Predicting Exercise Routine from Fitbit Data

by kakilima

Executive Summary

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).

According to this information, the variable «classe» has five possible values: A (correct exercise), B-E (common mistakes).

Prepare R Environment & Load Data

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.

Loading Data

```
setwd('c://coursera')
traindata <- 'pml-training.csv'
testdata <- 'pml-testing.csv'

trainURL <- 'https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv'
testURL <- 'https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv'

if(!file.exists(traindata)) {
    download.file(trainURL, destfile = traindata)
}
if(!file.exists(testdata)) {
    download.file(testURL, destfile = testdata)
}

trainn <- read.csv(traindata)
test <- read.csv(testdata)
```

Loading Required Libraries

```
library(plyr)
library(caret) # Machine learning library
library(ggplot2)
library(randomForest) # classification and regression
library(rpart) # regression partitioning and trees
library(rpart.plot) #pretty printing trees
library(rattle)
```

Preliminary Data Processing

Train and test dataset have different columns.

```
classe <- names(train)[colnames(train)!=colnames(test)]
problem_id <- names(test)[colnames(train)!=colnames(test)]
```

We are going to remove unnecesary and steady columns and detect columns with missing values

```
#Remove first five columns
trainn <- trainn[,6:dim(trainn)[2]]
test <- test[,6:dim(test)[2]]

# Select not steady columns
not_steady <- !nearZeroVar(trainn, saveMetrics = T)$nzv
trainn <- trainn[,not_steady]
test <- test[,not_steady]

# Find columns > 70% na values
nmissing <- function(x) sum(is.na(x))
few_na<-!colwise(nmissing)(trainn)> 0.7*dim(trainn)[1]
trainn <- trainn[,few_na]
test <- test[,few_na]
```

Let us take a look at the subset of variables

```
names(trainn)
```

```
##  [1] "num_window"           "roll_belt"            "pitch_belt"
##  [4] "yaw_belt"             "total_accel_belt"     "gyros_belt_x"
##  [7] "gyros_belt_y"         "gyros_belt_z"         "accel_belt_x"
## [10] "accel_belt_y"         "accel_belt_z"         "magnet_belt_x"
## [13] "magnet_belt_y"        "magnet_belt_z"        "roll_arm"
## [16] "pitch_arm"            "yaw_arm"              "total_accel_arm"
## [19] "gyros_arm_x"          "gyros_arm_y"          "gyros_arm_z"
## [22] "accel_arm_x"          "accel_arm_y"          "accel_arm_z"
## [25] "magnet_arm_x"         "magnet_arm_y"         "magnet_arm_z"
## [28] "roll_dumbbell"        "pitch_dumbbell"       "yaw_dumbbell"
## [31] "total_accel_dumbbell" "gyros_dumbbell_x"     "gyros_dumbbell_y"
## [34] "gyros_dumbbell_z"     "accel_dumbbell_x"     "accel_dumbbell_y"
```

```
## [37] "accel_dumbbell_z"     "magnet_dumbbell_x"    "magnet_dumbbell_y"
## [40] "magnet_dumbbell_z"    "roll_forearm"         "pitch_forearm"
## [43] "yaw_forearm"          "total_accel_forearm"  "gyros_forearm_x"
## [46] "gyros_forearm_y"      "gyros_forearm_z"      "accel_forearm_x"
## [49] "accel_forearm_y"      "accel_forearm_z"      "magnet_forearm_x"
## [52] "magnet_forearm_y"     "magnet_forearm_z"     "classe"
```

Select training and cross validation sets.

```
set.seed(888)
inTrain <- createDataPartition(y=trainn$classe,p=0.75,list=FALSE)
training <- trainn[inTrain,]
validation <- trainn[-inTrain,]
```

Data Modelling

We are going to test the several models and compare the goodness of fit

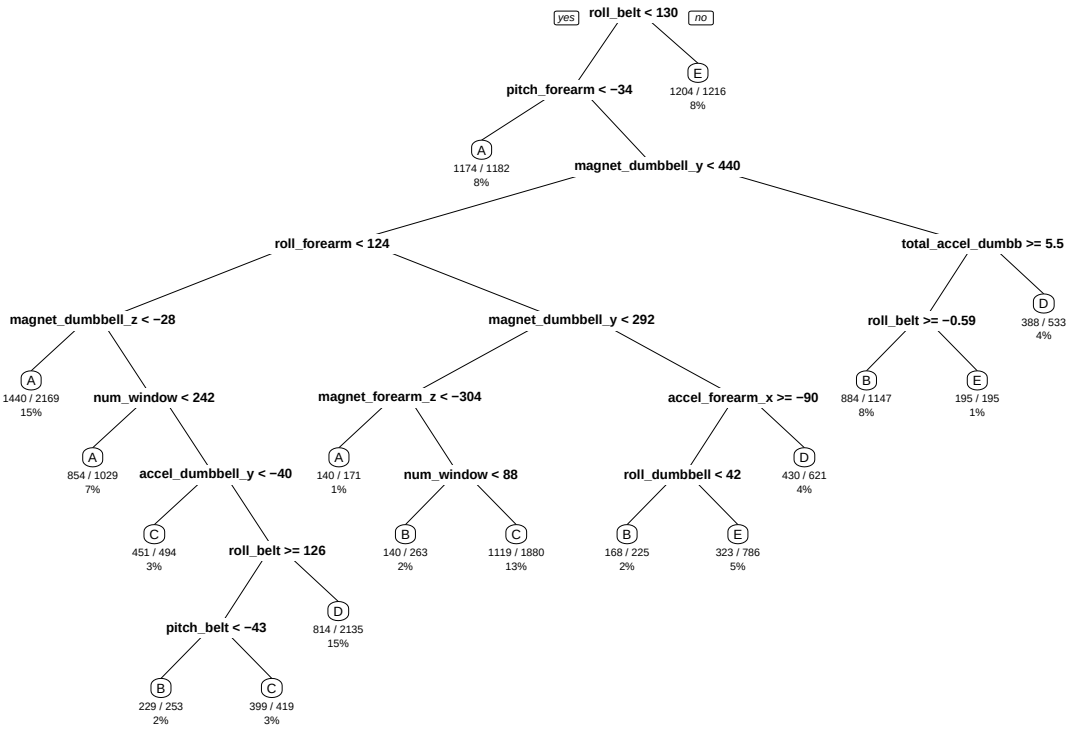
Decision Trees

```
model_dt <- rpart(classe ~ ., data=training, method="class")
predict_dt <- predict(model_dt, validation, type="class")
conf_dt<-confusionMatrix(predict_dt,validation$classe)
conf_dt
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1198  159    7   78   50
##          B   31  473   34   16   68
##          C   22   50  641  123   68
##          D  131  209  121  547   96
##          E   13   58   52   40  619
##
## Overall Statistics
##
##                Accuracy : 0.7092
##                  95% CI : (0.6963, 0.7219)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6322
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8588  0.49842   0.7497   0.6803   0.6870
## Specificity            0.9162  0.96233   0.9350   0.8641   0.9593
```

```
## Pos Pred Value         0.8029  0.76045   0.7091   0.4955   0.7916
## Neg Pred Value         0.9423  0.88884   0.9465   0.9324   0.9316
## Prevalence             0.2845  0.19352   0.1743   0.1639   0.1837
## Detection Rate         0.2443  0.09645   0.1307   0.1115   0.1262
## Detection Prevalence   0.3042  0.12684   0.1843   0.2251   0.1595
## Balanced Accuracy      0.8875  0.73037   0.8424   0.7722   0.8231
```

```
rpart.plot(model_dt, extra=102, under = TRUE, faclen=0)
```



The accuracy of the decision tree is 0.7092.

Random Forests

```
set.seed(888)
model_rf <- randomForest(classe ~ ., data=training, type="class")
predict_rf <- predict(model_rf, validation)
conf_rf<-confusionMatrix(predict_rf,validation$classe)
conf_rf
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1395    3    0    0    0
```

```
##          B    0  946    2    0    0
##          C    0    0  853    1    0
##          D    0    0    0  803    2
##          E    0    0    0    0  899
##
## Overall Statistics
##
##                Accuracy : 0.9984
##                  95% CI : (0.9968, 0.9993)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9979
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9968   0.9977   0.9988   0.9978
## Specificity            0.9991   0.9995   0.9998   0.9995   1.0000
## Pos Pred Value         0.9979   0.9979   0.9988   0.9975   1.0000
## Neg Pred Value         1.0000   0.9992   0.9995   0.9998   0.9995
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2845   0.1929   0.1739   0.1637   0.1833
## Detection Prevalence   0.2851   0.1933   0.1741   0.1642   0.1833
## Balanced Accuracy      0.9996   0.9982   0.9987   0.9991   0.9989
```

The accuracy of the random forest is 0.9984. The comparison between the two methods shows that **random forests** performs better. There's a big improvement in model accuracy.

Test data prediction

```
predict_rt_test <- predict(model_rf, test)
predict_rt_test
```

```
##  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
```

Prepare Submission (Coursera)

```
predictions <- as.vector(predict_rt_test)
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(predictions)
```