Activity Data Prediction

From Coursera:

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).

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://groupware.les.inf.puc-rio.br/har. If you use the document you create for this class for any purpose please cite them as they have been very generous in allowing their data to be used for this kind of assignment. ##Data Processing

```r
library(ggplot2)
library(lattice)
library(caret)
library(stats)
library(rpart)
library(RColorBrewer)
library(rattle)
```

```
## Rattle: A free graphical interface for data mining with R.
## Version 3.4.1 Copyright (c) 2006-2014 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```r
library(rpart.plot)
library(e1071)
library(randomForest)
```

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

```
set.seed(930)
```

```
download.file('https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv','pml-training.csv'

download.file('https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv
','pml-testing.csv', method = "curl")

# read data
train<-read.csv("./pml-training.csv", na.strings = c("NA", "#DIV/0!",""))
test<-read.csv("./pml-testing.csv", na.strings = c("NA", "#DIV/0!",""))
```

Omiting variables with over 60% NA

```
omit <- which((colSums(!is.na(train)) >= 0.6*nrow(train)))
train.omit <- train[,omit]
test.omit <- test[,omit]
```

Partition training data 60:40 into ‘train_part’ and ‘test_part’ for purposes of model building

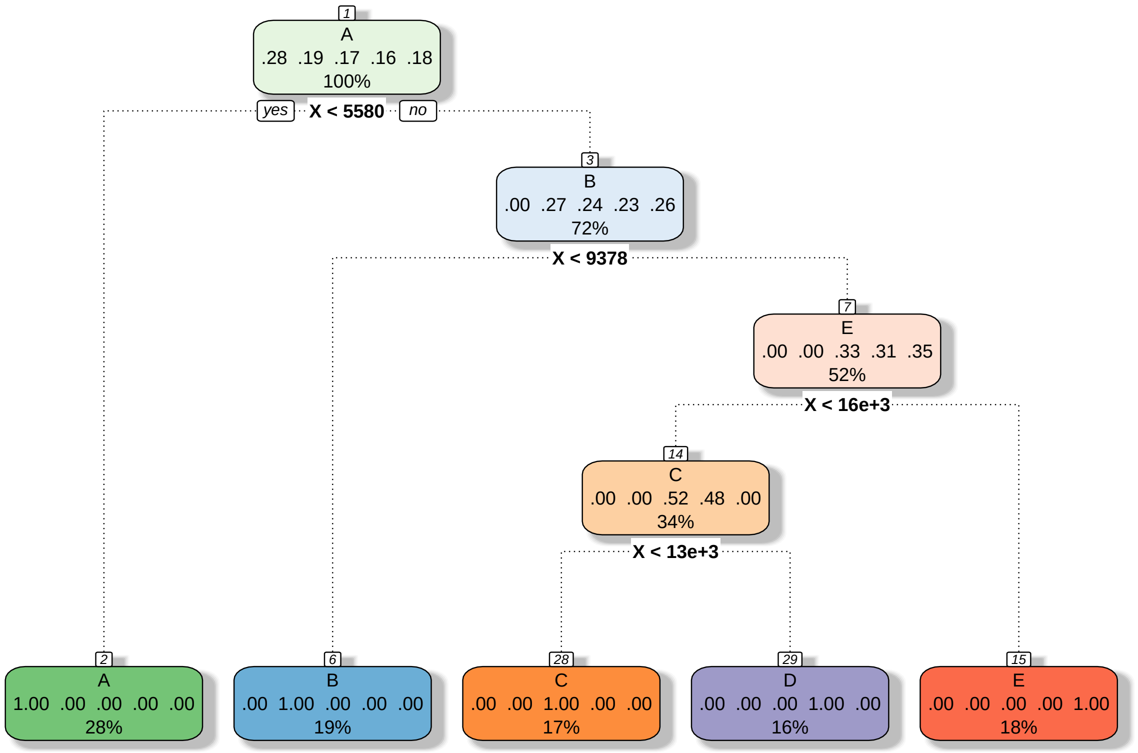
```
train_div  <- createDataPartition(train.omit$classe, p = 0.6, list = FALSE)
train_part    <- train.omit[train_div, ]
test_part     <- train.omit[-train_div, ]
```

Decision Tree

Let’s start the model building process with a decision tree algorithim.

```
class<-train_part$classe

dtree_mod <- rpart(classe ~ ., data=train_part, method="class")
fancyRpartPlot(dtree_mod)
```



Rattle 2015−Jun−21 14:25:03 gabrielaolemberg

```
prediction <- predict(dtree_mod, test_part, type = "class")
```

```
confusionMatrix(prediction, test_part$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2232    0    0    0    0
##          B    0 1518    0    0    0
##          C    0    0 1368    1    0
##          D    0    0    0 1285    1
##          E    0    0    0    0 1441
##
## Overall Statistics
##
##                Accuracy : 0.9997
##                  95% CI : (0.9991, 1)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9997
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   1.0000   1.0000   0.9992   0.9993
## Specificity            1.0000   1.0000   0.9998   0.9998   1.0000
## Pos Pred Value         1.0000   1.0000   0.9993   0.9992   1.0000
## Neg Pred Value         1.0000   1.0000   1.0000   0.9998   0.9998
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2845   0.1935   0.1744   0.1638   0.1837
## Detection Prevalence   0.2845   0.1935   0.1745   0.1639   0.1837
## Balanced Accuracy      1.0000   1.0000   0.9999   0.9995   0.9997
```

The confusion matrix looks pretty good. Acccuracy is at 99.97%. However, let's see if we can improve prediciton accuracy by generating a larger amount of bootstrapped trees, i.e. Random Forest.

Random Forest: Generating a larger number of bootstrapped trees

Model accuracy is now at 100%

```
random_forest<-randomForest(classe~., data = train_part)
```

```
predictition_RF<- predict(random_forest, test_part, type = "class")
```

```
confusionMatrix(predictition_RF, test_part$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2232    0    0    0    0
##          B    0 1518    0    0    0
##          C    0    0 1368    0    0
##          D    0    0    0 1286    0
##          E    0    0    0    0 1442
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (0.9995, 1)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   1.0000   1.0000   1.0000   1.0000
## Specificity            1.0000   1.0000   1.0000   1.0000   1.0000
## Pos Pred Value         1.0000   1.0000   1.0000   1.0000   1.0000
## Neg Pred Value         1.0000   1.0000   1.0000   1.0000   1.0000
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Prevalence   0.2845   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      1.0000   1.0000   1.0000   1.0000   1.0000
```

Submit results for test set

```
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(predictition_RF)
```