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.

The data consists of a Training data and a Test data (to be used to validate the selected model).

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. You may use any of the other variables to predict with.

Note: The dataset used in this project is a courtesy of "Ugulino, W.; Cardador, D.; Vega, K.; Velloso, E.; Milidiu, R.; Fuks, H. Wearable Computing: Accelerometers' Data Classification of Body Postures and Movements"

Download Data

Training data was obtained from https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv

Testing data was obtained from https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

Set working directory

```
setwd("D:/Handy/Coursera/Modul 8")
```

Load the dataset

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.6.2
```

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

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

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

```
## Warning: package 'rpart.plot' was built under R version 3.6.2
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.6.2
```

```
## randomForest 4.6-14
```

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

```
##
## Attaching package: 'randomForest'
```

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

```
library(rattle)
```

```
## Warning: package 'rattle' was built under R version 3.6.2
```

```
## Rattle: A free graphical interface for data science with R.
## Version 5.3.0 Copyright (c) 2006-2018 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
##
## Attaching package: 'rattle'
```

```
## The following object is masked from 'package:randomForest':
##
##     importance
```

```
set.seed(12345)
training <- read.csv("D:\\Handy\\Coursera\\Modul 8\\pml-training.csv")
testing <- read.csv("D:\\Handy\\Coursera\\Modul 8\\pml-testing.csv")
```

Data Cleaning

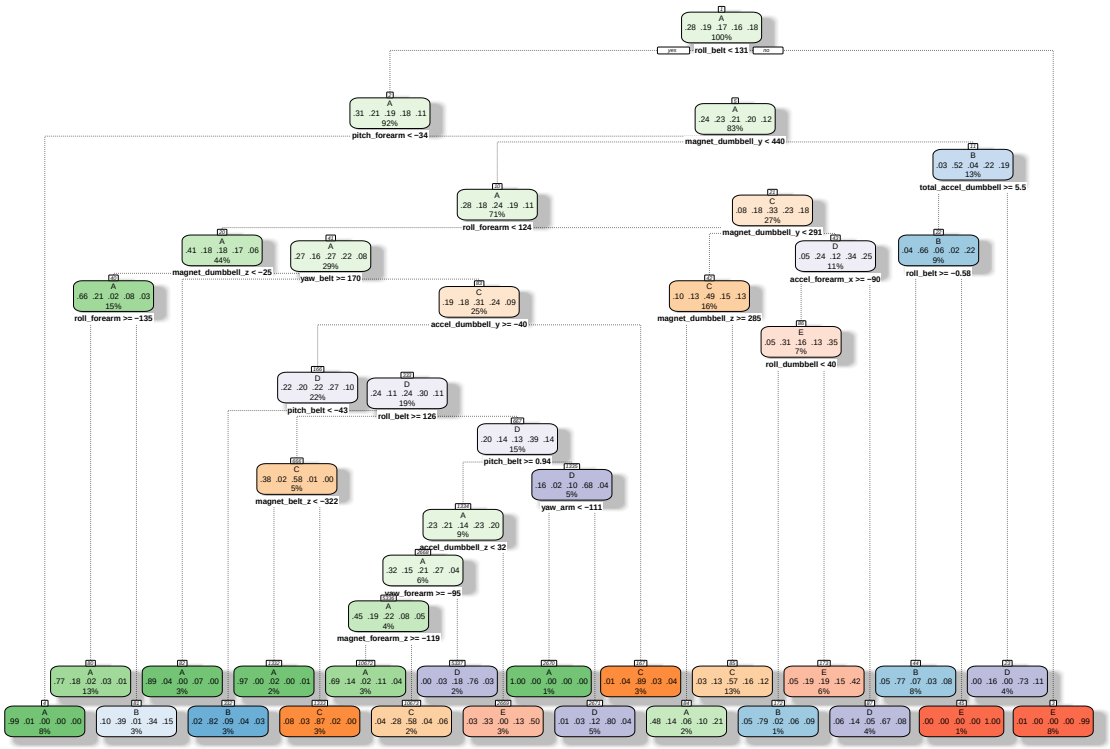
```
features <- names(testing[,colSums(is.na(testing)) == 0])[8:59]
trainclasse <- training[,c(features,"classe")]
testproblem <- testing[,c(features,"problem_id")]
```

Partitioning the data

```
inTrain <- createDataPartition(trainclasse$classe, p=0.7, list = FALSE)
myTraining <- trainclasse[inTrain,]
myTesting <- trainclasse[-inTrain,]
```

Decision Tree Prediction

```
set.seed(12345)
DTmodel <- rpart(classe ~ ., data = myTraining, method = "class")
fancyRpartPlot(DTmodel)
```



Rattle 2020−Jan−14 16:06:03 handyg062831

```
DTpredict <- predict(DTmodel, myTesting, type = "class")
confusionMatrix(DTpredict, myTesting$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1532  176   28   48   41
##          B   54  585   57   64   76
##          C   35  154  819  134  126
##          D   25   76   58  631   56
##          E   28  148   64   87  783
##
## Overall Statistics
##
##                Accuracy : 0.7392
##                  95% CI : (0.7277, 0.7503)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6692
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
```

```
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9152  0.51361   0.7982   0.6546   0.7237
## Specificity            0.9304  0.94711   0.9076   0.9563   0.9319
## Pos Pred Value         0.8395  0.69976   0.6459   0.7459   0.7054
## Neg Pred Value         0.9650  0.89028   0.9552   0.9339   0.9374
## Prevalence             0.2845  0.19354   0.1743   0.1638   0.1839
## Detection Rate         0.2603  0.09941   0.1392   0.1072   0.1331
## Detection Prevalence   0.3101  0.14206   0.2155   0.1438   0.1886
## Balanced Accuracy      0.9228  0.73036   0.8529   0.8054   0.8278
```

Random Forest Prediction

```
RFmodel <- randomForest(classe ~ ., data = myTraining)
RFpredict <- predict(RFmodel, myTesting, type = "class")
confusionMatrix(RFpredict, myTesting$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1673    4    0    0    0
##          B    1 1134    2    0    0
##          C    0    1 1024   11    0
##          D    0    0    0  953    1
##          E    0    0    0    0 1081
##
## Overall Statistics
##
##                Accuracy : 0.9966
##                  95% CI : (0.9948, 0.9979)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9957
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9994   0.9956   0.9981   0.9886   0.9991
## Specificity            0.9991   0.9994   0.9975   0.9998   1.0000
## Pos Pred Value         0.9976   0.9974   0.9884   0.9990   1.0000
## Neg Pred Value         0.9998   0.9989   0.9996   0.9978   0.9998
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2843   0.1927   0.1740   0.1619   0.1837
## Detection Prevalence   0.2850   0.1932   0.1760   0.1621   0.1837
## Balanced Accuracy      0.9992   0.9975   0.9978   0.9942   0.9995
```

Since the random forest model's accuracy was 99.3%, the out of sample error is 0.007.

We will use the random forest model to submit our predictions.

```
FinalPredict <- predict(RFmodel, testing, type = "class")
FinalPredict
```

```
##  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
```

```
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(FinalPredict)
```