# Final project

*Luis Galvis*

*Sunday, September 21, 2014*

**Executive summary**

This report shows the analysis on the data from Accelerometers on the belt, forearm, arm, and dumbell of 6 participants. The goal of the analysis is to predict the manner in which they did the exercise given the data in file "pml-training.csv". Then we will use the created model in 20 new test cases, given in the file "pml-testing.csv".

```
## Warning: package 'caret' was built under R version 3.1.1
```

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

```
## Warning: package 'ggplot2' was built under R version 3.1.1
## Warning: package 'randomForest' was built under R version 3.1.1
```

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

**First step - Choosing main Features**

The first step is to choose the features to build the model as there are 160 variables many of them with missing information. For that, we'll built the correlation matrix of the numerical variables and take the ones more than 0.68.

It's important to recall that we are interested in predict the feature classe, which gives the how persons made the excersice, so information like the name of who did the exercise is irrelevant.

```
cor1 <- mytrainData[sapply(mytrainData, function(x) sum(is.na(x)) == 0)]
# 93 features remaining after taking out all the features that have missing information

cor1 <- cor1[sapply(cor1, is.numeric)];
# 56 features remaining after taking out all the non numeric colums

corrbest <- cor(cor1, use = "pairwise.complete.obs");
corbestind<-findCorrelation(corrbest, 0.68);
# 25 features after choosing the best, the ones that are above 0.68

bestfeatnames <- names(cor1)[corbestind];
besttrainData <- mytrainData[,bestfeatnames]
# besttrainData is the data frame with the best 25 features (above 0.68)

# Now we add the classe variable back
besttrainData$classe <- mytrainData$classe
besttrain2 <- besttrainData[complete.cases(besttrainData),]
# Comparing besttrain2 and besttrainData they have the same number of rows, meaning matrix is complete,
```

```
cvdata <- mytestData[,bestfeatnames]

unique(mytrainData$classe)
```

```
## [1] A B C D E
## Levels: A B C D E
```

```
# Number of possible results is 5
```

### Creating training and test for the model

Now we create the training and testing as instructed in the lectures. This testing set though, will be our cross validation set. We'll take 70% for training and 30% for cross validation.

```
inTrain<-createDataPartition (y=besttrainData$classe,p=0.7,list=FALSE)
training<- besttrainData[inTrain,]
testing <- besttrainData[-inTrain,]
```

### Fitting and evaluating the Model

Given that there is more than 13000 observations in the training set, only 26 features(variables), and only 5 possible classification results (A,B,C,D,E), we should expect a very high degree of accuracy, more than 97%.

```
model <- randomForest(classe ~ .,   data=training, ntree = 50)
trainpredict <- predict(model, testing)
confusionMatrix(trainpredict, testing$classe)
```

```
## Warning: package 'e1071' was built under R version 3.1.1

## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1672    0    0    0    0
##          B    2 1139    3    0    0
##          C    0    0 1019    1    0
##          D    0    0    4  962    1
##          E    0    0    0    1 1081
##
## Overall Statistics
##
##                Accuracy : 0.998
##                  95% CI : (0.996, 0.999)
##     No Information Rate : 0.284
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.997
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
```

```
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity             0.999    1.000    0.993    0.998    0.999
## Specificity             1.000    0.999    1.000    0.999    1.000
## Pos Pred Value          1.000    0.996    0.999    0.995    0.999
## Neg Pred Value          1.000    1.000    0.999    1.000    1.000
## Prevalence              0.284    0.194    0.174    0.164    0.184
## Detection Rate          0.284    0.194    0.173    0.163    0.184
## Detection Prevalence    0.284    0.194    0.173    0.164    0.184
## Balanced Accuracy       0.999    0.999    0.996    0.998    0.999
```

So we got more than 99.7% accuracy on the cross validation set as expected, and very good specificity and sensitivity of our model as well. So our model is actually very good, and allows us to predict on the new testing set with confidence. Now we will use the model to get the predictions for the supplied testing set.

```
answers <- predict(model, cvdata);
#confusionMatrix(answers, cvdata$classe)
```

It's important to recall that supplied testing data doesn't contain a variable classe, so we can't confirm our accuracy, although we expect it will be very close to 99.5%. Finally, we will write our result files, as instructed.

```
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(answers)
#write the files, according instructions
```