Practical Machine Learning Project

Lori Ziegelmeier

Sunday, August 24, 2014

Introduction

Data related to personal activity may now be readily collected using devices such as Jawbone Up, Nike FuelBand, and Fitbit. Often, people are interested in quantifying how much of a particular activity they do. However, quantifying how well they do a particular activity is quite important as well.

In this project, we consider quantifying how well 6 individuals perform dumbbell bicep curls. These individuals were instructed to perform this activity in five different fashions (1 correct method and 4 incorrect): exactly according to the specification (Class A), throwing the elbows to the front (Class B), lifting the dumbbell only halfway (Class C), lowering the dumbbell only halfway (Class D), and throwing the hips to the front (Class E). Data from accelerometers located on the belt, forearm, arm, and dumbell were collected. This data may be found in the Weight Lifting Exercise Dataset http://groupware.les.inf.puc-rio.br/har [1].

Loading and Processing the Data

First, we obtain the data from the links provided in the project description. As the data takes a bit of time to download, we cache previously loaded data.

```
setwd("C:/Users/lziegel1/Documents/CourseraDataScience/R/")
if(!file.exists("PMLTrainData.csv")){
fileUrl1 <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
download.file(fileUrl1,destfile="./PMLTrainData.csv")
}
if(!file.exists("PMLTestData.csv")){
fileUrl2 <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
download.file(fileUrl2,destfile="./PMLTestData.csv")
}
TrainData=read.csv("PMLTrainData.csv")
TestData=read.csv("PMLTestData.csv")
dtrain=dim(TrainData)
dtest=dim(TestData)
str(TrainData)
```

```
## 'data.frame':    19622 obs. of  160 variables:
##  $ X                       : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ user_name               : Factor w/ 6 levels "adelmo","carlitos",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ raw_timestamp_part_1    : int  1323084231 1323084231 1323084231 1323084232 1323084232 1323084232 1
##  $ raw_timestamp_part_2    : int  788290 808298 820366 120339 196328 304277 368296 440390 484323 484
##  $ cvtd_timestamp          : Factor w/ 20 levels "02/12/2011 13:32",..: 9 9 9 9 9 9 9 9 9 9 ...
##  $ new_window              : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ 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      : Factor w/ 397 levels "","-0.016850",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ kurtosis_picth_belt     : Factor w/ 317 levels "","-0.021887",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```
##  $ kurtosis_yaw_belt       : Factor w/ 2 levels "","#DIV/0!": 1 1 1 1 1 1 1 1 1 1 ...
##  $ skewness_roll_belt      : Factor w/ 395 levels "","-0.003095",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ skewness_roll_belt.1    : Factor w/ 338 levels "","-0.005928",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ skewness_yaw_belt       : Factor w/ 2 levels "","#DIV/0!": 1 1 1 1 1 1 1 1 1 1 ...
##  $ 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            : Factor w/ 68 levels "","-0.1","-0.2",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ 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            : Factor w/ 68 levels "","-0.1","-0.2",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ 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      : Factor w/ 4 levels "","#DIV/0!","0.00",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ 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      : Factor w/ 330 levels "","-0.02438",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ kurtosis_picth_arm     : Factor w/ 328 levels "","-0.00484",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ kurtosis_yaw_arm       : Factor w/ 395 levels "","-0.01548",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ skewness_roll_arm      : Factor w/ 331 levels "","-0.00051",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ skewness_pitch_arm     : Factor w/ 328 levels "","-0.00184",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ skewness_yaw_arm       : Factor w/ 395 levels "","-0.00311",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ 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 : Factor w/ 398 levels "","-0.0035","-0.0073",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ kurtosis_picth_dumbbell: Factor w/ 401 levels "","-0.0163","-0.0233",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ kurtosis_yaw_dumbbell  : Factor w/ 2 levels "","#DIV/0!": 1 1 1 1 1 1 1 1 1 1 ...
##  $ skewness_roll_dumbbell : Factor w/ 401 levels "","-0.0082","-0.0096",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ skewness_pitch_dumbbell: Factor w/ 402 levels "","-0.0053","-0.0084",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ skewness_yaw_dumbbell  : Factor w/ 2 levels "","#DIV/0!": 1 1 1 1 1 1 1 1 1 1 ...
##  $ 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       : Factor w/ 73 levels "","-0.1","-0.2",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ 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       : Factor w/ 73 levels "","-0.1","-0.2",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ amplitude_roll_dumbbell: num  NA NA NA NA NA NA NA NA NA NA ...
##   [list output truncated]
```

Notice that there are 19622 entries in the training data set with 160 variables and only 20 entries in the testing data set. The last variable in the test data is the *classe* response variable, and the last variable in the training data corresponds to the 20 test values. Our goal is to use the remaining variables to predict an appropriate response for each entry.

Note that several features appear to contain mostly NA values, no values at all, or are not relevant to the analysis (such as the user and time variables). Thus, we select features to remove.

```
#Replace missing values with NA
TrainData[TrainData==""]=NA
cs=colSums(is.na(TrainData))
cs
```

```
##                        X                user_name     raw_timestamp_part_1
##                        0                        0                        0
##     raw_timestamp_part_2           cvtd_timestamp               new_window
##                        0                        0                        0
##               num_window               roll_belt              pitch_belt
##                        0                        0                        0
##                yaw_belt        total_accel_belt       kurtosis_roll_belt
```

```
##                        0                        0                    19216
##      kurtosis_picth_belt       kurtosis_yaw_belt      skewness_roll_belt
##                    19216                    19216                    19216
##     skewness_roll_belt.1       skewness_yaw_belt           max_roll_belt
##                    19216                    19216                    19216
##          max_picth_belt            max_yaw_belt           min_roll_belt
##                    19216                    19216                    19216
##          min_pitch_belt            min_yaw_belt     amplitude_roll_belt
##                    19216                    19216                    19216
##     amplitude_pitch_belt      amplitude_yaw_belt    var_total_accel_belt
##                    19216                    19216                    19216
##            avg_roll_belt         stddev_roll_belt           var_roll_belt
##                    19216                    19216                    19216
##          avg_pitch_belt       stddev_pitch_belt          var_pitch_belt
##                    19216                    19216                    19216
##            avg_yaw_belt         stddev_yaw_belt           var_yaw_belt
##                    19216                    19216                    19216
##            gyros_belt_x             gyros_belt_y           gyros_belt_z
##                       0                        0                       0
##            accel_belt_x             accel_belt_y           accel_belt_z
##                       0                        0                       0
##           magnet_belt_x            magnet_belt_y          magnet_belt_z
##                       0                        0                       0
##                roll_arm                pitch_arm                yaw_arm
##                       0                        0                       0
##          total_accel_arm            var_accel_arm           avg_roll_arm
##                       0                    19216                    19216
##          stddev_roll_arm             var_roll_arm           avg_pitch_arm
##                    19216                    19216                    19216
##         stddev_pitch_arm            var_pitch_arm             avg_yaw_arm
##                    19216                    19216                    19216
##           stddev_yaw_arm             var_yaw_arm             gyros_arm_x
##                    19216                    19216                       0
##             gyros_arm_y              gyros_arm_z             accel_arm_x
##                       0                        0                       0
##             accel_arm_y              accel_arm_z            magnet_arm_x
##                       0                        0                       0
##            magnet_arm_y             magnet_arm_z       kurtosis_roll_arm
##                       0                        0                   19216
##       kurtosis_picth_arm         kurtosis_yaw_arm       skewness_roll_arm
##                    19216                    19216                    19216
##       skewness_pitch_arm         skewness_yaw_arm            max_roll_arm
##                    19216                    19216                    19216
##            max_picth_arm             max_yaw_arm            min_roll_arm
##                    19216                    19216                    19216
##            min_pitch_arm             min_yaw_arm      amplitude_roll_arm
##                    19216                    19216                    19216
##      amplitude_pitch_arm       amplitude_yaw_arm           roll_dumbbell
##                    19216                    19216                       0
##           pitch_dumbbell             yaw_dumbbell   kurtosis_roll_dumbbell
##                       0                        0                   19216
##  kurtosis_picth_dumbbell   kurtosis_yaw_dumbbell   skewness_roll_dumbbell
##                    19216                    19216                    19216
##  skewness_pitch_dumbbell   skewness_yaw_dumbbell       max_roll_dumbbell
```

```
##                    19216                    19216                    19216
##       max_picth_dumbbell         max_yaw_dumbbell        min_roll_dumbbell
##                    19216                    19216                    19216
##       min_pitch_dumbbell         min_yaw_dumbbell  amplitude_roll_dumbbell
##                    19216                    19216                    19216
## amplitude_pitch_dumbbell   amplitude_yaw_dumbbell     total_accel_dumbbell
##                    19216                    19216                        0
##       var_accel_dumbbell        avg_roll_dumbbell     stddev_roll_dumbbell
##                    19216                    19216                    19216
##        var_roll_dumbbell       avg_pitch_dumbbell    stddev_pitch_dumbbell
##                    19216                    19216                    19216
##       var_pitch_dumbbell         avg_yaw_dumbbell      stddev_yaw_dumbbell
##                    19216                    19216                    19216
##         var_yaw_dumbbell         gyros_dumbbell_x         gyros_dumbbell_y
##                    19216                        0                        0
##         gyros_dumbbell_z         accel_dumbbell_x         accel_dumbbell_y
##                        0                        0                        0
##         accel_dumbbell_z        magnet_dumbbell_x        magnet_dumbbell_y
##                        0                        0                        0
##        magnet_dumbbell_z             roll_forearm            pitch_forearm
##                        0                        0                        0
##              yaw_forearm    kurtosis_roll_forearm   kurtosis_picth_forearm
##                        0                    19216                    19216
##     kurtosis_yaw_forearm    skewness_roll_forearm   skewness_pitch_forearm
##                    19216                    19216                    19216
##     skewness_yaw_forearm         max_roll_forearm        max_picth_forearm
##                    19216                    19216                    19216
##          max_yaw_forearm         min_roll_forearm        min_pitch_forearm
##                    19216                    19216                    19216
##          min_yaw_forearm   amplitude_roll_forearm  amplitude_pitch_forearm
##                    19216                    19216                    19216
##    amplitude_yaw_forearm      total_accel_forearm        var_accel_forearm
##                    19216                        0                    19216
##         avg_roll_forearm      stddev_roll_forearm         var_roll_forearm
##                    19216                    19216                    19216
##        avg_pitch_forearm     stddev_pitch_forearm        var_pitch_forearm
##                    19216                    19216                    19216
##          avg_yaw_forearm       stddev_yaw_forearm          var_yaw_forearm
##                    19216                    19216                    19216
##          gyros_forearm_x          gyros_forearm_y          gyros_forearm_z
##                        0                        0                        0
##          accel_forearm_x          accel_forearm_y          accel_forearm_z
##                        0                        0                        0
##         magnet_forearm_x         magnet_forearm_y         magnet_forearm_z
##                        0                        0                        0
##                   classe
##                        0
```

```
#Notice features either appear to be mostly NAs or contain no NAs

#Remove columns with NA values
TrainData=TrainData[,cs==0]
TestData=TestData[,cs==0]
```

```
#Remove the First 7 columns as seem irrelevant
TrainData=TrainData[,8:60]
TestData=TestData[,8:60]

d=dim(TrainData)
```

After removing these features, there are now 53 variables on which to predict the response, classe, variable.

Model Fitting

Now, we will split the original training into a testing and training data set with 60% of the data and 40%, respectively. This will be used for cross validation of our model.

```
library(caret)
```

```
## Loading required package: lattice
## Loading required package: ggplot2
```

```
set.seed(62433)
##Partition training set into train/test
inTrain = createDataPartition(TrainData$classe, p = .6)[[1]]
training = TrainData[ inTrain,]
testing = TrainData[-inTrain,]
```

We propose to fit our data to three different models using the methods of random forest, boosting, and using a stacked model of the two. We first consider the two individual methods.

```
#Random Forest
modfitRF <- train(classe~. , data = training , method = "rf")#,trControl=trainControl(method="cv",numbe
```

```
## Loading required package: randomForest
## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug fixes.
```

```
modfitRF
```

```
## Random Forest
##
## 11776 samples
##    52 predictors
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
##
## Summary of sample sizes: 11776, 11776, 11776, 11776, 11776, 11776, ...
##
## Resampling results across tuning parameters:
##
##   mtry  Accuracy  Kappa  Accuracy SD  Kappa SD
```

```
##   2     1          1      0.002        0.003
##   30    1          1      0.003        0.003
##   50    1          1      0.003        0.004
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 2.
```

```
#Boost
modfitGBM <- train(classe~. , data = training, method = "gbm",verbose=FALSE)#,trControl=trainControl(me
```

```
## Loading required package: gbm
## Loading required package: survival
## Loading required package: splines
##
## Attaching package: 'survival'
##
## The following object is masked from 'package:caret':
##
##     cluster
##
## Loading required package: parallel
## Loaded gbm 2.1
## Loading required package: plyr
```

```
modfitGBM
```

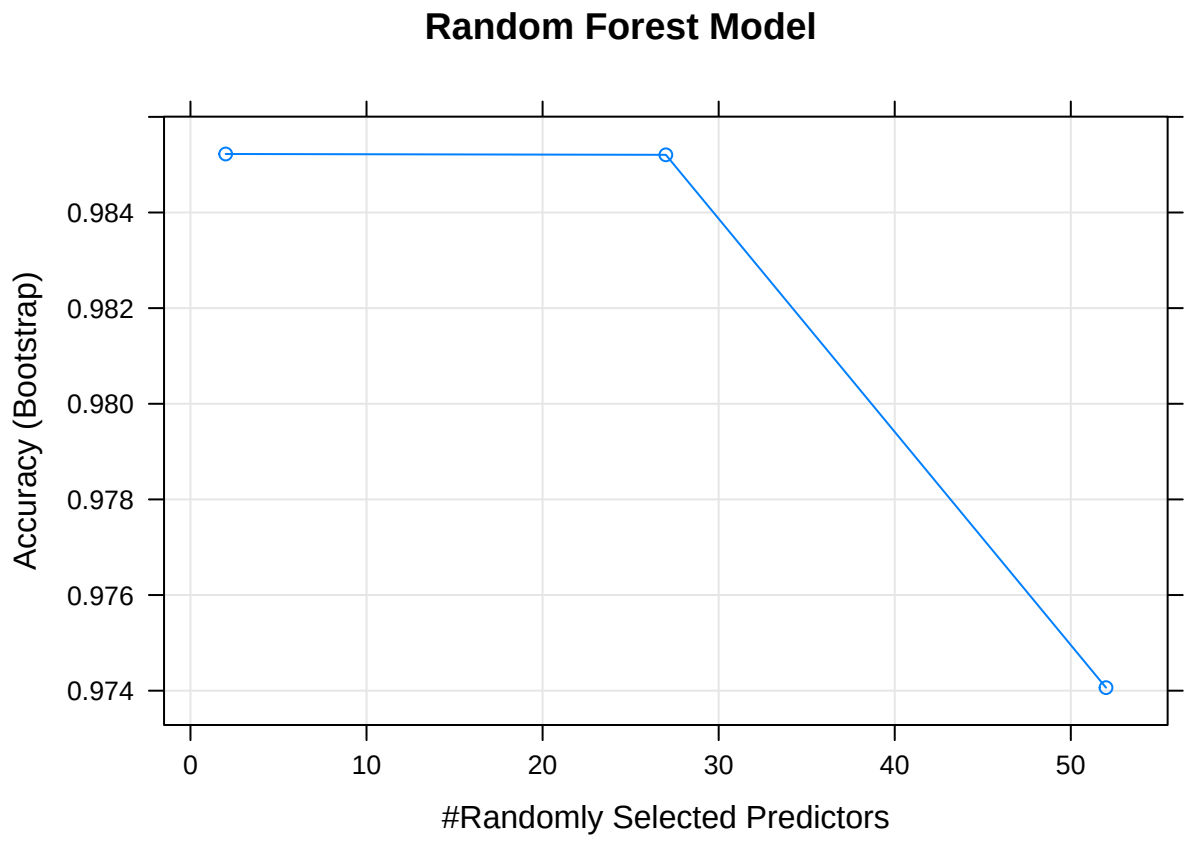
```
## Stochastic Gradient Boosting
##
## 11776 samples
##    52 predictors
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
##
## Summary of sample sizes: 11776, 11776, 11776, 11776, 11776, 11776, ...
##
## Resampling results across tuning parameters:
##
##   interaction.depth  n.trees  Accuracy  Kappa  Accuracy SD  Kappa SD
##   1                  50       0.8       0.7    0.007        0.009
##   1                  100      0.8       0.8    0.006        0.007
##   1                  200      0.9       0.8    0.005        0.006
##   2                  50       0.9       0.8    0.006        0.007
##   2                  100      0.9       0.9    0.005        0.006
##   2                  200      0.9       0.9    0.005        0.006
##   3                  50       0.9       0.9    0.005        0.006
##   3                  100      0.9       0.9    0.004        0.006
##   3                  200      1         0.9    0.003        0.004
##
## Tuning parameter 'shrinkage' was held constant at a value of 0.1
## Accuracy was used to select the optimal model using  the largest value.
## The final values used for the model were n.trees = 150,
##  interaction.depth = 3 and shrinkage = 0.1.
```

Cross Validation

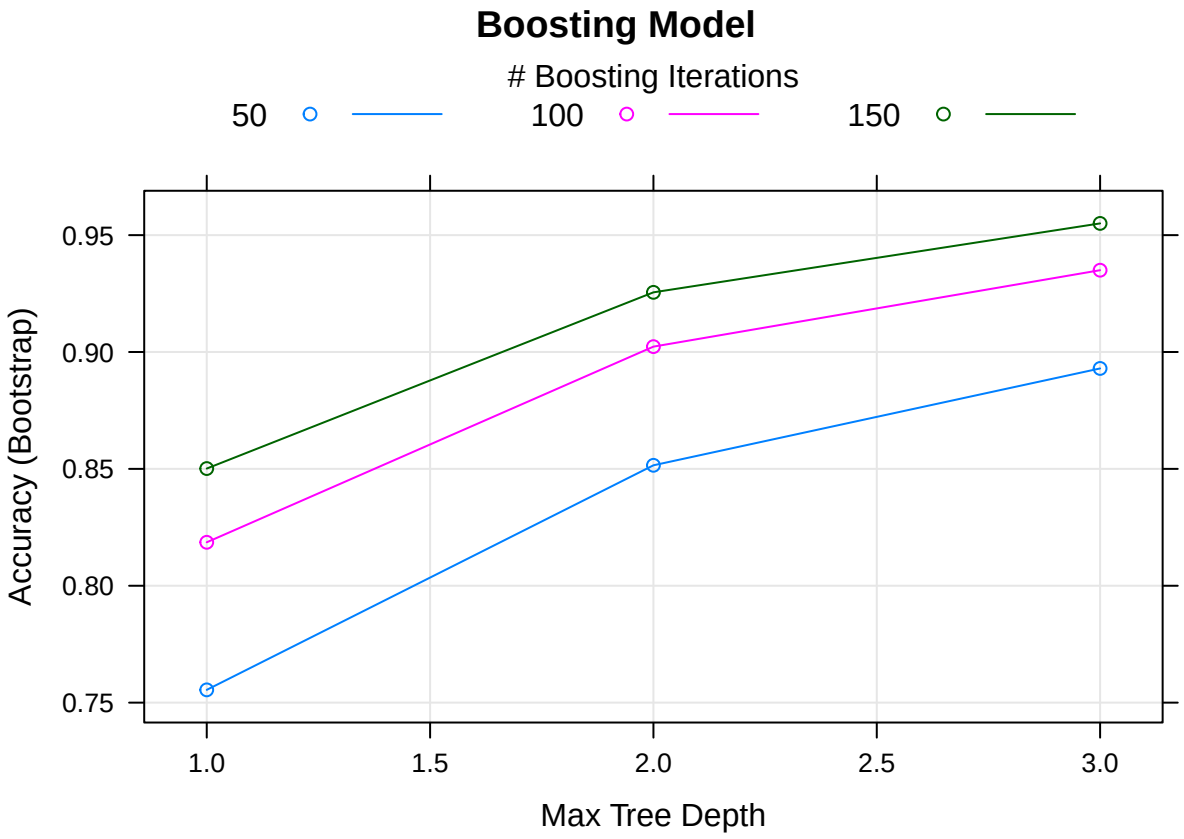
Now, we predict using our model on the cross validation set, compute confusion matrices, and report the accuracy i.e. the out of sample error.

```
# Use the models to predict the test set
predictRFtest <- predict(modfitRF, newdata = testing)
predictGBMtest <- predict(modfitGBM, newdata = testing)

#Plot of each model
plot(modfitRF, main="Random Forest Model")
```




```
plot(modfitGBM, main="Boosting Model")
```




```
#Compute confusion matrices
table(predictRFtest,testing$classe)
```

```
##
## predictRFtest    A    B    C    D    E
##             A 2229   14    1    0    0
##             B    3 1501   16    0    0
##             C    0    3 1349   32    1
##             D    0    0    2 1251    4
##             E    0    0    0    3 1437
```

```
table(predictGBMtest,testing$classe)
```

```
##
## predictGBMtest    A    B    C    D    E
##               A 2201   38    0    0    6
##               B   20 1446   61    5   11
##               C    7   32 1293   47   16
##               D    4    0   11 1231   18
##               E    0    2    3    3 1391
```

```
# Compute accuracy of the 2 models on the test set
RFAccuracy=table(predictRFtest == testing$classe)/length(testing$classe)
GBMAccuracy=table(predictGBMtest == testing$classe)/length(testing$classe)
```

Notice that our accuracy for the random forest model is 0.9899! The accuracy for the boosting model is not quite as good at 0.9638.

Model Fitting Using a Stacked Model

Next, we consider a stacked model of the two methods. We build a data frame of the predicted training data using the two models, then fit our stacked model to this new data frame.

```
#Compute the predicted training for the stacked model
predictRFtrain <- predict(modfitRF, newdata = training)
predictGBMtrain <- predict(modfitGBM, newdata = training)

# Use the predictions on the training set as 2 features for a stacked model
predictionDataFrame <- data.frame(varRF = predictRFtrain, varGBM = predictGBMtrain, classe = training$cl
modfitStacked <- train(classe~., method = "rf", data = predictionDataFrame)
```

Cross Validation on Stacked Model

A similar testing data frame is built in order to perform cross validation on the stacked model. We again compute the confusion matrix and the accuracy.

```
# Now build the test data frame containing the features for the testing set for the stacked model
testingStacked <- data.frame(varRF = predictRFtest, varGBM= predictGBMtest, classe = testing$classe)
dim(testingStacked)
```

```
## [1] 7846    3
```

```
predictStacked <- predict(modfitStacked, newdata = testingStacked)
# Get accuracy of stacked model on the test set

#Display table
table(predictStacked,testing$classe)
```

```
##
## predictStacked    A    B    C    D    E
##              A 2229   14    1    0    0
##              B    3 1501   16    0    0
##              C    0    3 1349   32    1
##              D    0    0    2 1251    4
##              E    0    0    0    3 1437
```

```
#Accuracy
StackedAccuracy=table(predictStacked == testing$classe)/length(testing$classe)
```

Notice that the accuracy for this stacked model is 0.9899. Since this accuracy is the same as the random forest model, we simply use the random forest model as it is a simpler model.

Applying Model to Test Cases

Finally, we use our random forest model to predict the 20 test cases and display the answers as below.

```
FinalTest=predict(modfitRF,newdata=TestData)
FinalTest
```

```
##  [1] 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
```

```
Answers=as.character(FinalTest)
```

Conclusion

In this report, we have used three different models (random forest, boosting, and a stacked model of the two) to quantify dumbbell curls of six individuals. We observe that all three methods perform quite well on our cross validation set, but choose the random forest model as the simplest model with highest accuracy.

Bibliography

[1] Velloso, E.; Bulling, A.; Gellersen, H.; Ugulino, W.; Fuks, H. Qualitative Activity Recognition of Weight Lifting Exercises. Proceedings of 4th International Conference in Cooperation with SIGCHI (Augmented Human '13) . Stuttgart, Germany: ACM SIGCHI, 2013.