Prediction Assignment Writeup

Amadou Barry

5/2/2020

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, 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://groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

Data

The training data for this project are available here: [https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv]

The test data are available here: [https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv]

The data for this project come from this source: [http://groupware.les.inf.puc-rio.br/har]. If you use the document you create for this class for any purpose please cite them as they have been very generous in allowing their data to be used for this kind of assignment.

Project Goal The goal of your 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.

Your submission should consist of a link to a Github repo with your R markdown and compiled HTML file describing your analysis. Please constrain the text of the writeup to < 2000 words and the number of figures to be less than 5. It will make it easier for the graders if you submit a repo with a gh-pages branch so the HTML page can be viewed online (and you always want to make it easy on graders :-). You should also apply your machine learning algorithm to the 20 test cases available in the test data above. Please submit your predictions in appropriate format to the programming assignment for automated grading. See the programming assignment for additional details.

Approach

Our outcome variable is 'classe', which is a factor variable that describe how well the candidates perfomed. For this data set, participants were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in 5 different fashions: - 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) - throwing the hips to the front (Class E).

Two models will be tested using decision tree and random forest. The model with the highest accuracy with the cross validation technique will be chosen as our final model, also the expected out-of sample error will be determined.

Reading the training and testing data set

```
# Reading the data and replacing all the missing data with NA
train <- read.csv("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv", na.strings=c(
test <- read.csv("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv", na.strings=c("NA
dim(train); dim(test)
```

```
## [1] 19622   160
```

```
## [1]  20 160
```

Preprocessing the data

Let's remove some obvious useless variables. These variables are: "X", "user_name", "raw_timestamp_part_1" "raw_timestamp_part_2" "cvtd_timestamp" and "new_window"

```
#Removing some variables from the training and testing set
train <- subset(train, select = -c(1:7))   #1:6 correspond to those first 6 variables
test <- subset(test, select = -c(1:7))
# deleting the last test colum "problem_id"
test <- subset(test, select = -c(problem_id))
# Cheking the dimensions
dim(train); dim(test)
```

```
## [1] 19622   153
```

```
## [1]  20 152
```

Deleting colums with NA's values

```
train <- train[,colSums(is.na(train))==0]
test <- test[,colSums(is.na(test))==0]
 dim(train); dim(test)
```

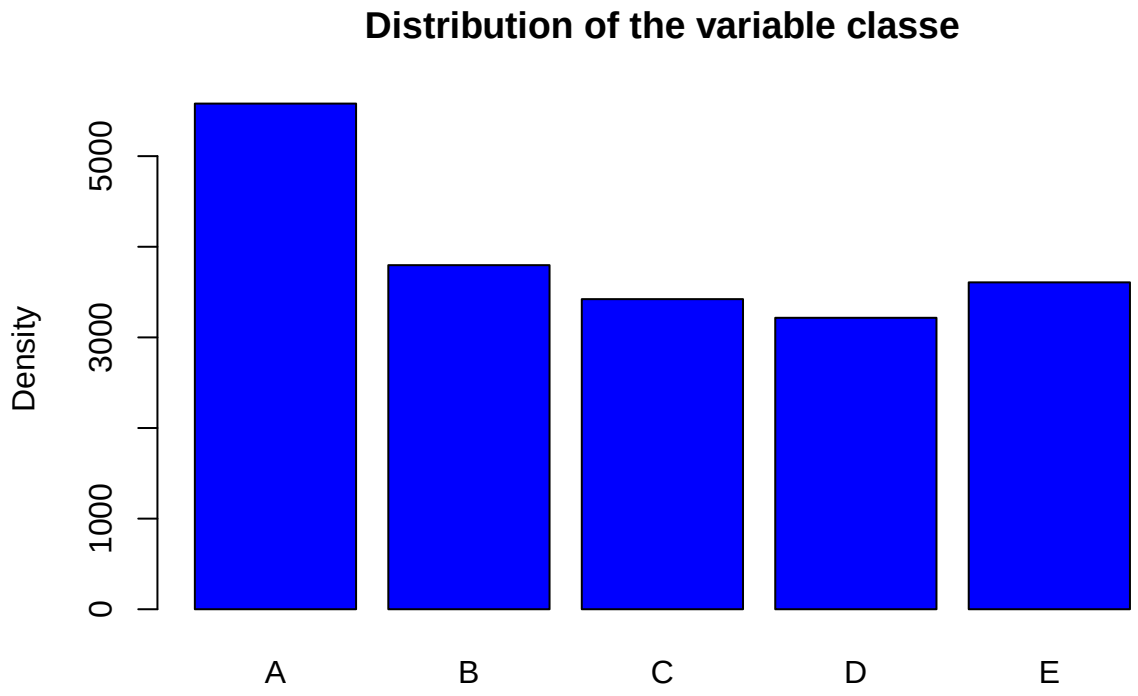
```
## [1] 19622    53
```

```
## [1] 20 52
```

Now we have only 53 colomns for our training set and 52 for our testing.

Plot of the train data according to the "classe" variable

```
plot(train$classe, col="blue", ylab= "Density", main="Distribution of the variable classe")
```



Libraries

```
set.seed(1234)
library(caret)
library(lattice)
library(ggplot2)
library(dplyr)
library(rpart)
library(randomForest)
library(rattle)
```

Partition of the data set into subTrain and subTest to allow cross validation later on

```
inTrain <- createDataPartition(y=train$classe, p=0.95, list = FALSE)
subTrain <- train[inTrain,]
subTest <- train[-inTrain,]
```

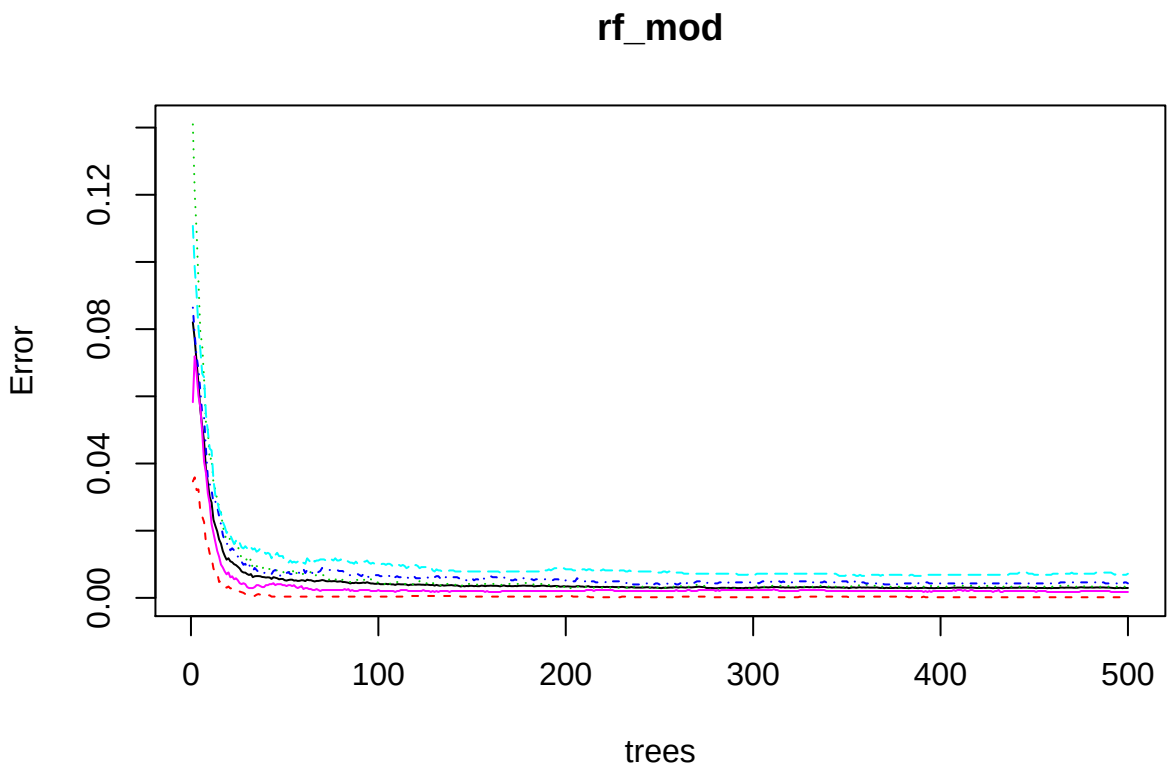
Models

Training model using random forest

```
rf_mod <- randomForest(classe ~., data= subTrain)
rf_mod
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = subTrain)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 0.3%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 5300    1    0    0    0 0.0001886437
## B    9 3596    3    0    0 0.0033259424
## C    0   11 3237    3    0 0.0043063673
## D    0    0   20 3034    2 0.0071989529
## E    0    0    1    5 3421 0.0017508025
```

```
plot(rf_mod)
```



Confusion matrix for random forest

```
# Prediction using random Forest model
rf_pre <- predict(rf_mod, newdata = subTest)
# confusion matrix
rf_confMX <- confusionMatrix(rf_pre, subTest$classe)
rf_confMX
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   A   B   C   D   E
##          A 279   1   0   0   0
##          B   0 188   0   0   0
##          C   0   0 171   2   0
##          D   0   0   0 157   0
##          E   0   0   0   1 180
##
## Overall Statistics
##
##                Accuracy : 0.9959
##                  95% CI : (0.9896, 0.9989)
##     No Information Rate : 0.285
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9948
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9947   1.0000   0.9812   1.0000
## Specificity            0.9986   1.0000   0.9975   1.0000   0.9987
## Pos Pred Value         0.9964   1.0000   0.9884   1.0000   0.9945
## Neg Pred Value         1.0000   0.9987   1.0000   0.9964   1.0000
## Prevalence             0.2850   0.1931   0.1747   0.1634   0.1839
## Detection Rate         0.2850   0.1920   0.1747   0.1604   0.1839
## Detection Prevalence   0.2860   0.1920   0.1767   0.1604   0.1849
## Balanced Accuracy      0.9993   0.9974   0.9988   0.9906   0.9994
```

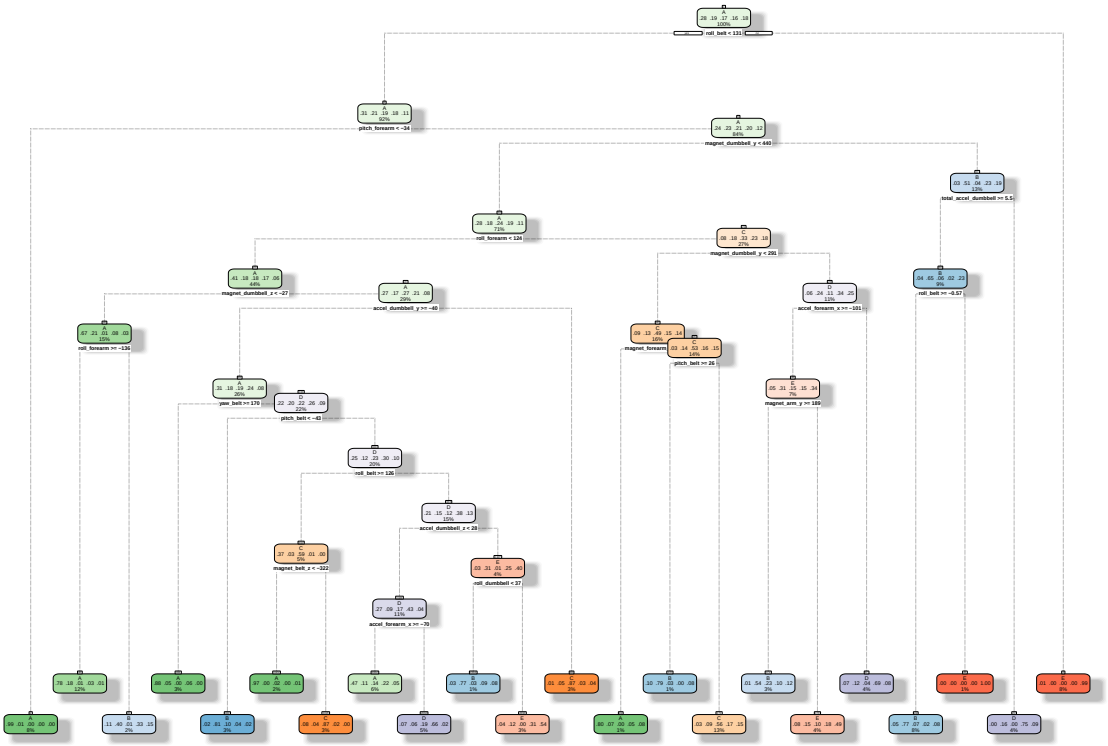
Our random forest model has 99.6% accuracy which is pretty good.

Training model using Decision Trees

```
dt_mod <- rpart(classe~., data = subTrain, method ="class") # method= "class" cause y is a factor
# fancyplot of the model

fancyRpartPlot(dt_mod)
```

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



Confusion matrix for Decision Trees

```
# prediction for the decision trees model
dt_pre <- predict(dt_mod, newdata= subTest, type="class")
# Confusion Matrix
dt_confMX <- confusionMatrix(dt_pre, subTest$classe)
dt_confMX
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   A   B   C   D   E
##          A 253  29  13  18   7
##          B  13 123  20  17  17
##          C   7  14 118  21  23
##          D   4  15  12  84  14
##          E   2   8   8  20 119
##
## Overall Statistics
##
##                Accuracy : 0.712
```

```
##                  95% CI : (0.6825, 0.7402)
##     No Information Rate : 0.285
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6334
##
##  Mcnemar's Test P-Value : 0.0001363
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9068   0.6508   0.6901   0.5250   0.6611
## Specificity            0.9043   0.9152   0.9196   0.9451   0.9524
## Pos Pred Value         0.7906   0.6474   0.6448   0.6512   0.7580
## Neg Pred Value         0.9605   0.9163   0.9334   0.9106   0.9258
## Prevalence             0.2850   0.1931   0.1747   0.1634   0.1839
## Detection Rate         0.2584   0.1256   0.1205   0.0858   0.1216
## Detection Prevalence   0.3269   0.1941   0.1869   0.1318   0.1604
## Balanced Accuracy      0.9055   0.7830   0.8048   0.7350   0.8068
```

The Decision Trees model has 71% accuracy on the test data.

Decision

With 99.6% accuracy our random forest model prediction is way better than than the Decision Trees model which is only about 71%. So obviously we choosing the random forest model to predict the "classe" of our test data from the begining. The expected out-of-sample error is estimated at 0.004, or 0.4%. The expected out-of-sample error is calculated as 1 - accuracy for predictions made against the cross-validation set.

Prediction of the random forest model on the test dataset

```
test_pred <- predict(rf_mod, test)
test_pred
```

```
##  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
```