Course 8 - Assigment

Synopsis

The goal of this project is to predict the manner in which people did the exercise in the testing set. You may use any of the other variables to predict with. You should create a report describing how you built your model, how you used cross validation, what you think the expected out of sample error is, and why you made the choices you did. You will also use your prediction model to predict 20 different test cases.

Data

```
rm(list=ls())
library(knitr)
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.4.4
```

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

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

Download and load the raw training data:

```
path <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
download.file(path, 'Training'); system("bunzip2 StormData.csv.bz2")
```

```
## Warning: comando ejecutado 'bunzip2 StormData.csv.bz2' tiene estatus 127
```

```
Training <- read.csv("Training", header = TRUE, sep = ",")
rm(path)
```

Download and load the raw test data:

```
path <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
download.file(path, 'Testing'); system("bunzip2 StormData.csv.bz2")
```

```
## Warning: comando ejecutado 'bunzip2 StormData.csv.bz2' tiene estatus 127
```

```
Testing <- read.csv("Testing", header = TRUE, sep = ",")
rm(path)
```

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

Clean Training and Testing

The timestamps and identity variables will not be used in the model, so they are removed from the data. Several of the columns contain mostly NA values or blanks for their observations, so these columns are also removed from the data. I set the threshold to filter down to only columns with less than 10% of NAs or blanks.

```
Training <- subset(Training, select=-c(X,user_name,raw_timestamp_part_1,raw_timestamp_part_2,cvtd_timestamp,new_window,num_w
indow))
nans <- colSums(is.na(Training))
Training <- subset(Training[nans==0])
Training <- Training[,colMeans(Training == "", na.rm = TRUE) <= .1]

Testing <- subset(Testing, select=-c(X,user_name,raw_timestamp_part_1,raw_timestamp_part_2,cvtd_timestamp,new_window,num_win
dow))
nans <- colSums(is.na(Testing))
Testing <- subset(Testing[nans==0])
rm(nans)
```

There are enough samples to split the Testing dataset in a training and testing.

```
table(Training$classe)
```

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

```
set.seed(12345)
inTrain <- createDataPartition(Training$classe, p = .75, list = FALSE)
training <- Training[inTrain,]
testing <- Training[-inTrain,]
```

Decision tree

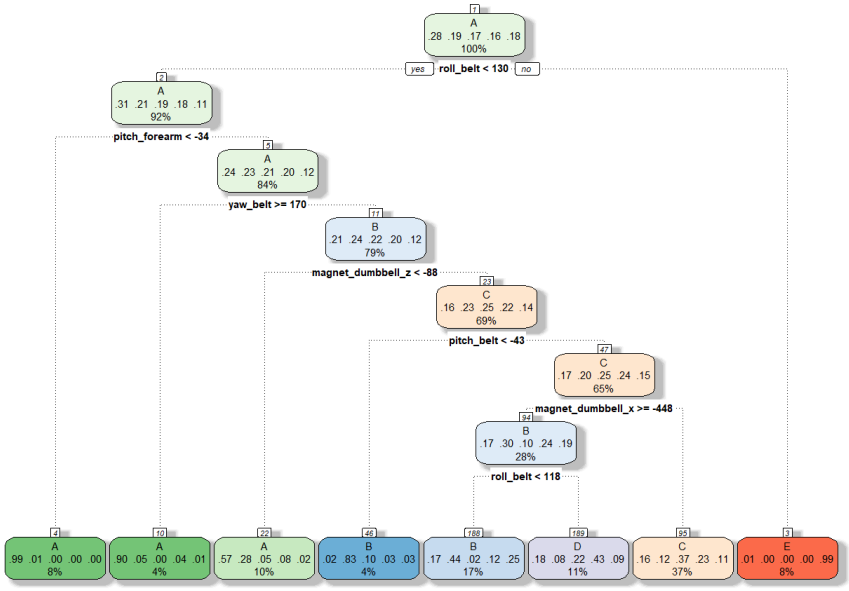
Since we want to classify which kind of exercise did the people, le’ts try with a tree.

```
modFit <- train(classe ~., method="rpart",data=training)
library(rattle)
```

```
## Warning: package 'rattle' was built under R version 3.4.4
```

```
## Rattle: A free graphical interface for data science with R.
## Versión 5.1.0 Copyright (c) 2006-2017 Togaware Pty Ltd.
## Escriba 'rattle()' para agitar, sacudir y  rotar sus datos.
```

```
fancyRpartPlot(modFit$finalModel)
```



Rattle 2018-jul.-05 19:45:41 xadel

Let's use the testing dataset to see if the tree decision is working.

```
confusionMatrix(testing$classe,predict(modFit,testing))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   A   B   C   D   E
##          A 870 159 273  88   5
##          B 162 530 214  43   0
##          C  29  36 674 116   0
##          D  46 136 429 193   0
##          E  16 221 224  51 389
##
## Overall Statistics
##
##                Accuracy : 0.5416
##                  95% CI : (0.5275, 0.5556)
##     No Information Rate : 0.3699
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4245
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.7747   0.4898   0.3716  0.39308  0.98731
## Specificity            0.8611   0.8904   0.9414  0.86155  0.88647
## Pos Pred Value         0.6237   0.5585   0.7883  0.24005  0.43174
## Neg Pred Value         0.9279   0.8604   0.7184  0.92732  0.99875
## Prevalence             0.2290   0.2206   0.3699  0.10012  0.08034
## Detection Rate         0.1774   0.1081   0.1374  0.03936  0.07932
## Detection Prevalence   0.2845   0.1935   0.1743  0.16395  0.18373
## Balanced Accuracy      0.8179   0.6901   0.6565  0.62731  0.93689
```

As we can observed, only the exercise E is well discriminate and the Accuracy in this case is really low. For these reason, a more complex algorithm is needed. So, let's try with a random forest

Random Forest

```
modFit2 <- train(classe ~., method="rf",data=training, ntrees=100)
confusionMatrix(testing$classe,predict(modFit2,testing))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1395    0    0    0    0
##          B    6  938    5    0    0
##          C    0    3  848    4    0
##          D    0    0   11  793    0
##          E    0    0    2    5  894
##
## Overall Statistics
##
##                Accuracy : 0.9927
##                  95% CI : (0.9899, 0.9949)
##     No Information Rate : 0.2857
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9907
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9957   0.9968   0.9792   0.9888   1.0000
## Specificity            1.0000   0.9972   0.9983   0.9973   0.9983
## Pos Pred Value         1.0000   0.9884   0.9918   0.9863   0.9922
## Neg Pred Value         0.9983   0.9992   0.9956   0.9978   1.0000
## Prevalence             0.2857   0.1919   0.1766   0.1635   0.1823
## Detection Rate         0.2845   0.1913   0.1729   0.1617   0.1823
## Detection Prevalence   0.2845   0.1935   0.1743   0.1639   0.1837
## Balanced Accuracy      0.9979   0.9970   0.9887   0.9930   0.9991
```

Now the accuracy is 99% with really high sensitivity and specificity for the 5 classes of exercise.

Testing

The random forest model is used to predict the 20 classe variables in the Testing data.

```
exercises <- data.frame(id = Testing$problem_id, Prediction = c(as.character(predict(modFit2, Testing))))
```