Course8Project

John David

8/23/2019

## Weight Lifting - Machine Learning Project

Using devices such as Jawbone Up, Nike FuelBand, and Fitbit it is now possible to collect a large amount of data about personal activity relatively inexpensively. These type of devices are part of the quantified self movement – a group of enthusiasts who take measurements about themselves regularly to improve their health, to find patterns in their behavior, or because they are tech geeks. One thing that people regularly do is quantify how much of a particular activity they do, but they rarely quantify how well they do it. In this project, your goal will be to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants. They were asked to perform barbell lifts correctly and incorrectly in 5 different ways. More information is available from the website here: <http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har> (see the section on the Weight Lifting Exercise Dataset).

Approach: My approach was to explore the data set and analyze what actions best describe how to correctly curl weight. So I explored cleaned and explored the data and used that subset of data to form my models. I did a K fold cross validation, K=20 and I built an qda model and a lda model and had well ok results then I used a random forest model and tuned it to achieve a 98.65% accuracy on my test data.

Conclusion: It was fun to analyze the data myself but I could have used the randomforest to do it. All in all I enjoyed this lab and used all my previous John Hopkins classes to find this success.

References: <http://www-personal.umich.edu/~johannb/Papers/paper63.pdf> <https://www.google.com/search?q=pitch+roll+and+yaw&tbm=isch&source=iu&ictx=1&fir=4R5jctF0uP_q5M%253A%252CrwdN0Ut4Lf6FUM%252C%252Fm%252F04gmp6w&vet=1&usg=AI4_-kR4vLK3GkH5rhntURhlMELz4kTd0g&sa=X&ved=2ahUKEwiF-4jEhZnkAhVldt8KHU2fB3UQ_B0wG3oECAEQAw#imgrc=4R5jctF0uP_q5M>:

library("dplyr")

## Warning: package 'dplyr' was built under R version 3.5.2

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

library("ggplot2")  
library("mlbench")  
library("caret")

## Warning: package 'caret' was built under R version 3.5.2

## Loading required package: lattice

#library("MASS")  
  
apply.type.rules <- function(tmpDF){  
 if (tmpDF["classe"] == "A"){  
 return (1)  
 }else if (tmpDF["classe"] == "B"){  
 return (2)  
 }else if (tmpDF["classe"] == "C"){  
 return (3)  
 }else if (tmpDF["classe"] == "D"){  
 return (4)  
 }else if (tmpDF["classe"] == "E"){  
 return (5)  
 }else if (tmpDF["classe"] == "F"){  
 return (6)  
 }else{  
 return (100)  
 }  
   
}  
  
  
  
  
trainingDF <- read.csv(file="/Course\_8\_Machine\_Learning/Project/pml-training.csv", stringsAsFactors = FALSE)  
  
#summary(trainingDF)  
#str(trainingDF)  
names(trainingDF)

## [1] "X" "user\_name"   
## [3] "raw\_timestamp\_part\_1" "raw\_timestamp\_part\_2"   
## [5] "cvtd\_timestamp" "new\_window"   
## [7] "num\_window" "roll\_belt"   
## [9] "pitch\_belt" "yaw\_belt"   
## [11] "total\_accel\_belt" "kurtosis\_roll\_belt"   
## [13] "kurtosis\_picth\_belt" "kurtosis\_yaw\_belt"   
## [15] "skewness\_roll\_belt" "skewness\_roll\_belt.1"   
## [17] "skewness\_yaw\_belt" "max\_roll\_belt"   
## [19] "max\_picth\_belt" "max\_yaw\_belt"   
## [21] "min\_roll\_belt" "min\_pitch\_belt"   
## [23] "min\_yaw\_belt" "amplitude\_roll\_belt"   
## [25] "amplitude\_pitch\_belt" "amplitude\_yaw\_belt"   
## [27] "var\_total\_accel\_belt" "avg\_roll\_belt"   
## [29] "stddev\_roll\_belt" "var\_roll\_belt"   
## [31] "avg\_pitch\_belt" "stddev\_pitch\_belt"   
## [33] "var\_pitch\_belt" "avg\_yaw\_belt"   
## [35] "stddev\_yaw\_belt" "var\_yaw\_belt"   
## [37] "gyros\_belt\_x" "gyros\_belt\_y"   
## [39] "gyros\_belt\_z" "accel\_belt\_x"   
## [41] "accel\_belt\_y" "accel\_belt\_z"   
## [43] "magnet\_belt\_x" "magnet\_belt\_y"   
## [45] "magnet\_belt\_z" "roll\_arm"   
## [47] "pitch\_arm" "yaw\_arm"   
## [49] "total\_accel\_arm" "var\_accel\_arm"   
## [51] "avg\_roll\_arm" "stddev\_roll\_arm"   
## [53] "var\_roll\_arm" "avg\_pitch\_arm"   
## [55] "stddev\_pitch\_arm" "var\_pitch\_arm"   
## [57] "avg\_yaw\_arm" "stddev\_yaw\_arm"   
## [59] "var\_yaw\_arm" "gyros\_arm\_x"   
## [61] "gyros\_arm\_y" "gyros\_arm\_z"   
## [63] "accel\_arm\_x" "accel\_arm\_y"   
## [65] "accel\_arm\_z" "magnet\_arm\_x"   
## [67] "magnet\_arm\_y" "magnet\_arm\_z"   
## [69] "kurtosis\_roll\_arm" "kurtosis\_picth\_arm"   
## [71] "kurtosis\_yaw\_arm" "skewness\_roll\_arm"   
## [73] "skewness\_pitch\_arm" "skewness\_yaw\_arm"   
## [75] "max\_roll\_arm" "max\_picth\_arm"   
## [77] "max\_yaw\_arm" "min\_roll\_arm"   
## [79] "min\_pitch\_arm" "min\_yaw\_arm"   
## [81] "amplitude\_roll\_arm" "amplitude\_pitch\_arm"   
## [83] "amplitude\_yaw\_arm" "roll\_dumbbell"   
## [85] "pitch\_dumbbell" "yaw\_dumbbell"   
## [87] "kurtosis\_roll\_dumbbell" "kurtosis\_picth\_dumbbell"   
## [89] "kurtosis\_yaw\_dumbbell" "skewness\_roll\_dumbbell"   
## [91] "skewness\_pitch\_dumbbell" "skewness\_yaw\_dumbbell"   
## [93] "max\_roll\_dumbbell" "max\_picth\_dumbbell"   
## [95] "max\_yaw\_dumbbell" "min\_roll\_dumbbell"   
## [97] "min\_pitch\_dumbbell" "min\_yaw\_dumbbell"   
## [99] "amplitude\_roll\_dumbbell" "amplitude\_pitch\_dumbbell"  
## [101] "amplitude\_yaw\_dumbbell" "total\_accel\_dumbbell"   
## [103] "var\_accel\_dumbbell" "avg\_roll\_dumbbell"   
## [105] "stddev\_roll\_dumbbell" "var\_roll\_dumbbell"   
## [107] "avg\_pitch\_dumbbell" "stddev\_pitch\_dumbbell"   
## [109] "var\_pitch\_dumbbell" "avg\_yaw\_dumbbell"   
## [111] "stddev\_yaw\_dumbbell" "var\_yaw\_dumbbell"   
## [113] "gyros\_dumbbell\_x" "gyros\_dumbbell\_y"   
## [115] "gyros\_dumbbell\_z" "accel\_dumbbell\_x"   
## [117] "accel\_dumbbell\_y" "accel\_dumbbell\_z"   
## [119] "magnet\_dumbbell\_x" "magnet\_dumbbell\_y"   
## [121] "magnet\_dumbbell\_z" "roll\_forearm"   
## [123] "pitch\_forearm" "yaw\_forearm"   
## [125] "kurtosis\_roll\_forearm" "kurtosis\_picth\_forearm"   
## [127] "kurtosis\_yaw\_forearm" "skewness\_roll\_forearm"   
## [129] "skewness\_pitch\_forearm" "skewness\_yaw\_forearm"   
## [131] "max\_roll\_forearm" "max\_picth\_forearm"   
## [133] "max\_yaw\_forearm" "min\_roll\_forearm"   
## [135] "min\_pitch\_forearm" "min\_yaw\_forearm"   
## [137] "amplitude\_roll\_forearm" "amplitude\_pitch\_forearm"   
## [139] "amplitude\_yaw\_forearm" "total\_accel\_forearm"   
## [141] "var\_accel\_forearm" "avg\_roll\_forearm"   
## [143] "stddev\_roll\_forearm" "var\_roll\_forearm"   
## [145] "avg\_pitch\_forearm" "stddev\_pitch\_forearm"   
## [147] "var\_pitch\_forearm" "avg\_yaw\_forearm"   
## [149] "stddev\_yaw\_forearm" "var\_yaw\_forearm"   
## [151] "gyros\_forearm\_x" "gyros\_forearm\_y"   
## [153] "gyros\_forearm\_z" "accel\_forearm\_x"   
## [155] "accel\_forearm\_y" "accel\_forearm\_z"   
## [157] "magnet\_forearm\_x" "magnet\_forearm\_y"   
## [159] "magnet\_forearm\_z" "classe"

str(trainingDF)

## 'data.frame': 19622 obs. of 160 variables:  
## $ X : int 1 2 3 4 5 6 7 8 9 10 ...  
## $ user\_name : chr "carlitos" "carlitos" "carlitos" "carlitos" ...  
## $ raw\_timestamp\_part\_1 : int 1323084231 1323084231 1323084231 1323084232 1323084232 1323084232 1323084232 1323084232 1323084232 1323084232 ...  
## $ raw\_timestamp\_part\_2 : int 788290 808298 820366 120339 196328 304277 368296 440390 484323 484434 ...  
## $ cvtd\_timestamp : chr "05/12/2011 11:23" "05/12/2011 11:23" "05/12/2011 11:23" "05/12/2011 11:23" ...  
## $ new\_window : chr "no" "no" "no" "no" ...  
## $ num\_window : int 11 11 11 12 12 12 12 12 12 12 ...  
## $ roll\_belt : num 1.41 1.41 1.42 1.48 1.48 1.45 1.42 1.42 1.43 1.45 ...  
## $ pitch\_belt : num 8.07 8.07 8.07 8.05 8.07 8.06 8.09 8.13 8.16 8.17 ...  
## $ yaw\_belt : num -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 ...  
## $ total\_accel\_belt : int 3 3 3 3 3 3 3 3 3 3 ...  
## $ kurtosis\_roll\_belt : chr "" "" "" "" ...  
## $ kurtosis\_picth\_belt : chr "" "" "" "" ...  
## $ kurtosis\_yaw\_belt : chr "" "" "" "" ...  
## $ skewness\_roll\_belt : chr "" "" "" "" ...  
## $ skewness\_roll\_belt.1 : chr "" "" "" "" ...  
## $ skewness\_yaw\_belt : chr "" "" "" "" ...  
## $ max\_roll\_belt : num NA NA NA NA NA NA NA NA NA NA ...  
## $ max\_picth\_belt : int NA NA NA NA NA NA NA NA NA NA ...  
## $ max\_yaw\_belt : chr "" "" "" "" ...  
## $ min\_roll\_belt : num NA NA NA NA NA NA NA NA NA NA ...  
## $ min\_pitch\_belt : int NA NA NA NA NA NA NA NA NA NA ...  
## $ min\_yaw\_belt : chr "" "" "" "" ...  
## $ amplitude\_roll\_belt : num NA NA NA NA NA NA NA NA NA NA ...  
## $ amplitude\_pitch\_belt : int NA NA NA NA NA NA NA NA NA NA ...  
## $ amplitude\_yaw\_belt : chr "" "" "" "" ...  
## $ var\_total\_accel\_belt : num NA NA NA NA NA NA NA NA NA NA ...  
## $ avg\_roll\_belt : num NA NA NA NA NA NA NA NA NA NA ...  
## $ stddev\_roll\_belt : num NA NA NA NA NA NA NA NA NA NA ...  
## $ var\_roll\_belt : num NA NA NA NA NA NA NA NA NA NA ...  
## $ avg\_pitch\_belt : num NA NA NA NA NA NA NA NA NA NA ...  
## $ stddev\_pitch\_belt : num NA NA NA NA NA NA NA NA NA NA ...  
## $ var\_pitch\_belt : num NA NA NA NA NA NA NA NA NA NA ...  
## $ avg\_yaw\_belt : num NA NA NA NA NA NA NA NA NA NA ...  
## $ stddev\_yaw\_belt : num NA NA NA NA NA NA NA NA NA NA ...  
## $ var\_yaw\_belt : num NA NA NA NA NA NA NA NA NA NA ...  
## $ gyros\_belt\_x : num 0 0.02 0 0.02 0.02 0.02 0.02 0.02 0.02 0.03 ...  
## $ gyros\_belt\_y : num 0 0 0 0 0.02 0 0 0 0 0 ...  
## $ gyros\_belt\_z : num -0.02 -0.02 -0.02 -0.03 -0.02 -0.02 -0.02 -0.02 -0.02 0 ...  
## $ accel\_belt\_x : int -21 -22 -20 -22 -21 -21 -22 -22 -20 -21 ...  
## $ accel\_belt\_y : int 4 4 5 3 2 4 3 4 2 4 ...  
## $ accel\_belt\_z : int 22 22 23 21 24 21 21 21 24 22 ...  
## $ magnet\_belt\_x : int -3 -7 -2 -6 -6 0 -4 -2 1 -3 ...  
## $ magnet\_belt\_y : int 599 608 600 604 600 603 599 603 602 609 ...  
## $ magnet\_belt\_z : int -313 -311 -305 -310 -302 -312 -311 -313 -312 -308 ...  
## $ roll\_arm : num -128 -128 -128 -128 -128 -128 -128 -128 -128 -128 ...  
## $ pitch\_arm : num 22.5 22.5 22.5 22.1 22.1 22 21.9 21.8 21.7 21.6 ...  
## $ yaw\_arm : num -161 -161 -161 -161 -161 -161 -161 -161 -161 -161 ...  
## $ total\_accel\_arm : int 34 34 34 34 34 34 34 34 34 34 ...  
## $ var\_accel\_arm : num NA NA NA NA NA NA NA NA NA NA ...  
## $ avg\_roll\_arm : num NA NA NA NA NA NA NA NA NA NA ...  
## $ stddev\_roll\_arm : num NA NA NA NA NA NA NA NA NA NA ...  
## $ var\_roll\_arm : num NA NA NA NA NA NA NA NA NA NA ...  
## $ avg\_pitch\_arm : num NA NA NA NA NA NA NA NA NA NA ...  
## $ stddev\_pitch\_arm : num NA NA NA NA NA NA NA NA NA NA ...  
## $ var\_pitch\_arm : num NA NA NA NA NA NA NA NA NA NA ...  
## $ avg\_yaw\_arm : num NA NA NA NA NA NA NA NA NA NA ...  
## $ stddev\_yaw\_arm : num NA NA NA NA NA NA NA NA NA NA ...  
## $ var\_yaw\_arm : num NA NA NA NA NA NA NA NA NA NA ...  
## $ gyros\_arm\_x : num 0 0.02 0.02 0.02 0 0.02 0 0.02 0.02 0.02 ...  
## $ gyros\_arm\_y : num 0 -0.02 -0.02 -0.03 -0.03 -0.03 -0.03 -0.02 -0.03 -0.03 ...  
## $ gyros\_arm\_z : num -0.02 -0.02 -0.02 0.02 0 0 0 0 -0.02 -0.02 ...  
## $ accel\_arm\_x : int -288 -290 -289 -289 -289 -289 -289 -289 -288 -288 ...  
## $ accel\_arm\_y : int 109 110 110 111 111 111 111 111 109 110 ...  
## $ accel\_arm\_z : int -123 -125 -126 -123 -123 -122 -125 -124 -122 -124 ...  
## $ magnet\_arm\_x : int -368 -369 -368 -372 -374 -369 -373 -372 -369 -376 ...  
## $ magnet\_arm\_y : int 337 337 344 344 337 342 336 338 341 334 ...  
## $ magnet\_arm\_z : int 516 513 513 512 506 513 509 510 518 516 ...  
## $ kurtosis\_roll\_arm : chr "" "" "" "" ...  
## $ kurtosis\_picth\_arm : chr "" "" "" "" ...  
## $ kurtosis\_yaw\_arm : chr "" "" "" "" ...  
## $ skewness\_roll\_arm : chr "" "" "" "" ...  
## $ skewness\_pitch\_arm : chr "" "" "" "" ...  
## $ skewness\_yaw\_arm : chr "" "" "" "" ...  
## $ max\_roll\_arm : num NA NA NA NA NA NA NA NA NA NA ...  
## $ max\_picth\_arm : num NA NA NA NA NA NA NA NA NA NA ...  
## $ max\_yaw\_arm : int NA NA NA NA NA NA NA NA NA NA ...  
## $ min\_roll\_arm : num NA NA NA NA NA NA NA NA NA NA ...  
## $ min\_pitch\_arm : num NA NA NA NA NA NA NA NA NA NA ...  
## $ min\_yaw\_arm : int NA NA NA NA NA NA NA NA NA NA ...  
## $ amplitude\_roll\_arm : num NA NA NA NA NA NA NA NA NA NA ...  
## $ amplitude\_pitch\_arm : num NA NA NA NA NA NA NA NA NA NA ...  
## $ amplitude\_yaw\_arm : int NA NA NA NA NA NA NA NA NA NA ...  
## $ roll\_dumbbell : num 13.1 13.1 12.9 13.4 13.4 ...  
## $ pitch\_dumbbell : num -70.5 -70.6 -70.3 -70.4 -70.4 ...  
## $ yaw\_dumbbell : num -84.9 -84.7 -85.1 -84.9 -84.9 ...  
## $ kurtosis\_roll\_dumbbell : chr "" "" "" "" ...  
## $ kurtosis\_picth\_dumbbell : chr "" "" "" "" ...  
## $ kurtosis\_yaw\_dumbbell : chr "" "" "" "" ...  
## $ skewness\_roll\_dumbbell : chr "" "" "" "" ...  
## $ skewness\_pitch\_dumbbell : chr "" "" "" "" ...  
## $ skewness\_yaw\_dumbbell : chr "" "" "" "" ...  
## $ max\_roll\_dumbbell : num NA NA NA NA NA NA NA NA NA NA ...  
## $ max\_picth\_dumbbell : num NA NA NA NA NA NA NA NA NA NA ...  
## $ max\_yaw\_dumbbell : chr "" "" "" "" ...  
## $ min\_roll\_dumbbell : num NA NA NA NA NA NA NA NA NA NA ...  
## $ min\_pitch\_dumbbell : num NA NA NA NA NA NA NA NA NA NA ...  
## $ min\_yaw\_dumbbell : chr "" "" "" "" ...  
## $ amplitude\_roll\_dumbbell : num NA NA NA NA NA NA NA NA NA NA ...  
## [list output truncated]

classeADF <- trainingDF[trainingDF['classe'] == "A",]  
classeBDF <- trainingDF[trainingDF['classe'] == "B",]  
classeCDF <- trainingDF[trainingDF['classe'] == "C",]  
classeDDF <- trainingDF[trainingDF['classe'] == "D",]  
classeEDF <- trainingDF[trainingDF['classe'] == "E",]  
classeFDF <- trainingDF[trainingDF['classe'] == "F",]  
  
#classADumbbellDF <- select(classeADF,roll\_dumbbell,pitch\_dumbbell,yaw\_dumbbell)  
#colnames(classADumbbellDF) <- c("Roll.Dumbbell","Pitch.Dumbbell","Yaw.Dumbbell")  
  
#head(classADumbbellDF)  
#dim(classADumbbellDF)  
  
#n = 3510:(3510+470)  
  
#classeAForearmDF <- select(classeADF,roll\_forearm, pitch\_forearm, yaw\_forearm)  
#colnames(classeAForearmDF) <- c("Roll.Forearm","Pitch.Forearm","Yaw.Forearm")  
#plot(classeAForearmDF$pitch\_forearm[n])  
  
#minPV <- apply(classeAForearmDF,2,min)  
#abs(minPV)  
#tempDF2 <- mutate(classeAForearmDF, log2Pitch = log2(pitch\_forearm+abs(minPV)))  
  
#head(tempDF2,10)  
#plot(tempDF2$log2Pitch[n])  
#useless  
#classAArmDF <- select(classeADF, roll\_arm, pitch\_arm, yaw\_arm)  
#plot(classAArmDF$pitch\_arm)  
  
#classeABeltDF <- select(classeADF,roll\_belt,pitch\_belt,yaw\_belt)  
#plot(classeABeltDF$pitch\_belt[n])  
  
#useless  
#classeAGyroForearmDF <- select(classeADF, gyros\_forearm\_x, gyros\_forearm\_y, gyros\_forearm\_z)  
#plot(classeAGyroForearmDF$gyros\_forearm\_z)  
  
#classeAGyroArmDF <- select(classeADF, gyros\_arm\_x, gyros\_arm\_y, gyros\_arm\_z)  
#plot(classeAGyroArmDF$gyros\_arm\_z)  
  
#classeAGyroBeltDF <- select(classeADF, gyros\_belt\_x, gyros\_belt\_y, gyros\_belt\_z)  
#plot(classeAGyroBeltDF$gyros\_belt\_z)  
  
  
  
#correlMtrx <- cor(trainingDF[,8:159])  
#highlyCorrel <- findCorrelation(correlMtrx, cutoff=0.5)  
#print(highlyCorrel)  
  
  
  
selectedInputDF = select(trainingDF, roll\_forearm, pitch\_forearm, yaw\_forearm, roll\_arm, pitch\_arm, yaw\_arm, roll\_belt, pitch\_belt, yaw\_belt, classe)  
  
typeAry <- c()  
for (k in 1:nrow(selectedInputDF)){  
 typeAry <- c(typeAry,"NA")  
}  
  
#selectedInputDF$Type=typeAry  
  
head(selectedInputDF,10)

## roll\_forearm pitch\_forearm yaw\_forearm roll\_arm pitch\_arm yaw\_arm  
## 1 28.4 -63.9 -153 -128 22.5 -161  
## 2 28.3 -63.9 -153 -128 22.5 -161  
## 3 28.3 -63.9 -152 -128 22.5 -161  
## 4 28.1 -63.9 -152 -128 22.1 -161  
## 5 28.0 -63.9 -152 -128 22.1 -161  
## 6 27.9 -63.9 -152 -128 22.0 -161  
## 7 27.9 -63.9 -152 -128 21.9 -161  
## 8 27.8 -63.8 -152 -128 21.8 -161  
## 9 27.7 -63.8 -152 -128 21.7 -161  
## 10 27.7 -63.8 -152 -128 21.6 -161  
## roll\_belt pitch\_belt yaw\_belt classe  
## 1 1.41 8.07 -94.4 A  
## 2 1.41 8.07 -94.4 A  
## 3 1.42 8.07 -94.4 A  
## 4 1.48 8.05 -94.4 A  
## 5 1.48 8.07 -94.4 A  
## 6 1.45 8.06 -94.4 A  
## 7 1.42 8.09 -94.4 A  
## 8 1.42 8.13 -94.4 A  
## 9 1.43 8.16 -94.4 A  
## 10 1.45 8.17 -94.4 A

tail(selectedInputDF)

## roll\_forearm pitch\_forearm yaw\_forearm roll\_arm pitch\_arm yaw\_arm  
## 19617 0 0 0 -99.1 -33.7 79.4  
## 19618 0 0 0 -99.4 -33.8 79.0  
## 19619 0 0 0 -99.6 -34.5 77.3  
## 19620 0 0 0 -99.6 -35.1 76.3  
## 19621 0 0 0 -98.6 -36.7 73.5  
## 19622 0 0 0 -97.6 -37.7 71.5  
## roll\_belt pitch\_belt yaw\_belt classe  
## 19617 148 -34.7 129 E  
## 19618 147 -34.8 129 E  
## 19619 145 -35.3 130 E  
## 19620 145 -35.5 130 E  
## 19621 143 -35.9 131 E  
## 19622 143 -36.0 132 E

#str(selectedInputDF)  
  
Type <- apply(X=selectedInputDF,MARGIN=1,FUN=apply.type.rules)  
  
str(Type)

## num [1:19622] 1 1 1 1 1 1 1 1 1 1 ...

finalDF <- cbind(selectedInputDF,Type)  
  
head(finalDF)

## roll\_forearm pitch\_forearm yaw\_forearm roll\_arm pitch\_arm yaw\_arm  
## 1 28.4 -63.9 -153 -128 22.5 -161  
## 2 28.3 -63.9 -153 -128 22.5 -161  
## 3 28.3 -63.9 -152 -128 22.5 -161  
## 4 28.1 -63.9 -152 -128 22.1 -161  
## 5 28.0 -63.9 -152 -128 22.1 -161  
## 6 27.9 -63.9 -152 -128 22.0 -161  
## roll\_belt pitch\_belt yaw\_belt classe Type  
## 1 1.41 8.07 -94.4 A 1  
## 2 1.41 8.07 -94.4 A 1  
## 3 1.42 8.07 -94.4 A 1  
## 4 1.48 8.05 -94.4 A 1  
## 5 1.48 8.07 -94.4 A 1  
## 6 1.45 8.06 -94.4 A 1

tail(finalDF)

## roll\_forearm pitch\_forearm yaw\_forearm roll\_arm pitch\_arm yaw\_arm  
## 19617 0 0 0 -99.1 -33.7 79.4  
## 19618 0 0 0 -99.4 -33.8 79.0  
## 19619 0 0 0 -99.6 -34.5 77.3  
## 19620 0 0 0 -99.6 -35.1 76.3  
## 19621 0 0 0 -98.6 -36.7 73.5  
## 19622 0 0 0 -97.6 -37.7 71.5  
## roll\_belt pitch\_belt yaw\_belt classe Type  
## 19617 148 -34.7 129 E 5  
## 19618 147 -34.8 129 E 5  
## 19619 145 -35.3 130 E 5  
## 19620 145 -35.5 130 E 5  
## 19621 143 -35.9 131 E 5  
## 19622 143 -36.0 132 E 5

str(finalDF)

## 'data.frame': 19622 obs. of 11 variables:  
## $ roll\_forearm : num 28.4 28.3 28.3 28.1 28 27.9 27.9 27.8 27.7 27.7 ...  
## $ pitch\_forearm: num -63.9 -63.9 -63.9 -63.9 -63.9 -63.9 -63.9 -63.8 -63.8 -63.8 ...  
## $ yaw\_forearm : num -153 -153 -152 -152 -152 -152 -152 -152 -152 -152 ...  
## $ roll\_arm : num -128 -128 -128 -128 -128 -128 -128 -128 -128 -128 ...  
## $ pitch\_arm : num 22.5 22.5 22.5 22.1 22.1 22 21.9 21.8 21.7 21.6 ...  
## $ yaw\_arm : num -161 -161 -161 -161 -161 -161 -161 -161 -161 -161 ...  
## $ roll\_belt : num 1.41 1.41 1.42 1.48 1.48 1.45 1.42 1.42 1.43 1.45 ...  
## $ pitch\_belt : num 8.07 8.07 8.07 8.05 8.07 8.06 8.09 8.13 8.16 8.17 ...  
## $ yaw\_belt : num -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 ...  
## $ classe : chr "A" "A" "A" "A" ...  
## $ Type : num 1 1 1 1 1 1 1 1 1 1 ...

summary(finalDF)

## roll\_forearm pitch\_forearm yaw\_forearm roll\_arm   
## Min. :-180.0000 Min. :-72.50 Min. :-180.00 Min. :-180.00   
## 1st Qu.: -0.7375 1st Qu.: 0.00 1st Qu.: -68.60 1st Qu.: -31.77   
## Median : 21.7000 Median : 9.24 Median : 0.00 Median : 0.00   
## Mean : 33.8265 Mean : 10.71 Mean : 19.21 Mean : 17.83   
## 3rd Qu.: 140.0000 3rd Qu.: 28.40 3rd Qu.: 110.00 3rd Qu.: 77.30   
## Max. : 180.0000 Max. : 89.80 Max. : 180.00 Max. : 180.00   
## pitch\_arm yaw\_arm roll\_belt pitch\_belt   
## Min. :-88.800 Min. :-180.0000 Min. :-28.90 Min. :-55.8000   
## 1st Qu.:-25.900 1st Qu.: -43.1000 1st Qu.: 1.10 1st Qu.: 1.7600   
## Median : 0.000 Median : 0.0000 Median :113.00 Median : 5.2800   
## Mean : -4.612 Mean : -0.6188 Mean : 64.41 Mean : 0.3053   
## 3rd Qu.: 11.200 3rd Qu.: 45.8750 3rd Qu.:123.00 3rd Qu.: 14.9000   
## Max. : 88.500 Max. : 180.0000 Max. :162.00 Max. : 60.3000   
## yaw\_belt classe Type   
## Min. :-180.00 Length:19622 Min. :1.000   
## 1st Qu.: -88.30 Class :character 1st Qu.:1.000   
## Median : -13.00 Mode :character Median :3.000   
## Mean : -11.21 Mean :2.769   
## 3rd Qu.: 12.90 3rd Qu.:4.000   
## Max. : 179.00 Max. :5.000

tempDF2 <- finalDF[,1:10]  
Type <- finalDF['Type']  
#tdf <- data.frame({"Type"=typeLst})  
#tempDF2 <- cbind(tempDF2, Type)  
  
head(tempDF2)

## roll\_forearm pitch\_forearm yaw\_forearm roll\_arm pitch\_arm yaw\_arm  
## 1 28.4 -63.9 -153 -128 22.5 -161  
## 2 28.3 -63.9 -153 -128 22.5 -161  
## 3 28.3 -63.9 -152 -128 22.5 -161  
## 4 28.1 -63.9 -152 -128 22.1 -161  
## 5 28.0 -63.9 -152 -128 22.1 -161  
## 6 27.9 -63.9 -152 -128 22.0 -161  
## roll\_belt pitch\_belt yaw\_belt classe  
## 1 1.41 8.07 -94.4 A  
## 2 1.41 8.07 -94.4 A  
## 3 1.42 8.07 -94.4 A  
## 4 1.48 8.05 -94.4 A  
## 5 1.48 8.07 -94.4 A  
## 6 1.45 8.06 -94.4 A

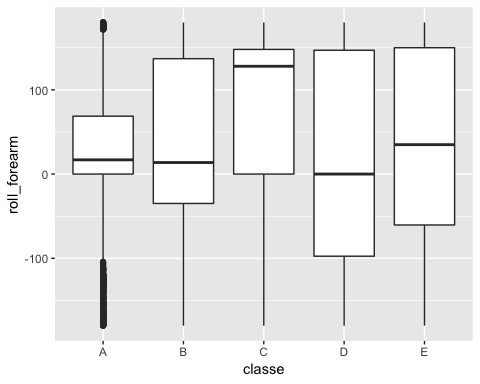
correlMtrx <- cor(tempDF2[,1:9])  
highlyCorrel <- findCorrelation(correlMtrx, cutoff=0.5)  
print(highlyCorrel)

## [1] 9

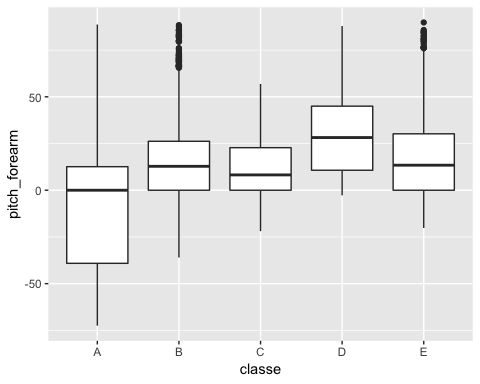
#data(PimaIndiansDiabetes)  
#head(PimaIndiansDiabetes)  
  
  
#control <- trainControl(method = "repeatedcv",number = 10,repeats=3)  
#model <- train(classe~.,data=tempDF2,method="lvq",preProcess="scale",trControl=control)  
#importance <- varImp(model,scale=FALSE)  
#print(importance)  
#plot(importance)  
  
################################ Exploring the Data ######################################  
  
plotDF <- finalDF  
plotDF$classe <- as.factor(plotDF$classe)  
str(plotDF)

## 'data.frame': 19622 obs. of 11 variables:  
## $ roll\_forearm : num 28.4 28.3 28.3 28.1 28 27.9 27.9 27.8 27.7 27.7 ...  
## $ pitch\_forearm: num -63.9 -63.9 -63.9 -63.9 -63.9 -63.9 -63.9 -63.8 -63.8 -63.8 ...  
## $ yaw\_forearm : num -153 -153 -152 -152 -152 -152 -152 -152 -152 -152 ...  
## $ roll\_arm : num -128 -128 -128 -128 -128 -128 -128 -128 -128 -128 ...  
## $ pitch\_arm : num 22.5 22.5 22.5 22.1 22.1 22 21.9 21.8 21.7 21.6 ...  
## $ yaw\_arm : num -161 -161 -161 -161 -161 -161 -161 -161 -161 -161 ...  
## $ roll\_belt : num 1.41 1.41 1.42 1.48 1.48 1.45 1.42 1.42 1.43 1.45 ...  
## $ pitch\_belt : num 8.07 8.07 8.07 8.05 8.07 8.06 8.09 8.13 8.16 8.17 ...  
## $ yaw\_belt : num -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 ...  
## $ classe : Factor w/ 5 levels "A","B","C","D",..: 1 1 1 1 1 1 1 1 1 1 ...  
## $ Type : num 1 1 1 1 1 1 1 1 1 1 ...

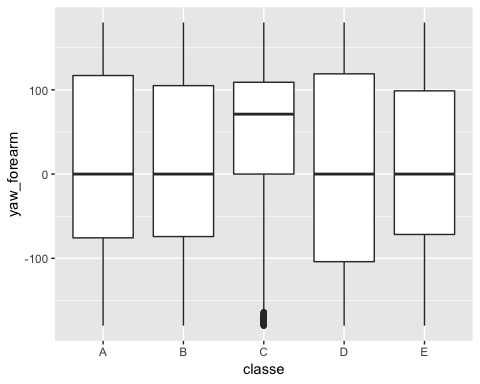
#print(names(plotDF))  
  
bxpRFA <- ggplot(plotDF,aes(x=classe,y=roll\_forearm)) + geom\_boxplot()  
plot(bxpRFA)



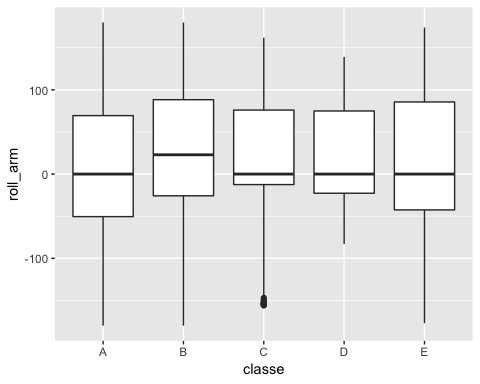
bxpPFA <- ggplot(plotDF,aes(x=classe,y=pitch\_forearm)) + geom\_boxplot()  
plot(bxpPFA)



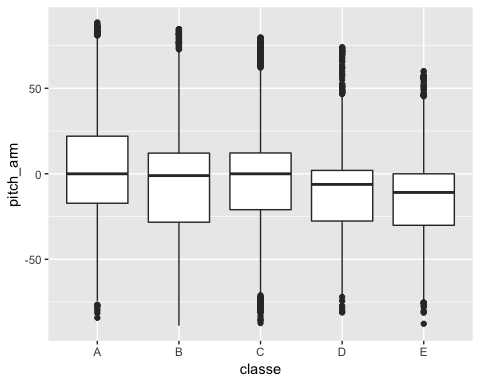
bxpYFA <- ggplot(plotDF,aes(x=classe,y=yaw\_forearm)) + geom\_boxplot()  
plot(bxpYFA)



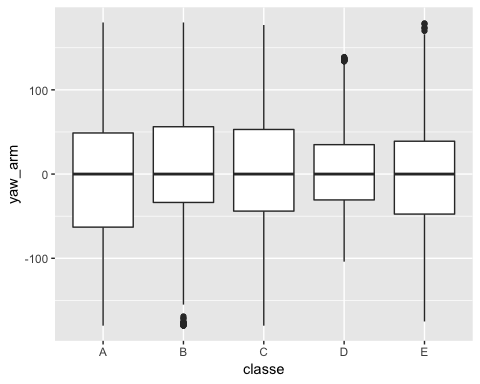
bxpRA <- ggplot(plotDF,aes(x=classe,y=roll\_arm)) + geom\_boxplot()  
plot(bxpRA)



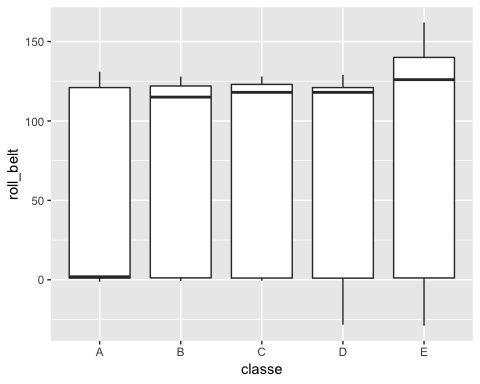
bxpPA <- ggplot(plotDF,aes(x=classe,y=pitch\_arm)) + geom\_boxplot()  
plot(bxpPA)



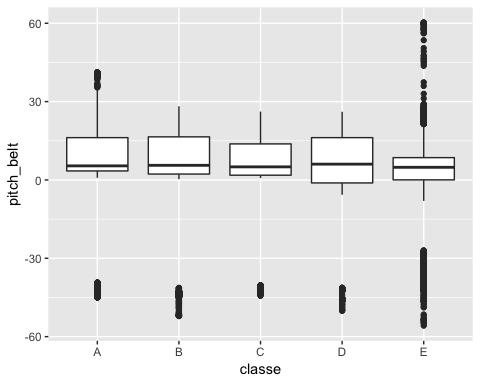
bxpYA <- ggplot(plotDF,aes(x=classe,y=yaw\_arm)) + geom\_boxplot()  
plot(bxpYA)



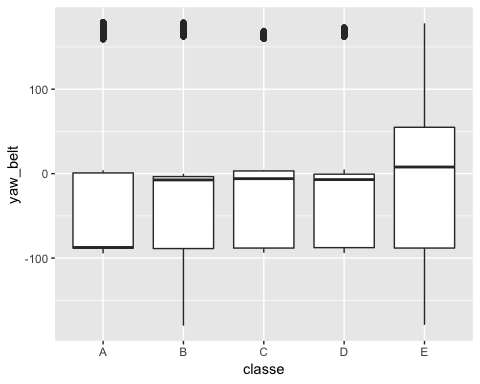
bxpRB <- ggplot(plotDF,aes(x=classe,y=roll\_belt)) + geom\_boxplot()  
plot(bxpRB)



bxpPB <- ggplot(plotDF,aes(x=classe,y=pitch\_belt)) + geom\_boxplot()  
plot(bxpPB)



bxpYB <- ggplot(plotDF,aes(x=classe,y=yaw\_belt)) + geom\_boxplot()  
plot(bxpYB)



###################### K cross over validation and qda and lda analysis  
  
  
  
library(MASS)

## Warning: package 'MASS' was built under R version 3.5.2

##   
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':  
##   
## select

tempDF2$classe=factor(tempDF2$classe)  
str(tempDF2)

## 'data.frame': 19622 obs. of 10 variables:  
## $ roll\_forearm : num 28.4 28.3 28.3 28.1 28 27.9 27.9 27.8 27.7 27.7 ...  
## $ pitch\_forearm: num -63.9 -63.9 -63.9 -63.9 -63.9 -63.9 -63.9 -63.8 -63.8 -63.8 ...  
## $ yaw\_forearm : num -153 -153 -152 -152 -152 -152 -152 -152 -152 -152 ...  
## $ roll\_arm : num -128 -128 -128 -128 -128 -128 -128 -128 -128 -128 ...  
## $ pitch\_arm : num 22.5 22.5 22.5 22.1 22.1 22 21.9 21.8 21.7 21.6 ...  
## $ yaw\_arm : num -161 -161 -161 -161 -161 -161 -161 -161 -161 -161 ...  
## $ roll\_belt : num 1.41 1.41 1.42 1.48 1.48 1.45 1.42 1.42 1.43 1.45 ...  
## $ pitch\_belt : num 8.07 8.07 8.07 8.05 8.07 8.06 8.09 8.13 8.16 8.17 ...  
## $ yaw\_belt : num -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 ...  
## $ classe : Factor w/ 5 levels "A","B","C","D",..: 1 1 1 1 1 1 1 1 1 1 ...

K <- 20  
folds <- cut(seq(1,nrow(tempDF2)),breaks = K,labels=FALSE)  
head(folds)

## [1] 1 1 1 1 1 1

set.seed(1)  
cv.qda <- sapply(1:K, FUN = function(i) {  
 testID <- which(folds == i, arr.ind = TRUE)  
 test <- tempDF2[testID,]  
 train <- tempDF2[-testID,]  
 qdaf <- qda(classe~.,data=train)  
 qda.pred <- predict(qdaf, test)  
 cv.est.qda <- mean(qda.pred$class != test$classe)  
 return(cv.est.qda)  
})  
print(cv.qda)

## [1] 0.6568228 0.6065240 0.2548420 0.4740061 0.6870540 0.5932722 1.0000000  
## [8] 0.9418960 1.0000000 0.9724771 0.7930683 0.9714577 0.7431193 0.6309888  
## [15] 0.7339450 0.6697248 0.6095821 0.9633028 0.8623853 0.7474542

print(mean(cv.qda))

## [1] 0.7455961

K <- 20  
folds <- cut(seq(1,nrow(tempDF2)),breaks = K,labels=FALSE)  
head(folds)

## [1] 1 1 1 1 1 1

set.seed(1)  
cv.lda <- sapply(1:K, FUN = function(i) {  
 testID <- which(folds == i, arr.ind = TRUE)  
 test <- tempDF2[testID,]  
 train <- tempDF2[-testID,]  
 ldaf <- lda(classe~.,data=train)  
 lda.pred <- predict(ldaf, test)  
 cv.est.lda <- mean(lda.pred$class != test$classe)  
 return(cv.est.lda)  
})  
print(cv.lda)

## [1] 0.5488798 0.7237513 0.2721713 0.4485219 0.7543323 0.6075433 1.0000000  
## [8] 1.0000000 1.0000000 0.9714577 1.0000000 1.0000000 0.9653415 0.8695209  
## [15] 0.9785933 0.9653415 0.7747197 0.9633028 0.9989806 0.9725051

print(mean(cv.lda))

## [1] 0.8407481

str(finalDF)

## 'data.frame': 19622 obs. of 11 variables:  
## $ roll\_forearm : num 28.4 28.3 28.3 28.1 28 27.9 27.9 27.8 27.7 27.7 ...  
## $ pitch\_forearm: num -63.9 -63.9 -63.9 -63.9 -63.9 -63.9 -63.9 -63.8 -63.8 -63.8 ...  
## $ yaw\_forearm : num -153 -153 -152 -152 -152 -152 -152 -152 -152 -152 ...  
## $ roll\_arm : num -128 -128 -128 -128 -128 -128 -128 -128 -128 -128 ...  
## $ pitch\_arm : num 22.5 22.5 22.5 22.1 22.1 22 21.9 21.8 21.7 21.6 ...  
## $ yaw\_arm : num -161 -161 -161 -161 -161 -161 -161 -161 -161 -161 ...  
## $ roll\_belt : num 1.41 1.41 1.42 1.48 1.48 1.45 1.42 1.42 1.43 1.45 ...  
## $ pitch\_belt : num 8.07 8.07 8.07 8.05 8.07 8.06 8.09 8.13 8.16 8.17 ...  
## $ yaw\_belt : num -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 ...  
## $ classe : chr "A" "A" "A" "A" ...  
## $ Type : num 1 1 1 1 1 1 1 1 1 1 ...

names(finalDF)

## [1] "roll\_forearm" "pitch\_forearm" "yaw\_forearm" "roll\_arm"   
## [5] "pitch\_arm" "yaw\_arm" "roll\_belt" "pitch\_belt"   
## [9] "yaw\_belt" "classe" "Type"

################################ Random Forest #################################3  
  
  
DF51 <- finalDF[,1:9]  
DF52 <- finalDF[,10]  
names(DF51)

## [1] "roll\_forearm" "pitch\_forearm" "yaw\_forearm" "roll\_arm"   
## [5] "pitch\_arm" "yaw\_arm" "roll\_belt" "pitch\_belt"   
## [9] "yaw\_belt"

names(DF52)

## NULL

rndforestDF <- cbind(DF51,Type)  
rndforestDF$Type <- as.factor(rndforestDF$Type)  
names(rndforestDF)

## [1] "roll\_forearm" "pitch\_forearm" "yaw\_forearm" "roll\_arm"   
## [5] "pitch\_arm" "yaw\_arm" "roll\_belt" "pitch\_belt"   
## [9] "yaw\_belt" "Type"

str(rndforestDF)

## 'data.frame': 19622 obs. of 10 variables:  
## $ roll\_forearm : num 28.4 28.3 28.3 28.1 28 27.9 27.9 27.8 27.7 27.7 ...  
## $ pitch\_forearm: num -63.9 -63.9 -63.9 -63.9 -63.9 -63.9 -63.9 -63.8 -63.8 -63.8 ...  
## $ yaw\_forearm : num -153 -153 -152 -152 -152 -152 -152 -152 -152 -152 ...  
## $ roll\_arm : num -128 -128 -128 -128 -128 -128 -128 -128 -128 -128 ...  
## $ pitch\_arm : num 22.5 22.5 22.5 22.1 22.1 22 21.9 21.8 21.7 21.6 ...  
## $ yaw\_arm : num -161 -161 -161 -161 -161 -161 -161 -161 -161 -161 ...  
## $ roll\_belt : num 1.41 1.41 1.42 1.48 1.48 1.45 1.42 1.42 1.43 1.45 ...  
## $ pitch\_belt : num 8.07 8.07 8.07 8.05 8.07 8.06 8.09 8.13 8.16 8.17 ...  
## $ yaw\_belt : num -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 ...  
## $ Type : Factor w/ 5 levels "1","2","3","4",..: 1 1 1 1 1 1 1 1 1 1 ...

table(rndforestDF$Tpe)

## < table of extent 0 >

set.seed(325)  
ind <- sample(2,nrow(rndforestDF),replace = TRUE,prob = c(0.7, 0.3))  
train <- rndforestDF[ind==1,]  
test <- rndforestDF[ind==2,]  
  
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:ggplot2':  
##   
## margin

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

set.seed(757)  
rf <- randomForest(Type~.,data=train)  
print(rf)

##   
## Call:  
## randomForest(formula = Type ~ ., data = train)   
## Type of random forest: classification  
## Number of trees: 500  
## No. of variables tried at each split: 3  
##   
## OOB estimate of error rate: 1.28%  
## Confusion matrix:  
## 1 2 3 4 5 class.error  
## 1 3893 10 1 0 0 0.002817623  
## 2 16 2557 56 3 0 0.028495441  
## 3 0 21 2370 15 4 0.016597510  
## 4 0 1 11 2189 5 0.007706256  
## 5 1 12 7 12 2503 0.012623274

############################### Predict using the train set  
attributes(rf)

## $names  
## [1] "call" "type" "predicted"   
## [4] "err.rate" "confusion" "votes"   
## [7] "oob.times" "classes" "importance"   
## [10] "importanceSD" "localImportance" "proximity"   
## [13] "ntree" "mtry" "forest"   
## [16] "y" "test" "inbag"   
## [19] "terms"   
##   
## $class  
## [1] "randomForest.formula" "randomForest"

p1 <- predict(rf, train)  
head(p1)

## 1 3 4 5 6 7   
## 1 1 1 1 1 1   
## Levels: 1 2 3 4 5

head(train$Type)

## [1] 1 1 1 1 1 1  
## Levels: 1 2 3 4 5

confusionMatrix(p1,train$Type)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 1 2 3 4 5  
## 1 3904 0 0 0 0  
## 2 0 2632 0 0 0  
## 3 0 0 2410 0 0  
## 4 0 0 0 2206 0  
## 5 0 0 0 0 2535  
##   
## Overall Statistics  
##   
## Accuracy : 1   
## 95% CI : (0.9997, 1)  
## No Information Rate : 0.2852   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 1   
##   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: 1 Class: 2 Class: 3 Class: 4 Class: 5  
## Sensitivity 1.0000 1.0000 1.0000 1.0000 1.0000  
## Specificity 1.0000 1.0000 1.0000 1.0000 1.0000  
## Pos Pred Value 1.0000 1.0000 1.0000 1.0000 1.0000  
## Neg Pred Value 1.0000 1.0000 1.0000 1.0000 1.0000  
## Prevalence 0.2852 0.1923 0.1761 0.1612 0.1852  
## Detection Rate 0.2852 0.1923 0.1761 0.1612 0.1852  
## Detection Prevalence 0.2852 0.1923 0.1761 0.1612 0.1852  
## Balanced Accuracy 1.0000 1.0000 1.0000 1.0000 1.0000

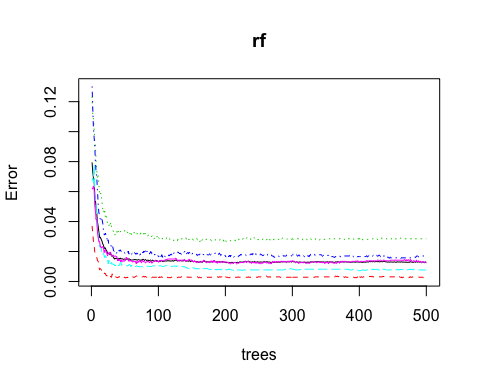
############################## Predict using the test set  
  
p2 <- predict(rf, test)  
confusionMatrix(p2,test$Type)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 1 2 3 4 5  
## 1 1669 7 0 0 0  
## 2 7 1128 14 1 1  
## 3 0 26 992 8 1  
## 4 0 4 4 1000 5  
## 5 0 0 2 1 1065  
##   
## Overall Statistics  
##   
## Accuracy : 0.9864   
## 95% CI : (0.9831, 0.9891)  
## No Information Rate : 0.2824   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.9827   
##   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: 1 Class: 2 Class: 3 Class: 4 Class: 5  
## Sensitivity 0.9958 0.9682 0.9802 0.9901 0.9935  
## Specificity 0.9984 0.9952 0.9929 0.9974 0.9994  
## Pos Pred Value 0.9958 0.9800 0.9659 0.9872 0.9972  
## Neg Pred Value 0.9984 0.9923 0.9959 0.9980 0.9986  
## Prevalence 0.2824 0.1963 0.1705 0.1702 0.1806  
## Detection Rate 0.2812 0.1901 0.1671 0.1685 0.1794  
## Detection Prevalence 0.2824 0.1939 0.1730 0.1707 0.1799  
## Balanced Accuracy 0.9971 0.9817 0.9866 0.9937 0.9964

names(train)

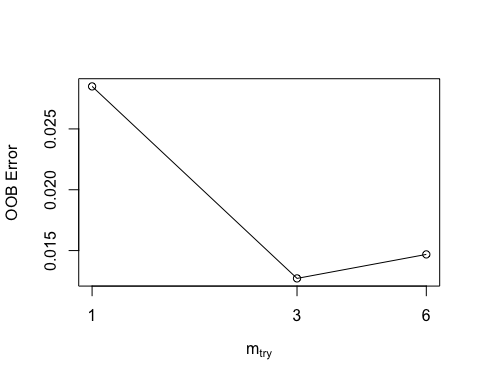
## [1] "roll\_forearm" "pitch\_forearm" "yaw\_forearm" "roll\_arm"   
## [5] "pitch\_arm" "yaw\_arm" "roll\_belt" "pitch\_belt"   
## [9] "yaw\_belt" "Type"

########################################### Tuning the Random Forest #######################3  
  
plot(rf)



t <- tuneRF(train[,-10],train[,10],stepFactor = 0.5,plot = TRUE,ntreeTry = 250,trace=TRUE,improve=0.05)

## mtry = 3 OOB error = 1.27%   
## Searching left ...  
## mtry = 6 OOB error = 1.47%   
## -0.1551724 0.05   
## Searching right ...  
## mtry = 1 OOB error = 2.85%   
## -1.241379 0.05



########################################## Tuning using the train sret  
  
rf2 <- randomForest(Type~.,data=train,ntree = 250,mtry = 3, importance = TRUE, proximity = TRUE)  
print(rf2)

##   
## Call:  
## randomForest(formula = Type ~ ., data = train, ntree = 250, mtry = 3, importance = TRUE, proximity = TRUE)   
## Type of random forest: classification  
## Number of trees: 250  
## No. of variables tried at each split: 3  
##   
## OOB estimate of error rate: 1.28%  
## Confusion matrix:  
## 1 2 3 4 5 class.error  
## 1 3894 10 0 0 0 0.002561475  
## 2 15 2555 58 4 0 0.029255319  
## 3 0 24 2368 14 4 0.017427386  
## 4 0 1 9 2191 5 0.006799637  
## 5 0 11 6 14 2504 0.012228797

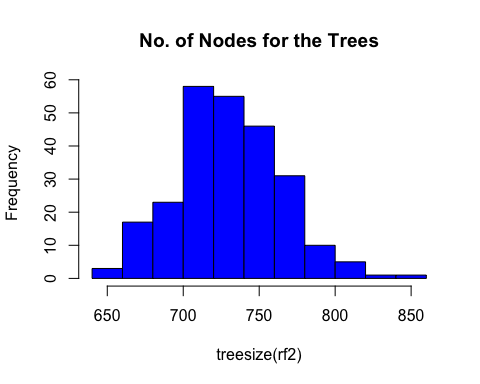
p21 <- predict(rf2,train)  
confusionMatrix(p21, train$Type)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 1 2 3 4 5  
## 1 3904 0 0 0 0  
## 2 0 2632 0 0 0  
## 3 0 0 2410 0 0  
## 4 0 0 0 2206 0  
## 5 0 0 0 0 2535  
##   
## Overall Statistics  
##   
## Accuracy : 1   
## 95% CI : (0.9997, 1)  
## No Information Rate : 0.2852   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 1   
##   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: 1 Class: 2 Class: 3 Class: 4 Class: 5  
## Sensitivity 1.0000 1.0000 1.0000 1.0000 1.0000  
## Specificity 1.0000 1.0000 1.0000 1.0000 1.0000  
## Pos Pred Value 1.0000 1.0000 1.0000 1.0000 1.0000  
## Neg Pred Value 1.0000 1.0000 1.0000 1.0000 1.0000  
## Prevalence 0.2852 0.1923 0.1761 0.1612 0.1852  
## Detection Rate 0.2852 0.1923 0.1761 0.1612 0.1852  
## Detection Prevalence 0.2852 0.1923 0.1761 0.1612 0.1852  
## Balanced Accuracy 1.0000 1.0000 1.0000 1.0000 1.0000

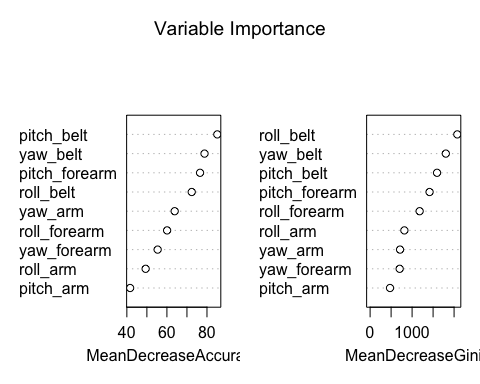
########################################## Tuning using the predict set  
  
p22 <- predict(rf2,test)  
confusionMatrix(p22,test$Type)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 1 2 3 4 5  
## 1 1668 7 0 0 0  
## 2 8 1128 12 1 1  
## 3 0 26 995 9 1  
## 4 0 4 5 999 5  
## 5 0 0 0 1 1065  
##   
## Overall Statistics  
##   
## Accuracy : 0.9865   
## 95% CI : (0.9833, 0.9893)  
## No Information Rate : 0.2824   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.983   
##   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: 1 Class: 2 Class: 3 Class: 4 Class: 5  
## Sensitivity 0.9952 0.9682 0.9832 0.9891 0.9935  
## Specificity 0.9984 0.9954 0.9927 0.9972 0.9998  
## Pos Pred Value 0.9958 0.9809 0.9651 0.9862 0.9991  
## Neg Pred Value 0.9981 0.9923 0.9965 0.9978 0.9986  
## Prevalence 0.2824 0.1963 0.1705 0.1702 0.1806  
## Detection Rate 0.2810 0.1901 0.1676 0.1683 0.1794  
## Detection Prevalence 0.2822 0.1938 0.1737 0.1707 0.1796  
## Balanced Accuracy 0.9968 0.9818 0.9879 0.9931 0.9966

hist(treesize(rf2),main = "No. of Nodes for the Trees", col="blue")



varImpPlot(rf2, main = "Variable Importance")



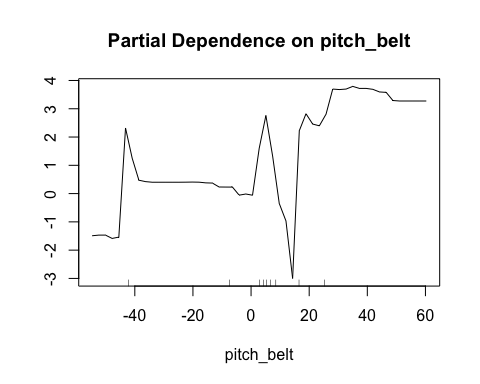
importance(rf2)

## 1 2 3 4 5  
## roll\_forearm 42.85669 51.26740 76.88669 50.68116 49.68210  
## pitch\_forearm 43.50019 57.18189 62.42005 85.07532 57.29369  
## yaw\_forearm 34.63824 41.26679 50.80003 42.62494 45.06413  
## roll\_arm 26.39943 42.96575 39.21809 43.95813 31.43289  
## pitch\_arm 21.72842 36.78679 39.61017 34.61318 33.93804  
## yaw\_arm 43.65869 50.29782 51.96163 45.27550 31.45772  
## roll\_belt 45.45110 61.69194 57.25247 49.89291 58.52817  
## pitch\_belt 50.46518 76.36023 68.35774 60.21233 42.53164  
## yaw\_belt 66.17040 65.77808 58.22214 68.60817 33.66951  
## MeanDecreaseAccuracy MeanDecreaseGini  
## roll\_forearm 60.09964 1179.6245  
## pitch\_forearm 76.53942 1416.2912  
## yaw\_forearm 55.37273 704.0048  
## roll\_arm 49.37771 816.1285  
## pitch\_arm 41.62751 474.1469  
## yaw\_arm 63.88046 713.1944  
## roll\_belt 72.39013 2077.0343  
## pitch\_belt 85.16214 1594.0007  
## yaw\_belt 78.77275 1802.9883

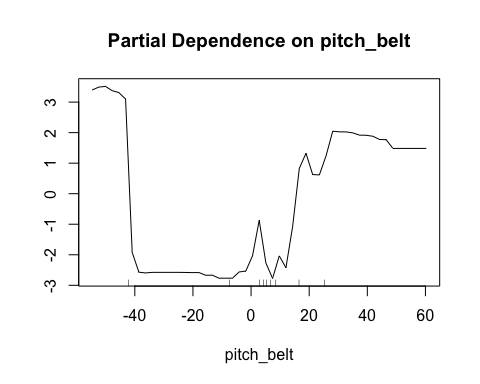
varUsed(rf2)

## [1] 19386 19848 18608 18456 16435 16639 21963 25299 25660

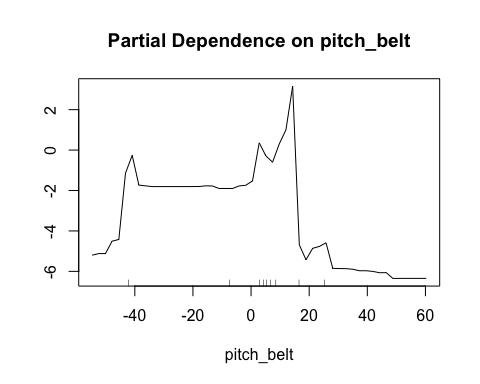
###########################3 How does pitch belt predict the various classe  
  
############# classe = A means Type = 1  
############# classe = B means Type = 2  
############# classe = C means Type = 3  
############# classe = D means Type = 4  
############# classe = E means Type = 5  
  
partialPlot(rf2, train, pitch\_belt, "1")



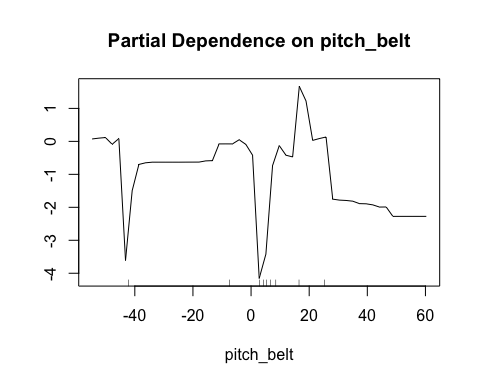
partialPlot(rf2, train, pitch\_belt, "2")



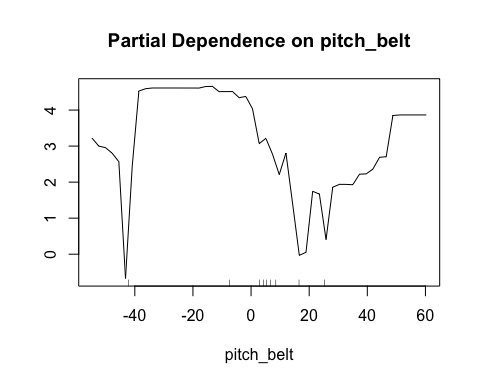
partialPlot(rf2, train, pitch\_belt, "3")



partialPlot(rf2, train, pitch\_belt, "4")



partialPlot(rf2, train, pitch\_belt, "5")



## Including Plots

Note that the echo = FALSE parameter was added to the code chunk to prevent printing of the R code that generated the plot.