# Machine Learning: Prediction

*Ben Straub*

*11/22/2017*

## 1 Project Overview

### 1.1 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://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

### 1.2 Objective

The goal of this project is to predict the manner in which particpants did the exercise. The report contains how I built the model, how you I did cross validation, what the expected out of sample error is, and why I made the choices you did. Finally, I will also use my final prediction model to predict 20 different test cases.

```
#useful packages
library(caret);library(randomForest);library(rpart);library(rpart.plot);
library(RColorBrewer);library(knitr)
#setting global chunks
knitr::opts_chunk$set(echo = TRUE)
knitr::opts_chunk$set(message = FALSE)
knitr::opts_chunk$set(warning = FALSE)
knitr::opts_chunk$set(comment = "")
knitr::opts_chunk$set(cache = TRUE)
```

### 1.3 Data

```
setwd("/Users/benStraub/Desktop/Coursera_Machine_Learning")
#Read in train and test sets
#data to build the model
train <- read.csv("pml-training.csv", na.strings=c("NA","#DIV/0!",""), header=TRUE)
#data for the final run
final_test <- read.csv("pml-testing.csv", na.strings=c("NA","#DIV/0!",""), header=TRUE)
```

I have split the provided training data into two data sets, one as a training set (60%) for the model and one as a testing set (40%) to assess the performance of the model. I use the createDataPartitiion() function from the caret package. The data is partitioned by the classe variable, which is the variable that will be used in the prediction.

```
#Split data into testing and training sets
inTrain <- caret::createDataPartition(y=train$classe, p = 0.60, list=FALSE)
training <- train[inTrain,]
testing <- train[-inTrain,]

dim(training)
```

```
[1] 11776   160
```

```
dim(testing)
```

```
[1] 7846  160
```

```
dim(final_test)
```

```
[1]  20 160
```

Training data has 11776 observation with 160 variables. Test data has 7846 observations with 160 variables. Final test data has 20 observations with 160 variables.

The classe variable is the response variabes. It contains 5 different ways barbell lifts were performed correctly and incorrectly:

The report "Qualitative Activity Recognition of Weight Lifting Exercises" has the classe variables defined as follows:

Participants were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in five 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) and throwing the hips to the front (Class E). Class A corresponds to the specified execution of the exercise, while the other 4 classes correspond to common mistakes.

```
table(train$classe)
```

```
   A    B    C    D    E
5580 3797 3422 3216 3607
```

Above is the distribtion of the response variable classee

### 1.4 Data PreProcessing

I removed timestamp, windows, names and extra id as they will not help in building the final model, i.e. information won't help increase prediction accuracy.

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

I removed any column that had na's greater than 90 percent as these are almost empty columns and will not help improve the model and reduce computation time.

```
#found on stackoverflow! so nice!
new_training <- training[, -which(colMeans(is.na(training)) > 0.9)]
#View(new_training)
dim(training)
```

```
[1] 11776   153
```

```
dim(new_training)
```

```
[1] 11776    53
```

The training data has been reduced from 160 variables to 53 variables.

## 2 Prediction

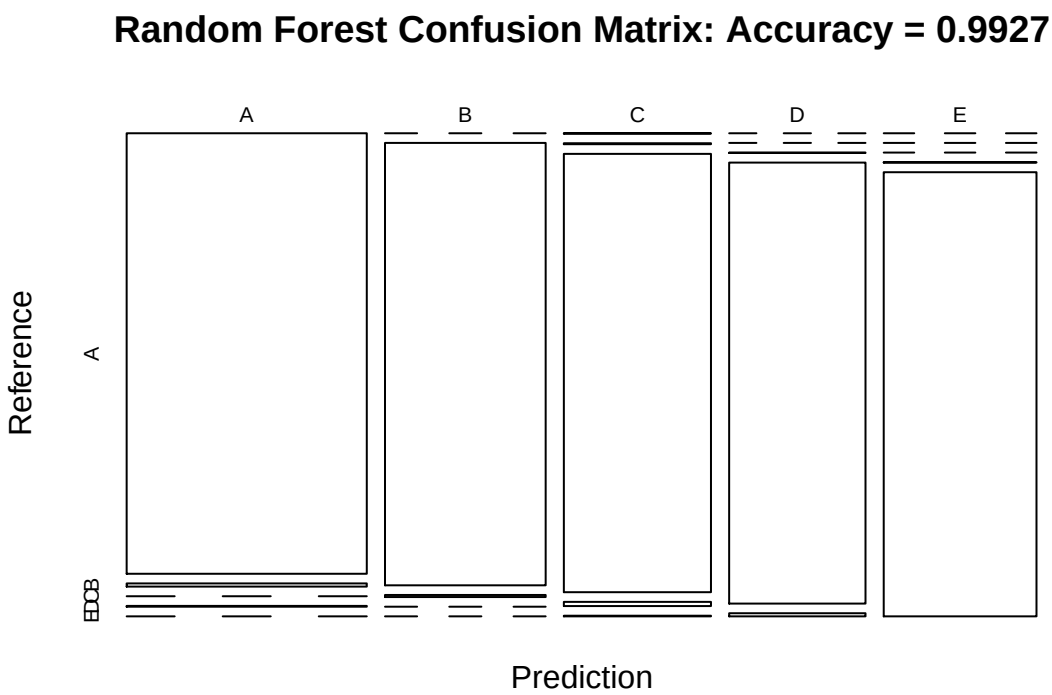
### 2.1 Prediction with Random Forest

```
set.seed(999)
model <- randomForest(classe ~ ., data=new_training)
prediction <- predict(model, testing)
cm <- caret::confusionMatrix(prediction, testing$classe)
kable(cm$table, caption="Confusion Table")
```

Table 1: Confusion Table

| | A | B | C | D | E |
|---|---|---|---|---|---|
| A | 2231 | 16 | 0 | 4 | 0 |
| B | 0 | 1499 | 7 | 0 | 0 |
| C | 1 | 3 | 1360 | 13 | 2 |
| D | 0 | 0 | 1 | 1268 | 9 |
| E | 0 | 0 | 0 | 1 | 1431 |

```
#kable(round(cm$byClass,3))
plot(cm$table, col = cm$byClass, main = paste("Random Forest Confusion Matrix: Accuracy =", round(cm$ov
```




```
overall.accuracy <- round(cm$overall['Accuracy'] * 100, 2)
sam.err <- round(1 - cm$overall['Accuracy'],2)
```

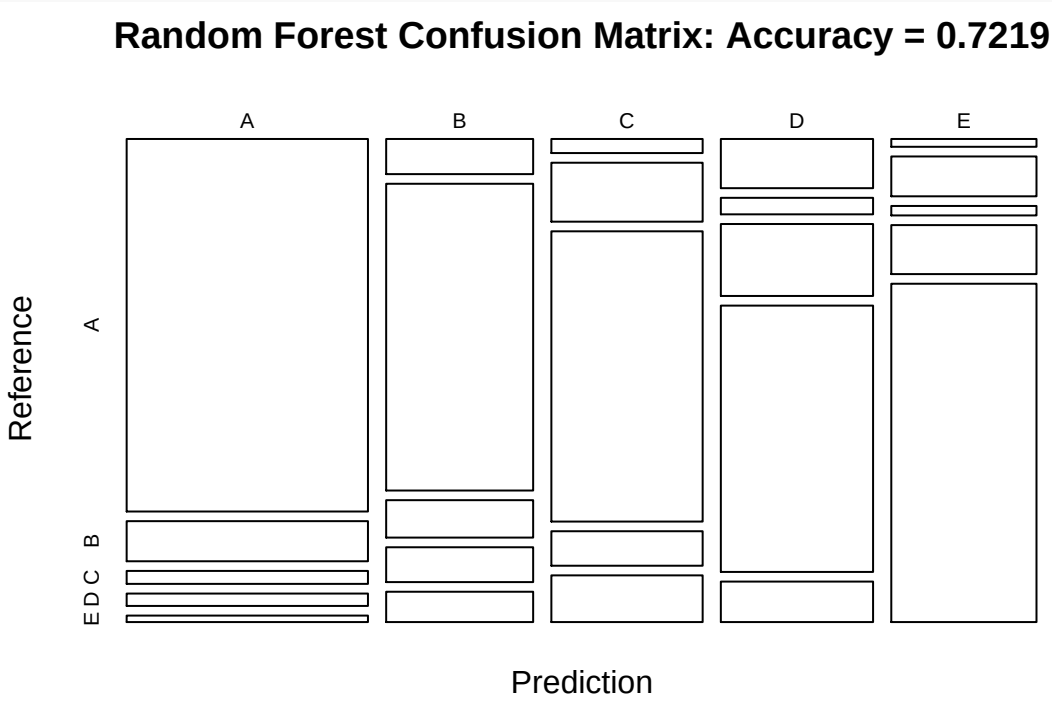
### 2.2 Prediction with Decision Trees

```
set.seed(999)
model2 <- rpart::rpart(classe ~ ., data=training, method="class")
prediction2 <- predict(model2, testing, type="class")
cm2 <- caret::confusionMatrix(prediction2, testing$classe)
kable(cm2$table, caption="Confusion Table")
```

Table 2: Confusion Table

| | A | B | C | D | E |
|---|---|---|---|---|---|
| A | 1896 | 204 | 66 | 63 | 33 |
| B | 109 | 951 | 116 | 108 | 94 |
| C | 45 | 188 | 926 | 110 | 149 |
| D | 158 | 53 | 231 | 855 | 130 |
| E | 24 | 122 | 29 | 150 | 1036 |

```
#kable(round(cm2$byClass,3))
plot(cm2$table, col = cm2$byClass, main = paste("Random Forest Confusion Matrix: Accuracy =", round(cm2
```




```
overall.accuracy2 <- round(cm2$overall['Accuracy'] * 100, 2)
sam.err2 <- round(1 - cm2$overall['Accuracy'],2)
```

### 2.3 Summary

Random forest performs much better than decision trees. The accuracy for the random forest technique is 99.27 with an out of sample error rate of 0.01 while the decision tree technique has an accuracy of 72.19 with an out of sample error rate of 0.28. Therefore, I will use random forest on the final 20 test set.

## 3 Final 20

```
final_20_prediction <- predict(model, final_test, type="class")
final_20_prediction
```

```
 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
```