Practical Machine Learning Final Assignment

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.

The data for this project come from this source: [Link]: http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har

Loading the data

```
library(randomForest)
library(rpart)
library(rpart.plot)
library(RColorBrewer)
library(rattle)
library(e1071)
library(caret)
library(dplyr)

trainUrl <-"https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
testUrl <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"

train <- read.csv(trainUrl)
test <- read.csv(testUrl)
```

Cleaning data

```
train$classe<-as.factor(train$classe)
Train <- train %>% tibble::as_tibble() %>% sample_n(size = 1000)
Train <- Train[, -(which((colSums(is.na(train)) / nrow(train)) > 0.75))]
Train <- Train[, -grep("cvtd_timestamp",names(Train))]
Train <- Train[, -grep("X|user_name",names(Train))]
Train <- Train[, -nearZeroVar(Train)]
```

Possible predictors

```
predictors <- names(Train)
pred <- predictors[-grep("classe", predictors)]
pred
```

```
##  [1] "raw_timestamp_part_1" "raw_timestamp_part_2" "num_window"
##  [4] "roll_belt"            "pitch_belt"           "yaw_belt"
##  [7] "total_accel_belt"     "gyros_belt_x"         "gyros_belt_y"
## [10] "gyros_belt_z"         "accel_belt_x"         "accel_belt_y"
## [13] "accel_belt_z"         "magnet_belt_x"        "magnet_belt_y"
## [16] "magnet_belt_z"        "roll_arm"             "pitch_arm"
## [19] "yaw_arm"              "total_accel_arm"      "gyros_arm_x"
## [22] "gyros_arm_y"          "gyros_arm_z"          "accel_arm_x"
## [25] "accel_arm_y"          "accel_arm_z"          "magnet_arm_x"
## [28] "magnet_arm_y"         "magnet_arm_z"         "roll_dumbbell"
## [31] "pitch_dumbbell"       "yaw_dumbbell"         "total_accel_dumbbell"
## [34] "gyros_dumbbell_x"     "gyros_dumbbell_y"     "gyros_dumbbell_z"
## [37] "accel_dumbbell_x"     "accel_dumbbell_y"     "accel_dumbbell_z"
## [40] "magnet_dumbbell_x"    "magnet_dumbbell_y"    "magnet_dumbbell_z"
## [43] "roll_forearm"         "pitch_forearm"        "yaw_forearm"
## [46] "total_accel_forearm"  "gyros_forearm_x"      "gyros_forearm_y"
## [49] "gyros_forearm_z"      "accel_forearm_x"      "accel_forearm_y"
## [52] "accel_forearm_z"      "magnet_forearm_x"     "magnet_forearm_y"
## [55] "magnet_forearm_z"
```

Building the model

```
trainning <- train[, predictors]
model <- randomForest(classe ~., data = trainning, type = "class")
prediction_train <- predict(model, newdata = train)
confusionMatrix(prediction_train, train$classe)$table
```

```
##          Reference
## Prediction    A    B    C    D    E
##          A 5580    0    0    0    0
##          B    0 3797    0    0    0
##          C    0    0 3422    0    0
##          D    0    0    0 3216    0
##          E    0    0    0    0 3607
```

```
model
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = trainning, type = "class")
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 0.06%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 5579    1    0    0    0 0.0001792115
## B    2 3795    0    0    0 0.0005267316
## C    0    5 3417    0    0 0.0014611338
## D    0    0    2 3213    1 0.0009328358
## E    0    0    0    1 3606 0.0002772387
```

Test set prediction

Applying the model to the test set will produce the following results:

```
prediction_test <- predict(model, newdata = test, type = "class")
prediction_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
```

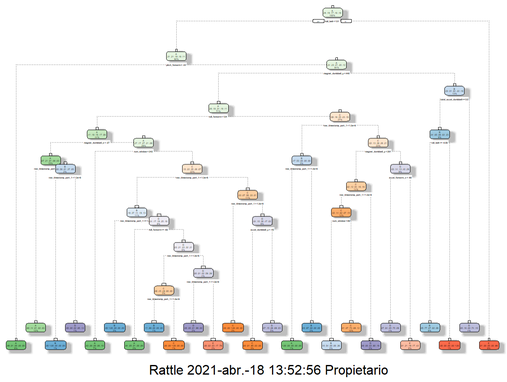
Conclusion

The random forests algorythm achived a confusion Matrix close to a 100% accuracy, which means that it performs very well when you need to hadle a larga number of inputs and you dont know the interactions between variables.

Appendix

```
treeModel <- rpart(classe ~ ., data=trainning, method="class")
fancyRpartPlot(treeModel)
```

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



Rattle 2021-abr.-18 13:52:56 Propietario