Human Activity Recognition: Accuracy of Activities in Weight Lifting

Introduction

Considerable amount of research have been conducted by Universities and businesses on human wear devices for monitoring human's physical activities. A branch of the HAR research is focusing on accuracy of the intended activities, for example, daily routine activities of elderly people and athletics body movement during their regular exercise.

We will use the data set in http://groupware.les.inf.puc-rio.br/har to predict the accuracy by which the weight lifters are using dumbbell during their exercise. Six subjects are monitored while carrying out the unilateral dumbbell biceps curl in five different ways: 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). Class A corresponds to proper execution of the activity, whereas other four classes correspond to common mistakes. During these activities accelerometer attached to the subject arm, forearm, belt, and dumbbell are used to record the movement. We will build a prediction learning model from the data in the training set. The prediction model will be used to classify the observations in the test set.

Following packages and libraries are required: caret, Random Forest, AppliedPredictiveModeling.

```r
library(caret)
```

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

```r
library(randomForest)
```

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

```r
library(AppliedPredictiveModeling)
```

Data preparation

The data set are partitioned into two sets: Training and Testing, which are transferred from the above website to the working directory and then imported to the appropriate data frame.

```r
train <- read.csv2("pml-training.csv", header = TRUE, sep = ",")
test <- read.csv2("pml-testing.csv", header = TRUE, sep = ",")
for (i in ncol(train):1)
    if (sum(is.na(train[,i])) > 19000) {
        train[,i] <- NULL
        test[,i] <- NULL
    }
for (i in ncol(test):1) {
    s <- sum(is.na(test[,i]))
    if (s == nrow(test)) {
        test[,i] <- NULL
        train[,i] <- NULL
    }
}
for (i in 8:59) { train[,i] <- as.numeric(train[,i])
                  test[,i] <- as.numeric(test[,i])
}
table(train$user_name,train$classe)
```

```
##
##               A    B    C    D    E
##   adelmo   1165  776  750  515  686
##   carlitos  834  690  493  486  609
##   charles   899  745  539  642  711
##   eurico    865  592  489  582  542
##   jeremy   1177  489  652  522  562
##   pedro     640  505  499  469  497
```

```
train <- train[,-c(1:7)]
test <- test[,-c(1:7)]
```

Predictors with large number of NA (> 19000) are removed from both data sets and the remaining ones except classe in train and problem_id in test, are covnerted to numeric. The cross-table between user_name and classe in train set shows a uniform distribution of observations for each pair of class (classe and user_name) values. This is an indication that the model should be able to predict an outcome for every combination of class pairs. Highly correlated predictors are also removed from the data sets. There were no duplicate observation in the training set.

```
Corr <- cor(train[,-53])
highCorr <- findCorrelation(Corr, 0.90)
strain <- train[, -highCorr]
stest <- test[, -highCorr]
```

The Model

Considering the number of predictors and the inherent dependencies among them, we used Random Forest to build the prediction model, which exhaustively generates multiple trees and selects the best tree model from the permutation of predictors, averaged over accuracy on the out-of-bag portion of the data.

```
set.seed(123)
rfmodel <- randomForest(classe~., data=strain, importance=T, ntree=500)
print(rfmodel)
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = strain, importance = T,      ntree = 500)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 0.37%
## Confusion matrix:
```

```
##      A    B    C    D    E class.error
## A 5576    4    0    0    0   0.0007168
## B    9 3780    8    0    0   0.0044772
## C    0   15 3406    1    0   0.0046756
## D    1    0   23 3189    3   0.0083955
## E    0    0    1    8 3598   0.0024951
```

```
predict(rfmodel, newdata=stest)
```

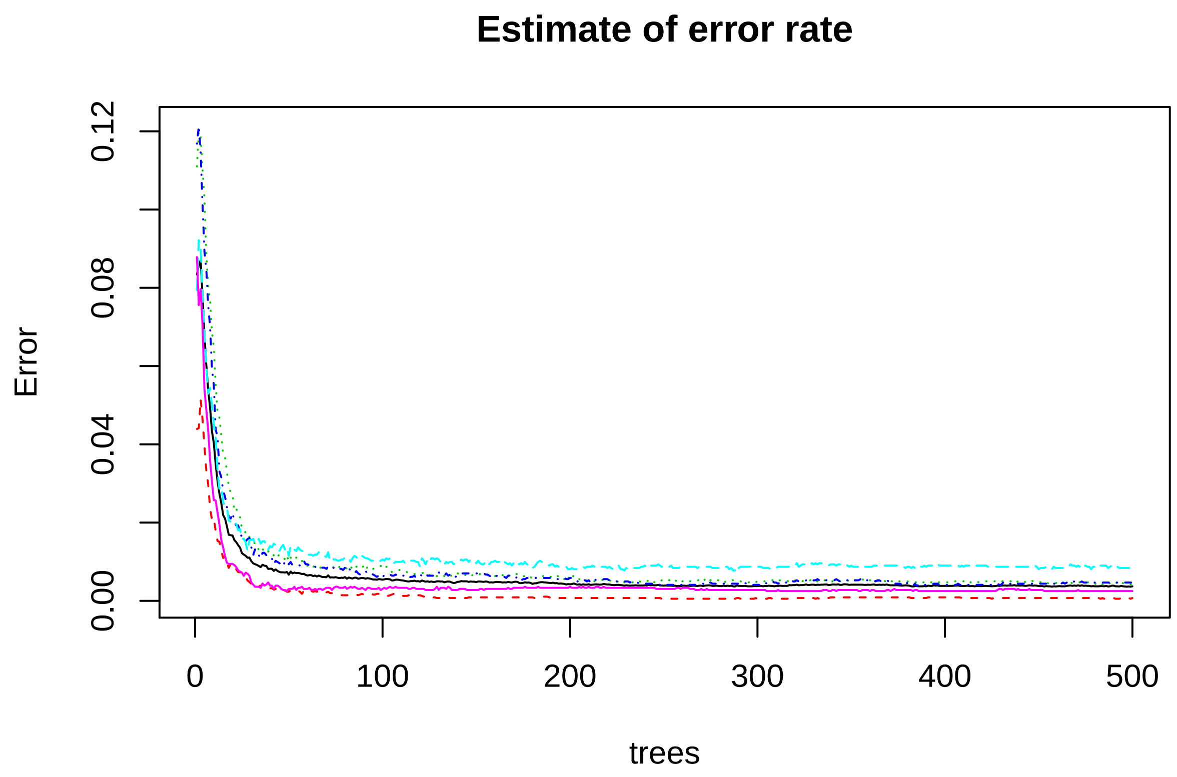
```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  B  A  A  A  A  E  E  E  A  A  A  E  A  A  E  A  A  D  B  B
## Levels: A B C D E
```

The estimate of error rate of 0.37% and near zero classification errors for each classe is an indication of a good fit, perhaps somewhat an overfit. The prediction from the test data is also displayed. We will look at the output of few statistics of this model.

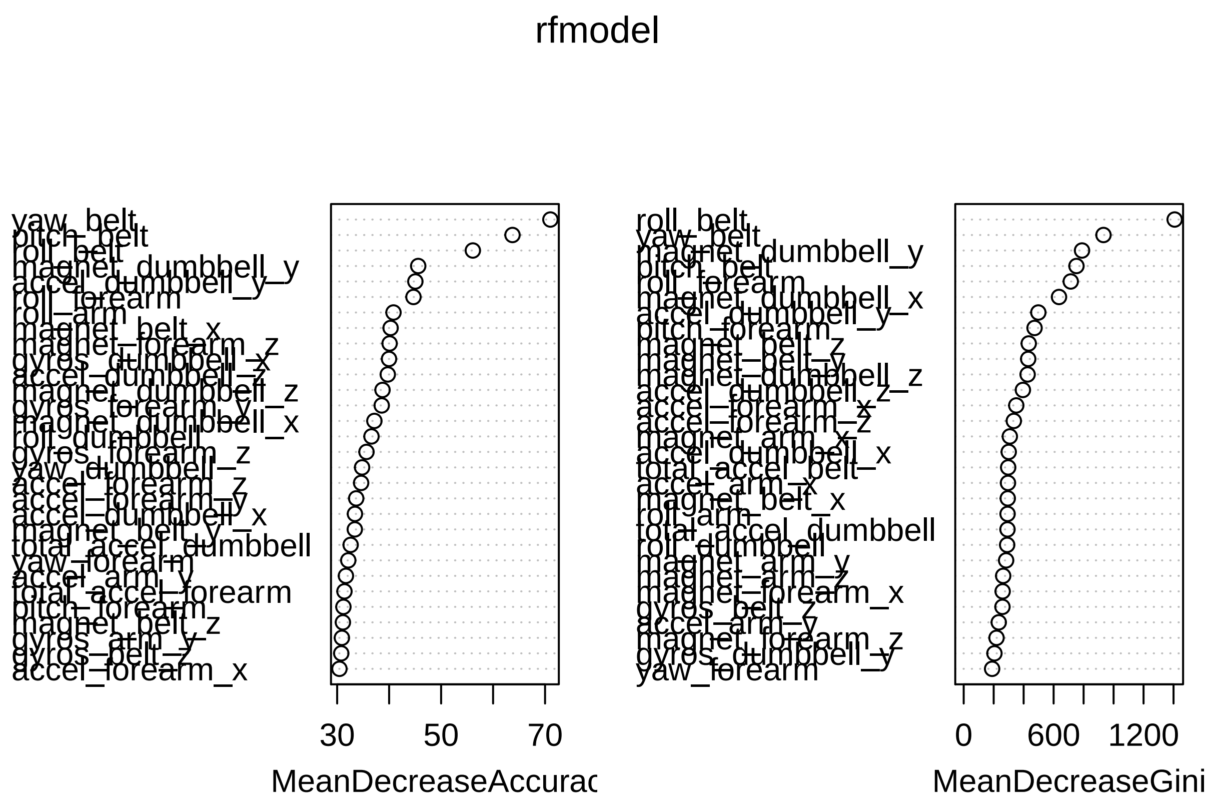
```
head(varImp(rfmodel, sort=T, scale=F))
```

```
##                        A        B       C       D        E
## roll_belt        0.10386 0.123599 0.14901 0.16466 0.238025
## pitch_belt       0.09857 0.079791 0.16154 0.15131 0.047394
## yaw_belt         0.09735 0.107848 0.12593 0.13824 0.090971
## total_accel_belt 0.02514 0.029686 0.03575 0.03261 0.035702
## gyros_belt_x     0.01335 0.008935 0.01522 0.01208 0.003760
## gyros_belt_y     0.02241 0.020379 0.02637 0.01877 0.008315
```

```
plot(rfmodel, main="Estimate of error rate")
```



```r
varImpPlot(rfmodel, sort=T)
```



```r
cv <- rfcv(stest[,-1],stest[,1],cv.fold=10)
cv$error.cv
```

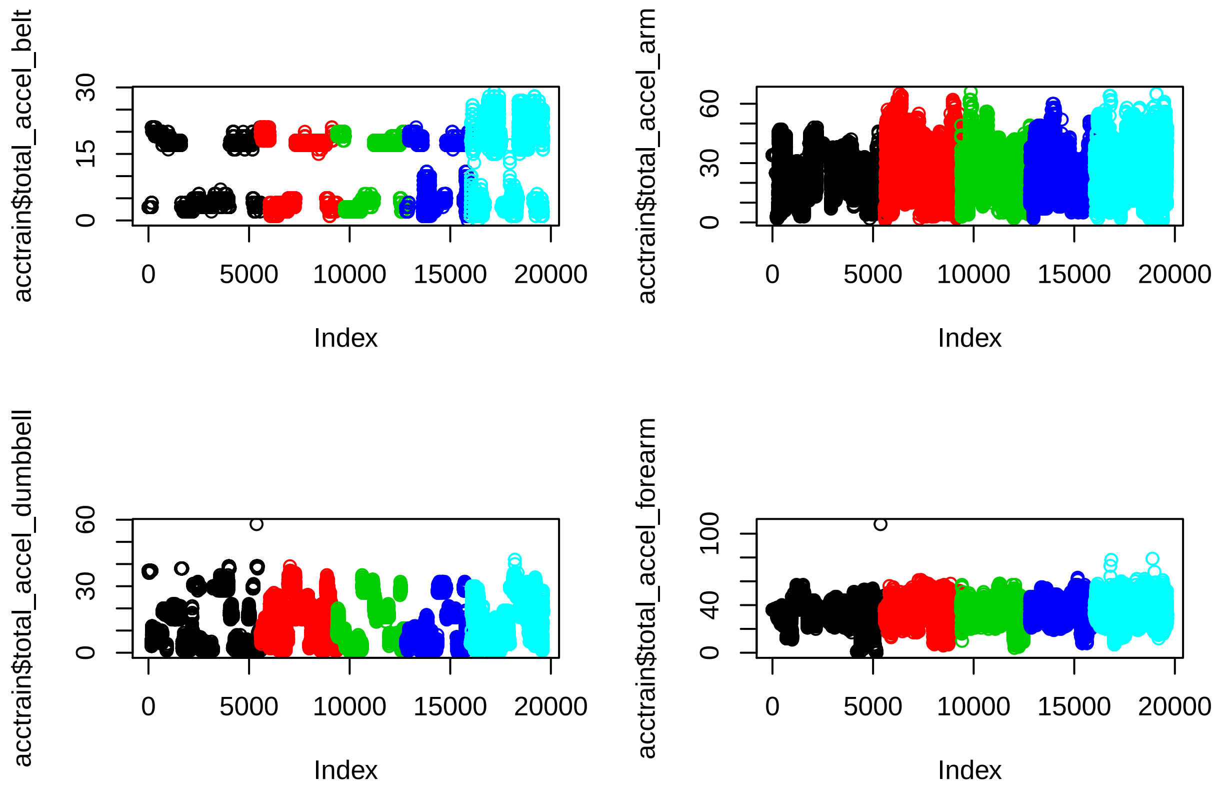
```
##    50    25    12     6     3     1
## 24.07 25.37 28.09 28.56 28.12 27.01
```

The plot of estimate error shows the errors for each class, trending down as the number of trees grow and the plot of varImp displays the contribution of the predictors to the model. Results of cross-validation shows the error for the number of variables used in each step of building the model.

Refining the Model

The above random forest model, though has a small miss-classification error did not reflect a uniform prediction of all classe (e.g. no prediction for Class C and more than 50% of the test observation are of Class A, correct execution of dumbbell exercise) of test data. To explore for better model, we used the list of important predictors and divided to different groups of accelerator, Gyroscope, and Mag. and for each group or combination of them generated a random forest model and measured the estimated error. It appeared that the predictors of accelerometers has the minimal error rate and fully predicted all classes in the test data.

```r
acctrain <- train[,c(53,4,8:10,17,21:23,30,34:36,43,47:49)]
acctest <- test[,c(53,4,8:10,17,21:23,30,34:36,43,47:49)]
par(mfrow=c(2,2))
plot(acctrain$total_accel_belt,col=acctrain$classe)
plot(acctrain$total_accel_arm,col=acctrain$classe)
plot(acctrain$total_accel_dumbbell,col=acctrain$classe)
plot(acctrain$total_accel_forearm,col=acctrain$classe)
```

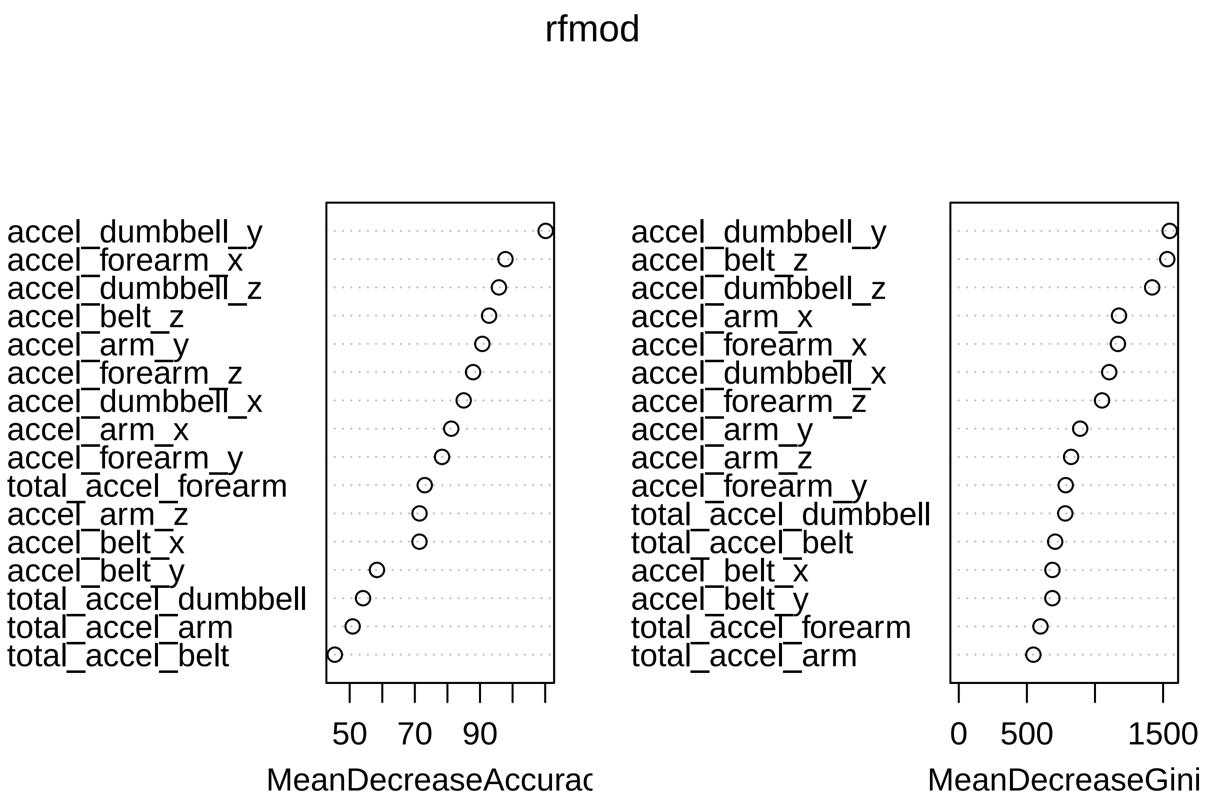


Plot of four predictors shown above depicts a clear separation of the data for total acceleration.

```r
rfmod <- randomForest(classe~., data=acctrain, importance=T, ntree=500)
print(rfmod)
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = acctrain, importance = T,      ntree = 500)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 4
##
##         OOB estimate of  error rate: 4.31%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 5414   29   63   72    2     0.02975
## B  106 3558   86   21   26     0.06294
## C   51   70 3266   30    5     0.04559
## D   58   16  110 3017   15     0.06188
## E    5   37   17   26 3522     0.02357
```

```
varImpPlot(rfmod,sort=T)
```



```
predict(rfmod,newdata=acctest, type="class")
```

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

Summary

As it appears to build the best random forest model with high accuracy we need all the quantitative predictors, giving ~100% accuracy (miss-classifications error of 0.37%). To predict the data in the test set we need a model that can predict for all classes (A, B, C, D, E), for which the initial random forest model did not accomplish. However, this model divided the training data into two groups with proper use of dumbbell (28%) and without proper use of the dumbbell (78%). we felt that the model has overfit the training data and is not providing a good prediction for the test data.

Using the importance() variables we narrowed down the model to the set of acceleration predictors and produced a random forest model for validation of the test data. This model exhibited a 4% miss-classifications error but gave an accurate prediction for the test data.