# Machine Learning Course Project

Maximiliano Fernández

23/08/2020

## Analysing human body movement

### Executive Summary

This project is from Coursera Machine Learning Course assingment on predicting the outcome of physical excercise.

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, the goal is to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants. Then, with the previous data, predict the manner in which they did the exercise. This is the "classe" variable in the training set. The final model will be used to predict 20 different test cases.

#### 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).

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://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har.

Workflow of the project. The project is divided in three parts:

- Getting and cleaning data: where the data is downloaded from the Internet, identigy how it is composed and then cleaned, because the original data has incomplete data or NA
- Modeling in order to find the best model that can predict the Classe (factor variable) of the data
- Final discussion

The object is to predict the Class variable, which can be: A, B, C, D or E (factor variables). Class A corresponds to the specified execution of the exercise, while the other 4 classes correspond to common

mistakes. Participants were supervised by an experienced weight lifter to make sure the execution complied to the manner they were supposed to simulate. The exercises were performed by six male participants aged between 20-28 years, with little weight lifting experience. We made sure that all participants could easily simulate the mistakes in a safe and controlled manner by using a relatively light dumbbell (1.25kg).

The accuracy indicator is used to identify and select the best model.

For more information in Human Activity Recognition: + http://groupware.les.inf.puc-rio.br/har#ixzz6VtX3Luqn

### 1. Getting and cleaning data

First download and process the raw data from the

```
if(!file.exists("pml-training.csv")) {
  download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv",
                destfile = "pml-training.csv", method = "curl")
}

if(!file.exists("pml-testing.csv")) {
  download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv",
                destfile = "pml-testing.csv", method = "curl")
}

# Load libraries
#suppressMessages(library(dplyr))
suppressMessages(library(ggplot2))
```

```
## Warning: package 'ggplot2' was built under R version 3.6.3
```

```
suppressMessages(library(caret))
```

```
## Warning: package 'caret' was built under R version 3.6.3
```

```
suppressMessages(library(rattle))
```

```
## Warning: package 'rattle' was built under R version 3.6.3
```

```
suppressMessages(library(rpart.plot))
```

```
## Warning: package 'rpart.plot' was built under R version 3.6.3
```

```
suppressMessages(library(randomForest))
```

```
## Warning: package 'randomForest' was built under R version 3.6.3
```

```
#Read data from working directory
training <-read.csv("pml-training.csv", sep=",", header=T, na.strings=c("NA","#DIV/0!", ""))
testing <-read.csv("pml-testing.csv", sep=",", header=T, na.strings=c("NA","#DIV/0!", ""))
```

Search how is composed the training data

```r
dim(training)
```

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

There are 19.622 observations and 160 columns. However, there might be NAs, which should be takem into account

```r
sum(is.na(training))
```

```
## [1] 1925102
```

As there are 1.925.102 missing values. It might seem that various columns containg all NA.

In order to clean the data, the previous columns will be extracted from the training and testing set

```r
training <- training [,colSums(is.na(training)) == 0]
testing <- testing [,colSums(is.na(training)) == 0]
```

In addition to deleting the columns with NA values, other columns do not add information to the analysis. These columns are: user_name, raw_timestamp_part_1, raw_timestamp_part_,2 cvtd_timestamp, new_window, and num_window, and are the first seven columns of the data frame.

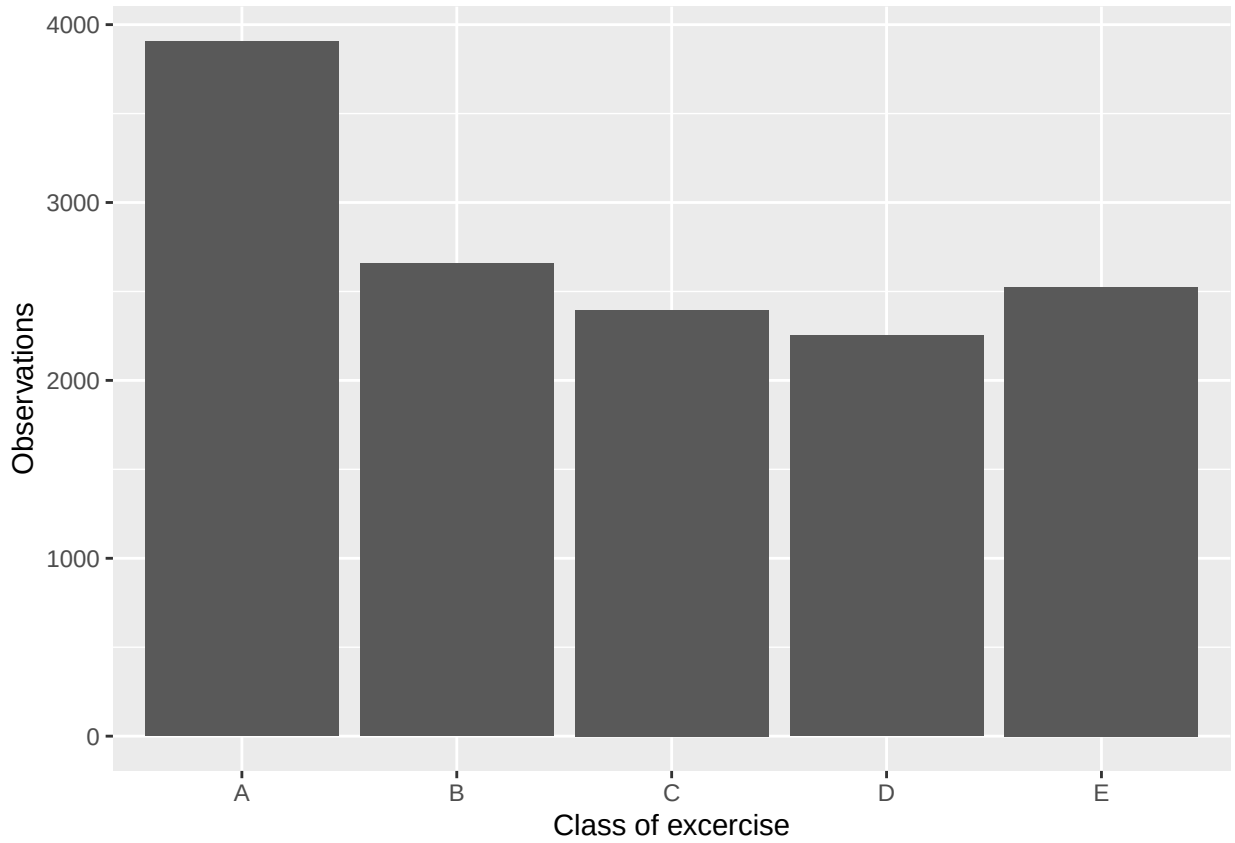
```r
training <- training[,-c(1:7)]
testing <- testing[,-c(1:7)]
```

**Create new training and testing set** After cleaning the trainig data set, another training set is created, now only with 70% of the original set, and the other 30% to a new testing set. The new training data will be use to create various models which will then be tested in the new testing data to perfom cross validation. Once the best model is defined, using the accuracy indicator, it will be tested in the original testing data, which contains 20 samples The new training data will be created taking randomly 70% of the original training set (without replacement)

```r
# Set seed for reproducibility
set.seed(555)
inTrain = createDataPartition(y=training$classe, p = 0.7,list=FALSE)
training_2 = training[ inTrain,]
testing_2 = training[-inTrain,]
```

See how is distributed

```r
qplot(training_2$classe, ylab = "Observations", xlab = "Class of excercise")
```




From the previous model, the most common class of excercise is type A. However, the other types (B, C, D and E) have more observations than the first one, that is, most of the excercises where not completed using the optimal movement of the body parts.

### 2. Modeling

The first model to use will be a decision tree, which is useful approach to try when the data has many predictors is to model a deccission tree (https://en.wikipedia.org/wiki/Decision_tree).

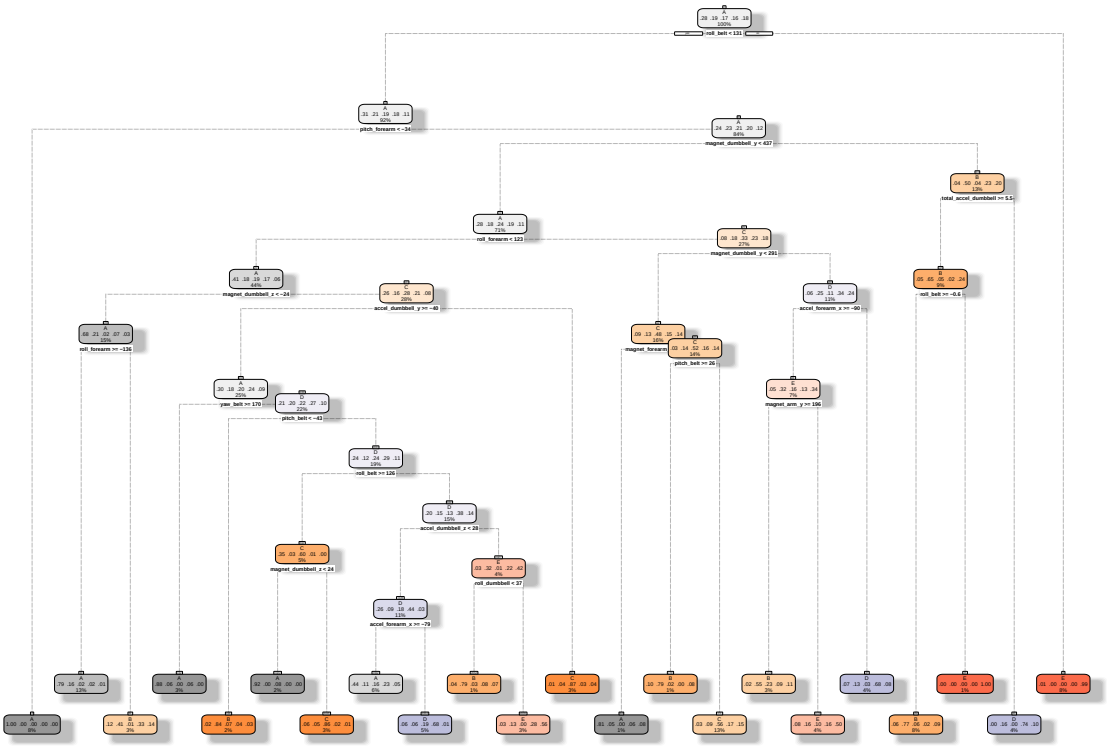
```
modFit_Dec_Tree <- rpart(classe ~ ., data=training_2, method="class")
print(modFit_Dec_Tree, digits = 2)
```

```
## n= 13737
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##    1) root 13737 9800 A (0.28 0.19 0.17 0.16 0.18)
##      2) roll_belt< 1.3e+02 12604 8700 A (0.31 0.21 0.19 0.18 0.11)
##        4) pitch_forearm< -34 1110    5 A (1 0.0045 0 0 0) *
##        5) pitch_forearm>=-34 11494 8700 A (0.24 0.23 0.21 0.2 0.12)
##         10) magnet_dumbbell_y< 4.4e+02 9685 7000 A (0.28 0.18 0.24 0.19 0.11)
##           20) roll_forearm< 1.2e+02 5979 3500 A (0.41 0.18 0.19 0.17 0.062)
##             40) magnet_dumbbell_z< -24 2085  680 A (0.68 0.21 0.017 0.073 0.028)
##               80) roll_forearm>=-1.4e+02 1730  360 A (0.79 0.16 0.019 0.021 0.0052) *
##               81) roll_forearm< -1.4e+02 355  210 B (0.12 0.41 0.0085 0.33 0.14) *
```

```
##             41) magnet_dumbbell_z>=-24 3894 2800 C (0.26 0.16 0.28 0.21 0.081)
##               82) accel_dumbbell_y>=-40 3422 2400 A (0.3 0.18 0.2 0.24 0.086)
##                164) yaw_belt>=1.7e+02 436   52 A (0.88 0.055 0 0.062 0.0023) *
##                165) yaw_belt< 1.7e+02 2986 2200 D (0.21 0.2 0.22 0.27 0.098)
##                  330) pitch_belt< -43 322   53 B (0.019 0.84 0.075 0.04 0.031) *
##                  331) pitch_belt>=-43 2664 1900 D (0.24 0.12 0.24 0.29 0.11)
##                    662) roll_belt>=1.3e+02 638  260 C (0.35 0.034 0.6 0.013 0.0047)
##                     1324) magnet_dumbbell_z< 24 215   18 A (0.92 0 0.084 0 0) *
##                     1325) magnet_dumbbell_z>=24 423   58 C (0.059 0.052 0.86 0.019 0.0071) *
##                    663) roll_belt< 1.3e+02 2026 1300 D (0.2 0.15 0.13 0.38 0.14)
##                     1326) accel_dumbbell_z< 28 1479  830 D (0.26 0.089 0.18 0.44 0.034)
##                       2652) accel_forearm_x>=-80 793  440 A (0.44 0.11 0.16 0.23 0.054) *
##                       2653) accel_forearm_x< -80 686  220 D (0.057 0.061 0.19 0.68 0.012) *
##                     1327) accel_dumbbell_z>=28 547  320 E (0.033 0.32 0.0073 0.22 0.42)
##                       2654) roll_dumbbell< 37 157   33 B (0.038 0.79 0.025 0.076 0.07) *
##                       2655) roll_dumbbell>=37 390  170 E (0.031 0.13 0 0.28 0.56) *
##               83) accel_dumbbell_y< -40 472   61 C (0.011 0.044 0.87 0.034 0.04) *
##             21) roll_forearm>=1.2e+02 3706 2500 C (0.079 0.18 0.33 0.23 0.18)
##               42) magnet_dumbbell_y< 2.9e+02 2160 1100 C (0.094 0.13 0.48 0.15 0.14)
##                 84) magnet_forearm_z< -2.5e+02 170   33 A (0.81 0.053 0 0.059 0.082) *
##                 85) magnet_forearm_z>=-2.5e+02 1990  950 C (0.033 0.14 0.52 0.16 0.14)
##                  170) pitch_belt>=26 145   30 B (0.1 0.79 0.021 0 0.083) *
##                  171) pitch_belt< 26 1845  810 C (0.027 0.089 0.56 0.17 0.15) *
##               43) magnet_dumbbell_y>=2.9e+02 1546 1000 D (0.058 0.25 0.11 0.34 0.24)
##                 86) accel_forearm_x>=-90 966  640 E (0.052 0.32 0.16 0.13 0.34)
##                  172) magnet_arm_y>=2e+02 403  180 B (0.017 0.55 0.23 0.092 0.11) *
##                  173) magnet_arm_y< 2e+02 563  280 E (0.076 0.16 0.1 0.16 0.5) *
##                 87) accel_forearm_x< -90 580  180 D (0.069 0.13 0.034 0.68 0.084) *
##           11) magnet_dumbbell_y>=4.4e+02 1809  900 B (0.036 0.5 0.036 0.23 0.2)
##             22) total_accel_dumbbell>=5.5 1283  460 B (0.051 0.65 0.051 0.018 0.24)
##               44) roll_belt>=-0.6 1082  250 B (0.06 0.77 0.06 0.021 0.093) *
##               45) roll_belt< -0.6 201    0 E (0 0 0 0 1) *
##             23) total_accel_dumbbell< 5.5 526  140 D (0 0.16 0.0019 0.74 0.1) *
##        3) roll_belt>=1.3e+02 1133   12 E (0.011 0 0 0 0.99) *
```

```
fancyRpartPlot(modFit_Dec_Tree, palettes=c("Greys", "Oranges"))
```

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




After modeling the decission tree with the training set, it is possible to predict the outcome (Class of the excercise) with the testing set. After it, a consufion matrix (https://honingds.com/blog/confusion-matrix-in-r/) gives the accuracy, and other indicatior, on how well can the model predict with new data.

```
test_predict <- predict(modFit_Dec_Tree,newdata=testing_2, type = "class")
confusionMatrix(test_predict, testing_2$classe)$overall[1]
```

```
##  Accuracy
## 0.7259133
```

The accuracy of the model is 73.63%, that is, the model can correctly identify the class of the excercise 73.63% of the time. Moreover, the Kappa indicator is 0.6657, which is a subtancial indicator of the inter-rater reliability (https://en.wikipedia.org/wiki/Cohen%27s_kappa)

A model that should fit better to the data is a Random Forest, which is one of the most commonly used and the most powerful machine learning techniques. It is a special type of bagging applied to decision trees. Compared to the standard decision tree algorithm, the random forest provides a strong improvement, which consists of applying bagging to the data and bootstrap sampling to the predictor variables at each split (James et al. 2014, P. Bruce and Bruce (2017)). This means that at each splitting step of the tree algorithm, a random sample of n predictors is chosen as split candidates from the full set of the predictors.

Random forest can be used for both classification (predicting a categorical variable) and regression (predicting a continuous variable). More information: http://www.sthda.com/english/articles/35-statistical-machine-learning-essentials/140-bagging-and-random-forest-essentials

### Prediction with random forest

```
modFit_Ran_Forest <- randomForest(classe ~ ., data=training_2, method="class")
```

```
test_predict_Ran_forest <- predict(modFit_Ran_Forest,newdata=testing_2, type = "class")
confusionMatrix(test_predict_Ran_forest, testing_2$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1672    8    0    0    0
##          B    1 1131   16    0    0
##          C    0    0 1010   10    0
##          D    0    0    0  954    1
##          E    1    0    0    0 1081
##
## Overall Statistics
##
##                Accuracy : 0.9937
##                  95% CI : (0.9913, 0.9956)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.992
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9988   0.9930   0.9844   0.9896   0.9991
## Specificity            0.9981   0.9964   0.9979   0.9998   0.9998
## Pos Pred Value         0.9952   0.9852   0.9902   0.9990   0.9991
## Neg Pred Value         0.9995   0.9983   0.9967   0.9980   0.9998
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2841   0.1922   0.1716   0.1621   0.1837
## Detection Prevalence   0.2855   0.1951   0.1733   0.1623   0.1839
## Balanced Accuracy      0.9985   0.9947   0.9912   0.9947   0.9994
```

The random forest model has an accuracy of 99.5% which is better than the 73.6% of the decision tree model. Furthermothe, the kappa coeficient, increased from 0.66 from the decision tree to 0.9938 in the random forest.

#### Prediction on the origina test set

```
predict_test <- predict(modFit_Ran_Forest, newdata = testing, type = "class")
```

## Discussion

Although a decision tree usually provides an adecuate model to predict values with many predictors, in the previous example, this model provided an accuracy of 76% against a 99.5% accuracy of the random forest model. The last model is the best of both and should be used in the case more data need to be predicted. The original data have an important issue, which is that the data was colected from 6 different males from ages between 20 and 28 years old. Because of this, the model could not fit well with new information from other types of test subjects, for example, elderly people might not get the same observations of the accelerometers and then lead the model to incorrect assumptions.