# Assessment : How to exercise efficiently ?

# 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).

# Library

library(caret)

## Warning: package 'caret' was built under R version 3.2.2

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

## Warning: package 'ggplot2' was built under R version 3.2.2

library(rpart)  
library(rpart.plot)

## Warning: package 'rpart.plot' was built under R version 3.2.2

library(RColorBrewer)  
library(rattle)

## Warning: package 'rattle' was built under R version 3.2.2

## Loading required package: RGtk2

## Warning: package 'RGtk2' was built under R version 3.2.2

## Rattle: A free graphical interface for data mining with R.  
## Version 3.5.0 Copyright (c) 2006-2015 Togaware Pty Ltd.  
## Type 'rattle()' to shake, rattle, and roll your data.

library(randomForest)

## Warning: package 'randomForest' was built under R version 3.2.2

## randomForest 4.6-10  
## Type rfNews() to see new features/changes/bug fixes.

# Random Number Generation

Integer vector, containing the random number generator (RNG) state for random number generation in R

set.seed(12345)

# 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://groupware.les.inf.puc-rio.br/har>.

Download both training dataset :-

curdir <-getwd()  
file.url<-'http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv'  
download.file(file.url,destfile=paste(curdir,'/pml-training.csv',sep=""))  
  
curdir <-getwd()  
file.url<-'http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv'  
download.file(file.url,destfile=paste(curdir,'/pml-testing.csv',sep=""))

Load both dataset and change the missing value "#DIV/0!" to "NA" .

training <-read.csv(paste(curdir,'/pml-training.csv',sep=""),na.strings=c("NA","#DIV/0!",""))  
testing <-read.csv(paste(curdir,'/pml-testing.csv',sep=""), na.strings=c("NA","#DIV/0!", ""))

Delete column which has missing values.

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

Cheking the dimension of training and test dataset :-

dim(training)

## [1] 19622 60

dim(testing)

## [1] 20 60

Checking the columns which have all missing values

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

We remove 6 of the variables which is irrelevant like :-

1. user\_name
2. raw\_timestamp\_part\_1
3. raw\_timestamp\_part\_2
4. cvtd\_timestamp
5. new\_window
6. num\_window

which resides on the column 1-7.

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

Check again the dimension

dim(training)

## [1] 19622 46

dim(testing)

## [1] 20 46

Now we obtain the several rows to preview

head(training)

## accel\_belt\_x accel\_belt\_y accel\_belt\_z magnet\_belt\_x magnet\_belt\_y  
## 1 -21 4 22 -3 599  
## 2 -22 4 22 -7 608  
## 3 -20 5 23 -2 600  
## 4 -22 3 21 -6 604  
## 5 -21 2 24 -6 600  
## 6 -21 4 21 0 603  
## magnet\_belt\_z roll\_arm pitch\_arm yaw\_arm total\_accel\_arm gyros\_arm\_x  
## 1 -313 -128 22.5 -161 34 0.00  
## 2 -311 -128 22.5 -161 34 0.02  
## 3 -305 -128 22.5 -161 34 0.02  
## 4 -310 -128 22.1 -161 34 0.02  
## 5 -302 -128 22.1 -161 34 0.00  
## 6 -312 -128 22.0 -161 34 0.02  
## gyros\_arm\_y gyros\_arm\_z accel\_arm\_x accel\_arm\_y accel\_arm\_z magnet\_arm\_x  
## 1 0.00 -0.02 -288 109 -123 -368  
## 2 -0.02 -0.02 -290 110 -125 -369  
## 3 -0.02 -0.02 -289 110 -126 -368  
## 4 -0.03 0.02 -289 111 -123 -372  
## 5 -0.03 0.00 -289 111 -123 -374  
## 6 -0.03 0.00 -289 111 -122 -369  
## magnet\_arm\_y magnet\_arm\_z roll\_dumbbell pitch\_dumbbell yaw\_dumbbell  
## 1 337 516 13.05217 -70.49400 -84.87394  
## 2 337 513 13.13074 -70.63751 -84.71065  
## 3 344 513 12.85075 -70.27812 -85.14078  
## 4 344 512 13.43120 -70.39379 -84.87363  
## 5 337 506 13.37872 -70.42856 -84.85306  
## 6 342 513 13.38246 -70.81759 -84.46500  
## total\_accel\_dumbbell gyros\_dumbbell\_x gyros\_dumbbell\_y gyros\_dumbbell\_z  
## 1 37 0 -0.02 0.00  
## 2 37 0 -0.02 0.00  
## 3 37 0 -0.02 0.00  
## 4 37 0 -0.02 -0.02  
## 5 37 0 -0.02 0.00  
## 6 37 0 -0.02 0.00  
## accel\_dumbbell\_x accel\_dumbbell\_y accel\_dumbbell\_z magnet\_dumbbell\_x  
## 1 -234 47 -271 -559  
## 2 -233 47 -269 -555  
## 3 -232 46 -270 -561  
## 4 -232 48 -269 -552  
## 5 -233 48 -270 -554  
## 6 -234 48 -269 -558  
## magnet\_dumbbell\_y magnet\_dumbbell\_z roll\_forearm pitch\_forearm  
## 1 293 -65 28.4 -63.9  
## 2 296 -64 28.3 -63.9  
## 3 298 -63 28.3 -63.9  
## 4 303 -60 28.1 -63.9  
## 5 292 -68 28.0 -63.9  
## 6 294 -66 27.9 -63.9  
## yaw\_forearm total\_accel\_forearm gyros\_forearm\_x gyros\_forearm\_y  
## 1 -153 36 0.03 0.00  
## 2 -153 36 0.02 0.00  
## 3 -152 36 0.03 -0.02  
## 4 -152 36 0.02 -0.02  
## 5 -152 36 0.02 0.00  
## 6 -152 36 0.02 -0.02  
## gyros\_forearm\_z accel\_forearm\_x accel\_forearm\_y accel\_forearm\_z  
## 1 -0.02 192 203 -215  
## 2 -0.02 192 203 -216  
## 3 0.00 196 204 -213  
## 4 0.00 189 206 -214  
## 5 -0.02 189 206 -214  
## 6 -0.03 193 203 -215  
## magnet\_forearm\_x magnet\_forearm\_y magnet\_forearm\_z classe  
## 1 -17 654 476 A  
## 2 -18 661 473 A  
## 3 -18 658 469 A  
## 4 -16 658 469 A  
## 5 -17 655 473 A  
## 6 -9 660 478 A

head(testing)

## accel\_belt\_x accel\_belt\_y accel\_belt\_z magnet\_belt\_x magnet\_belt\_y  
## 1 -38 69 -179 -13 581  
## 2 -13 11 39 43 636  
## 3 1 -1 49 29 631  
## 4 46 45 -156 169 608  
## 5 -8 4 27 33 566  
## 6 -11 -16 38 31 638  
## magnet\_belt\_z roll\_arm pitch\_arm yaw\_arm total\_accel\_arm gyros\_arm\_x  
## 1 -382 40.7 -27.80 178 10 -1.65  
## 2 -309 0.0 0.00 0 38 -1.17  
## 3 -312 0.0 0.00 0 44 2.10  
## 4 -304 -109.0 55.00 -142 25 0.22  
## 5 -418 76.1 2.76 102 29 -1.96  
## 6 -291 0.0 0.00 0 14 0.02  
## gyros\_arm\_y gyros\_arm\_z accel\_arm\_x accel\_arm\_y accel\_arm\_z magnet\_arm\_x  
## 1 0.48 -0.18 16 38 93 -326  
## 2 0.85 -0.43 -290 215 -90 -325  
## 3 -1.36 1.13 -341 245 -87 -264  
## 4 -0.51 0.92 -238 -57 6 -173  
## 5 0.79 -0.54 -197 200 -30 -170  
## 6 0.05 -0.07 -26 130 -19 396  
## magnet\_arm\_y magnet\_arm\_z roll\_dumbbell pitch\_dumbbell yaw\_dumbbell  
## 1 385 481 -17.73748 24.96085 126.23596  
## 2 447 434 54.47761 -53.69758 -75.51480  
## 3 474 413 57.07031 -51.37303 -75.20287  
## 4 257 633 43.10927 -30.04885 -103.32003  
## 5 275 617 -101.38396 -53.43952 -14.19542  
## 6 176 516 62.18750 -50.55595 -71.12063  
## total\_accel\_dumbbell gyros\_dumbbell\_x gyros\_dumbbell\_y gyros\_dumbbell\_z  
## 1 9 0.64 0.06 -0.61  
## 2 31 0.34 0.05 -0.71  
## 3 29 0.39 0.14 -0.34  
## 4 18 0.10 -0.02 0.05  
## 5 4 0.29 -0.47 -0.46  
## 6 29 -0.59 0.80 1.10  
## accel\_dumbbell\_x accel\_dumbbell\_y accel\_dumbbell\_z magnet\_dumbbell\_x  
## 1 21 -15 81 523  
## 2 -153 155 -205 -502  
## 3 -141 155 -196 -506  
## 4 -51 72 -148 -576  
## 5 -18 -30 -5 -424  
## 6 -138 166 -186 -543  
## magnet\_dumbbell\_y magnet\_dumbbell\_z roll\_forearm pitch\_forearm  
## 1 -528 -56 141 49.30  
## 2 388 -36 109 -17.60  
## 3 349 41 131 -32.60  
## 4 238 53 0 0.00  
## 5 252 312 -176 -2.16  
## 6 262 96 150 1.46  
## yaw\_forearm total\_accel\_forearm gyros\_forearm\_x gyros\_forearm\_y  
## 1 156.0 33 0.74 -3.34  
## 2 106.0 39 1.12 -2.78  
## 3 93.0 34 0.18 -0.79  
## 4 0.0 43 1.38 0.69  
## 5 -47.9 24 -0.75 3.10  
## 6 89.7 43 -0.88 4.26  
## gyros\_forearm\_z accel\_forearm\_x accel\_forearm\_y accel\_forearm\_z  
## 1 -0.59 -110 267 -149  
## 2 -0.18 212 297 -118  
## 3 0.28 154 271 -129  
## 4 1.80 -92 406 -39  
## 5 0.80 131 -93 172  
## 6 1.35 230 322 -144  
## magnet\_forearm\_x magnet\_forearm\_y magnet\_forearm\_z problem\_id  
## 1 -714 419 617 1  
## 2 -237 791 873 2  
## 3 -51 698 783 3  
## 4 -233 783 521 4  
## 5 375 -787 91 5  
## 6 -300 800 884 6

In order to run cross-validation , the training dataset need to partition into 2 sets . We set the 1st partition for training dataset to 75% and test dataset to 25%. Training dataset contains 53 variables with 19622 obs and test dataset contains 53 variables with 20 obs.

This will do the randomize sub-sampling without replacement

chunks <- createDataPartition(y=training$classe, p=0.75, list=FALSE)  
chunks\_training <- training[chunks, ];   
chunks\_testing <- training[-chunks, ]  
dim(chunks\_training);

## [1] 14718 46

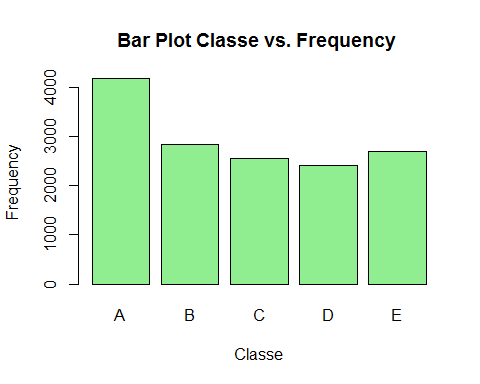
dim(chunks\_testing);

## [1] 4904 46

# Visualization

We try to plot into the histogram to see the trending frequency of each sub-training & test dataset by comparing with each other. The variable classe contains 5 levels which is A,B,C,D & E

plot(chunks\_training$classe, col="lightgreen", main="Bar Plot Classe vs. Frequency ", xlab="Classe", ylab="Frequency")



The graph above shows that A ~ 4000x occurrences is most frequent while D is the lest frequent ~ 2500x occurrences

# Decision Tree

Decision Tree machine learning algorithm as a support tool that uses a tree-like graph or model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility.

Fit\_Model\_1 <- rpart(classe ~ ., data=chunks\_training, method="class")

Displays the (Complexity) cp table for fitted model .

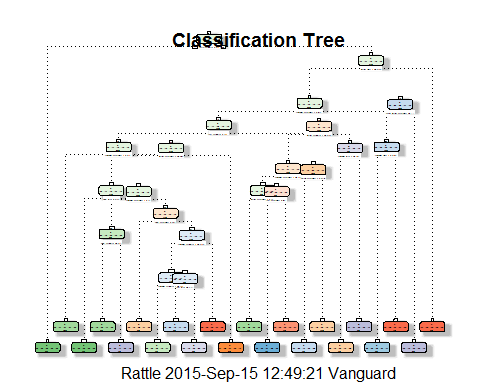
printcp(Fit\_Model\_1)

##   
## Classification tree:  
## rpart(formula = classe ~ ., data = chunks\_training, method = "class")  
##   
## Variables actually used in tree construction:  
## [1] accel\_belt\_z accel\_dumbbell\_y accel\_dumbbell\_z   
## [4] accel\_forearm\_x gyros\_dumbbell\_y magnet\_arm\_y   
## [7] magnet\_belt\_z magnet\_dumbbell\_y magnet\_dumbbell\_z   
## [10] magnet\_forearm\_x magnet\_forearm\_z pitch\_forearm   
## [13] roll\_arm roll\_dumbbell roll\_forearm   
## [16] total\_accel\_dumbbell yaw\_arm yaw\_dumbbell   
## [19] yaw\_forearm   
##   
## Root node error: 10533/14718 = 0.71565  
##   
## n= 14718   
##   
## CP nsplit rel error xerror xstd  
## 1 0.062826 0 1.00000 1.00000 0.0051957  
## 2 0.033134 4 0.74869 0.74983 0.0057435  
## 3 0.031615 5 0.71556 0.72335 0.0057553  
## 4 0.021172 6 0.68395 0.68546 0.0057579  
## 5 0.017089 8 0.64160 0.62708 0.0057286  
## 6 0.015190 9 0.62451 0.58815 0.0056864  
## 7 0.013956 10 0.60932 0.57211 0.0056637  
## 8 0.013909 11 0.59537 0.55255 0.0056316  
## 9 0.013102 15 0.52027 0.52995 0.0055885  
## 10 0.011266 17 0.49407 0.48941 0.0054946  
## 11 0.010728 20 0.46027 0.47318 0.0054508  
## 12 0.010000 21 0.44954 0.46596 0.0054301

To visualize the decision tree , we use this fancyRpartPlot command below :-

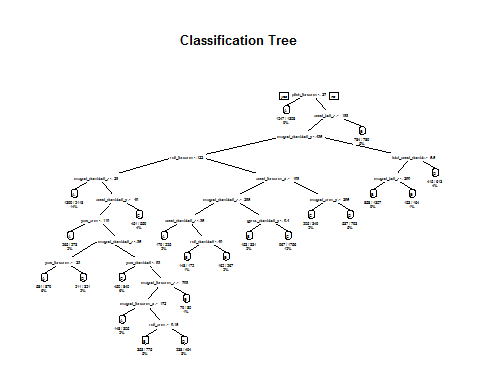
fancyRpartPlot(Fit\_Model\_1,main="Classification Tree")

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



Green nodes represent individuals classified by the tree as A, blue nodes are those classified as B and orange nodes are classified as C. The gradient is a visual representation of the three numbers in the middle of the nodes: bearing in mind that levels of a factor are by default in alphabetical order, the first of these three numbers is the proportion of individuals in that node that were actually classified as the first level, (A), in train\_part ; the second number is the proportion that were actually classified as B, and the third the proportion that were C.

rpart.plot(Fit\_Model\_1,main="Classification Tree",extra=102, under=TRUE, faclen=0)



Now we predict the fit model for test dataset .

Prediction\_Model1 <- predict(Fit\_Model\_1, chunks\_testing, type = "class")

# Confusion Matrix

Confusion matrix, also known as a contingency table or an error matrix , is a specific table layout that allows visualization of the performance of an algorithm, typically a supervised learning one (in unsupervised learning it is usually called a matching matrix). Each column of the matrix represents the instances in a predicted class while each row represents the instances in an actual class (or vice-versa).

confusionMatrix(Prediction\_Model1, chunks\_testing$classe)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 1277 217 40 115 92  
## B 61 483 83 68 160  
## C 36 150 674 141 155  
## D 19 83 56 445 128  
## E 2 16 2 35 366  
##   
## Overall Statistics  
##   
## Accuracy : 0.6617   
## 95% CI : (0.6483, 0.6749)  
## No Information Rate : 0.2845   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.5685   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Statistics by Class:  
##   
## Class: A Class: B Class: C Class: D Class: E  
## Sensitivity 0.9154 0.50896 0.7883 0.55348 0.40622  
## Specificity 0.8678 0.90594 0.8810 0.93024 0.98626  
## Pos Pred Value 0.7335 0.56491 0.5830 0.60876 0.86936  
## Neg Pred Value 0.9627 0.88491 0.9517 0.91397 0.88066  
## Prevalence 0.2845 0.19352 0.1743 0.16395 0.18373  
## Detection Rate 0.2604 0.09849 0.1374 0.09074 0.07463  
## Detection Prevalence 0.3550 0.17435 0.2357 0.14906 0.08585  
## Balanced Accuracy 0.8916 0.70745 0.8346 0.74186 0.69624

# Random Forest

Random forests are an ensemble learning method for classification, regression and other tasks, that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random forests correct for decision trees' habit of overfitting to their training set.

Fit\_Model\_2 <- randomForest(classe ~. , data=chunks\_training)

Now we predict the fit model for test dataset .

Prediction\_Model2 <- predict(Fit\_Model\_2, chunks\_testing, type = "class")

Below is the confusion matrix of the test results

confusionMatrix(Prediction\_Model2, chunks\_testing$classe)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 1394 5 0 0 0  
## B 1 939 5 0 0  
## C 0 5 850 11 1  
## D 0 0 0 790 4  
## E 0 0 0 3 896  
##   
## Overall Statistics  
##   
## Accuracy : 0.9929   
## 95% CI : (0.9901, 0.995)  
## No Information Rate : 0.2845   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.991   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: A Class: B Class: C Class: D Class: E  
## Sensitivity 0.9993 0.9895 0.9942 0.9826 0.9945  
## Specificity 0.9986 0.9985 0.9958 0.9990 0.9993  
## Pos Pred Value 0.9964 0.9937 0.9804 0.9950 0.9967  
## Neg Pred Value 0.9997 0.9975 0.9988 0.9966 0.9988  
## Prevalence 0.2845 0.1935 0.1743 0.1639 0.1837  
## Detection Rate 0.2843 0.1915 0.1733 0.1611 0.1827  
## Detection Prevalence 0.2853 0.1927 0.1768 0.1619 0.1833  
## Balanced Accuracy 0.9989 0.9940 0.9950 0.9908 0.9969

# Conclusion

From the machine learning method above , the cross validation accuracy of the Decision Tree is ~ 66.17% and the Random Forest is ~ 99.3% which is better and the sample error rate rather small around ~ 0.07% .

Final\_Prediction <- predict(Fit\_Model\_2, testing, type = "class")

Random Forests generally needs larger number of instances to work its randomization concept well and generalize to the novel data. In addition, in one way or another, random forests works with combination of some kind of soft linear boundaries at the decision surface

# Prediction files generator for assignment submission code

pml\_write\_files = function(x){  
 n = length(x)  
 for(i in 1:n){  
 filename = paste0("problem\_id\_",i,".txt")  
 write.table(x[i],file=filename,quote=FALSE,row.names=FALSE,col.names=FALSE)  
 }  
}  
  
pml\_write\_files(Final\_Prediction)

# Reference

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. Read more: <http://groupware.les.inf.puc-rio.br/har#ixzz3lj0hACeI>