Analysis on Weight Lifting Exercises Dataset

Kason M. Joo

May 26, 2016

Background

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, my 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.

Prepare packages

```r
library(mlbench)
```

```
## Warning: package 'mlbench' was built under R version 3.2.5
```

```r
library(reshape2)
```

```
## Warning: package 'reshape2' was built under R version 3.2.5
```

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

```
## Warning: package 'rpart.plot' was built under R version 3.2.5
```

```r
library(RColorBrewer)
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.2.5
```

```
## randomForest 4.6-12
```

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

```r
library(data.table)
```

```
## Warning: package 'data.table' was built under R version 3.2.5
```

```
##
## Attaching package: 'data.table'
## The following objects are masked from 'package:reshape2':
##
##     dcast, melt
```

```
library(rattle)
```

```
## Warning: package 'rattle' was built under R version 3.2.5
```

```
## Rattle: A free graphical interface for data mining with R.
## XXXX 4.1.0 Copyright (c) 2006-2015 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
library(e1071)
```

```
## Warning: package 'e1071' was built under R version 3.2.5
```

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.2.5
```

```
##
## Attaching package: 'ggplot2'
```

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

```
library(munsell)
```

```
## Warning: package 'munsell' was built under R version 3.2.5
```

Data Loading

Subset the training dataset to include only the predictors and the outcome variable. Exclude all variables that contain most NA values.

```
training <- read.csv("pml-training.csv", header = T, sep = ",")
testing <- read.csv("pml-testing.csv", header = T, sep = ",")
# summary(training)
# summary(testing)
# Remove empty columns
isAnyMissing <- sapply(training, function (x) any(is.na(x) | x == ""))
isPredictor <- !isAnyMissing & grepl("belt|[^(fore)]arm|dumbbell|forearm", names(isAnyMissing))
predCandidates <- names(isAnyMissing)[isPredictor]
predCandidates
```

```
##  [1] "roll_belt"            "pitch_belt"           "yaw_belt"
##  [4] "total_accel_belt"     "gyros_belt_x"         "gyros_belt_y"
##  [7] "gyros_belt_z"         "accel_belt_x"         "accel_belt_y"
## [10] "accel_belt_z"         "magnet_belt_x"        "magnet_belt_y"
## [13] "magnet_belt_z"        "roll_arm"             "pitch_arm"
## [16] "yaw_arm"              "total_accel_arm"      "gyros_arm_x"
## [19] "gyros_arm_y"          "gyros_arm_z"          "accel_arm_x"
## [22] "accel_arm_y"          "accel_arm_z"          "magnet_arm_x"
## [25] "magnet_arm_y"         "magnet_arm_z"         "roll_dumbbell"
## [28] "pitch_dumbbell"       "yaw_dumbbell"         "total_accel_dumbbell"
## [31] "gyros_dumbbell_x"     "gyros_dumbbell_y"     "gyros_dumbbell_z"
## [34] "accel_dumbbell_x"     "accel_dumbbell_y"     "accel_dumbbell_z"
## [37] "magnet_dumbbell_x"    "magnet_dumbbell_y"    "magnet_dumbbell_z"
## [40] "roll_forearm"         "pitch_forearm"        "yaw_forearm"
## [43] "total_accel_forearm"  "gyros_forearm_x"      "gyros_forearm_y"
## [46] "gyros_forearm_z"      "accel_forearm_x"      "accel_forearm_y"
## [49] "accel_forearm_z"      "magnet_forearm_x"     "magnet_forearm_y"
## [52] "magnet_forearm_z"
```

```
varToInclude <- c("classe", predCandidates)
training <- training[, varToInclude]
dim(training)
```

```
## [1] 19622    53
```

```
# Make sure classe is a factor variable
class(training$classe)
```

```
## [1] "factor"
```

Split Into training and validation dataset

```
require(caret)
```

```
## Loading required package: caret
```

```
## Warning: package 'caret' was built under R version 3.2.5
```

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

```
inTrain <- createDataPartition(training$classe, p=0.6, list=FALSE)
my_Training <- training[inTrain, ]
my_Testing <- training[-inTrain, ]
dim(my_Training); dim(my_Testing)
```

```
## [1] 11776    53
```

```
## [1] 7846   53
```

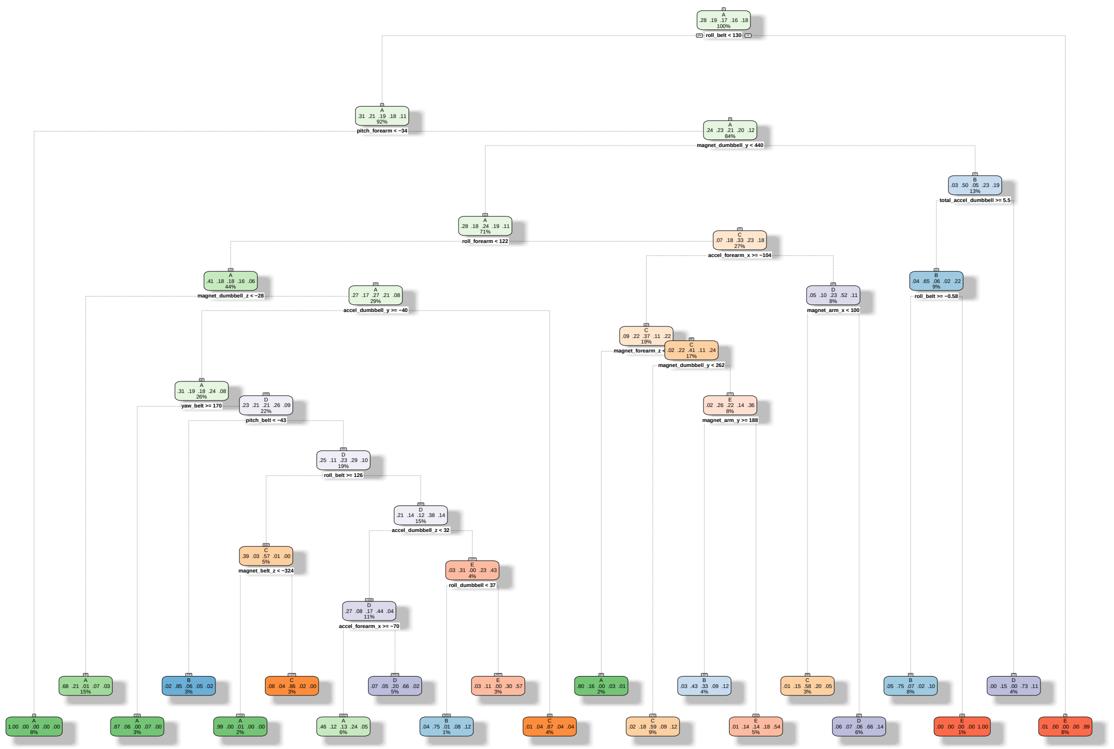
Train Model on Training dataset and cross validation

Using random forest, the out of sample error should be small. The error will be estimated using the 40% probing sample. I would be quite happy with an error estimate of 3% or less.

Prediction with Decision Trees

```
set.seed(12345)
modFitA1 <- rpart(classe ~ ., data=my_Training, method="class")
fancyRpartPlot(modFitA1, sub="")
```

```
## Warning: labs do not fit even at cex 0.15, there may be some overplotting
```



```
predictionsA1 <- predict(modFitA1, my_Testing, type = "class")
cmtree <- confusionMatrix(predictionsA1, my_Testing$classe)
cmtree
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2040  349   85  220   66
##          B   51  823  153   50   94
##          C   60  161  969  102  101
##          D   64  103  114  776   98
##          E   17   82   47  138 1083
##
## Overall Statistics
##
##                Accuracy : 0.7253
##                  95% CI : (0.7153, 0.7352)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6496
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9140   0.5422   0.7083   0.6034   0.7510
## Specificity            0.8717   0.9450   0.9345   0.9422   0.9557
## Pos Pred Value         0.7391   0.7028   0.6956   0.6719   0.7922
## Neg Pred Value         0.9622   0.8959   0.9382   0.9238   0.9446
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2600   0.1049   0.1235   0.0989   0.1380
## Detection Prevalence   0.3518   0.1492   0.1775   0.1472   0.1742
## Balanced Accuracy      0.8929   0.7436   0.8214   0.7728   0.8533
```

```
# plot(modFitA1)
```

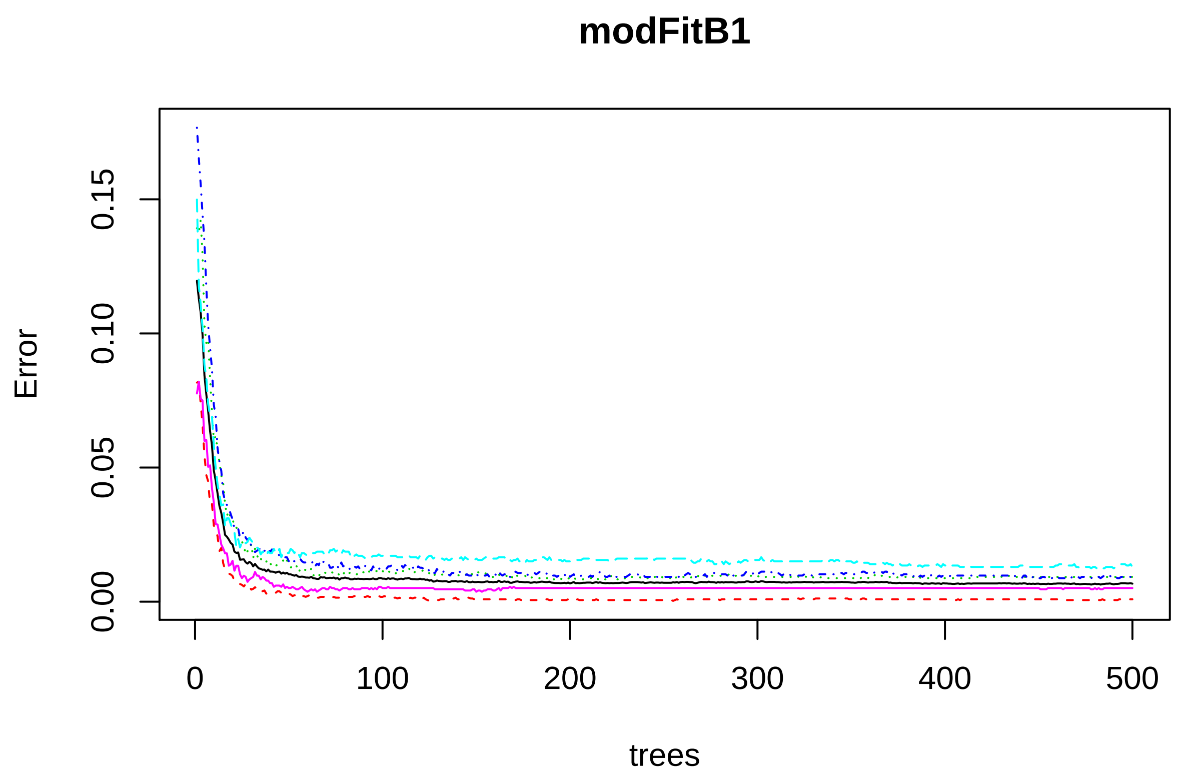
Prediction with Random Forests

```
set.seed(12345)
modFitB1 <- randomForest(classe ~ ., data=my_Training)
predictionB1 <- predict(modFitB1, my_Testing, type = "class")
cmrf <- confusionMatrix(predictionB1, my_Testing$classe)
cmrf
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2231    9    0    0    0
##          B    1 1509    8    0    0
##          C    0    0 1358   16    2
##          D    0    0    2 1269    4
##          E    0    0    0    1 1436
##
## Overall Statistics
##
##                Accuracy : 0.9945
##                  95% CI : (0.9926, 0.996)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9931
##  Mcnemar's Test P-Value : NA
```

```
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9996   0.9941   0.9927   0.9868   0.9958
## Specificity            0.9984   0.9986   0.9972   0.9991   0.9998
## Pos Pred Value         0.9960   0.9941   0.9869   0.9953   0.9993
## Neg Pred Value         0.9998   0.9986   0.9985   0.9974   0.9991
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2843   0.1923   0.1731   0.1617   0.1830
## Detection Prevalence   0.2855   0.1935   0.1754   0.1625   0.1832
## Balanced Accuracy      0.9990   0.9963   0.9950   0.9929   0.9978
```

```
plot(modFitB1)
```



The estimated error rate is less than 1%.

Predicting Results on the Test Data

Random Forests gave an Accuracy in the Validation dataset of 99.89%, which was more accurate that what I got from the Decision Trees. The expected out-of-sample error is 100-99.89 = 0.11%.

Prediction with Random Forests on testing dataset. Get the result for Course project quiz.

```
predictionB2 <- predict(modFitB1, testing, type = "class")
predictionB2
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  B  A  B  A  A  E  D  B  A  A  B  C  B  A  E  E  A  B  B  B
## Levels: A B C D E
```

```
# Write the results to a text file for submission
pml_write_files = function(x){
    n = length(x)
    for(i in 1:n){
        filename = paste0("problem_id_",i,".txt")
        write.table(x[i],file=filename,quote=FALSE,row.names=FALSE,col.names=FALSE)
    }
}
pml_write_files(predictionB2)
```

Conclusion

The prediction result of the test data is: B A A A A E D B A A B C B A E E A B B B.