Summary

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 analysis we're going to look at accelerometer data (belt, forearm, arm, and dumbell) of 6 different individuals performing the Unilateral Dumbbell Biceps Curl. They were asked to perform barbell lifts correctly and incorrectly in 5 different ways (exercise class). Each form of the barbell lift that was preformed was labeled as A,B,C,D,E depending on the way the barbell lift was preformed.

The goal is to build a model, using the accelerometer data, to predict the exercise class based on the data with high accuracy.

Data preparation and processing

Required packages are loaded, data are downloaded from the provided sources.

```
library(caret)
```

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

```
library(randomForest)
```

```
## randomForest 4.6-12
## Type rfNews() to see new features/changes/bug fixes.
```

```
library(corrplot)
```

```
download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv", destfile = "./pml-trainin
g.csv", method="curl")
```

```
## Warning: running command 'curl "https://d396qusza40orc.cloudfront.net/
## predmachlearn/pml-training.csv" -o "./pml-training.csv"' had status 127
```

```
## Warning in download.file("https://d396qusza40orc.cloudfront.net/
## predmachlearn/pml-training.csv", : download had nonzero exit status
```

```
download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv", destfile = "./pml-testing.cs
v", method="curl")
```

```
## Warning: running command 'curl "https://d396qusza40orc.cloudfront.net/
## predmachlearn/pml-testing.csv" -o "./pml-testing.csv"' had status 127
```

```
## Warning in download.file("https://d396qusza40orc.cloudfront.net/
## predmachlearn/pml-testing.csv", : download had nonzero exit status
```

```
# read the csv file for training and clean from NA and filter out non needed columns
training_data <- read.csv("./pml-training.csv", na.strings= c("NA","","#DIV/0!"))
training_clean <- training_data[,which(apply(training_data, 2, function(x) {sum(is.na(x))}) == 0)]
training_clean <- training_clean[8:length(training_clean)]
```

Building the model

In this section, we will build a machine learning model for predicting the classe value based on the other features of the dataset. We will split the training data between an actual “training” set (75%) keeping a 25% to test the performance of the model (to estimate the out-of-sample accuracy).

We are going to use the random forest algorithm (full model), since the purpose of the analysis is to have the highest accuracy and also because of it being suitable with non linearity, absence of constraints in the problem definition as far as the parameter selection is concerned, robustness to outliers.

```
# set seed for reproducible results
set.seed(100)
# partitioning the cleaned testing data
inTrain <- createDataPartition(y = training_clean$classe, p = 0.75, list = FALSE)
training <- training_clean[inTrain, ]
test <- training_clean[-inTrain, ]

# predict the classe using the random forest algorithm, full (everything else as a predictor)
model <- randomForest(classe ~ ., data = training)
model$confusion
```

```
##      A    B    C    D    E  class.error
## A 4181    3    0    0    1 0.0009557945
## B   10 2834    4    0    0 0.0049157303
## C    0   15 2548    4    0 0.0074016362
## D    0    0   19 2391    2 0.0087064677
## E    0    0    1    8 2697 0.0033259424
```

Validating the model

We can now use the remaining 25% of the data of the training set, since it has not been used to create our model, to get an unbiased estimation of the out of sample error rate.

```
prediction <- predict(model, newdata=test)
confusionMatrix(prediction, test$classe)
```

```
## Confusion Matrix and Statistics
##
##          Reference
## Prediction    A    B    C    D    E
##         A 1394    3    0    0    0
##         B    1  942    4    0    0
##         C    0    4  849    9    1
##         D    0    0    2  795    2
##         E    0    0    0    0  898
##
## Overall Statistics
##
##               Accuracy : 0.9947
##                 95% CI : (0.9922, 0.9965)
##    No Information Rate : 0.2845
##    P-Value [Acc > NIR] : < 2.2e-16
##
##                  Kappa : 0.9933
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity           0.9993   0.9926   0.9930   0.9888   0.9967
## Specificity           0.9991   0.9987   0.9965   0.9990   1.0000
## Pos Pred Value        0.9979   0.9947   0.9838   0.9950   1.0000
## Neg Pred Value        0.9997   0.9982   0.9985   0.9978   0.9993
## Prevalence            0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate        0.2843   0.1921   0.1731   0.1621   0.1831
## Detection Prevalence  0.2849   0.1931   0.1760   0.1629   0.1831
## Balanced Accuracy     0.9992   0.9957   0.9948   0.9939   0.9983
```

```
print(confusionMatrix(prediction, test$classe), digits=4)
```

```
## Confusion Matrix and Statistics
##
##          Reference
## Prediction    A    B    C    D    E
##         A 1394    3    0    0    0
##         B    1  942    4    0    0
##         C    0    4  849    9    1
##         D    0    0    2  795    2
##         E    0    0    0    0  898
##
## Overall Statistics
##
##               Accuracy : 0.9947
##                 95% CI : (0.9922, 0.9965)
##    No Information Rate : 0.2845
##    P-Value [Acc > NIR] : < 2.2e-16
##
##                  Kappa : 0.9933
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9993   0.9926   0.9930   0.9888   0.9967
## Specificity            0.9991   0.9987   0.9965   0.9990   1.0000
## Pos Pred Value         0.9979   0.9947   0.9838   0.9950   1.0000
## Neg Pred Value         0.9997   0.9982   0.9985   0.9978   0.9993
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2843   0.1921   0.1731   0.1621   0.1831
## Detection Prevalence   0.2849   0.1931   0.1760   0.1629   0.1831
## Balanced Accuracy      0.9992   0.9957   0.9948   0.9939   0.9983
```

Conclusions

The model built in this way seems to deliver very good results (out of sample accuracy of more than 99%), therefore it is our choice for the prediction of the classe of exercises (second part of the assignment).

Credits.

Full credits and many thanks to:

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.

Appendix. Prediction assigment.

```
# apply the same treatment to the final testing data
data_test <- read.csv("./pml-testing.csv", na.strings= c("NA","","#DIV/0!"))
data_test_clean <- data_test[,which(apply(data_test, 2, function(x) {sum(is.na(x))}) == 0)]
data_test_clean <- data_test_clean[8:length(data_test_clean)]

# predict the classes of the test set and write the results in the requested txt files
answers <- predict(model, data_test_clean)
answers <- as.character(answers)
answers
```

```
##  [1] "B" "A" "B" "A" "A" "E" "D" "B" "A" "A" "B" "C" "B" "A" "E" "E" "A"
## [18] "B" "B" "B"
```

```
# use the function suggested to write the answers
source("pml_write_files.R")
pml_write_files(answers)
```