# “Practical Machine Learning - Prediction Assigment

## Summary

The goal of your project is to predict the manner in which they did the exercise. This is the “classe” variable in the training set. Prediction models on the training set were built using three different algorithms, namely ‘Decision Tree’, ‘General Boosted Model’ and ‘Random Forest’. These prediction models were applied to the 20 test cases from the test data. Since the “Random Forest” model had the highest accuracy, the predictions of this model should be the most reliable.

## 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). Read more: http://groupware.les.inf.puc-rio.br/har#ixzz3xsbS5bVX

## Data

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. With kind acknowledgment to the authors.

Description of the datasets as provided by the authors:

“Six young health participants were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in five different fashions: exactly according to the specification (Class A), throwing the elbows to the front (Class B), lifting the dumbbell only halfway (Class C), lowering the dumbbell only halfway (Class D) and throwing the hips to the front (Class E).

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)."

Following (down)loading the dataset, the relevant variables were extracted from the training and test sets, respectivley, resulting in a total of 53 variables The training set was split into a training and a test set.

```r
# load libraries
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.5.3

## Loading required package: lattice

## Loading required package: ggplot2
## Warning: package 'ggplot2' was built under R version 3.5.3
```

```
library(kernlab)
```

```
##
## Attaching package: 'kernlab'

## The following object is masked from 'package:ggplot2':
##
##     alpha
```

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

```
## Warning: package 'rpart.plot' was built under R version 3.5.3
```

```
library(rattle)
```

```
## Warning: package 'rattle' was built under R version 3.5.3

## Rattle: A free graphical interface for data science with R.
## Version 5.2.0 Copyright (c) 2006-2018 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
library(randomForest)
```

```
## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:rattle':
##
##     importance

## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(ggcorrplot)
```

```
## Warning: package 'ggcorrplot' was built under R version 3.5.3
```

```
library(factoextra)
```

```
## Warning: package 'factoextra' was built under R version 3.5.3

## Welcome! Related Books: `Practical Guide To Cluster Analysis in R` at https://goo.gl/13EFCZ
```

```
set.seed(32345)

# (down)load datasets
download.file(url = "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv", destfile =
training <- read.csv("./UrlTrain.csv", na.strings=c("NA","#DIV/0!",""))

download.file(url = "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv",destfile = ".
testing <- read.csv("./UrlTest.csv", na.strings=c("NA","#DIV/0!",""))

# Data cleaning
# Extraction of relevant variables from the testset excluding all variables with NA's (which are static
variables <- names(testing[,colSums(is.na(testing)) == 0])[8:59]

training <- training[,c(variables,"classe")]
testing <- testing[,c(variables,"problem_id")]
dim(training)
```

```
## [1] 19622    53
```

```
dim(testing)
```

```
## [1] 20 53
```

```
# Split training set in training (70%) and test sets (30%)
inTrain  <- createDataPartition(training$classe, p=0.7, list=FALSE)
Train <- training[inTrain, ]
Test  <- training[-inTrain, ]
dim(Train)
```

```
## [1] 13737    53
```

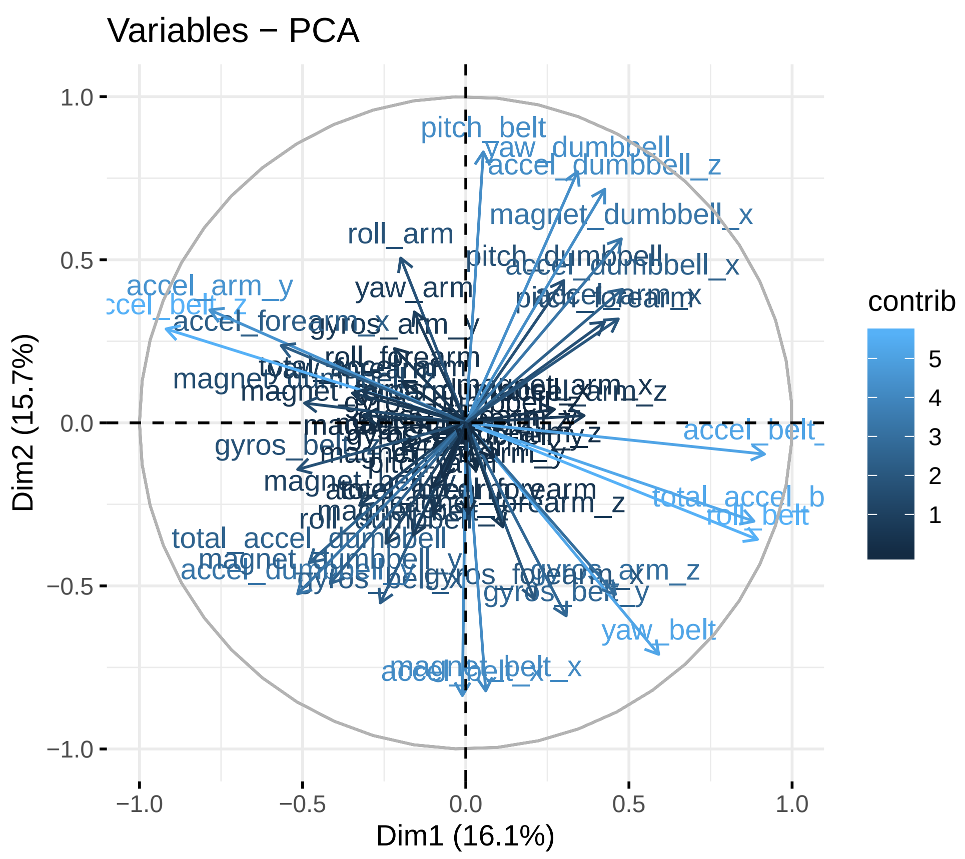
```
dim (Test)
```

```
## [1] 5885   53
```

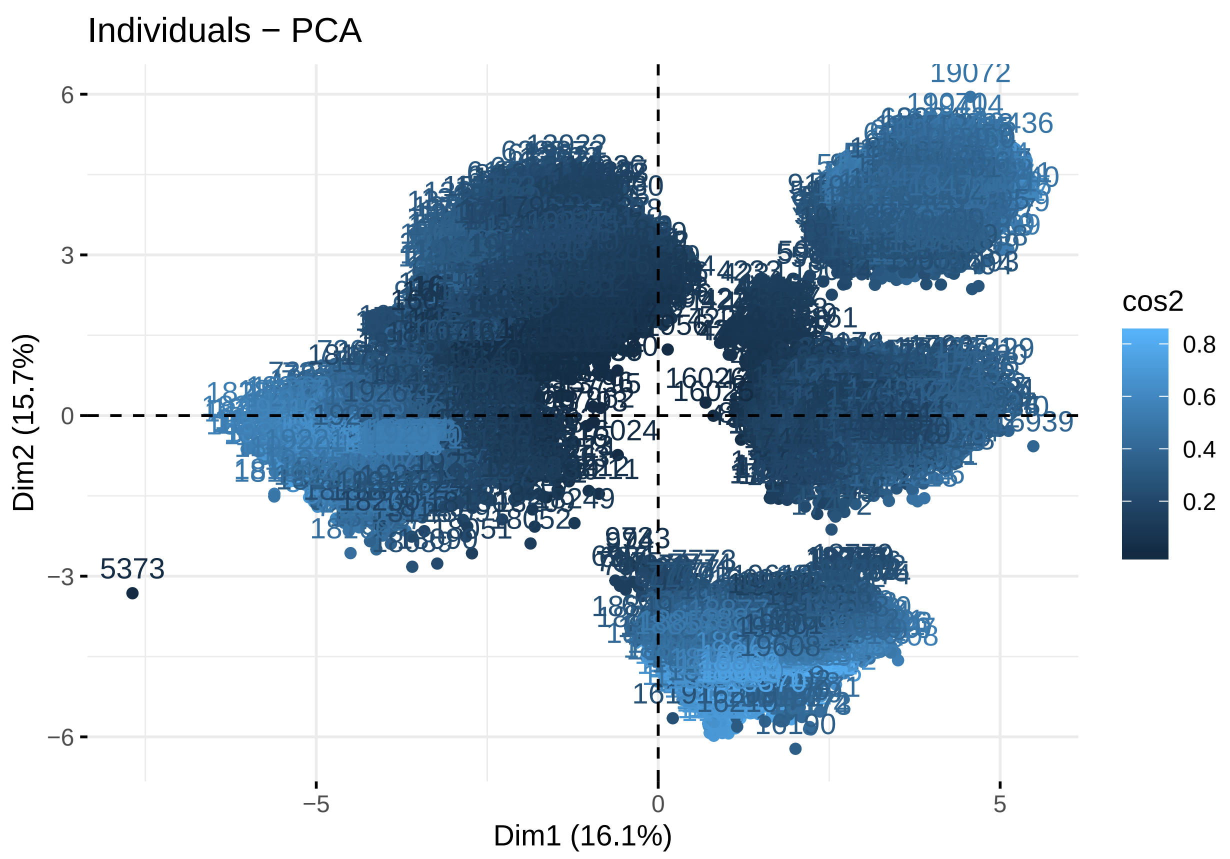
## Exploratory Analysis

Principal component analysis was performed to show the dependencies between the variables in a multi-dimensional space. The scores plot shows a clear separation between the 5 groups and the loading plot shows the variables driving the separation.

```
# PCA
pca <- prcomp(Train[, -53],scale.=T, center=T)
fviz_pca_var(pca, col.var = "contrib")
```



```r
fviz_pca_ind(pca, col.ind = "cos2")
```



## Prediction Models

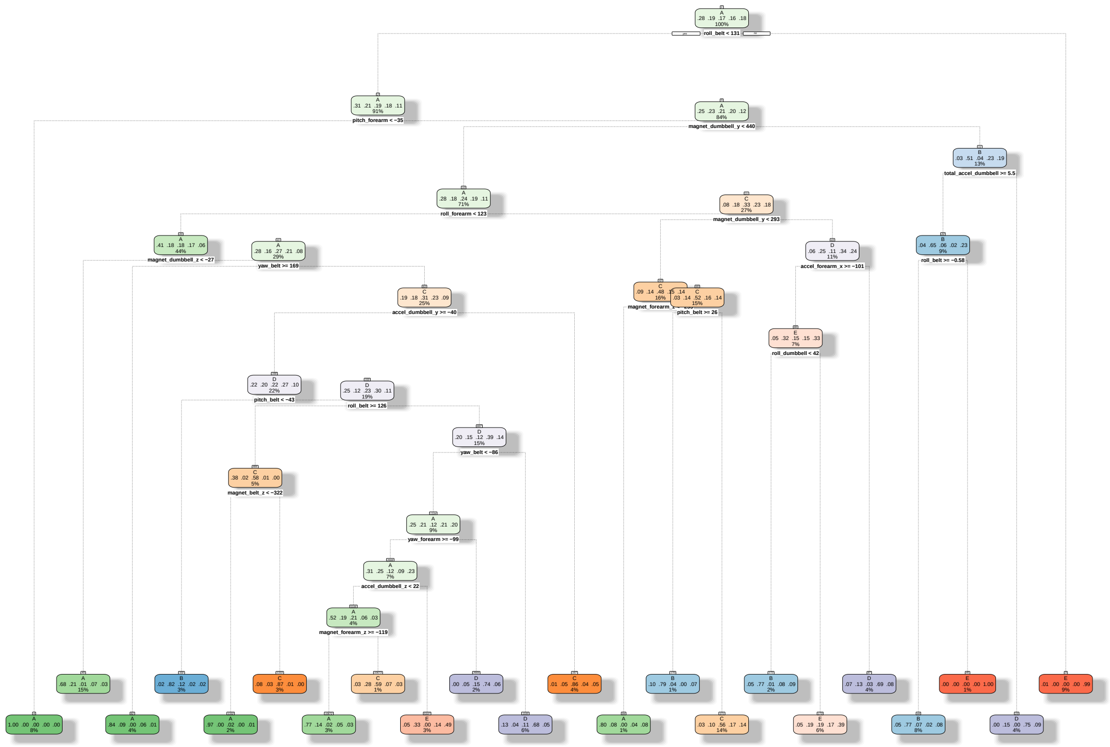
Prediction models were built using three different methods.

1. Method: Decision Tree

Performance of training set

```
set.seed(32345)
# Building model on training set
modelDT <- rpart(classe ~ ., data = Train, method="class")
fancyRpartPlot(modelDT, sub="classification based on decision tree")
```

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



classification based on decision tree

Performance of test set on the training model

```r
# Predicting on Test dataset
predictDT <- predict(modelDT, newdata=Test, type="class")
confusion_matrix_DT <- confusionMatrix(predictDT, Test$classe)
confusion_matrix_DT
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1505  225   17   89   42
##          B   42  601   53   22   61
##          C   45   93  829  150  149
##          D   49   91   63  617   60
##          E   33  129   64   86  770
##
## Overall Statistics
##
##                Accuracy : 0.7344
##                  95% CI : (0.7229, 0.7457)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6629
##
##  Mcnemar's Test P-Value : < 2.2e-16
```

```
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8990   0.5277   0.8080   0.6400   0.7116
## Specificity            0.9114   0.9625   0.9101   0.9466   0.9350
## Pos Pred Value         0.8014   0.7715   0.6548   0.7011   0.7116
## Neg Pred Value         0.9578   0.8946   0.9574   0.9307   0.9350
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2557   0.1021   0.1409   0.1048   0.1308
## Detection Prevalence   0.3191   0.1324   0.2151   0.1495   0.1839
## Balanced Accuracy      0.9052   0.7451   0.8590   0.7933   0.8233
```

2. Method: Generalized Boosted Model

Performance of training set

```r
set.seed(32345)
# Building model on training set
ctrlGBM <- trainControl(method = "repeatedcv", number = 5, repeats = 1)
modelGBM <- train(classe ~ ., data=Train, method = "gbm", trControl = ctrlGBM, verbose = FALSE)
modelGBM$finalModel
```

```
## A gradient boosted model with multinomial loss function.
## 150 iterations were performed.
## There were 52 predictors of which 52 had non-zero influence.
```

Performance of test set on the training model

```r
# Predicting on Test dataset
predictGBM <- predict(modelGBM, newdata=Test)
confusion_matrix_GBM <- confusionMatrix(predictGBM, Test$classe)
confusion_matrix_GBM
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1656   26    0    1    3
##          B   10 1085   33    6   16
##          C    2   27  978   33    6
##          D    4    1   14  922    7
##          E    2    0    1    2 1050
##
## Overall Statistics
##
##                Accuracy : 0.967
##                  95% CI : (0.9622, 0.9714)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9583
##
##  Mcnemar's Test P-Value : 1.91e-06
##
## Statistics by Class:
##
```

```
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9892   0.9526   0.9532   0.9564   0.9704
## Specificity            0.9929   0.9863   0.9860   0.9947   0.9990
## Pos Pred Value         0.9822   0.9435   0.9350   0.9726   0.9953
## Neg Pred Value         0.9957   0.9886   0.9901   0.9915   0.9934
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2814   0.1844   0.1662   0.1567   0.1784
## Detection Prevalence   0.2865   0.1954   0.1777   0.1611   0.1793
## Balanced Accuracy      0.9911   0.9694   0.9696   0.9756   0.9847
```

3. Method: Random Forest

Performance of training set

```
set.seed(32345)
# Building model on training set
ctrRF <- trainControl(method="cv", number=3, verboseIter=FALSE)
modelRF <- train(classe ~ ., data=Train, method="rf", trControl=ctrRF)
modelRF$finalModel
```

```
##
## Call:
##  randomForest(x = x, y = y, mtry = param$mtry)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 27
##
##         OOB estimate of  error rate: 0.66%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 3900    3    2    0    1 0.001536098
## B   18 2634    5    1    0 0.009029345
## C    0   14 2373    9    0 0.009599332
## D    0    1   22 2227    2 0.011101243
## E    0    1    7    5 2512 0.005148515
```

Performance of test set on the training model

```
# Predicting on Test dataset
predictRF <- predict(modelRF, newdata=Test)
confusion_matrix_RF <- confusionMatrix(predictRF, Test$classe)
confusion_matrix_RF
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1673    9    0    0    0
##          B    1 1127    5    1    1
##          C    0    3 1015   10    1
##          D    0    0    6  952    1
##          E    0    0    0    1 1079
##
## Overall Statistics
##
##                Accuracy : 0.9934
##                  95% CI : (0.991, 0.9953)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9916
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9994   0.9895   0.9893   0.9876   0.9972
## Specificity            0.9979   0.9983   0.9971   0.9986   0.9998
## Pos Pred Value         0.9946   0.9930   0.9864   0.9927   0.9991
## Neg Pred Value         0.9998   0.9975   0.9977   0.9976   0.9994
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2843   0.1915   0.1725   0.1618   0.1833
## Detection Prevalence   0.2858   0.1929   0.1749   0.1630   0.1835
## Balanced Accuracy      0.9986   0.9939   0.9932   0.9931   0.9985
```

Conclusion: The accuracy of the 3 models based on 3 different methods are:

Decision Tree : 0.7344 GBM : 0.963 Random Forest : 0.9944

The model based on Random Forest is the most accurate model and thus should deliver the most accurate predictions. The model based on Decision Tree is less accurate when compared to GGM and Random Forest.

## Application of 3 models to test dataset

Application to Decision Tree model

```
predictTestSet_DT <- predict(modelDT, newdata=testing, type="class")
predictTestSet_DT
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  B  A  E  D  A  C  D  A  A  A  C  E  C  A  E  E  A  A  A  B
## Levels: A B C D E
```

Application to Generalized Boosted Model

```
##  [1] 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
```

Application to Random Forest model

```
predictTestSet_RF <- predict(modelRF, newdata=testing)
predictTestSet_RF
```

```
##  [1] 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: Both the ‘Generalized Boosted’ and the ‘Random Forest’ models resulted in the same classifications of the test case, while the ‘Decision Tree’ resulted provided different classifications in 9 cases. The predictions of the ‘Generalized Boosted’ and the ‘Random Forest’ models were used for the quiz.