Wearables Workout Prediction - Machine Learning

Adam Moreno

February 14, 2019

R Markdown Document

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, 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.

Data

Training Data: "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"

Test Data: "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"

More information is available from the website here: http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

Goal

The goal of this project is to predict the manner in which users did an exercise. This is the "Classe" variable in the training set. Use any of the other variables to predict with. Create a report describing the model build, the use of cross validation, the expected out of sample error, and why these choices were made. Use this prediction model to predict 20 different test cases.

Project Outline

Data Processing

Working within this data, you'll quickly find blanks, NAs, and other values that represent missing/insufficient data. Processing this data into a matching variable "NA", allows us to remove columns with any type of insufficient data effectively. I chose to remove columns containing more than 15% NA values.

Reproducibility

Setting the seed at the beginning of analysis will create reproducible results, as well as downloading all libraries listed below and installing packages where needed on your own machine. I have suppressed warnings and messages when referencing libraries to reduce clutter within the report.

Building the Model

The model was built based on the Classe variable within our data. The Classe variable describes five different fashions of how a health participant performed a Unilateral Dumbbell Bicep Curl:

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

All other variables, after excluding columns with excessive NA values, are used for our prediction model. Two models were built using decision tree and random forest algorithms. Whichever algorithm produces the more accurate results will be used as our final model against the out of sample dataset.

Cross-validation

Creating data partitions within our training data, 70% training [myTraining] and 30% testing [myTesting], cross validates our original training dataset. Both models will be tested on the training dataset then the more accurate model will be run on the testing. Finally, we will test predictions on the out of sample testing dataset containing 20 instances.

Out of Sample Error

To test our predictions, I am running a Confusion Matrix to determine accuracy within a 95% confidence interval across each Classe variable. I will be using accuracy to determine the validity of the model. The expected out of sample error correlates with any misclassification of the workout to our out of sample data set [og_testing].

Report

Load in Data and Libraries

```r
## Read in Data Files
#Tons of NA, Blank, DIV/0 data, convert to NA at read in

#Web URL download
training_Url <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
og_training <- read.csv(url(training_Url), na.strings=c("NA","#DIV/0!",""))
test_Url <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
og_testing <- read.csv(url(test_Url), na.strings=c("NA","#DIV/0!",""))

dim(og_training); dim(og_testing)
```

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

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

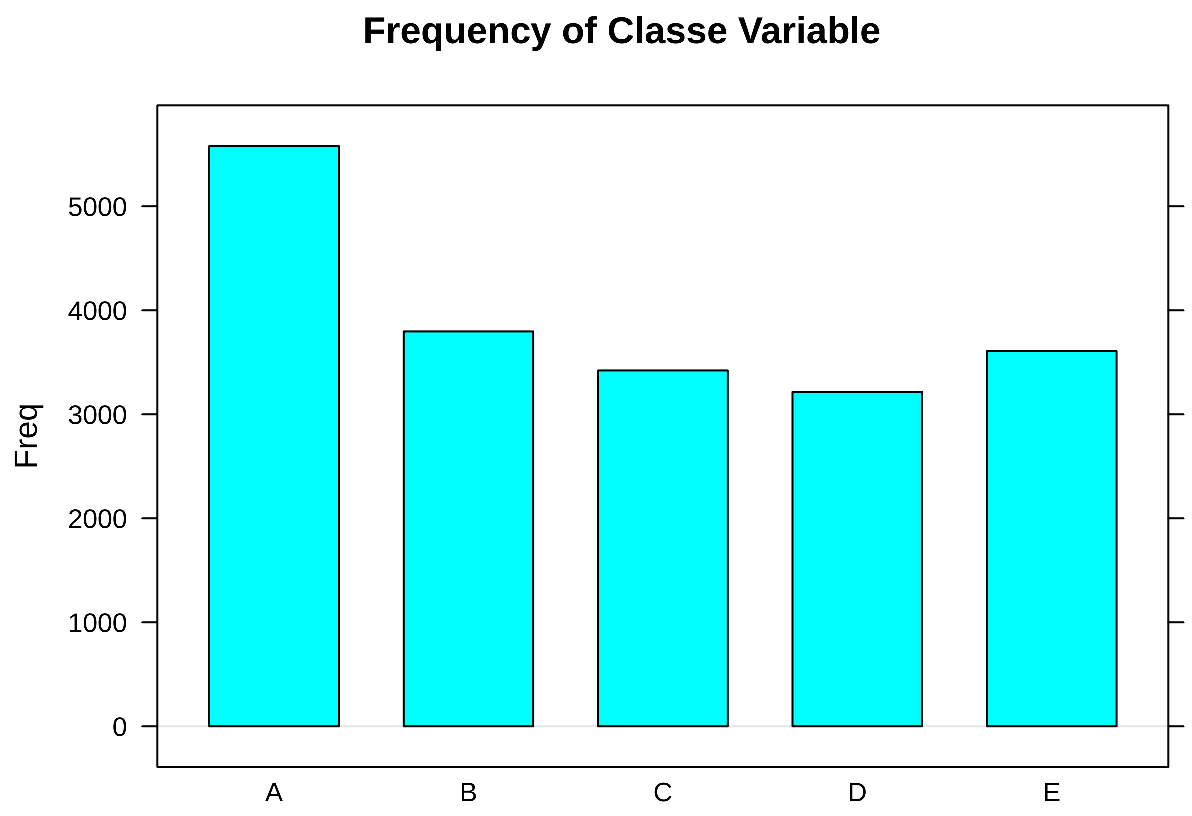
```
#Load Libraries

suppressWarnings(suppressMessages(library(caret)))
suppressWarnings(suppressMessages(library(dplyr)))
suppressWarnings(suppressMessages(library(rpart)))
#library(rpart.plot)
suppressWarnings(suppressMessages(library(randomForest)))
```

Removing NA data and Further Dataset Manipulation

Frequency Plot of Classe Variable

```
#Taking a quick look at frequency for our classe variable
barchart(f_trainingDF$classe,
         horizontal = FALSE,
         main = "Frequency of Classe Variable")
```



Creating Machine Learning Datasets and Running Models

```
#Create Data Partitions of the training group
inTrain <- createDataPartition(y = f_trainingDF$classe, p = 0.7, list = FALSE)
myTraining <- f_trainingDF[inTrain, ]; myTesting <- f_trainingDF[-inTrain, ]

###
#Spot holder for potential NZV variable removals
###
```

```r
#Training a model with Decision Tree
set.seed(123)
decisionTreeModel <- rpart(classe ~ ., data = myTraining, method = "class")

#Predict using Training Data
DTprediction1 <- predict(decisionTreeModel, myTraining, type = "class")

#Plot Tree to visualize isn't helpful, overplotted and too many variables
##rpart.plot(decisionTreeModel, main = "Variable Classification")

#Test accuracy
confusionMatrix(DTprediction1, myTraining$classe)
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction    A    B    C    D    E
##          A 3576  564   94  246  145
##          B   73 1421   76   42  100
##          C   25  351 2067  238   52
##          D  222  296  157 1618  357
##          E   10   26    2  108 1871
##
## Overall Statistics
##
##                Accuracy : 0.7682
##                  95% CI : (0.7611, 0.7753)
##     No Information Rate : 0.2843
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7054
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9155   0.5346   0.8627   0.7185   0.7410
## Specificity            0.8933   0.9737   0.9413   0.9101   0.9870
## Pos Pred Value         0.7732   0.8300   0.7563   0.6106   0.9276
## Neg Pred Value         0.9638   0.8971   0.9701   0.9428   0.9442
## Prevalence             0.2843   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2603   0.1034   0.1505   0.1178   0.1362
## Detection Prevalence   0.3367   0.1246   0.1990   0.1929   0.1468
## Balanced Accuracy      0.9044   0.7542   0.9020   0.8143   0.8640
```

```r
#.77% on training data

#Train a model with Random Forest
randomForestModel <- randomForest(classe ~., data = myTraining)

#Predict using Training Data
RFprediction2 <- predict(randomForestModel, myTraining, type = "class")
```

```r
#Test accuracy
confusionMatrix(RFprediction2, myTraining$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 3906    0    0    0    0
##          B    0 2658    0    0    0
##          C    0    0 2396    0    0
##          D    0    0    0 2252    0
##          E    0    0    0    0 2525
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (0.9997, 1)
##     No Information Rate : 0.2843
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   1.0000   1.0000   1.0000   1.0000
## Specificity            1.0000   1.0000   1.0000   1.0000   1.0000
## Pos Pred Value         1.0000   1.0000   1.0000   1.0000   1.0000
## Neg Pred Value         1.0000   1.0000   1.0000   1.0000   1.0000
## Prevalence             0.2843   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2843   0.1935   0.1744   0.1639   0.1838
## Detection Prevalence   0.2843   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      1.0000   1.0000   1.0000   1.0000   1.0000
```

```r
#Accuracy 99%
#Random Forest is much more accurate, now use on testing data

#Train a model with Random Forest
randomForestModel <- randomForest(classe ~., data = myTraining)

#Predict using Testing Data
RFprediction2 <- predict(randomForestModel, myTesting, type = "class")

#Test accuracy
confusionMatrix(RFprediction2, myTesting$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1673    0    0    0    0
##           B    0 1139    1    0    0
##           C    0    0 1025    1    0
##           D    0    0    0  963    2
##           E    1    0    0    0 1080
##
## Overall Statistics
##
##                Accuracy : 0.9992
##                  95% CI : (0.998, 0.9997)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9989
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9994   1.0000   0.9990   0.9990   0.9982
## Specificity            1.0000   0.9998   0.9998   0.9996   0.9998
## Pos Pred Value         1.0000   0.9991   0.9990   0.9979   0.9991
## Neg Pred Value         0.9998   1.0000   0.9998   0.9998   0.9996
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2843   0.1935   0.1742   0.1636   0.1835
## Detection Prevalence   0.2843   0.1937   0.1743   0.1640   0.1837
## Balanced Accuracy      0.9997   0.9999   0.9994   0.9993   0.9990
```

```
#Accuracy 99%
```

Since the accuracy of the Random Forest prediction was the best, we will test this predictor on our out of sample data.

```
#Find predictions on out of sample data
assignmentPredictions <- predict(randomForestModel, og_testing, type = "class")
assignmentPredictions
```

```
##  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
```