Assignment: Prediction Assignment Writeup

Gayathri Kulathumani

March 31, 2016

Executive Summary

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, that goal is 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.

Exploratory Data Analysis

```r
#loading library

library(lattice)
library(ggplot2)
library(caret)
library(rpart)
library(randomForest)
```

```
## randomForest 4.6-12

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':
##
##     margin
```

```r
#library(ElemStatLearn)
library(data.table)

#reproducibility
set.seed(321)

# read data
trainUrl <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
testUrl <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
training <- read.csv(url(trainUrl))
testing <- read.csv(url(testUrl))
```

Cleaning Data

Before we do anything, we need to clean the data as there are many null or empty values in the data

```r
#Removing missing data
RemoveMissing <- function(d) {
  noMiss <- !sapply(d, function(x) any(is.na(x)))
  d <- d[, noMiss]

  noMiss <- !sapply(d, function(x) any(x==""))
  d <- d[, noMiss]
  return(d)
}


trainD<- RemoveMissing(training)
testD<- RemoveMissing(testing)

# To clean anything that isnt a predictor variable

col.rm <- c("X", "user_name", "raw_timestamp_part_1", "raw_timestamp_part_2",
            "cvtd_timestamp", "new_window", "num_window")

d.rm <- which(colnames(trainD) %in% col.rm)
trainD <- trainD[, -d.rm]

d.rm <- which(colnames(testD) %in% col.rm)
testD <- testD[, -d.rm]

trainD$classe <- as.factor(trainD$classe)

testD <- testD[,-ncol(testD)]
```

Partioning training and testing data set

Splitting test and training data sets

```r
inTrain = createDataPartition(trainD$classe, p=0.60, list=FALSE)
trainingD = trainD[inTrain,]
validatingD = trainD[-inTrain,]
```

```r
preObj <- preProcess(trainingD[, -ncol(trainingD)], method=c("center","scale"))
preObj
```

```
## Created from 11776 samples and 52 variables
##
## Pre-processing:
##   - centered (52)
##   - ignored (0)
##   - scaled (52)
```

```r
preClass<-predict(preObj,trainingD[, -ncol(trainingD)])
DTrainClass <- data.table(trainingD$classe, preClass)
names(DTrainClass)
```

```
##  [1] "V1"                   "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"
```

```r
preObjV <- preProcess(validatingD[, -ncol(validatingD)], method=c("center","scale"))

preClassV<-predict(preObj,validatingD[, -ncol(validatingD)])
DValClass <- data.table(validatingD$classe, preClassV)
```

Random Forest Model

Using random forest model with the training data set. Estimated error rate is .65% and accuracy is 99% over validation dataset

```r
trainingmodel <- randomForest(classe ~ .,data=trainingD)
trainingmodel
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = trainingD)
##                Type of random forest: classification
```

```
##                      Number of trees: 500
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 0.65%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 3346    1    0    1    0 0.0005973716
## B   19 2255    5    0    0 0.0105309346
## C    0   18 2035    1    0 0.0092502434
## D    0    0   22 1907    1 0.0119170984
## E    0    0    2    6 2157 0.0036951501
```

```r
varImp(trainingmodel)
```

```
##                        Overall
## roll_belt            725.75490
## pitch_belt           407.12670
## yaw_belt             530.82015
## total_accel_belt     132.09995
## gyros_belt_x          59.43788
## gyros_belt_y          65.51850
## gyros_belt_z         199.53860
## accel_belt_x          73.16504
## accel_belt_y          82.36609
## accel_belt_z         228.88439
## magnet_belt_x        147.59263
## magnet_belt_y        245.64954
## magnet_belt_z        242.52763
## roll_arm             187.44497
## pitch_arm            103.50418
## yaw_arm              132.45875
## total_accel_arm       59.48481
## gyros_arm_x           83.96510
## gyros_arm_y           84.70040
## gyros_arm_z           38.92890
## accel_arm_x          143.78821
## accel_arm_y           94.59638
## accel_arm_z           72.93435
## magnet_arm_x         154.76762
## magnet_arm_y         138.48885
## magnet_arm_z         115.34414
## roll_dumbbell        258.48765
## pitch_dumbbell       109.00972
## yaw_dumbbell         152.08215
## total_accel_dumbbell 169.31862
## gyros_dumbbell_x      80.70639
## gyros_dumbbell_y     149.14279
## gyros_dumbbell_z      50.33055
## accel_dumbbell_x     149.40581
## accel_dumbbell_y     241.44858
## accel_dumbbell_z     207.84480
## magnet_dumbbell_x    300.11778
## magnet_dumbbell_y    408.41893
## magnet_dumbbell_z    452.29901
```

```
## roll_forearm         341.61910
## pitch_forearm        472.50995
## yaw_forearm           97.30256
## total_accel_forearm   70.91934
## gyros_forearm_x       47.77335
## gyros_forearm_y       81.32966
## gyros_forearm_z       53.43006
## accel_forearm_x      188.93404
## accel_forearm_y       86.34454
## accel_forearm_z      147.74235
## magnet_forearm_x     135.01405
## magnet_forearm_y     133.41883
## magnet_forearm_z     173.68048
```

```
m <- predict(trainingmodel,newdata=validatingD[,-ncol(validatingD)])
confusionMatrix(m,validatingD$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2229    8    0    0    0
##          B    2 1501   14    0    0
##          C    0    9 1354   24    1
##          D    0    0    0 1260   11
##          E    1    0    0    2 1430
##
## Overall Statistics
##
##                Accuracy : 0.9908
##                  95% CI : (0.9885, 0.9928)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9884
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9987   0.9888   0.9898   0.9798   0.9917
## Specificity            0.9986   0.9975   0.9948   0.9983   0.9995
## Pos Pred Value         0.9964   0.9895   0.9755   0.9913   0.9979
## Neg Pred Value         0.9995   0.9973   0.9978   0.9960   0.9981
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2841   0.1913   0.1726   0.1606   0.1823
## Detection Prevalence   0.2851   0.1933   0.1769   0.1620   0.1826
## Balanced Accuracy      0.9986   0.9931   0.9923   0.9891   0.9956
```

Prediction

```r
predictions <- predict(trainingmodel,newdata=testD)
predictions
```

```
##  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
```

For Submission

```r
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)
```

References

The training data for this project comes from: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv

The test data for this project comes from: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

The data for this project come from: http://groupware.les.inf.puc-rio.br/har