Practical Machine Learning - Course Final Peer Assignment

GustavoCruz

July 16, 2016

Table of Contents

- Problem Statement
- Loading and preparing the data
- Generating the model
- Saving the model and test data

Problem Statement

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, we use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants [@velloso2013]. They were asked to perform barbell lifts correctly and incorrectly in 5 different ways. Given data from accelerometers, the goal is to predict the class of action which is one of the following.

- exactly according to the specification (A)
- throwing elbows to the front (B)
- lifting the dumbbell only halfway (C)
- lowering the dumbbell only halfway (D)
- throwing the hips to the front (E).

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).

Loading and preparing the 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

```
# Loading Curl Libraries
library(RCurl)
```

```
## Loading required package: bitops
```

```
# Loading the data
train_url <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
test_url <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
train_data <- read.csv(text=getURL(train_url), na.strings=c("", "NA"))
test_data <- read.csv(text=getURL(test_url), na.strings=c("", "NA"))
```

The first column of the data is just index. We remove it from training data frame.

```
train_data$X <- NULL
```

Similarly the user and time information should not have any effect on whether barbell lifts are performed correctly or not. I's need to clean the data for analysis purposes.

```
cols_to_remove <- c("user_name", "raw_timestamp_part_1",
                    "raw_timestamp_part_2", "cvtd_timestamp")
for (col in cols_to_remove) {
    train_data[, col] <- NULL
}
```

Many columns in the dataset have missing values. We remove features from the training and testing data that have not too many missing values, if imputing data is not an option.

```
NAs <- apply(train_data,2,function(x) {sum(is.na(x))})
train_data <- train_data[,which(NAs == 0)]
```

We also remove features that don't have many missing values but have one unique value (i.e. zero variance predictors) or have few unique values relative to the number of samples and the ratio of frequency of the most common value to the frequency of second most common value is large.

```
library(caret)
nsv <- nearZeroVar(train_data)
train_data <- train_data[-nsv]
test_data <- test_data[-nsv]
```

The final set of predictors used for classification are as follows.

```
names(train_data)
```

```
##  [1] "num_window"           "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"     "classe"
```

Generating the model

We build a random forest classifier to predict the action class. To measure the accuracy of the model, we do 10-fold cross validation with 80:20 split, on each fold, 80% of the data is used for training the random forest and remaining 20% is used for testing.

```
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
```

```
set.seed(1)
obs <- c()
preds <- c()
for(i in 1:10) {
    intrain = sample(1:dim(train_data)[1], size=dim(train_data)[1] * 0.8, replace=F)
    train_cross = train_data[intrain,]
    test_cross = train_data[-intrain,]
    rf <- randomForest(classe ~ ., data=train_cross)
    obs <- c(obs, test_cross$classe)
    preds <- c(preds, predict(rf, test_cross))
}
```

The confusion matrix for predictions on cross validation folds is given below.

```
conf_mat <- confusionMatrix(table(preds, obs))
conf_mat$table
```

```
##       obs
## preds     1     2     3     4     5
##     1 11099     7     0     0     0
##     2     1  7456    10     0     0
##     3     0     3  6836    32     0
##     4     0     0     3  6470     7
##     5     2     0     0     2  7322
```

The proposed model seems classifying well enough. The accuracy is 99.8292994% and it misclassifies only few instances. Finally, we train the random forest with whole dataset so that the classifier can be used to predict the class of an action, given the set of activity measurements.

```
model <- randomForest(classe ~ ., data=train_data)
```

Saving the model and test data

Finally the model and test data is saved so they can be reused to perform further analysis.

```
# Saving the model and the test data to perform the quiz evaluation.
save(model, file = "Model.rda")
save(test_data, file = "TestData.rda")
```