Quantifying physical exercise performance

Joshua Poirier

Sunday, March 22, 2015

Executive Summary

Devices such as *Jawbone Up*, *Nike FuelBand*, and *Fitbit* make it possible to inexpensively collect large amounts of data about personal activity. The devices are part of the quantified self movement - a group of enthusiasts who take measurements about themselves to improve health, and/or identify behavioural patterns. People often quantify how *much* of particular activities they do; however, they do rarely measure *how well they do it*. This paper aims to use data from accelerometers worn on the belt, forearm, arm, and dumbell of six persons. They were asked to lift the barbell correctly and incorrectly in five different ways. Further information about the data is available at http://groupware.les.inf.puc-rio.br/har.

Our workflow is summarized as follows: (1) Load the training data, (2) Split the training data into training and test sets (for cross-validation), (3) Build a model on the training set, (4) Evaluate the model on the test set, and (5) Apply the model to the testing set.

Data Loading, Feature Selection, and Subsetting

Here we load the data, select features, and subset the data into training and test sets. Most of the features we want to include are related to the sensors installed on the belt, arm, and forearm of the six persons - as well as the dumbbell. This is shown visually in Figure 1 below.

~

```
library(dplyr); library(caret); library(randomForest)

# load the training and testing data sets
data <- read.csv("C:/Papers/Coursera/Machine Learning/data/pml-training.csv")
data$classe <- as.factor(data$classe)

test <- read.csv("C:/Papers/Coursera/Machine Learning/data/pml-testing.csv")
```

Select only features beginning with *raw_timestamp*, *roll*, *pitch*, *yaw*, or *total* or end with *x*, *y*, or *z*. We also select the outcome *classe*.

```
# select only fields with data throughout (other fields are processed by original authors)
data <- select(data, classe, starts_with("raw_timestamp"), starts_with("roll"),
               starts_with("pitch"), starts_with("yaw"), starts_with("total"),
               ends_with("_x"), ends_with("_y"), ends_with("_z"))

test <- select(test, starts_with("raw_timestamp"), starts_with("roll"),
               starts_with("pitch"), starts_with("yaw"), starts_with("total"),
               ends_with("_x"), ends_with("_y"), ends_with("_z"))
```

We subset the training data into subsets *training* and *testing*. This will be used to cross-validate our model.

```
inTrain <- createDataPartition(y=data$roll_belt, p=0.7, list=FALSE)
training <- data[inTrain,]
testing <- data[-inTrain,]
```

Model Building

We build a model using the Random Forest's algorithm using all predictors. The model is built on the training data subset.

```
modFit <- randomForest(classe ~ ., data=training)
```

Cross-Validation / Out-of-sample error estimation

In order to correctly predict the test data set (with-held), we need an accuracy of greater than 0.95. Let us examine the Confusion Matrix for the testing data subset. If the model is good, the accuracy will be very high (>0.95).

```
confusionMatrix(testing$classe, predict(modFit, testing[,-1]))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1682    0    0    0    0
##          B    4 1144    0    0    0
##          C    0    3  993    0    0
##          D    0    0    3  992    0
##          E    0    0    1    5 1058
##
## Overall Statistics
##
##                Accuracy : 0.997
##                  95% CI : (0.996, 0.998)
##     No Information Rate : 0.286
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.997
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity             0.998    0.997    0.996    0.995    1.000
## Specificity             1.000    0.999    0.999    0.999    0.999
## Pos Pred Value          1.000    0.997    0.997    0.997    0.994
## Neg Pred Value          0.999    0.999    0.999    0.999    1.000
## Prevalence              0.286    0.195    0.169    0.169    0.180
## Detection Rate          0.286    0.194    0.169    0.169    0.180
## Detection Prevalence    0.286    0.195    0.169    0.169    0.181
## Balanced Accuracy       0.999    0.998    0.998    0.997    0.999
```

The accuracy of the model on the testing data subset reaches 0.998 - approximating the out-of-sample error rate. This is a very good accuracy, and should predict the 20 test cases correctly!

Predicting Test Data

Given the high accuracy of the model developed (as shown above), we expect to accurately predict the 20 test cases for which classes were withheld. Let us now apply the model to the test data and print the classifications.

```r
print(predict(modFit, test))
```

```
##  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
```

References

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). Stuttgard, Germany: ACM SIGCHI, 2013.

Leek, Jeff; Peng, Roger; Caffo, Brian. Practical Machine Learning [Course Notes]. Coursera.org / Johns Hopkins School of Public Health. https://www.coursera.org/course/predmachlearn 2015.