Diagnosis of Abnormal Posture from Pelvic Measurements

Syed Imran

6/16/2019

Executive Summary

Diagnosis of abnormalities from medical measurements is a routine task that is performed by medical practicioners. One such diagnosis involves abnormal posture identification from measurements from the pelvis. Data from 310 patients was available from kaggle that was used to perform the classification task.

During exploratory data analysis, one of the features *pelvic_incidence* was observed to be highly correlated to *sacral_slope* and was eliminated from the machine learning models.

The data was partitioned into equal training and test sets. The training sets were used to train the machine learning methods. Validation was performed using the test sets.

Nineteen different machine learning methods were evaluated simultaneously. The methods with the highest accuracy as well as sensitivity were selected.

An ensemble model with four different methods was combined using **glm** method. However, it failed to give better results than the single method with the highest accuracy in the test set.

The final method selected was **Rborist**. It has an overall accuracy of 0.872 and sensitivity of 0.886 and specificity of 0.843. The confusion matrix on the test set was:

	Reference	
Prediction	Abnormal	Normal
Abnormal	93	8
Normal	12	43

Objective

The ojective of the study is to select appropriate machine learning models to diagnose patients with normal/abnormal posture based on pelvic measurements.

Methods Analysis

Data Description

The data have been organized to help in classifying patients as belonging to one out of two categories: Normal (100 patients) or Abnormal (210 patients). Each patient is represented in the data set by six biomechanical attributes (features) derived from the shape and orientation of the pelvis and lumbar spine (each one is a column): 1. pelvic incidence 2. pelvic tilt 3. lumbar lordosis angle 4. sacral slope 5. pelvic radius 6. grade of spondylolisthesis

Data Sources

The original dataset was downloaded from UCI ML repository: Lichman, M. (2013). [UCI Machine Learning Repository] (http://archive.ics.uci.edu/ml). Irvine, CA: University of California, School of Information and Computer Science Files were converted to CSV and downloaded from [Kaggle] (https://www.kaggle.com/uciml/biomechanical-features-of-orthopedic-patients).

Machine Learning Libraries

In this project, we will evaluate different machine learining algorithms (models) in caret. The *outcome* is categorical binary and can be *normal* or *abnormal*. There are six different numeric *features* which can help in the classification.
For this project the main libraries required are **tidyverse** and **caret**. Additional libraries such as **gridExtra** and **kableExtra** were required for outputting exploratory data. **caret** package also requires downloading different models used in this project if they are not available on the user's system. The library **caretEnsemble** was used to develop and test an ensemble model of multiple models.

First we load the required libraries

```
# Load requred libraries. If the required libraries are not
# installed on computer, install them from cran repos

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

Data Download

The original data was downloaded from Kaggle website and a copy of the data is stored on Github. In the next code chunk, we will download the data from Github and read it into a data frame object **data_weka**

```
#Download the data file from github

dl <- tempfile()
url_data_github <- "https://raw.githubusercontent.com/SyedAImran/
DYOP-DataScience/master/column_2C_weka.csv"
download.file(url_data_github,dl)
#Read the downloaded csv file
data_weka <- read.csv(dl)
```

Exploratory Data Analysis

Our first task is to have a look at the data and examine its basic structure.

```
#Check if the downloaded file is as expected
glimpse(data_weka)
```

```
## Observations: 310
## Variables: 7
## $ pelvic_incidence         <dbl> 63.02782, 39.05695, 68.83202, 69.2970...
## $ pelvic_tilt.numeric      <dbl> 22.552586, 10.060991, 22.218482, 24.6...
## $ lumbar_lordosis_angle    <dbl> 39.60912, 25.01538, 50.09219, 44.3112...
## $ sacral_slope             <dbl> 40.47523, 28.99596, 46.61354, 44.6441...
## $ pelvic_radius            <dbl> 98.67292, 114.40543, 105.98514, 101.8...
## $ degree_spondylolisthesis <dbl> -0.2544000, 4.5642586, -3.5303173, 11...
## $ class                    <fct> Abnormal, Abnormal, Abnormal, Abnorma...
```

We observe that there are 310 data with 7 columns. The columns are: pelvic_incidence, pelvic_tilt.numeric, lumbar_lordosis_angle, sacral_slope, pelvic_radius, degree_spondylolisthesis, class. The first six columns are numeric and the seventh column **class** is a factor type. Next we examine the summary statistics for the data set.

```
#Check data classes, range and summary statistics
summary(data_weka)
```

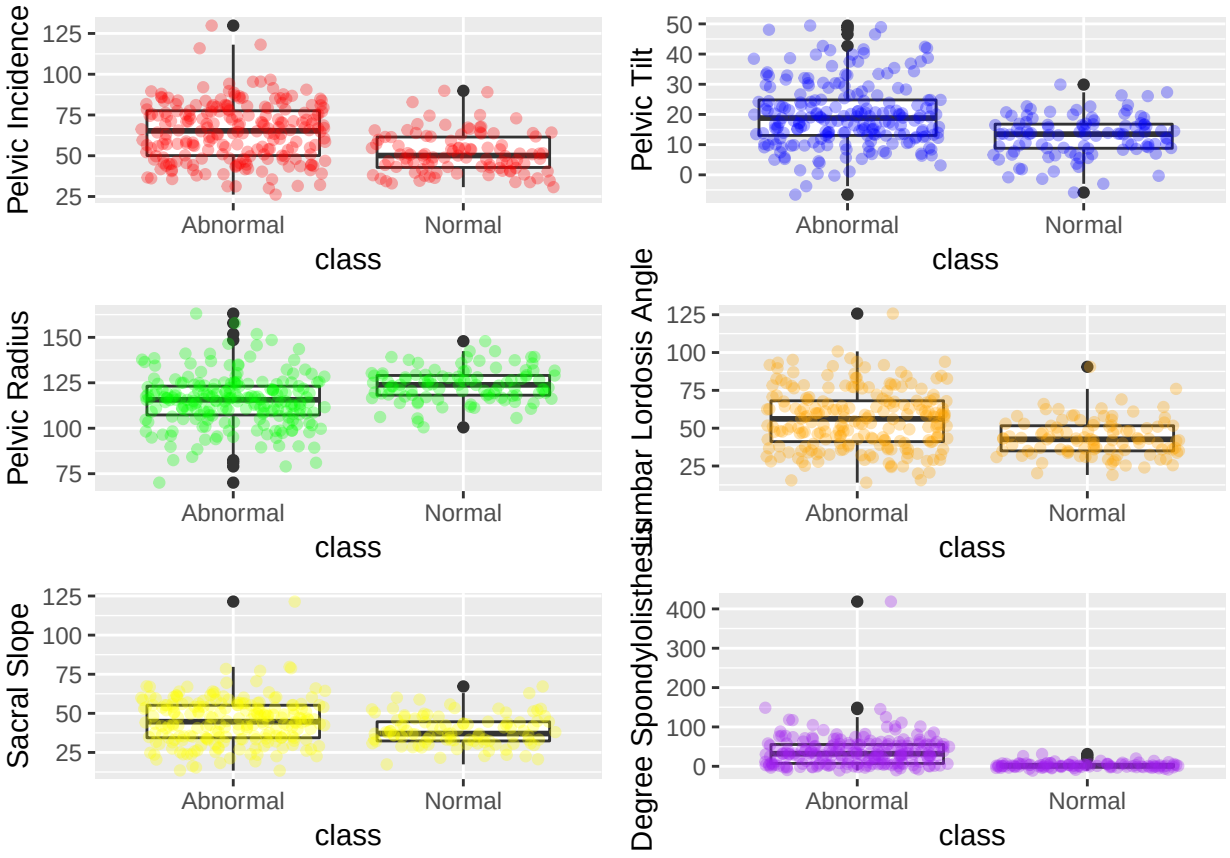
```
##  pelvic_incidence pelvic_tilt.numeric lumbar_lordosis_angle
##  Min.   : 26.15   Min.   :-6.555      Min.   : 14.00
##  1st Qu.: 46.43   1st Qu.:10.667      1st Qu.: 37.00
##  Median : 58.69   Median :16.358      Median : 49.56
##  Mean   : 60.50   Mean   :17.543      Mean   : 51.93
##  3rd Qu.: 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      class
##  Min.   : 13.37   Min.   : 70.08   Min.   :-11.058          Abnormal:210
##  1st Qu.: 33.35   1st Qu.:110.71   1st Qu.:  1.604          Normal  :100
##  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
```

It can be noticed that the data is approximately normal (as mean ~ median) for the first five features. The sixth feature **degree_spondylolisthesis* shows a slight skewness. This can be observed from the chart below:

```
require(gridExtra)
p1 <- data_weka %>% ggplot(aes(y=pelvic_incidence,x = class)) +
  geom_boxplot()+geom_jitter(alpha=0.3, color="red") +ylab("Pelvic Incidence")
p2 <- data_weka %>% ggplot(aes(y=pelvic_tilt.numeric,x = class)) +
  geom_boxplot()+geom_jitter(alpha=0.3, color="blue") + ylab("Pelvic Tilt")
p3 <- data_weka %>% ggplot(aes(y=pelvic_radius,x = class)) +
  geom_boxplot()+geom_jitter(alpha=0.3, color="green") + ylab("Pelvic Radius")
p4 <- data_weka %>% ggplot(aes(y=lumbar_lordosis_angle,x = class)) +
  geom_boxplot()+geom_jitter(alpha=0.3, color="orange") + ylab("Lumbar Lordosis Angle")
p5 <- data_weka %>% ggplot(aes(y=sacral_slope,x = class)) +
  geom_boxplot()+geom_jitter(alpha=0.3, color="yellow") + ylab("Sacral Slope")
p6 <- data_weka %>% ggplot(aes(y=degree_spondylolisthesis,x = class)) +
```

```
  geom_boxplot()+geom_jitter(alpha=0.3, color="purple") +ylab("Degree Spondylolisthesis")

grid.arrange(p1,p2,p3,p4,p5,p6, ncol = 2, widths = c(3, 3))
```



From the chart above, we can observe that there are several data points that can be regarded as outliers. Specifically, the datapoint for *degree_sponylosthesis* > 400. We can either remove this data-point or keep it. But since we are not sure of the medical significance of these extreme value, we will err on side of caution and leave this datapoint in the dataset.

Partitioning of Test And Training Sets

To ensure reproducible results, we use a *set.seed(2019)* before partitioning our data-set into two equal parts. One part is will be used as the training data and the other half will be used as the test data.

```
# Partition Data into two equal Training and Test sets
# set.seed(2019) to get comparable results from random partition

set.seed(2019)
test_index <- createDataPartition(y = data_weka$pelvic_incidence, times = 1,
                                  p = 0.5, list = FALSE)
train_weka <- data_weka[-test_index,]
test_weka <- data_weka[test_index,]
```

Now that we have partitioned our data, we will now used the training set exclusively for training our models.

But first, we would like to see if the exploratory data for the training set. We do so in a similar way as for the dataset above.

```
# Check if the partitioned data is similar (balanced) with respect
# to the overall data
glimpse(train_weka)
```

```
## Observations: 154
## Variables: 7
## $ pelvic_incidence         <dbl> 63.02782, 40.25020, 53.43293, 45.3667...
## $ pelvic_tilt.numeric      <dbl> 22.552586, 13.921907, 15.864336, 10.7...
## $ lumbar_lordosis_angle    <dbl> 39.60912, 25.12495, 37.16593, 29.0383...
## $ sacral_slope             <dbl> 40.47523, 26.32829, 37.56859, 34.6111...
## $ pelvic_radius            <dbl> 98.67292, 130.32787, 120.56752, 117.2...
## $ degree_spondylolisthesis <dbl> -0.2544000, 2.2306517, 5.9885507, -10...
## $ class                    <fct> Abnormal, Abnormal, Abnormal, Abnorma...
```

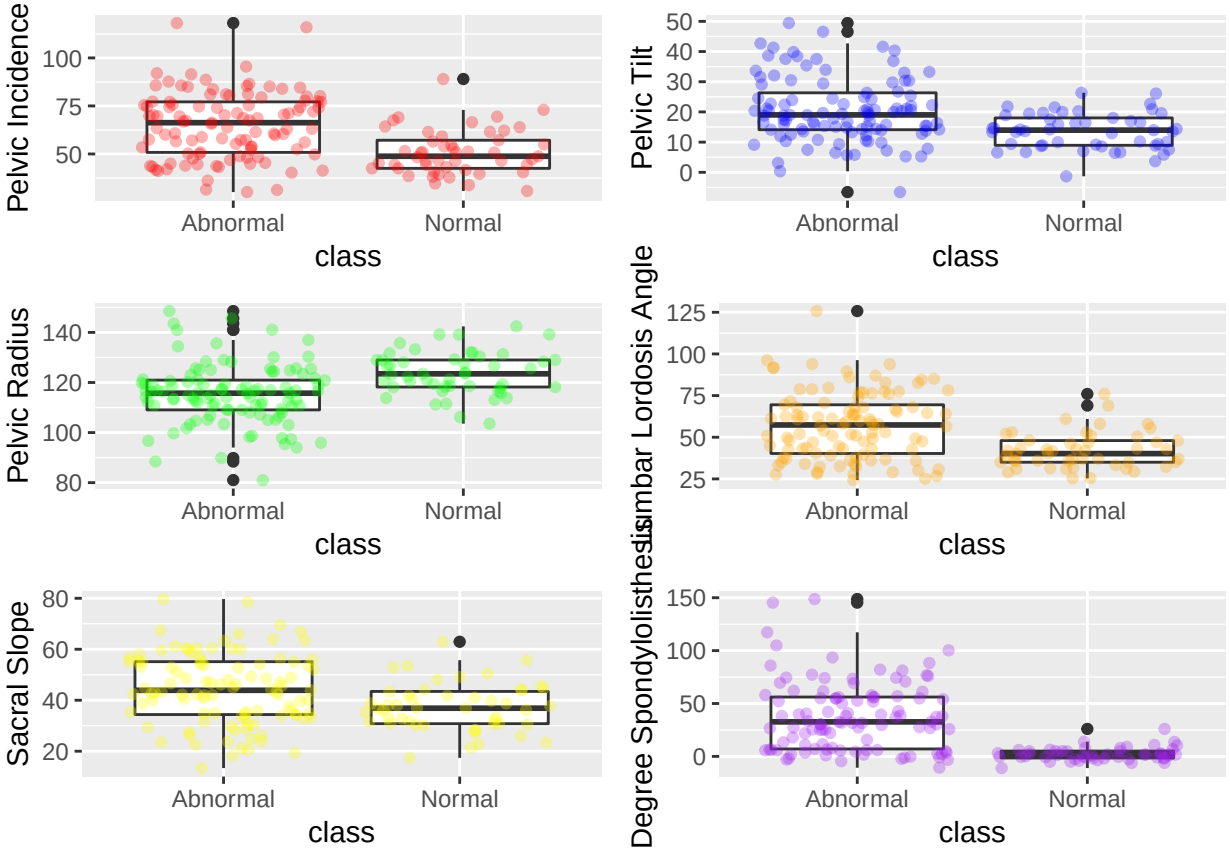
```
summary(train_weka)
```

```
##  pelvic_incidence pelvic_tilt.numeric lumbar_lordosis_angle
##  Min.   : 30.15   Min.   :-6.555      Min.   : 24.28
##  1st Qu.: 46.48   1st Qu.:12.313      1st Qu.: 36.68
##  Median : 58.47   Median :16.705      Median : 48.67
##  Mean   : 60.52   Mean   :18.394      Mean   : 52.29
##  3rd Qu.: 72.88   3rd Qu.:22.382      3rd Qu.: 63.83
##  Max.   :118.14   Max.   :49.432      Max.   :125.74
##   sacral_slope   pelvic_radius    degree_spondylolisthesis      class
##  Min.   :13.37   Min.   : 81.02   Min.   :-11.058          Abnormal:105
##  1st Qu.:32.94   1st Qu.:111.51   1st Qu.:  1.806          Normal  : 49
##  Median :41.80   Median :117.90   Median : 13.486
##  Mean   :42.13   Mean   :117.98   Mean   : 26.772
##  3rd Qu.:52.55   3rd Qu.:125.33   3rd Qu.: 42.700
##  Max.   :79.70   Max.   :148.53   Max.   :148.754
```

```
# Exploratory Data Visualization

require(gridExtra)
p1 <- train_weka %>% ggplot(aes(y=pelvic_incidence,x = class)) +
  geom_boxplot()+geom_jitter(alpha=0.3, color="red") +ylab("Pelvic Incidence")
p2 <- train_weka %>% ggplot(aes(y=pelvic_tilt.numeric,x = class)) +
  geom_boxplot()+geom_jitter(alpha=0.3, color="blue") + ylab("Pelvic Tilt")
p3 <- train_weka %>% ggplot(aes(y=pelvic_radius,x = class)) +
  geom_boxplot()+geom_jitter(alpha=0.3, color="green") + ylab("Pelvic Radius")
p4 <- train_weka %>% ggplot(aes(y=lumbar_lordosis_angle,x = class)) +
  geom_boxplot()+geom_jitter(alpha=0.3, color="orange") + ylab("Lumbar Lordosis Angle")
p5 <- train_weka %>% ggplot(aes(y=sacral_slope,x = class)) +
  geom_boxplot()+geom_jitter(alpha=0.3, color="yellow") + ylab("Sacral Slope")
p6 <- train_weka %>% ggplot(aes(y=degree_spondylolisthesis,x = class)) +
  geom_boxplot()+geom_jitter(alpha=0.3, color="purple") +ylab("Degree Spondylolisthesis")

grid.arrange(p1,p2,p3,p4,p5,p6, ncol = 2, widths = c(3, 3))
```



The training dataset looks good so far, with nothing of particular concern.

Pre-processing Data

```
require(kableExtra)
x <- train_weka[,1:6] %>% as.matrix()
landscape(knitr::kable(cor(x), digits =3, caption = "Correlation Matrix for Features"))
```

Table 2: Correlation Matrix for Features

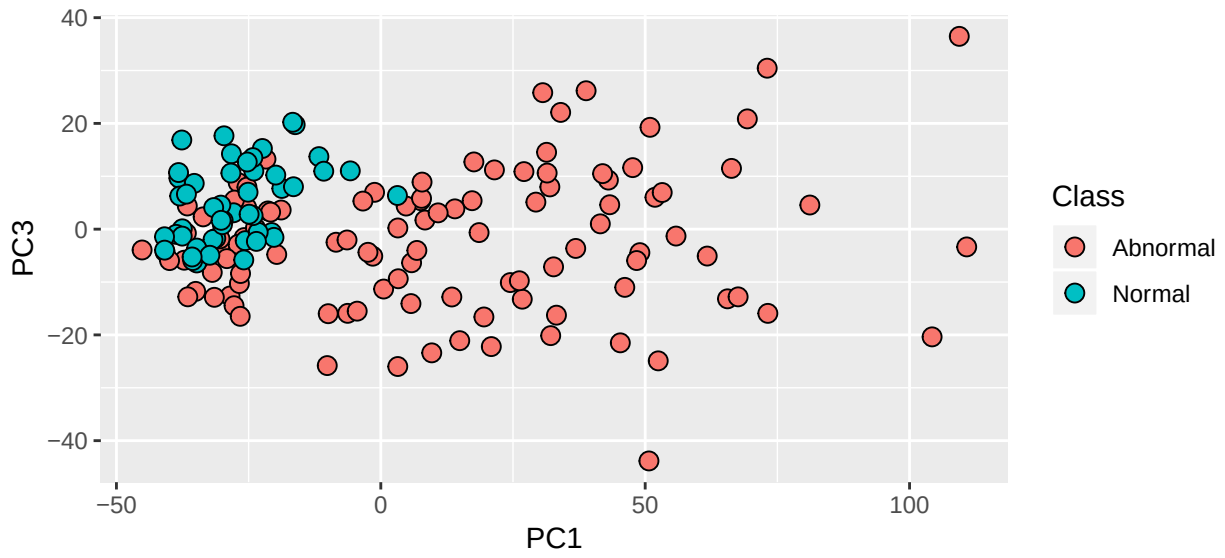
	pelvic_incidence	pelvic_tilt.numeric	lumbar_lordosis_angle	sacral_slope	pelvic_radius	degree_spondylolisthesis
pelvic_incidence	1.000	0.650	0.667	0.811	-0.288	0.607
pelvic_tilt.numeric	0.650	1.000	0.419	0.083	-0.007	0.507
lumbar_lordosis_angle	0.667	0.419	1.000	0.552	-0.005	0.695
sacral_slope	0.811	0.083	0.552	1.000	-0.372	0.405
pelvic_radius	-0.288	-0.007	-0.005	-0.372	1.000	0.002
degree_spondylolisthesis	0.607	0.507	0.695	0.405	0.002	1.000

We observe from the table above that there is a high correlation (> 0.81) between the features *pelvic_incidence* and *sacral_slope*. One of these features can be safely removed from further analysis.

```
x <- train_weka[,2:6] %>% as.matrix()
pca <- prcomp(x)
summary(pca)
```

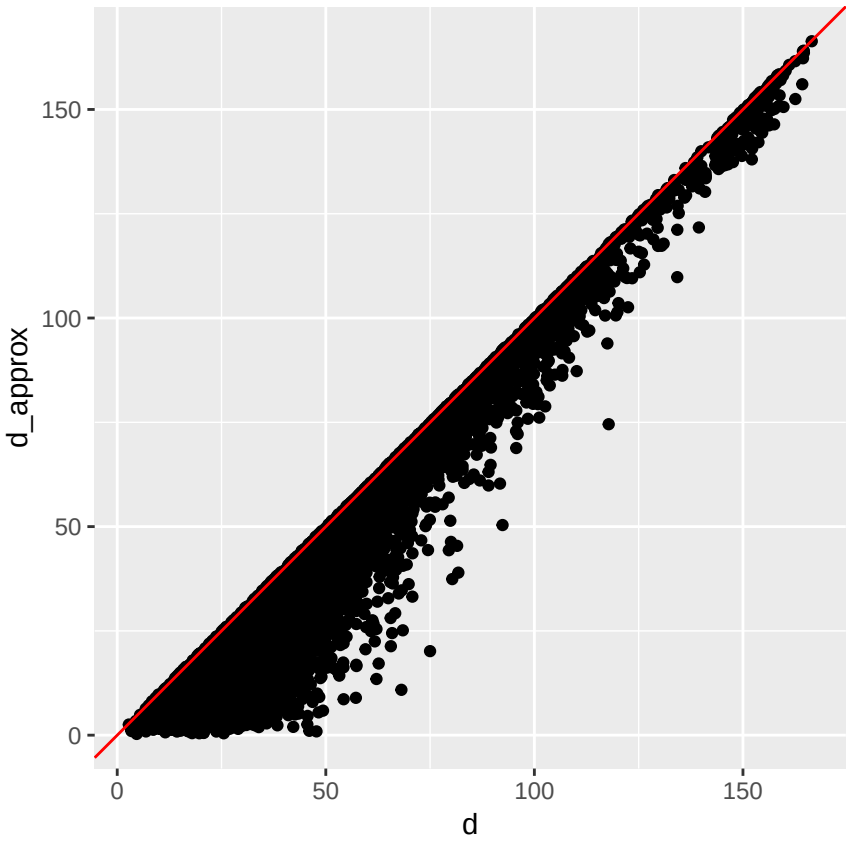
```
## Importance of components:
##                            PC1     PC2      PC3     PC4     PC5
## Standard deviation     35.9112 14.5741 12.05809 9.12570 7.04164
## Proportion of Variance  0.7244  0.1193  0.08167 0.04678 0.02785
## Cumulative Proportion   0.7244  0.8437  0.92537 0.97215 1.00000
```

```
data.frame(pca$x[,1:5], Class=train_weka$class) %>%
  ggplot(aes(PC1, PC3, fill=Class)) +
  geom_point(cex=3, pch=21) +
  coord_fixed(ratio=1)
```




```
d <- dist(x)
d_approx <- dist(pca$x[,c(1,2)])
qplot(d, d_approx) +geom_abline(color="red")+coord_fixed(ratio=1)
```

```
## Don't know how to automatically pick scale for object of type dist. Defaulting to continuous.
## Don't know how to automatically pick scale for object of type dist. Defaulting to continuous.
```



A principle component analysis indicates that 3 principle components are sufficient to predict more than 90% of data variability. Note: we have previously eliminated *pelvic_incidence* from our subsequent analysis

Evaluating Different Classification Models

Our next step is to run different machine learning models with their default settings to evaluate which model would be appropriate for further use. The models that are selected are *glm*, *lda*, *naive_bayes*, *svmLinear*, *gamboost*, *gamLoess*, *knn*, *kknn*, *loclda*, *gam*, *rf*, *ranger*, *wsrf*, *Rborist*, *mlp*, *adaboost*, *svmRadial*, *svmRadialCost*, *svmRadialSigma*. All models were run with the caret **train** function. Each model was evaluated on both the training dataset and the testing dataset using the **predict** function. The performance metrics of each model were obtained from the appropriate subsetting of the **confusionMatrix** function. The performance metrics were stored in a *tibble*.

```
# Select a number of models that will be evaluated

models <- c("glm", "lda", "naive_bayes", "svmLinear",
            "gamboost", "gamLoess",
            "knn", "kknn", "loclda", "gam",
            "rf", "ranger", "wsrf", "Rborist", "mlp", "adaboost",
            "svmRadial", "svmRadialCost",
            "svmRadialSigma")

# models <- c("glm", "lda")

# Loop over models and collect the accuracy, sensitivity and specificity metrics
```

```r
# for both the training set and testing set.
# This loop takes a few minutes/hours to run. It also asks if some packages need
# to be installed.

accuracy <- map_df(models, function(model) {

  print(paste("Training model ---- ", model))

  # Train the model selected in the loop with default parameters
  train_weka_model <- train(class~., method=model, data = train_weka[,2:7])

  # Check accuracy of selected model on training set and save the metrics
  y_hat_train <- predict(train_weka_model, train_weka, type="raw")
  cm_train <- confusionMatrix(y_hat_train, train_weka$class)
  Accuracy_train <- cm_train$overall["Accuracy"]
  Sensitivity_train <- cm_train$byClass[1]
  Specificity_train <- cm_train$byClass[2]

  # Check accuracy of selected model on test set and save the metrics
  y_hat_test <- predict(train_weka_model, test_weka, type="raw")
  cm_test <- confusionMatrix(y_hat_test, test_weka$class)
  Accuracy_test <- cm_test$overall["Accuracy"]
  Sensitivity_test <- cm_test$byClass[1]
  Specificity_test <- cm_test$byClass[2]

  tibble(Model= model, Train_acc = Accuracy_train, Train_sens = Sensitivity_train,
         Train_Spec=Specificity_train, Test_acc=Accuracy_test,
         Test_sens = Sensitivity_test, Test_spec = Specificity_test)
})
```

```
## [1] "Training model ----  glm"
## [1] "Training model ----  lda"
## [1] "Training model ----  naive_bayes"
## [1] "Training model ----  svmLinear"
## [1] "Training model ----  gamboost"
## [1] "Training model ----  gamLoess"
## [1] "Training model ----  knn"
## [1] "Training model ----  kknn"
## [1] "Training model ----  loclda"
## [1] "Training model ----  gam"
## [1] "Training model ----  rf"
## [1] "Training model ----  ranger"
## [1] "Training model ----  wsrf"
## [1] "Training model ----  Rborist"
## [1] "Training model ----  mlp"
## [1] "Training model ----  adaboost"
## [1] "Training model ----  svmRadial"
## [1] "Training model ----  svmRadialCost"
## [1] "Training model ----  svmRadialSigma"
```

Table 3: Summary of Results from Machine Learning Models

Model	Train_acc	Train_sens	Train_Spec	Test_acc	Test_sens	Test_spec
glm	0.8701	0.9048	0.7959	0.8269	0.8190	0.8431
lda	0.8571	0.8952	0.7755	0.7949	0.7905	0.8039
naive_bayes	0.7792	0.7238	0.8980	0.7756	0.7524	0.8235
svmLinear	0.8636	0.8857	0.8163	0.8462	0.8381	0.8627
gamboost	0.8636	0.9048	0.7755	0.8077	0.8190	0.7843
gamLoess	0.8896	0.9143	0.8367	0.8462	0.8476	0.8431
knn	0.8571	0.8857	0.7959	0.8526	0.8857	0.7843
kknn	0.8896	0.9143	0.8367	0.8397	0.8762	0.7647
loclda	0.8831	0.9333	0.7755	0.8462	0.8762	0.7843
gam	0.8636	0.8952	0.7959	0.8590	0.8667	0.8431
rf	1.0000	1.0000	1.0000	0.8654	0.8857	0.8235
ranger	1.0000	1.0000	1.0000	0.8718	0.8667	0.8824
wsrf	0.9870	1.0000	0.9592	0.8526	0.8571	0.8431
Rborist	1.0000	1.0000	1.0000	0.8718	0.8762	0.8627
mlp	0.6818	1.0000	0.0000	0.6731	1.0000	0.0000
adaboost	1.0000	1.0000	1.0000	0.8654	0.8857	0.8235
svmRadial	0.8636	0.9238	0.7347	0.8397	0.8857	0.7451
svmRadialCost	0.8636	0.9238	0.7347	0.8462	0.8762	0.7843
svmRadialSigma	0.8571	0.9143	0.7347	0.8269	0.8476	0.7843

Results

```
# Display in a friendly table

knitr::kable(accuracy, digits = 4,
             caption = "Summary of Results from Machine Learning Models")
```

From the table above, it can be observed that most of the machine learning models had a similar performance. It can be noticed that some of the models (*rf*, *ranger*, *Rborist*, *adaboost*) have very high accuracy, sensitivity and specificity (almost equal to 1) for the training sets. This indicates that these models are suffering from oover-fitting of the training set. A look at the performance of these models on the test set confirms our suspicion as the accuracy declines to comparable levels with the other models.

Ensemble Model

The highest accuracy and sensitivity come from the *Rborist* model. However, it is the most computationally extensive model in the set and takes several minutes to run. It would be nearly impossible to use for a larger dataset. Therefore, the next step is to build an ensemble model with faster models to see if it can give us an accuracy similar to the more computationally intensive model *RBorist*.

```
# Stacking Algorithms - Run multiple algos in one call.
trainControl <- trainControl(method="repeatedcv",
                             number=10,
                             repeats=3,
                             savePredictions="final",
                             classProbs=TRUE)
```

```r
algorithmList <- c("lda", "knn", "svmRadial", "ranger")

set.seed(2019)

train_models_list <- caretList(class ~ ., data=train_weka[,2:7],
                               trControl=trainControl, methodList=algorithmList)
```

```
## Warning in trControlCheck(x = trControl, y = target): indexes not defined
## in trControl. Attempting to set them ourselves, so each model in the
## ensemble will have the same resampling indexes.
```

```r
results <- resamples(train_models_list)

summary(results)
```

```
##
## Call:
## summary.resamples(object = results)
##
## Models: lda, knn, svmRadial, ranger
## Number of resamples: 30
##
## Accuracy
##             Min.   1st Qu.    Median      Mean   3rd Qu.   Max. NA's
## lda       0.6875 0.7625000 0.8666667 0.8561706 0.9333333 1.0000    0
## knn       0.6250 0.7500000 0.8125000 0.8206746 0.8750000 0.9375    0
## svmRadial 0.6875 0.8000000 0.8619048 0.8450198 0.9333333 1.0000    0
## ranger    0.5625 0.7589286 0.8348214 0.8153175 0.8666667 1.0000    0
##
## Kappa
##                  Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## lda         0.23076923 0.5065789 0.6734694 0.6700155 0.8543956 1.000000
## knn         0.05405405 0.4181818 0.5323887 0.5851874 0.7377049 0.862069
## svmRadial   0.13043478 0.4902746 0.6274510 0.6227213 0.8451417 1.000000
## ranger     -0.07692308 0.4431768 0.6293478 0.5706271 0.7000000 1.000000
##            NA's
## lda           0
## knn           0
## svmRadial     0
## ranger        0
```

```r
set.seed(101)

ensembleControl <- trainControl(method="repeatedcv",
                                number=10,
                                repeats=3,
                                savePredictions=TRUE,
                                classProbs=TRUE)

# Ensemble the predictions of `models` to form a new combined prediction based on glm

ensemble.glm <- caretStack(train_models_list, method="glm", metric="Accuracy",
```

```
                          trControl=ensembleControl)

print(ensemble.glm)
```

```
## A glm ensemble of 2 base models: lda, knn, svmRadial, ranger
##
## Ensemble results:
## Generalized Linear Model
##
## 462 samples
##   4 predictor
##   2 classes: 'Abnormal', 'Normal'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 416, 416, 415, 416, 416, 417, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.8429869  0.6310639
```

```
# Predict on testData

ensemble_predicteds <- predict(ensemble.glm, newdata=test_weka[,2:7])

#head(ensemble_predicteds)

cm_ensemble <- confusionMatrix(ensemble_predicteds,test_weka$class)
cm_ensemble$table
```

```
##           Reference
## Prediction Abnormal Normal
##   Abnormal       89     11
##   Normal         16     40
```

```
cm_ensemble$overall["Accuracy"]
```

```
##  Accuracy
## 0.8269231
```

Unexpectedly, our ensemble model which was a combination of *lda*, *knn*, *svmRadial*, and *ranger* models did not do any better than the *Rborist* model. Therefore, we will select the *Rborist* model for our classification/diagnosis of postural abnormalities.

Conclusions

```
  # Train the model selected in the loop with default parameters
  train_weka_model <- train(class~., method="Rborist", data = train_weka[,2:7])
```

Table 4: Confusion Matrix for Selected Method (Rborist)

	Abnormal	Normal
Abnormal	93	8
Normal	12	43

```
# Check accuracy of selected model on training set and save the metrics
y_hat_train <- predict(train_weka_model, train_weka, type="raw")
cm_train <- confusionMatrix(y_hat_train, train_weka$class)
Accuracy_train <- cm_train$overall["Accuracy"]
Sensitivity_train <- cm_train$byClass[1]
Specificity_train <- cm_train$byClass[2]

# Check accuracy of selected model on test set and save the metrics
y_hat_test <- predict(train_weka_model, test_weka, type="raw")
cm_test <- confusionMatrix(y_hat_test, test_weka$class)
Accuracy_test <- cm_test$overall["Accuracy"]
Sensitivity_test <- cm_test$byClass[1]
Specificity_test <- cm_test$byClass[2]
```

The model selected for our classification/diagnosis of postural abnormalities based on pelvic measurements is **Rborist**. This model had an overall accuracy of 1 in the training set. The overall accuracy in the test set was 0.8717949. The confusion matrix for the selected model is shown in table above.