Machine Learning Project

Ewen McKinnon

Sunday, May 15, 2016

Background

This project uses data from accelerometers on the belt, forearm, arm, and dumbbell of 6 participants in a physical exercise trial. Participants were asked to perform dumbbell lifts correctly and incorrectly in 5 different ways. The aim of this report is to see if we can accurately classify incorrect and correct performance from the accelerometer data - and then to run the final classifier on a set of 20 test questions.

Methodology

Given a very clear analytical question my methodological approach for this project was to:
* load and clean the data to reduce the data to useful features only
* split the data into training and testing datasets
* explore the potential features to determine if any pre-processing is required
* run some candidate classifiers - classification tree, random forest and gradient boosting
* perform cross-validation on test sets
* explore the possibility of blending classifiers to improve performance
* select a final classifier and run it on the project test questions

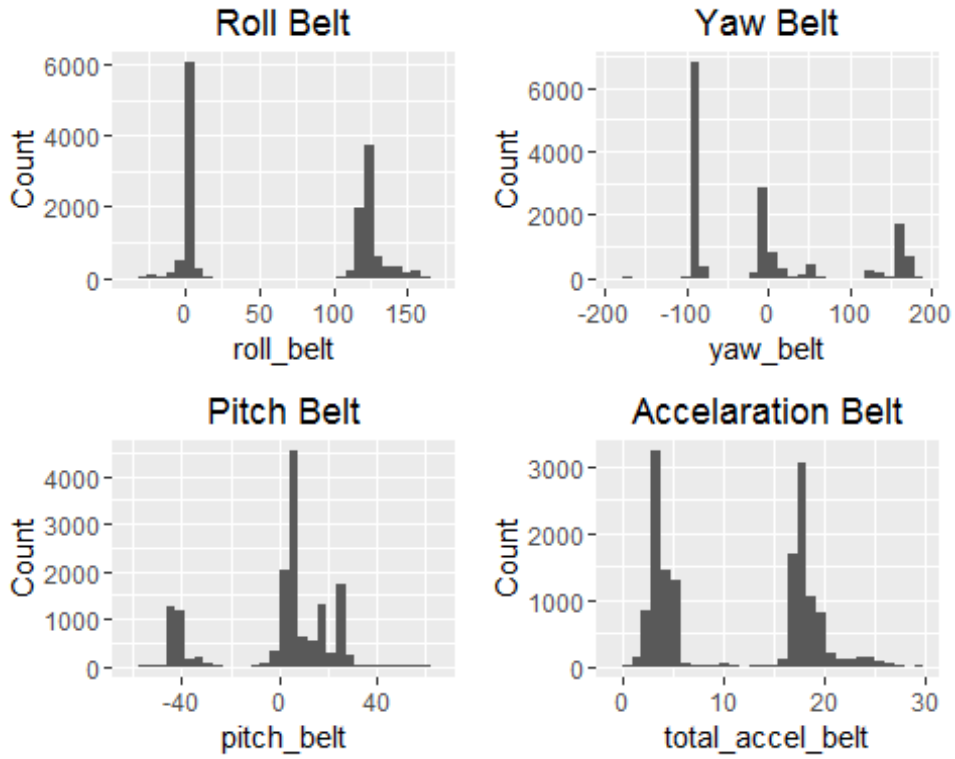
Data Cleansing and Preparation

The data can be downloaded from http://groupware.les.inf.puc-rio.br/har. In the training data there are 159 variables with an additional single classification variable 'classe'. However many of the variables in the training data are dominated by missing data. Furthermore, these variables are entirely missing in the final test question dataset and they needed to be removed. In addition, there are time stamp variables which intuitively should not contribute as features within the classifier and I have removed these too. Annex A provides a summary of the 54 useful variables I retained for classification analysis.

I have split the data into training and testing dataset on a 75:25 basis. So in total there are 14718 observations in the training dataset and 4904 observations in the testing dataset for cross-validation.

Exploratory Analysis

Exploratory analysis highlights non-normal distributions among the potential classifier features. The charts below illustrate a few histograms of features and show skewed and bi-modal distributions. These suggest pre-processing using centering and scaling might help with some classifications algorithms - particularly generalised linear regression modelling.



Development of Classifiers and Cross-validation Results

The classification variable consists of a factor variable with 5 levels - in other words one correct form of exercise and four incorrect. In practice this means binomial logistic regression is not possible because there are more than two classification possibilities. I therefore ran three types of models - classification tree, boosting and random forest, with two forms of pre-processing - centre & scaling and Principle Components Analysis.

The two best models which emerged from this analysis were Random Forest and Boosting, without any pre-processing. Centre and scaling, and PCA pre-processing in almost all cases produced poorer classification results. Classification tree algorithms performed particularly poorly and do not appear suited to this dataset.

Confusion Matrices for the two best models are presented in Annexes B and C. However the headline performance and error rates are as follows:

- Random Forest in sample accuracy 100% with error rate 0%

- Random Forest out of sample accuracy 99.69% with error rate 0.31%
- Boosting in sample accuracy 99.27% with error rate 0.73%
- Boosting out of sample sample accuracy 98.9% with error rate 1.1%

So both classifiers perform well but with the the Random Forest classifier performing marginally better.

Discussion on Combinational/ Ensemble Classification

There is clearly very little room for improvement given the high accuracy of the classifiers however it is possible to construct a majority voting combination of the two classifiers. On the training data set the two classifiers agree with each other on 99.27% of observations. On the testing dataset they agree on 98.98% of observations. So some dissagrement suggests there is a potential for a very small improvement using a combination classifier. Such small improvements in accuracy might help with thousands of classifications, however over a small set of test questions there will be little to gain from developing a combination classifier - in fact both classifiers give exactly the same results on the final test questions. Furthermore given the very high processing times for the two classifiers (both took take 3hrs to run on my computer) I would recommend using the random forest classifier which its slightly higher accuracy and almost halving the computation time that would be associated with tuning a combinational classifier.

Final Classifier Results

The table below provides the final results against the 20 quiz questions using the random forest classifier which has a measured out of sample accuracy of 99.69% and out of sample error rate of 0.31%.

```
##    problem_id rforrest_prediction
## 1           1                   B
## 2           2                   A
## 3           3                   B
## 4           4                   A
## 5           5                   A
## 6           6                   E
## 7           7                   D
## 8           8                   B
## 9           9                   A
## 10         10                   A
## 11         11                   B
## 12         12                   C
## 13         13                   B
## 14         14                   A
## 15         15                   E
## 16         16                   E
```

```
## 17         17                   A
## 18         18                   B
## 19         19                   B
## 20         20                   B
```

Annex A Summary of variables used in classification training set

```
## 'data.frame':    14718 obs. of  55 variables:
##  $ user_name           : Factor w/ 6 levels "adelmo","carlitos",..: 2 2 2
2 2 2 2 2 2 2 ...
##  $ num_window          : int  11 11 11 12 12 12 12 12 12 12 ...
##  $ roll_belt           : num  1.41 1.41 1.42 1.48 1.48 1.45 1.42 1.43 1.45
1.45 ...
##  $ pitch_belt          : num  8.07 8.07 8.07 8.05 8.07 8.06 8.09 8.16 8.17
8.18 ...
##  $ yaw_belt            : num  -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -9
4.4 -94.4 -94.4 ...
##  $ total_accel_belt    : int  3 3 3 3 3 3 3 3 3 3 ...
##  $ gyros_belt_x        : num  0 0.02 0 0.02 0.02 0.02 0.02 0.02 0.03 0.03
...
##  $ gyros_belt_y        : num  0 0 0 0 0.02 0 0 0 0 0 ...
##  $ gyros_belt_z        : num  -0.02 -0.02 -0.02 -0.03 -0.02 -0.02 -0.02 -0
.02 0 -0.02 ...
##  $ accel_belt_x        : int  -21 -22 -20 -22 -21 -21 -22 -20 -21 -21 ...
##  $ accel_belt_y        : int  4 4 5 3 2 4 3 2 4 2 ...
##  $ accel_belt_z        : int  22 22 23 21 24 21 21 24 22 23 ...
##  $ magnet_belt_x       : int  -3 -7 -2 -6 -6 0 -4 1 -3 -5 ...
##  $ magnet_belt_y       : int  599 608 600 604 600 603 599 602 609 596 ...
##  $ magnet_belt_z       : int  -313 -311 -305 -310 -302 -312 -311 -312 -308
-317 ...
##  $ roll_arm            : num  -128 -128 -128 -128 -128 -128 -128 -128 -128
-128 ...
##  $ pitch_arm           : num  22.5 22.5 22.5 22.1 22.1 22 21.9 21.7 21.6 2
1.5 ...
##  $ yaw_arm             : num  -161 -161 -161 -161 -161 -161 -161 -161 -161
-161 ...
##  $ total_accel_arm     : int  34 34 34 34 34 34 34 34 34 34 ...
##  $ gyros_arm_x         : num  0 0.02 0.02 0.02 0 0.02 0 0.02 0.02 0.02 ...
##  $ gyros_arm_y         : num  0 -0.02 -0.02 -0.03 -0.03 -0.03 -0.03 -0.03
-0.03 -0.03 ...
##  $ gyros_arm_z         : num  -0.02 -0.02 -0.02 0.02 0 0 0 -0.02 -0.02 0 .
..
##  $ accel_arm_x         : int  -288 -290 -289 -289 -289 -289 -289 -288 -288
-290 ...
##  $ accel_arm_y         : int  109 110 110 111 111 111 111 109 110 110 ...
##  $ accel_arm_z         : int  -123 -125 -126 -123 -123 -122 -125 -122 -124
-123 ...
##  $ magnet_arm_x        : int  -368 -369 -368 -372 -374 -369 -373 -369 -376
-366 ...
```

```
##  $ magnet_arm_y        : int  337 337 344 344 337 342 336 341 334 339 ...
##  $ magnet_arm_z        : int  516 513 513 512 506 513 509 518 516 509 ...
##  $ roll_dumbbell       : num  13.1 13.1 12.9 13.4 13.4 ...
##  $ pitch_dumbbell      : num  -70.5 -70.6 -70.3 -70.4 -70.4 ...
##  $ yaw_dumbbell        : num  -84.9 -84.7 -85.1 -84.9 -84.9 ...
##  $ total_accel_dumbbell: int  37 37 37 37 37 37 37 37 37 37 ...
##  $ gyros_dumbbell_x    : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ gyros_dumbbell_y    : num  -0.02 -0.02 -0.02 -0.02 -0.02 -0.02 -0.02 -0
.02 -0.02 -0.02 ...
##  $ gyros_dumbbell_z    : num  0 0 0 -0.02 0 0 0 0 0 0 ...
##  $ accel_dumbbell_x    : int  -234 -233 -232 -232 -233 -234 -232 -232 -235
-233 ...
##  $ accel_dumbbell_y    : int  47 47 46 48 48 48 47 47 48 47 ...
##  $ accel_dumbbell_z    : int  -271 -269 -270 -269 -270 -269 -270 -269 -270
-269 ...
##  $ magnet_dumbbell_x   : int  -559 -555 -561 -552 -554 -558 -551 -549 -558
-564 ...
##  $ magnet_dumbbell_y   : int  293 296 298 303 292 294 295 292 291 299 ...
##  $ magnet_dumbbell_z   : num  -65 -64 -63 -60 -68 -66 -70 -65 -69 -64 ...
##  $ roll_forearm        : num  28.4 28.3 28.3 28.1 28 27.9 27.9 27.7 27.7 2
7.6 ...
##  $ pitch_forearm       : num  -63.9 -63.9 -63.9 -63.9 -63.9 -63.9 -63.9 -6
3.8 -63.8 -63.8 ...
##  $ yaw_forearm         : num  -153 -153 -152 -152 -152 -152 -152 -152 -152
-152 ...
##  $ total_accel_forearm : int  36 36 36 36 36 36 36 36 36 36 ...
##  $ gyros_forearm_x     : num  0.03 0.02 0.03 0.02 0.02 0.02 0.02 0.03 0.02
0.02 ...
##  $ gyros_forearm_y     : num  0 0 -0.02 -0.02 0 -0.02 0 0 0 -0.02 ...
##  $ gyros_forearm_z     : num  -0.02 -0.02 0 0 -0.02 -0.03 -0.02 -0.02 -0.0
2 -0.02 ...
##  $ accel_forearm_x     : int  192 192 196 189 189 193 195 193 190 193 ...
##  $ accel_forearm_y     : int  203 203 204 206 206 203 205 204 205 205 ...
##  $ accel_forearm_z     : int  -215 -216 -213 -214 -214 -215 -215 -214 -215
-214 ...
##  $ magnet_forearm_x    : int  -17 -18 -18 -16 -17 -9 -18 -16 -22 -17 ...
##  $ magnet_forearm_y    : num  654 661 658 658 655 660 659 653 656 657 ...
##  $ magnet_forearm_z    : num  476 473 469 469 473 478 470 476 473 465 ...
##  $ classe              : Factor w/ 5 levels "A","B","C","D",..: 1 1 1 1 1
1 1 1 1 1 ...
```

Annex B Confusion Matrices for Random Forest Classifier

Insample Results

```
## Confusion Matrix and Statistics
##
##           Reference
```

```
## Prediction    A    B    C    D    E
##          A 4185    0    0    0    0
##          B    0 2848    0    0    0
##          C    0    0 2567    0    0
##          D    0    0    0 2412    0
##          E    0    0    0    0 2706
##
## 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   0.1839
## Detection Rate         0.2843   0.1935   0.1744   0.1639   0.1839
## Detection Prevalence   0.2843   0.1935   0.1744   0.1639   0.1839
## Balanced Accuracy      1.0000   1.0000   1.0000   1.0000   1.0000
```

Out of Sample Results

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1394    0    0    0    1
##          B    1  946    2    0    0
##          C    0    3  852    0    0
##          D    0    0    4  799    1
##          E    0    0    0    3  898
##
## Overall Statistics
##
##                Accuracy : 0.9969
##                  95% CI : (0.995, 0.9983)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9961
##  Mcnemar's Test P-Value : NA
```

```
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9993   0.9968   0.9930   0.9963   0.9978
## Specificity            0.9997   0.9992   0.9993   0.9988   0.9993
## Pos Pred Value         0.9993   0.9968   0.9965   0.9938   0.9967
## Neg Pred Value         0.9997   0.9992   0.9985   0.9993   0.9995
## Prevalence             0.2845   0.1935   0.1750   0.1635   0.1835
## Detection Rate         0.2843   0.1929   0.1737   0.1629   0.1831
## Detection Prevalence   0.2845   0.1935   0.1743   0.1639   0.1837
## Balanced Accuracy      0.9995   0.9980   0.9961   0.9975   0.9985
```

Annex C Confusion Matrices for Boosting Classifier

Insample Results

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 4184    1    0    0    0
##          B   14 2819   12    2    1
##          C    0   13 2547    4    3
##          D    0   12   18 2382    0
##          E    0    4    2   21 2679
##
## Overall Statistics
##
##                Accuracy : 0.9927
##                  95% CI : (0.9912, 0.994)
##     No Information Rate : 0.2852
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9908
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9967   0.9895   0.9876   0.9888   0.9985
## Specificity            0.9999   0.9976   0.9984   0.9976   0.9978
## Pos Pred Value         0.9998   0.9898   0.9922   0.9876   0.9900
## Neg Pred Value         0.9987   0.9975   0.9974   0.9978   0.9997
## Prevalence             0.2852   0.1936   0.1752   0.1637   0.1823
## Detection Rate         0.2843   0.1915   0.1731   0.1618   0.1820
## Detection Prevalence   0.2843   0.1935   0.1744   0.1639   0.1839
## Balanced Accuracy      0.9983   0.9935   0.9930   0.9932   0.9981
```

Out of Sample Results

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1390    5    0    0    0
##          B    7  928   14    0    0
##          C    0    4  850    1    0
##          D    0    4    4  796    0
##          E    1    0    1   13  886
##
## Overall Statistics
##
##                Accuracy : 0.989
##                  95% CI : (0.9857, 0.9917)
##     No Information Rate : 0.2851
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9861
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9943   0.9862   0.9781   0.9827   1.0000
## Specificity            0.9986   0.9947   0.9988   0.9980   0.9963
## Pos Pred Value         0.9964   0.9779   0.9942   0.9900   0.9834
## Neg Pred Value         0.9977   0.9967   0.9953   0.9966   1.0000
## Prevalence             0.2851   0.1919   0.1772   0.1652   0.1807
## Detection Rate         0.2834   0.1892   0.1733   0.1623   0.1807
## Detection Prevalence   0.2845   0.1935   0.1743   0.1639   0.1837
## Balanced Accuracy      0.9964   0.9904   0.9884   0.9904   0.9981
```