Final Project PML

Arturo Estrada

17 de marzo de 2017

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.

Instructions

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. The goal of this project is to predict the manner in which they did the exercise. This is the "classe" variable in the training set. You may use any of the other variables to predict with. You should create a report describing how you built your model, how you used cross validation, what you think the expected out of sample error is, and why you made the choices you did. You will also use your prediction model to predict 20 different test cases.

Data

```
train <- read.csv("C:/Users/Arturo/Documents/Data Science Specialization/Machine Learning/pml-training.
test <- read.csv("C:/Users/Arturo/Documents/Data Science Specialization/Machine Learning/pml-testing.cs
```

Cleaning Data

```
##Delete columns with more than 60% of NAs
train <- train[ ,colSums(is.na(train)) <= 0.6*nrow(train)]
test <- test[ ,colSums(is.na(test)) <= 0.6*nrow(test)]

##Delete columns not relevant as covariates
train <- train[ ,-c(1:7)];test <- test[ ,-c(1:7)]
dim(train)
```

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

Spliting Training Data Set

As Training data set is large, cross-validation will be carried out by splitting the original training set into 2 subsets (i.e. randomly without replacement). First Set (60%) and second set (40%).

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.3.2
## Loading required package: lattice
```

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

```
## Warning: package 'ggplot2' was built under R version 3.3.2
```

```
In_train_s1 <- createDataPartition(train$classe, p = 0.6, list = F)
train_s1 <- train[In_train_s1, ]; train_s2 <- train[-In_train_s1, ]
dim(train_s1); dim(train_s2)
```

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

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

Fitting Models

First, a CART model is fitted

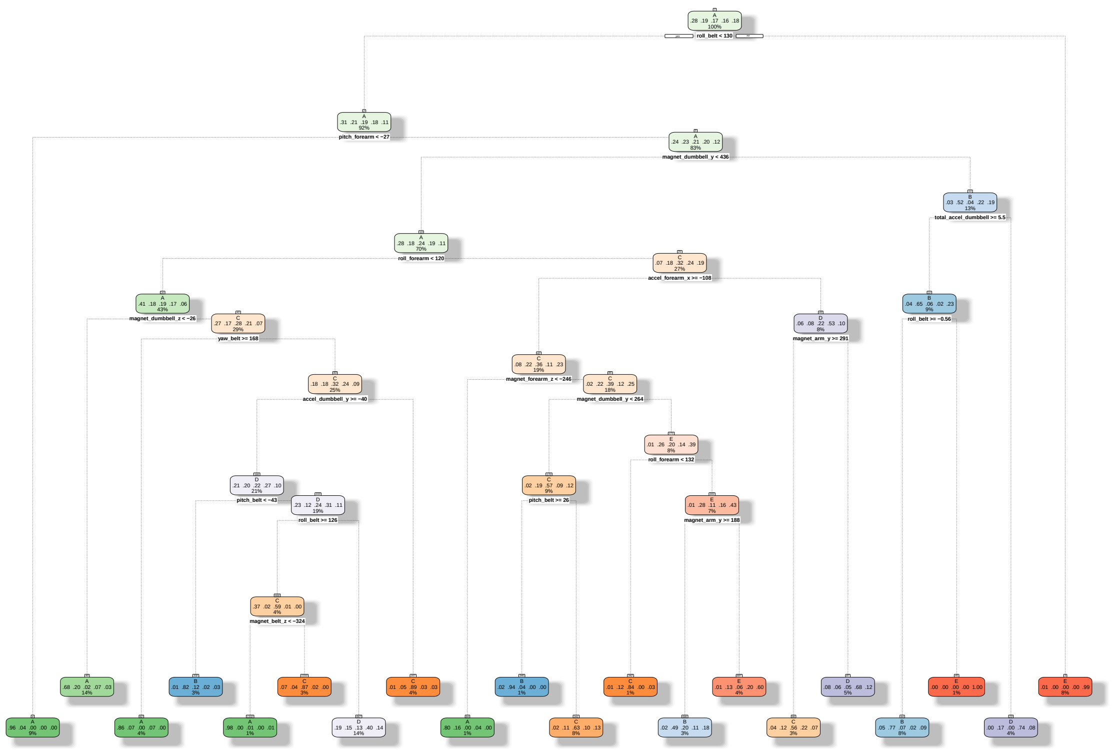
```
library(rattle)
```

```
## Warning: package 'rattle' was built under R version 3.3.3
```

```
## Rattle: A free graphical interface for data mining with R.
## Versión 4.1.0 Copyright (c) 2006-2015 Togaware Pty Ltd.
## Escriba 'rattle()' para agitar, sacudir y  rotar sus datos.
```

```
library(rpart)
mod1 <- rpart(classe ~., method = "class", data = train_s1)
fancyRpartPlot(mod1)
```

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



Rattle 2017−mar.−17 14:04:42 Arturo

```
Mod1_pred <- predict(mod1, train_s2, type = "class")
confusionMatrix(train_s2$classe, Mod1_pred)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1866   63   63  227   13
##          B  340  792  115  234   37
##          C   28  101 1030  183   26
##          D  110   50  103  944   79
##          E   37   84  114  249  958
##
## Overall Statistics
##
##                Accuracy : 0.7125
##                  95% CI : (0.7023, 0.7225)
##     No Information Rate : 0.3035
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6363
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.7837   0.7266   0.7228   0.5139   0.8607
## Specificity            0.9330   0.8925   0.9474   0.9431   0.9281
## Pos Pred Value         0.8360   0.5217   0.7529   0.7341   0.6644
## Neg Pred Value         0.9083   0.9529   0.9390   0.8639   0.9758
## Prevalence             0.3035   0.1389   0.1816   0.2341   0.1419
## Detection Rate         0.2378   0.1009   0.1313   0.1203   0.1221
## Detection Prevalence   0.2845   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      0.8584   0.8096   0.8351   0.7285   0.8944
```

A random Forest Model is fitted.

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.3.3
```

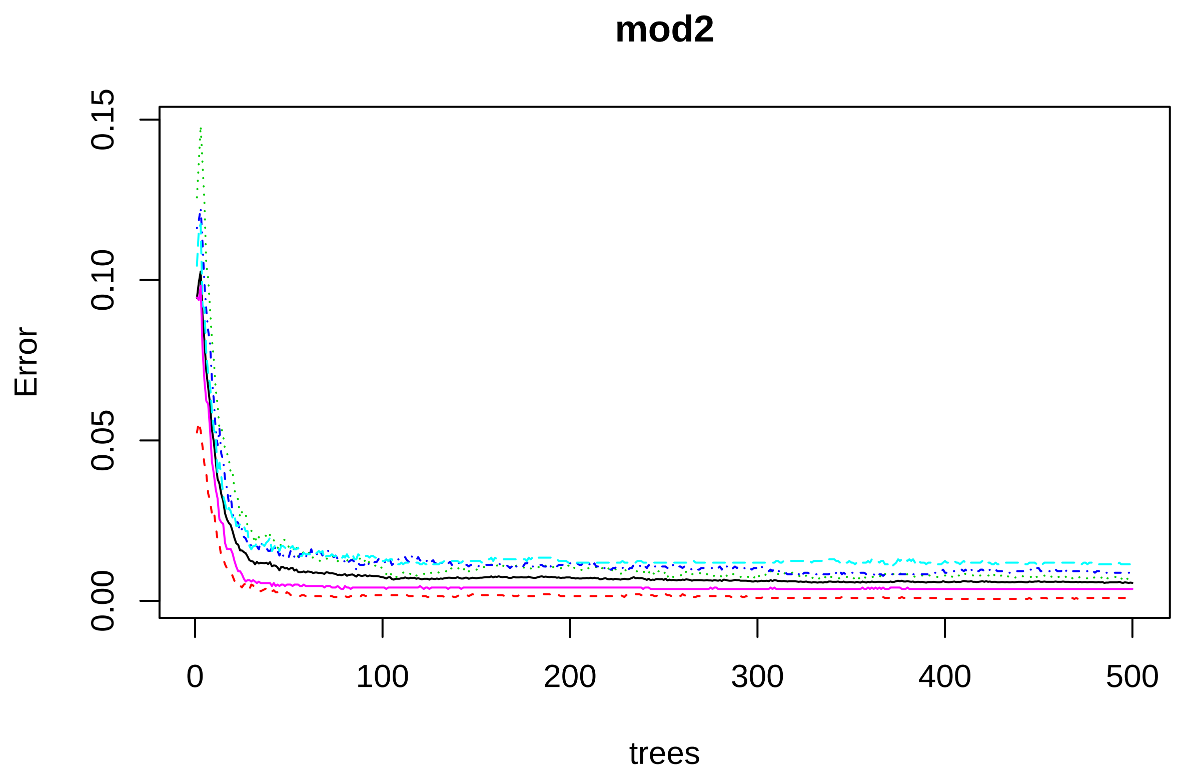
```
## randomForest 4.6-12
```

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

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
mod2 <- randomForest(classe ~., data = train_s1)
Mod2_pred <- predict(mod2, train_s2)
plot(mod2)
```



```r
confusionMatrix(Mod2_pred, train_s2$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2230   10    0    0    0
##          B    0 1501   10    0    0
##          C    2    7 1357    9    1
##          D    0    0    1 1277    7
##          E    0    0    0    0 1434
##
## Overall Statistics
##
##                Accuracy : 0.994
##                  95% CI : (0.992, 0.9956)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9924
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9991   0.9888   0.9920   0.9930   0.9945
## Specificity            0.9982   0.9984   0.9971   0.9988   1.0000
## Pos Pred Value         0.9955   0.9934   0.9862   0.9938   1.0000
## Neg Pred Value         0.9996   0.9973   0.9983   0.9986   0.9988
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2842   0.1913   0.1730   0.1628   0.1828
## Detection Prevalence   0.2855   0.1926   0.1754   0.1638   0.1828
## Balanced Accuracy      0.9987   0.9936   0.9945   0.9959   0.9972
```

Prediction on Test Data

Random Forests gave an Accuracy in the test_s2 dataset of 99.34%, which was more accurate than CART model. The expected out-of-sample error is 100-99.34 = 0.66%.

```r
pred_Test <- predict(mod2, test)
pred_Test
```

```
##  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
```