Practical Machine Learning Project

Walter Breno Theves

06/09/2020

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, 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.

Six young health participants were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in five different fashions: exactly according to the specification (Class A), throwing the elbows to the front (Class B), lifting the dumbbell only halfway (Class C), lowering the dumbbell only halfway (Class D) and throwing the hips to the front (Class E).

More information is available at http://groupware.les.inf.puc-rio.br/har#ixzz6XJnCqixX

Getting, Cleaning and Preparing the Data

```r
# libraries needed to run the code
library(caret)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```r
library(randomForest)
```

```
## randomForest 4.6-14
```

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

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

The first step of the process is to download and read the data.

```
URLtraining <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
URLtesting <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
setwd("C:/Users/Teste/Desktop/Projects/Project_MachineLearning")
# download.file(URLtraining, "training_data")
# download.file(URLtesting, "testing_data")
training <- read.csv("training_data")
testing <- read.csv("testing_data")
```

The first columns are information used to identify characteristics of the subject that hold no information regarding the actual exercise, so they will be excluded. Columns that hold too many NA values will also be excluded.

```
excluded_col <- grep("name|timestamp|window|X", colnames(training), value=FALSE)
training <- training[,-excluded_col]
testing <- testing[,-excluded_col]
# Columns with no or NA values will also be excluded
training[training==""] <- NA
testing[testing==""] <- NA
excluded_col <- colSums(is.na(training))/nrow(training)
NAs <- names(excluded_col[excluded_col > 0.75]) # names of the columns with NAs
NACols <- which(excluded_col > 0.75) # index of the NA columns
training <- training[,-NACols]
testing <- testing[,-NACols]
# Now the datasets are clean
```

In order to apply *Cross Validation* to the dataset, the training dataset will be splitted in two parts.

```
set.seed(1000)
InTrain <- createDataPartition(y=training$classe, p=0.75, list=FALSE)
training_train <- training[InTrain,]
training_test <- training[-InTrain,]
```

```
# Transforming classe column into factor
training_train[,"classe"] <- as.factor(training_train[,"classe"])
training_test[,"classe"] <- as.factor(training_test[,"classe"])
```

Now the dataset is ready to the actual machine learning algorithm.

Machine Learning Algorithm

Training the model The ML Algorithm used to train the model is the `randomforest` one.

```
model <- randomForest(classe ~. , data=training_train, method="class")
prediction <- predict(model, training_test, type = "class")
confusionMatrix(prediction, training_test$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1394    6    0    0    0
##          B    1  938    7    0    0
##          C    0    5  847   10    0
##          D    0    0    1  794    6
##          E    0    0    0    0  895
##
## Overall Statistics
##
##                Accuracy : 0.9927
##                  95% CI : (0.9899, 0.9949)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9907
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9993   0.9884   0.9906   0.9876   0.9933
## Specificity            0.9983   0.9980   0.9963   0.9983   1.0000
## Pos Pred Value         0.9957   0.9915   0.9826   0.9913   1.0000
## Neg Pred Value         0.9997   0.9972   0.9980   0.9976   0.9985
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2843   0.1913   0.1727   0.1619   0.1825
## Detection Prevalence   0.2855   0.1929   0.1758   0.1633   0.1825
## Balanced Accuracy      0.9988   0.9932   0.9935   0.9929   0.9967
```

The accuracy achieved with the model is of *99.27%* and the out of sample error is of *0.0073*.

```r
result <- predict(model, testing)

# Classifications of the test data set:
result
```

Applying the model to the train dataset

```
##  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
```