HAR Project

Shawnzoom

January 26, 2016

```
##
## Attaching package: 'dplyr'
##
## The following object is masked from 'package:stats':
##
##     filter
##
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

Using Sensor Data to Classify how well an Exercise Activity is Performed

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. 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, the goal was to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants, to classify how well an activity was performed. They were asked to perform barbell lifts correctly and incorrectly in 5 different ways.The "raw" data had 19622 observations and 160 features. After data exploration and cleansing, the data was split into training and test set, and the feature count was reduce to 40. The training data was then fitted to a Random Forest model using 5-fold cross validation. The model was then used to the classe variable in the test set and to answer the 20 quiz questions. The model correctlt classified the 20 quiz questions.

Question

The data analyis in this report sets out to answer the following question.

> Is is possible to classify the manner in in which a dumbbel execise was performed using sensor data from a glove, belt, arm-band and dumbbell?

Data

The training data for this project are available here:

The test data are available here:

The data for this project comes from this source:

Citation: This report is based on data from the following paper:

Velloso, E.; Bulling, A.; Gellersen, H.; Ugulino, W.; Fuks, H. Qualitative Activity Recognition of Weight Lifting Exercises. Proceedings of 4th International Conference in Cooperation with SIGCHI (Augmented Human '13) . Stuttgart, Germany: ACM SIGCHI, 2013.

Read more here

Assignment setup

The dataset has been placed in the directory `e:/rcode/chap08`

```
# Set the assignment working directory
project_dir <- setwd("e:/rcode/chap08")
```

Libraries used in analysis

```
# load libraries
library(dplyr, quietly = TRUE)
library(caret, quietly = TRUE)
library(YaleToolkit, quietly = TRUE)
library(parallel, quietly = TRUE)
library(doParallel)
library(iterators)
library(foreach)
```

Data Processing

This section describes (in words and code) how the data were loaded into R and processed for analysis. In particular, this shows how the analysis starts with a raw CSV file containing the data.

Preliminary data exploration

Read the first 5 rows and examine the data as it relates to the question

```
# Perform initial exploration

# Peek at frist 5 rows and find class of data
peek5.train <- read.table("pml-training.csv",  sep = ",", nrows = 5, header = TRUE)

# examine variables
str(peek5.train)

# Peek at frist 5 rows and find class of data
peek5.test <- read.table("pml-testing.csv",  sep = ",", nrows = 5, header = TRUE)

# examine variables
str(peek5.test)
```

Load all data

Load all data

```
# read data file
har.all <- read.table("pml-training.csv", sep = ",", header = TRUE, na.strings = "NA", stringsAsFactors

dim(har.all)
```

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

```
har.quiz <- read.table("pml-testing.csv", sep = ",", header = TRUE, na.strings = "NA", stringsAsFactors

dim(har.quiz)
```

```
## [1]  20 160
```

Use the whatis() function from the YaleToolKit to get additional perspective and insight on the data. Writing the data to a csv file and then browsing the data with Excel proved to be very useful.

```
var_explore_all <- whatis(har.all)

# create file var_exploration.csv
write.table(var_explore_all, file = "var_explore_all.csv", sep = ",", row.names = FALSE, col.names = TRU

var_explore_quiz <- whatis(har.quiz)

# create file var_exploration.csv
write.table(var_explore_quiz, file = "var_explore_quiz.csv", sep = ",", row.names = FALSE, col.names =
```

Feature Extraction and Selection

1. Based on data exploration, there are 19622 observations across 160 variables in the har.all. However, reviewing the csv created from var-exploration_all, 67 of the variables all have 19216 missing values. For this reason, all of these variables will not be considered for the model
2. Based on **Section 5.1 Feature Extraction and Selection** from the research paper , the following variables do not appear to have any roll in building the classification model: new_window, raw_timestamp_part1, raw_timestamp_part2, cvtd_timestamp, user_name, and V1. These variables will also not be considered for the model.
3. Additionally, Section 5.1 Feature Extraction and Selection makes no mention of using an of the kurtois, or skewness variables. These variables will also not be considered for the model.

To further help with feature selection, for a given sensor, the following additional assumptions were made from reading Section 5.1 Feature Extraction and Selection:

Additional assumptions

- Gyro, implies pitch, roll and yaw in the variable names
- magnetometer, implies magnet in the variable names
- range, implies amplitude
- If gyro was mentioned as being used, all the raw variables for that sensor were included. For example, for the belt, it says "variance of the gyro" and therefore pitch, roll and yaw variables are included for that sensor
- If magnetometer was mentioned as being used, the x, y, and z magnet variables for that sensor were included in the model

Data Prep and Cleaning

From the data exploration, it can be seen that some variables need to have there data types coerced. For example, character to numeric and logical to numeric.

```
# convert classes

har.all$classe <- as.factor(har.all$classe)
har.all$max_yaw_belt <- as.numeric(har.all$max_yaw_belt)
har.quiz$avg_roll_belt <- as.numeric(har.quiz$avg_roll_belt)
har.quiz$var_roll_belt <- as.numeric(har.quiz$var_roll_belt)
har.quiz$var_total_accel_belt <- as.numeric(har.quiz$var_total_accel_belt)
har.quiz$max_roll_belt <- as.numeric(har.quiz$max_roll_belt)
har.quiz$min_roll_belt <- as.numeric(har.quiz$min_roll_belt)
har.quiz$max_picth_belt <- as.numeric(har.quiz$max_picth_belt)
har.quiz$amplitude_pitch_belt <- as.numeric(har.quiz$amplitude_pitch_belt)
har.quiz$var_accel_arm <- as.numeric(har.quiz$var_accel_arm)
har.quiz$var_pitch_dumbbell <- as.numeric(har.quiz$var_pitch_dumbbell)
har.quiz$var_roll_dumbbell <- as.numeric(har.quiz$var_roll_dumbbell)
har.quiz$var_yaw_dumbbell <- as.numeric(har.quiz$var_yaw_dumbbell)
har.quiz$amplitude_roll_dumbbell <- as.numeric(har.quiz$amplitude_roll_dumbbell)

#dumbell
har.quiz$min_pitch_dumbbell <- as.numeric(har.quiz$min_pitch_dumbbell)
har.quiz$min_roll_dumbbell <- as.numeric(har.quiz$min_roll_dumbbell)
har.quiz$min_yaw_dumbbell <- as.numeric(har.quiz$min_yaw_dumbbell)
har.quiz$max_picth_dumbbell <- as.numeric(har.quiz$max_picth_dumbbell)
har.quiz$max_roll_dumbbell <- as.numeric(har.quiz$max_roll_dumbbell)
har.quiz$max_yaw_dumbbell <- as.numeric(har.quiz$max_yaw_dumbbell)

#arm
har.quiz$min_pitch_arm <- as.numeric(har.quiz$min_pitch_arm)
har.quiz$min_roll_arm <- as.numeric(har.quiz$min_roll_arm)
har.quiz$min_yaw_arm <- as.numeric(har.quiz$min_yaw_arm)
har.quiz$max_picth_arm <- as.numeric(har.quiz$max_picth_arm)
har.quiz$max_roll_arm <- as.numeric(har.quiz$max_roll_arm)
har.quiz$max_yaw_arm <- as.numeric(har.quiz$max_yaw_arm)

#belt
har.quiz$min_pitch_belt <- as.numeric(har.quiz$min_pitch_belt)
har.quiz$min_roll_belt <- as.numeric(har.quiz$min_roll_belt)
har.quiz$min_yaw_belt <- as.numeric(har.quiz$min_yaw_belt)
har.quiz$max_picth_belt <- as.numeric(har.quiz$max_picth_belt)
har.quiz$max_roll_belt <- as.numeric(har.quiz$max_roll_belt)
har.quiz$max_yaw_belt <- as.numeric(har.quiz$max_yaw_belt)

#forearm
har.quiz$min_pitch_forearm <- as.numeric(har.quiz$min_pitch_forearm)
har.quiz$min_roll_forearm <- as.numeric(har.quiz$min_roll_forearm)
har.quiz$min_yaw_forearm <- as.numeric(har.quiz$min_yaw_forearm)
har.quiz$max_picth_forearm <- as.numeric(har.quiz$max_picth_forearm)
har.quiz$max_roll_forearm <- as.numeric(har.quiz$max_roll_forearm)
har.quiz$max_yaw_forearm <- as.numeric(har.quiz$max_yaw_forearm)
```

```
# amplitude train
har.all$amplitude_yaw_arm <- as.numeric(har.all$amplitude_yaw_arm)
har.all$amplitude_yaw_belt <- as.numeric(har.all$amplitude_yaw_belt)
har.all$amplitude_yaw_forearm <- as.numeric(har.all$amplitude_yaw_forearm)

# amplitude test
har.quiz$amplitude_yaw_arm <- as.numeric(har.quiz$amplitude_yaw_arm)
har.quiz$amplitude_yaw_belt <- as.numeric(har.quiz$amplitude_yaw_belt)
har.quiz$amplitude_yaw_forearm <- as.numeric(har.quiz$amplitude_yaw_forearm)

# train yaw
har.all$max_yaw_belt <- as.numeric(har.all$max_yaw_belt)
har.all$max_yaw_dumbbell <- as.numeric(har.all$max_yaw_dumbbell)
har.all$max_yaw_forearm <- as.numeric(har.all$max_yaw_forearm)
har.all$min_yaw_belt <- as.numeric(har.all$min_yaw_belt)
har.all$min_yaw_dumbbell <- as.numeric(har.all$min_yaw_dumbbell)
har.all$min_yaw_forearm <- as.numeric(har.all$min_yaw_forearm)
```

Creating the training and test datasets

Even though the this assignment implies a specific training and test dataset, this analysis report partitions the data differently. For this report, the original test dataset is being used solely for the answering the quiz questions and is called har.quiz.

The original training dataset has been renamed to har.all. har.har.all is further devided into a new train and test dataset, har.train and har.test, respectively.

```
# create a training and test data set from the har.all data
set.seed(3465)
inTrain <- createDataPartition(y = har.all$classe, p=0.75, list=FALSE)

har.train <- har.all[inTrain,]
har.test <- har.all[-inTrain,]
```

Model Computation

```
# create a vector of variables bases on defined feature selection criteria
har.train.tmp <- select(har.train,
                        classe,
                        roll_belt,
                        pitch_belt,
                        yaw_belt,
                        total_accel_belt,
                        gyros_belt_x,
                        gyros_belt_y,
                        gyros_belt_z,
                        accel_belt_x,
                        accel_belt_y,
                        accel_belt_z,
                        magnet_belt_x,
                        magnet_belt_y,
```

```
                         magnet_belt_z,
                         gyros_forearm_x,
                         gyros_forearm_y,
                         gyros_forearm_z,
                         accel_forearm_x,
                         accel_forearm_y,
                         accel_forearm_z,
                         magnet_forearm_x,
                         magnet_forearm_y,
                         magnet_forearm_z,
                         gyros_arm_x,
                         gyros_arm_y,
                         gyros_arm_z,
                         accel_arm_x,
                         accel_arm_y,
                         accel_arm_z,
                         magnet_arm_x,
                         magnet_arm_y,
                         magnet_arm_z,
                         gyros_dumbbell_x,
                         gyros_dumbbell_y,
                         gyros_dumbbell_z,
                         accel_dumbbell_x,
                         accel_dumbbell_y,
                         accel_dumbbell_z,
                         magnet_dumbbell_x,
                         magnet_dumbbell_y,
                         magnet_dumbbell_z
                         )

# Citation:
# Information on how to improve performance of Random Forest in caret came from teh folling:  https://g

# setup and register cluster
cluster <- makeCluster(detectCores() - 1) # convention to leave 1 core for OS
registerDoParallel(cluster)

# configure train control parameters to include cross validation parallel computation
fitControl <- trainControl(method = "cv",
                           number = 5,
                           allowParallel = TRUE)

# train the model
modFit2 <- train(classe ~ .,method="rf", trControl = fitControl ,data=har.train.tmp)

# shutdown the cluster
stopCluster(cluster)
```

Results and Analysis

Show model results and use the model to predict the classe variable on the test data.

```r
# Show model results
modFit2
```

```
## Random Forest
##
## 14718 samples
##    40 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 11775, 11775, 11774, 11774, 11774
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa      Accuracy SD   Kappa SD
##    2    0.9902161  0.9876227  0.0016005094  0.0020258318
##   21    0.9890610  0.9861608  0.0019868724  0.0025144113
##   40    0.9858676  0.9821223  0.0007829572  0.0009904767
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 2.
```

```r
# use model to predict the classe variable on the test data
pred.modFit2.test <- predict(modFit2, newdata = har.test)

# generate confusion matraix and estimate of out-of-sample error
conf.matrix.test <- confusionMatrix(data = pred.modFit2.test, har.test$classe)
conf.matrix.test
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1395    1    0    1    0
##          B    0  946    6    0    0
##          C    0    2  848   11    0
##          D    0    0    1  791    0
##          E    0    0    0    1  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   0.9918   0.9838   1.0000
```

```
## Specificity            0.9994   0.9985   0.9968   0.9998   0.9998
## Pos Pred Value         0.9986   0.9937   0.9849   0.9987   0.9989
## Neg Pred Value         1.0000   0.9992   0.9983   0.9968   1.0000
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2845   0.1929   0.1729   0.1613   0.1837
## Detection Prevalence   0.2849   0.1941   0.1756   0.1615   0.1839
## Balanced Accuracy      0.9997   0.9977   0.9943   0.9918   0.9999
```

Model has an overall accuracy of 0.99531

What are the most important variables in the model?

```r
modFit2.importance <- as.data.frame(modFit2$finalModel$importance)

# move row names to a new column
modFit2.importance$new <- rownames(modFit2.importance)
rownames(modFit2.importance) <- NULL

modFit2.importance <- arrange(modFit2.importance, desc(MeanDecreaseGini))
modFit2.importance
```

```
##    MeanDecreaseGini               new
## 1          633.5613         roll_belt
## 2          562.0207          yaw_belt
## 3          543.2001 magnet_dumbbell_z
## 4          493.4976 magnet_dumbbell_y
## 5          474.0261        pitch_belt
## 6          440.2459 magnet_dumbbell_x
## 7          390.3627  accel_dumbbell_y
## 8          369.0083     magnet_belt_z
## 9          357.0062      accel_belt_z
## 10         353.0827  accel_dumbbell_z
## 11         339.1845     magnet_belt_y
## 12         320.1377  accel_dumbbell_x
## 13         313.3496   accel_forearm_x
## 14         312.2319      magnet_arm_x
## 15         305.9672       accel_arm_x
## 16         292.3523      magnet_arm_y
## 17         290.1414  magnet_forearm_x
## 18         283.4474      gyros_belt_z
## 19         272.6965  magnet_forearm_z
## 20         271.2225  gyros_dumbbell_y
## 21         268.3218  magnet_forearm_y
## 22         260.1735   accel_forearm_z
## 23         252.9213      magnet_arm_z
## 24         252.2037     magnet_belt_x
## 25         230.0715       accel_arm_y
## 26         226.1951       gyros_arm_x
## 27         224.4454   accel_forearm_y
## 28         222.9943  total_accel_belt
## 29         215.9327       accel_arm_z
## 30         210.8938       gyros_arm_y
```

```
## 31          208.5369   gyros_forearm_y
## 32          191.7140  gyros_dumbbell_x
## 33          184.2749      accel_belt_x
## 34          169.5078      accel_belt_y
## 35          165.8144      gyros_belt_x
## 36          156.0640   gyros_forearm_z
## 37          155.7811   gyros_forearm_x
## 38          152.5882      gyros_belt_y
## 39          142.7339  gyros_dumbbell_z
## 40          126.7566       gyros_arm_z
```

Use model to predict quiz answers

```
pred.modFit2.quiz <- predict(modFit2, newdata = har.quiz)
pred.modFit2.quiz
```

```
##  [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
```

Conclusion

This report showed how a dataset without a code book could be explored and analyzed to build a model to classify how an activity was performed. The model used 5-flod cross validation and showed out-of-sample accuracy/error on the test data. Also, the model correctly classified the 20 quiz questions.

However, this model would not generalize well as (1) the the HAR experiment was carried out in a very controlled environment. (2) There were not enough participants in the study.