Practical Machine Learning: Course Project

Jenina Halitsky

November 23, 2014 ==========

Synopsis

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).

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. 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.

1. Your submission should consist of a link to a Github repo with your R markdown and compiled HTML file describing your analysis. Please constrain the text of the writeup to < 2000 words and the number of figures to be less than 5. It will make it easier for the graders if you submit a repo with a gh-pages branch so the HTML page can be viewed online (and you always want to make it easy on graders :-).
2. You should also apply your machine learning algorithm to the 20 test cases available in the test data above. Please submit your predictions in appropriate format to the programming assignment for automated grading. See the programming assignment for additional details.

Note for Peer Analysis
Reproducibility - Due to security concerns with the exchange of R code, the code will not be run during the evaluation by classmates. If you download the repo, you will be able to view the compiled HTML version of my analysis.

Data Processing

The data for this analysis comes in the form of a comma-separated-value (CSV) file. The data files were downloaded from the Coursera Practical Machine Learning website on **November 23, 2014**.

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.
==========

```
        library(caret)
```

Setup Configurations & Libraries

```
## Warning: package 'caret' was built under R version 3.1.2

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

```
        library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.1.1

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

```
        library(foreach)
        set.seed(2048)
        options(warn=-1)
```

```
        training_data <- read.csv("pml-training.csv", na.strings=c("#DIV/0!") )
        evaluation_data <- read.csv("pml-testing.csv", na.strings=c("#DIV/0!") )
```

Load Data

```
        for(i in c(8:ncol(training_data)-1)) {training_data[,i] = as.numeric(as.character(training_data
        for(i in c(8:ncol(evaluation_data)-1)) {evaluation_data[,i] = as.numeric(as.character(evaluatio
```

Modify Columns to be Numeric ==========
Analysis

Determine Feature Set Because some of the columns were mostly blank, I determined that these will not contribute to the prediction. By removing user name, timestamps and windows, the feature set will have only complete columns.

```
        feature_set <- colnames(training_data[colSums(is.na(training_data)) == 0])[-(1:7)]
        model_data <- training_data[feature_set]
        feature_set
```

```
##  [1] "roll_belt"            "pitch_belt"           "yaw_belt"
##  [4] "total_accel_belt"     "gyros_belt_x"         "gyros_belt_y"
##  [7] "gyros_belt_z"         "accel_belt_x"         "accel_belt_y"
## [10] "accel_belt_z"         "magnet_belt_x"        "magnet_belt_y"
## [13] "magnet_belt_z"        "roll_arm"             "pitch_arm"
## [16] "yaw_arm"              "total_accel_arm"      "gyros_arm_x"
## [19] "gyros_arm_y"          "gyros_arm_z"          "accel_arm_x"
## [22] "accel_arm_y"          "accel_arm_z"          "magnet_arm_x"
## [25] "magnet_arm_y"         "magnet_arm_z"         "roll_dumbbell"
## [28] "pitch_dumbbell"       "yaw_dumbbell"         "total_accel_dumbbell"
## [31] "gyros_dumbbell_x"     "gyros_dumbbell_y"     "gyros_dumbbell_z"
## [34] "accel_dumbbell_x"     "accel_dumbbell_y"     "accel_dumbbell_z"
## [37] "magnet_dumbbell_x"    "magnet_dumbbell_y"    "magnet_dumbbell_z"
## [40] "roll_forearm"         "pitch_forearm"        "yaw_forearm"
## [43] "total_accel_forearm"  "gyros_forearm_x"      "gyros_forearm_y"
## [46] "gyros_forearm_z"      "accel_forearm_x"      "accel_forearm_y"
## [49] "accel_forearm_z"      "magnet_forearm_x"     "magnet_forearm_y"
## [52] "magnet_forearm_z"     "classe"
```

```
idx <- createDataPartition(y=model_data$classe, p=0.75, list=FALSE )
training <- model_data[idx,]
testing <- model_data[-idx,]
```

Build Model Data from Feature Set

```
x <- training[-ncol(training)]
y <- training$classe
rf <- foreach(ntree=rep(150, 6), .combine=randomForest::combine, .packages='randomForest') %dopa
        randomForest(x, y, ntree=ntree)
}
```

By using parallel processing to build the model, we using 5 random forests with 150 trees each.

```
predictions1 <- predict(rf, newdata=training)
confusionMatrix(predictions1,training$classe)
```

Confusion Matrix and Statistics

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 4185    0    0    0    0
```

```
##          B    0 2848    0    0    0
##          C    0    0 2567    0    0
##          D    0    0    0 2412    0
##          E    0    0    0    0 2706
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (1, 1)
##     No Information Rate : 0.284
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 1
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity             1.000    1.000    1.000    1.000    1.000
## Specificity             1.000    1.000    1.000    1.000    1.000
## Pos Pred Value          1.000    1.000    1.000    1.000    1.000
## Neg Pred Value          1.000    1.000    1.000    1.000    1.000
## Prevalence              0.284    0.194    0.174    0.164    0.184
## Detection Rate          0.284    0.194    0.174    0.164    0.184
## Detection Prevalence    0.284    0.194    0.174    0.164    0.184
## Balanced Accuracy       1.000    1.000    1.000    1.000    1.000
```

```
predictions2 <- predict(rf, newdata=testing)
confusionMatrix(predictions2,testing$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1394    4    0    0    0
##          B    1  945    8    0    0
##          C    0    0  844    6    0
##          D    0    0    3  797    1
##          E    0    0    0    1  900
##
## Overall Statistics
##
##                Accuracy : 0.995
##                  95% CI : (0.993, 0.997)
##     No Information Rate : 0.284
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.994
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity             0.999    0.996    0.987    0.991    0.999
```

```
## Specificity             0.999    0.998    0.999    0.999    1.000
## Pos Pred Value          0.997    0.991    0.993    0.995    0.999
## Neg Pred Value          1.000    0.999    0.997    0.998    1.000
## Prevalence              0.284    0.194    0.174    0.164    0.184
## Detection Rate          0.284    0.193    0.172    0.163    0.184
## Detection Prevalence    0.285    0.195    0.173    0.163    0.184
## Balanced Accuracy       0.999    0.997    0.993    0.995    0.999
```

==========

Conclusions

The test data was approximately 99% accurate when experimenting with the Confusion Matrix that I feel that the submitted test cases were correct.

==========

Submission

Prepare the submission. (using COURSERA provided code)

```
        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)
                }
        }


        x <- evaluation_data
        x <- x[feature_set[feature_set!='classe']]
        answers <- predict(rf, newdata=x)

        answers
```

```
##  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(answers)
```