Practical Machine Learning Coursera Course - project

Sunday, November 22, 2015

Executive summary

This document is the project for a Coursera class: Practical Machine Learning. It uses the personal activity data taken by personal devices such as Jawbone Up, Nike FuelBand, and Fitbit. The goal of this project is to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants and analyze/predict data taken when performing barbell lifts correctly and incorrectly in 5 different ways, Specifically, to predict the manner in which they did the exercise.

System info:

```
Sys.info()[1:2]
```

```
##   sysname   release
## "Windows"   "7 x64"
```

```
R.version.string
```

```
## [1] "R version 3.2.2 (2015-08-14)"
```

Data

The training data for this project are available here:

- https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv (https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv)
- contains sensor data and execution type data

The test data are available here:

- https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv (https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv)
- contains sensor data

The aim is to make prediction algorithm trained on the data from the pml-training.csv file to predict the execution type in the pml-testing.csv data file.

Loading and exploring data

We can see that there are 19622 observations and 160 observed variables. ALso from the bellow we can see that variables contain the name of a sensor (_belt, _arm, _dumbbell, _forearm). Second column shows the name of the watched person, the last column contains the values A to E, which show the classe type = the manner in which they did the exercise. This sais that not all the variables are generated by the sensors.

```
training_data <- read.table("pml-training.csv", header = TRUE, sep = ",", na.strings = c("NA", "#DIV/0!"))

dim(training_data)
```

```
## [1] 19622   160
```

```
head(names(training_data), 8)
```

```
## [1] "X"                    "user_name"            "raw_timestamp_part_1"
## [4] "raw_timestamp_part_2" "cvtd_timestamp"       "new_window"
## [7] "num_window"           "roll_belt"
```

```
head(names(training_data), 3)
```

```
## [1] "X"                    "user_name"            "raw_timestamp_part_1"
```

Data needed for the prediction model and removing NA values.

```
column_names = grep(pattern = "_belt|_arm|_dumbbell|_forearm", names(training_data))
data = training_data[, c(column_names,160)]

na_data = is.na(data)
na_cols = which(colSums(na_data) > 19000)
data = data[, -na_cols]
```

Algorithm

Split data into training (75%) and testing sets.

```
library(caret)
set.seed(1234)

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

Making predictor

Random forest is used to fit the predictor to the training set.

```
library(randomForest)

randForest <- randomForest(classe~., data=training)
randForest
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = training)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 0.44%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 4182    2    0    0    1 0.0007168459
## B   12 2832    4    0    0 0.0056179775
## C    0   13 2550    4    0 0.0066225166
## D    0    0   17 2393    2 0.0078772803
## E    0    0    4    6 2696 0.0036954915
```

The resulting predictor has a low OOB (out-of-bag) error estimate. The confusion matrix indicates that the predictor is accurate on that set.

Testing

Appling model on the testing sample (not the special testing set) to get an estimate of out of sample error.

```
predictionTest1 <- predict(randForest, newdata = testing, type = "class")
confusionMatrix(predictionTest1, testing$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1395    1    0    0    0
##          B    0  946   11    0    0
##          C    0    2  843    8    0
##          D    0    0    1  796    0
##          E    0    0    0    0  901
##
## Overall Statistics
##
##                Accuracy : 0.9953
##                  95% CI : (0.993, 0.997)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9941
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9968   0.9860   0.9900   1.0000
## Specificity            0.9997   0.9972   0.9975   0.9998   1.0000
## Pos Pred Value         0.9993   0.9885   0.9883   0.9987   1.0000
## Neg Pred Value         1.0000   0.9992   0.9970   0.9981   1.0000
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2845   0.1929   0.1719   0.1623   0.1837
## Detection Prevalence   0.2847   0.1951   0.1739   0.1625   0.1837
## Balanced Accuracy      0.9999   0.9970   0.9917   0.9949   1.0000
```

Accuracy and the Cohen's kappa indicator of concordance show that the predictor has a low out of sample error rate.

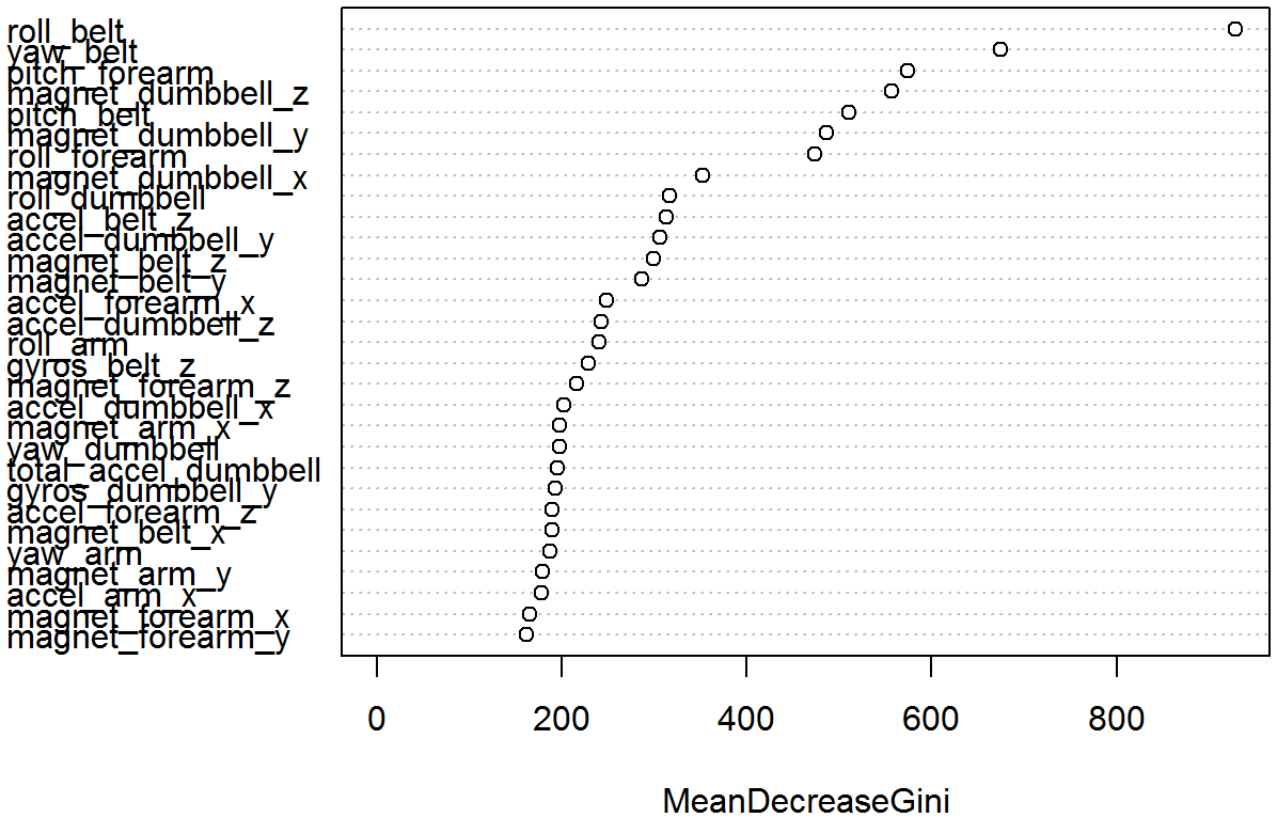
Testing on the special data test and preparing files for submission

```
testing_final <- read.csv("pml-testing.csv", na.strings=c("NA","#DIV/0!",""))
predictionTest2 <- predict(randForest, testing_final, type = "class")

files = function(x){
     n = length(x)
     for(i in 1:n){
         name = paste0("problem_id_",i,".txt")
         write.table(x[i],file=name,quote=FALSE,row.names=FALSE,col.names=FALSE)
     }
 }
files(predictionTest2)
```

Appendix

```
varImpPlot(randForest)
```




```
plot(randForest)
```