Effective Exercise Predictions Using Machine Learning

Kevin Clifford

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.

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

Data Preprocessing

```
library(caret)

## Warning: package 'caret' was built under R version 3.1.3

## Loading required package: lattice
## Loading required package: ggplot2

library(rpart)
library(rpart.plot)

## Warning: package 'rpart.plot' was built under R version 3.1.3

library(randomForest)

## Warning: package 'randomForest' was built under R version 3.1.3

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

library(rattle)

## Warning: package 'rattle' was built under R version 3.1.3

## 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.
```

Read the Data

```r
trainRaw <- read.csv("pml-training.csv")
testRaw <- read.csv("pml-testing.csv")
```

Select relevant predictors and setup train and test dataset

```r
missingCols <- sapply(testRaw, function (x) any(is.na(x) | x == ""))
predictors <- !missingCols & grepl("belt|[^(fore)]arm|dumbbell|forearm",
names(missingCols))
predictorCols <- names(missingCols)[predictors]
colsrelevant <- c("classe", predictorCols)
trainRaw <- trainRaw[, colsrelevant]
testRaw <- testRaw[, predictorCols]
```

Split data into train and validation sets

Split raw training set into training data set (75%) and a cross validation data set (25%)

```r
set.seed(12345)
inTrain <- createDataPartition(trainRaw$classe, p=0.75, list=F)
trainData <- trainRaw[inTrain, ]
valData <- trainRaw[-inTrain, ]
```

Model the Data

Using the Random Forest algorithm with 200 trees and 5-fold cross validation

```r
RFCtrl <- trainControl(method="cv", 5)
RFModel <- train(classe ~ ., data=trainData, method="rf", trControl=RFCtrl,
ntree=200)
RFModel

## Random Forest
##
## 14718 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
##
## Summary of sample sizes: 11773, 11773, 11775, 11775, 11776
##
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa      Accuracy SD  Kappa SD
##    2    0.9911670  0.9888261  0.001360691  0.001721172
##   27    0.9915749  0.9893409  0.003271811  0.004140123
##   52    0.9887891  0.9858175  0.002497638  0.003161333
##
```

```
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 27.
```

Validate the model against the validation dataset

```
RFPredict <- predict(RFModel, valData)
confusionMatrix(valData$classe, RFPredict)

## 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    2  850    3    0
##          D    0    0   11  793    0
##          E    0    0    2    5  894
## 
## Overall Statistics
## 
##                Accuracy : 0.9931
##                  95% CI : (0.9903, 0.9952)
##     No Information Rate : 0.2857
##     P-Value [Acc > NIR] : < 2.2e-16
## 
##                   Kappa : 0.9912
##  Mcnemar's Test P-Value : NA
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9957   0.9979   0.9793   0.9900   1.0000
## Specificity            1.0000   0.9972   0.9988   0.9973   0.9983
## Pos Pred Value         1.0000   0.9884   0.9942   0.9863   0.9922
## Neg Pred Value         0.9983   0.9995   0.9956   0.9980   1.0000
## Prevalence             0.2857   0.1917   0.1770   0.1633   0.1823
## Detection Rate         0.2845   0.1913   0.1733   0.1617   0.1823
## Detection Prevalence   0.2845   0.1935   0.1743   0.1639   0.1837
## Balanced Accuracy      0.9979   0.9975   0.9890   0.9937   0.9991
```

Accuracy:

```
as.numeric(confusionMatrix(valData$classe, RFPredict)$overall[1])

## [1] 0.9930669
```

Out-Of-Sample ERROR:

```
1 - as.numeric(confusionMatrix(valData$classe, RFPredict)$overall[1])

## [1] 0.006933116
```

Display the Model

```
varImp(RFModel)

## rf variable importance
##
##   only 20 most important variables shown (out of 52)
##
##                      Overall
## roll_belt             100.00
## pitch_forearm          57.68
## yaw_belt               52.96
## magnet_dumbbell_z      42.36
## pitch_belt             41.73
## magnet_dumbbell_y      40.87
## roll_forearm           39.59
## accel_dumbbell_y       20.22
## magnet_dumbbell_x      17.37
## roll_dumbbell          16.64
## accel_forearm_x        16.61
## total_accel_dumbbell   14.89
## magnet_belt_z          14.49
## accel_dumbbell_z       13.90
## magnet_forearm_z       13.60
## accel_belt_z           11.72
## magnet_belt_y          11.10
## yaw_arm                10.79
## gyros_belt_z           10.73
## magnet_belt_x          10.00
```

Predict using downloaded Test dataSet

Apply validated model against the Test data

```
result <- predict(RFModel, testRaw)
result

##  [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
```

Project Submission

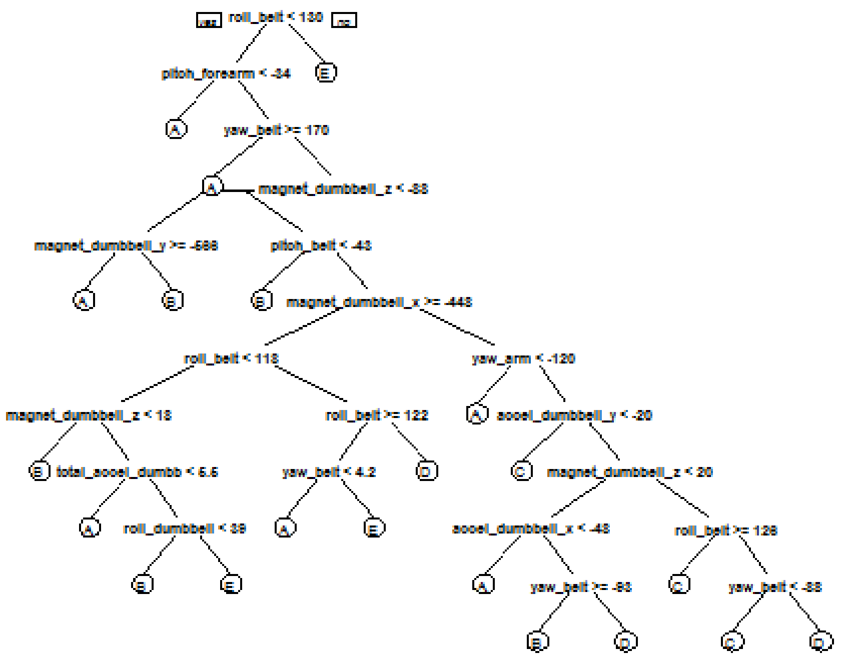
Generate files for submission

```
pml_write_files = function(x){
  n = length(x)
  path <- "results"
  for(i in 1:n){
    filename = paste0("problem_id_",i,".txt")
    write.table(x[i],file=file.path(path,
filename),quote=FALSE,row.names=FALSE,col.names=FALSE)
```

```
  }
}
pml_write_files(result)
```

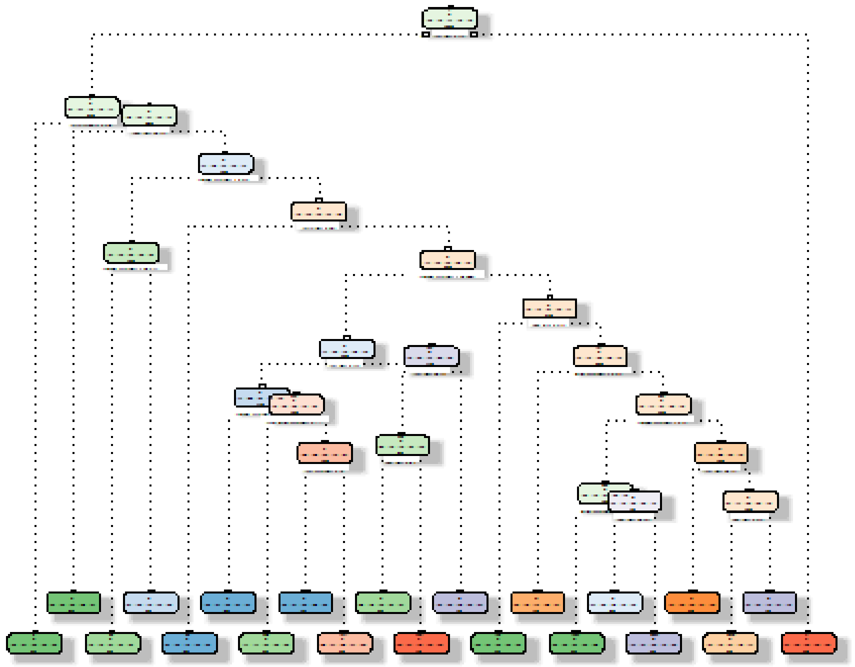
Appendix Of Figures

```
treeModel <- rpart(classe ~ ., data=trainData, method="class")
prp(treeModel)
```



```
fancyRpartPlot(treeModel)

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


Rattle 2015-Mar-18 10:38:29 usw131d