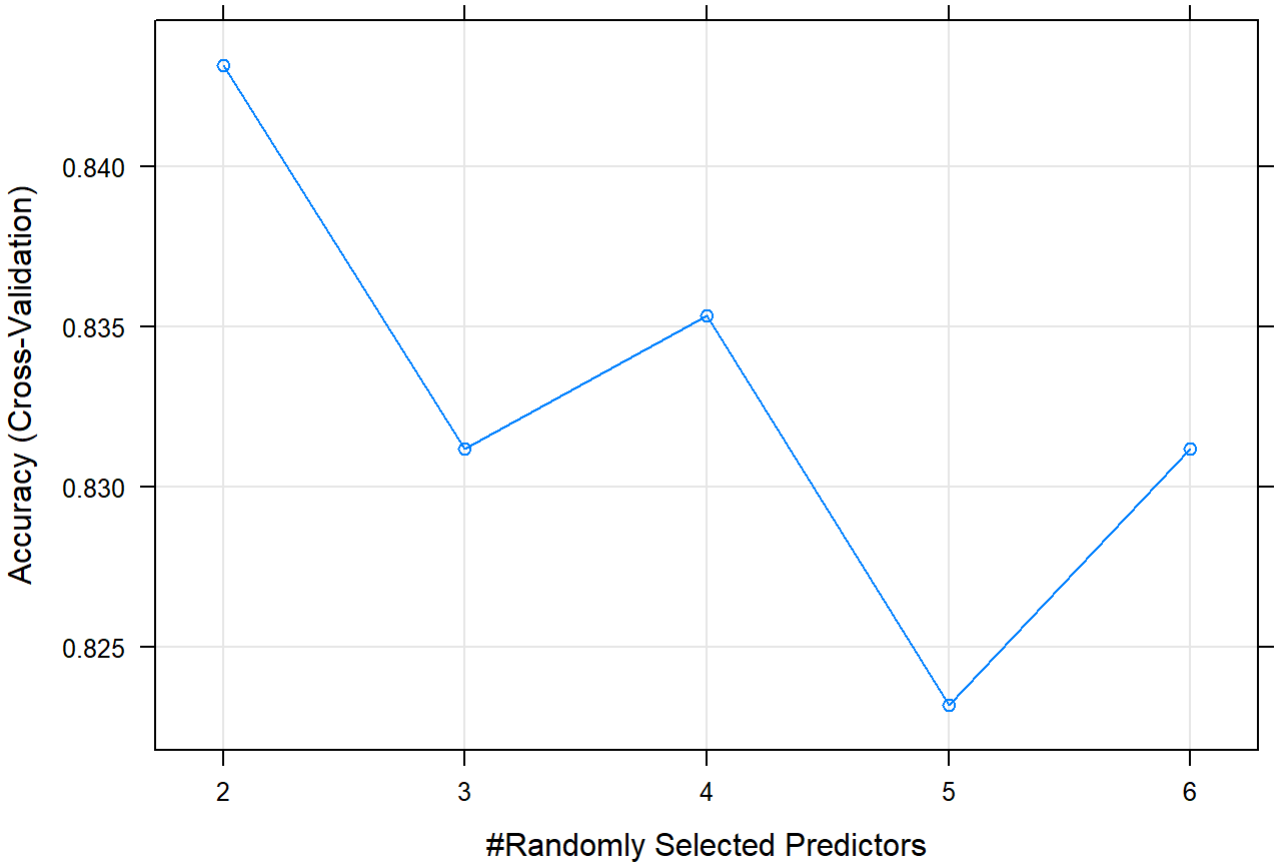
```
## # A tibble: 1 x 9
##   data  model accuracy recall_hernia precision_hernia F1_hernia recall_spond
##   <chr> <chr>    <dbl>         <dbl>            <dbl>     <dbl>        <dbl>
## 1 Data… rf_b…    0.839         0.583             0.96     0.667        0.933
## # … with 2 more variables: precision_spond <dbl>, F1_spond <dbl>
```

3.4.4. Data 2 - 10-fold cross validation

```
fit_rf_cv2<-train(class~degree_spondylolisthesis+pelvic_incidence+lumbar_lordosis_angle+sacral_slope+pelvic_radius+pelvic_ti
lt,
               method="rf",data=train_set2,tuneLength=5,trControl=trainControl(method="cv",number=10))
predictions_rf_cv2<-predict(fit_rf_cv2,newdata=test_set2,type="raw")
results_rf_cv2<-results_func2(predictions_rf_cv2,test_set2$class)
fit_rf_cv2$results%>%arrange(desc(Accuracy))%>%head
```

```
##   mtry  Accuracy     Kappa AccuracySD   KappaSD
## 1    2 0.8431667 0.7487161 0.06847749 0.1058539
## 2    4 0.8353333 0.7383262 0.08260347 0.1263715
## 3    3 0.8311667 0.7302096 0.07882646 0.1205292
## 4    6 0.8311667 0.7317866 0.07653794 0.1164644
## 5    5 0.8231667 0.7187276 0.08183260 0.1242143
```

```
plot(fit_rf_cv2)
```



Results for **Data 2** with **rf** and **10-fold cross validation**:

```
varImp(fit_rf_cv2)
```

```
## rf variable importance
##
##                           Overall
## degree_spondylolisthesis  100.00
## lumbar_lordosis_angle      16.83
## sacral_slope               16.37
## pelvic_incidence           15.85
## pelvic_radius              12.67
## pelvic_tilt                 0.00
```

```
## # A tibble: 1 x 9
##   data  model accuracy recall_hernia precision_hernia F1_hernia recall_spond
##   <chr> <chr>    <dbl>         <dbl>            <dbl>     <dbl>        <dbl>
## 1 Data… rf_cv    0.839           0.5             0.98     0.632        0.933
## # … with 2 more variables: precision_spond <dbl>, F1_spond <dbl>
```

4. Analysis and Optimization

Results obtained from fitted models for both Data 1 and Data 2 can be summarized into the following tables:

```
do.call("rbind",list(tab_benchmark1,tab_glm,tab_knn_boot1,tab_knn_cv1,tab_rf_boot1,tab_rf_cv1))
```

```
## # A tibble: 6 x 6
##   data   model         accuracy recall precision    F1
##   <chr>  <chr>            <dbl>  <dbl>     <dbl> <dbl>
## 1 Data 1 benchmark        0.774  0.762     0.889 0.821
## 2 Data 1 glm              0.887  0.929     0.907 0.918
## 3 Data 1 knn_bootstrap    0.839  0.810     0.944 0.872
## 4 Data 1 knn_cv           0.823  0.786     0.943 0.857
## 5 Data 1 rf_bootstrap     0.823  0.810     0.919 0.861
## 6 Data 1 rf_cv            0.839  0.810     0.944 0.872
```

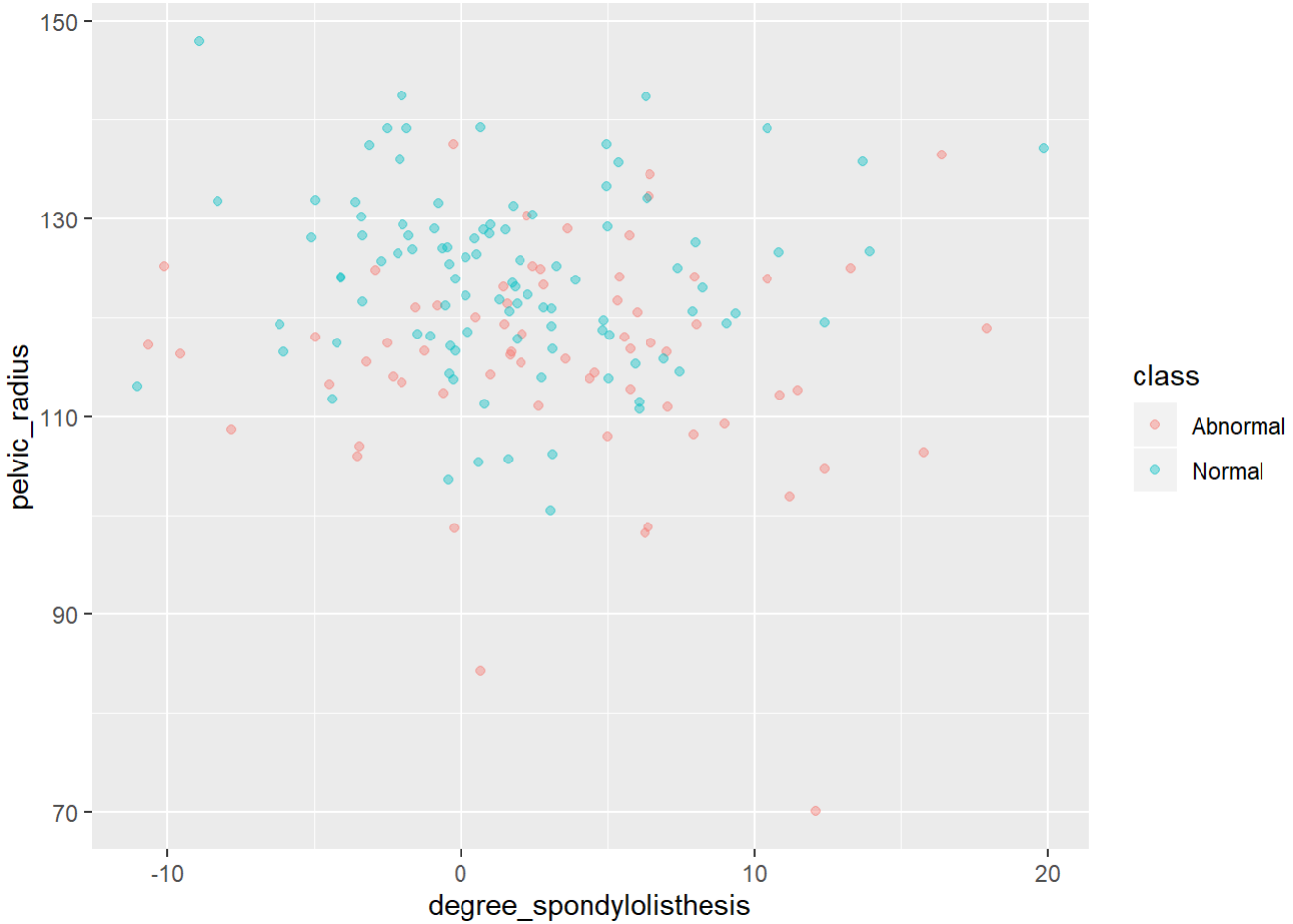
```
do.call("rbind",list(tab_knn_boot2,tab_knn_cv2,tab_rf_boot2,tab_rf_cv2))
```

```
## # A tibble: 4 x 9
##   data  model accuracy recall_hernia precision_hernia F1_hernia recall_spond
##   <chr> <chr>    <dbl>         <dbl>             <dbl>     <dbl>        <dbl>
## 1 Data… knn_…    0.855         0.583              0.96     0.667        0.933
## 2 Data… knn_…    0.823         0.417              0.94     0.5          0.933
## 3 Data… rf_b…    0.839         0.583              0.96     0.667        0.933
## 4 Data… rf_cv    0.839         0.5                0.98     0.632        0.933
## # … with 2 more variables: precision_spond <dbl>, F1_spond <dbl>
```

For Data 1, glm produces a model with the highest accuracy and recall which are prioritized metrics in the context of detecting abnormalities in observations.

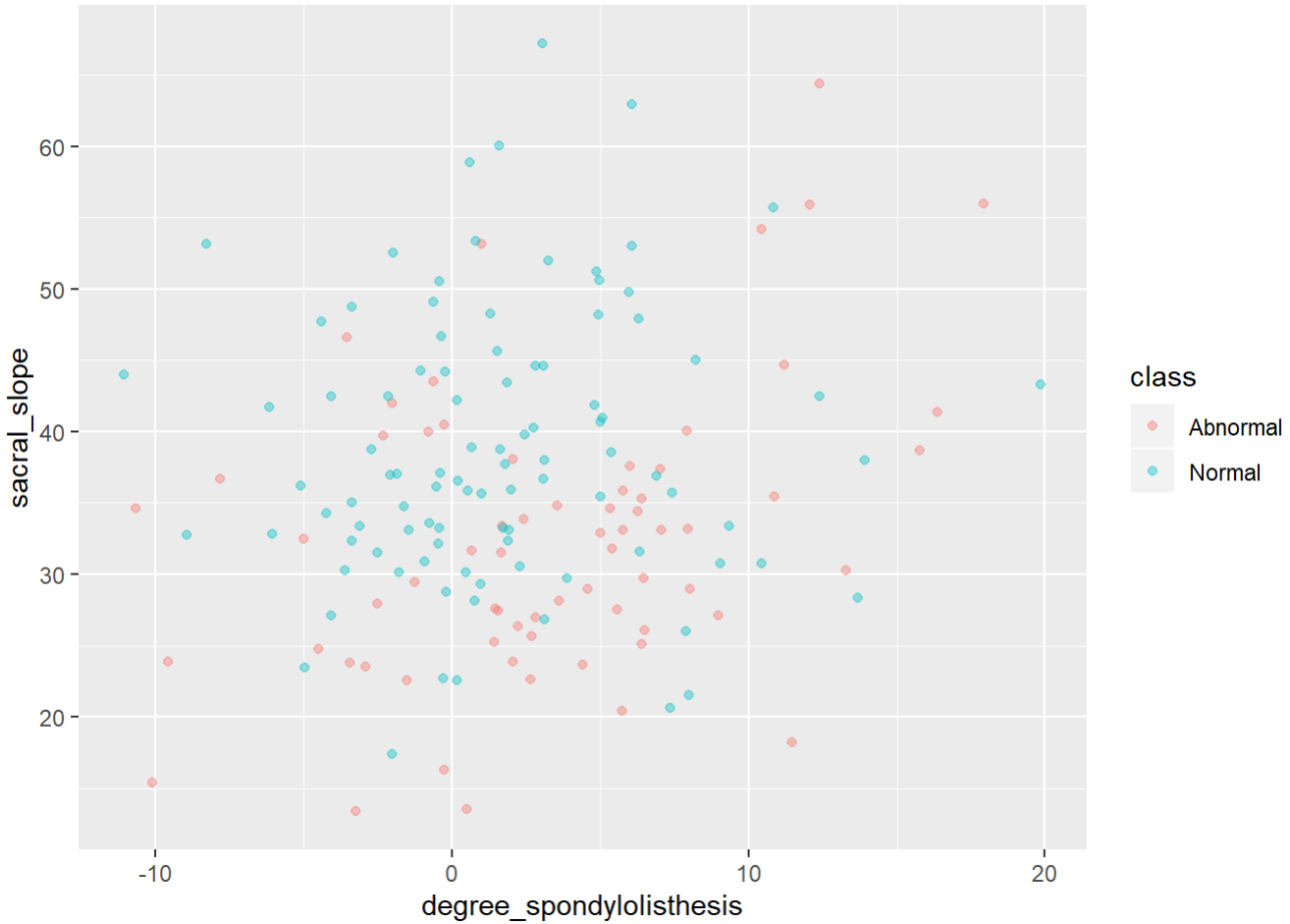
Due to the small size of the datasets, training models using all predictors is computationally manageable. However, we can see in Section 2's correlation chart, that only 1 predictor is not strongly correlated to degree_spondylolisthesis. Therefore, in theory, degree_spondylolisthesis and pelvic_radius are enough to reasonably predict abnormalities.

```
dat1%>%select(degree_spondylolisthesis,pelvic_radius,class)%>%filter(degree_spondylolisthesis<20)%>%
  ggplot(aes(degree_spondylolisthesis,pelvic_radius,color=class))+
  geom_point(alpha=0.4)
```



We can see from the plot that both outcomes are somehow clustered even when degree_spondylolisthesis is low. Higher pelvic_radius tends to indicate normal condition. Similar trend can be observed from combination of degree_spondylolisthesis and sacral_slope.

```
dat1%>%select(degree_spondylolisthesis,sacral_slope,class)%>%filter(degree_spondylolisthesis<20)%>%
  ggplot(aes(degree_spondylolisthesis,sacral_slope,color=class))+
  geom_point(alpha=0.4)
```



Unsurprisingly, random forest method confirms the importance of the three predictors in Data 1's models.