MachineLearningProject

Matthew Lessman

Saturday, November 22, 2014

Report Overview

Goal: predict the manner in which they did the exercise

Data Source: http://groupware.les.inf.puc-rio.br/har

You should create a report describing:

- how you built your model,
- how you used cross validation,
- what you think the expected out of sample error is,
- why you made the choices you did.
- You will also use your prediction model to predict 20 different test cases

From the paper:

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)
- throwing the hips to the front (Class E).

Read more: http://groupware.les.inf.puc-rio.br/har#ixzz3JozJ2800

Read the Data into rawTraining and finalTesting

```
rawTraining <- read.csv("pml-training.csv")
finalTesting <- read.csv("pml-testing.csv")
```

Load Packages

```
library(caret)
library(ggplot2)
library(randomForest)
```

Splitting the Data into Training and Testing Datasets

```
inTrain <- createDataPartition(y = rawTraining$classe, p = 0.75, list = FALSE)
training <- rawTraining[inTrain,]
testing <- rawTraining[-inTrain,]
dim(training)
```

```
## [1] 14718   160
```

```
dim(testing)
```

```
## [1] 4904  160
```

Reduce the Size of the Datasets

```
training <- training[, c(7:11,37:49, 60:68, 84:86, 102, 113:124, 140, 151:160)]
testing <- testing[, c(7:11,37:49, 60:68, 84:86, 102, 113:124, 140, 151:160)]
```

Check the Size of the Dataset:

```
dim(training)
```

```
## [1] 14718    54
```

```
dim(testing)
```

```
## [1] 4904   54
```

Summary Information and Exploratory Analysis

```
str(training)
```

```
## 'data.frame':    14718 obs. of  54 variables:
##  $ 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 1.42 1.43 1.45 1.45 1.42 ...
##  $ pitch_belt          : num  8.07 8.07 8.07 8.05 8.07 8.09 8.16 8.17 8.18 8.2 ...
##  $ yaw_belt            : num  -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -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 0.03 0.03 0.02 ...
##  $ 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 -0.02 0 ...
```

```
##  $ accel_belt_x        : int  -21 -22 -20 -22 -21 -22 -20 -21 -21 -22 ...
##  $ accel_belt_y        : int  4 4 5 3 2 3 2 4 2 4 ...
##  $ accel_belt_z        : int  22 22 23 21 24 21 24 22 23 21 ...
##  $ magnet_belt_x       : int  -3 -7 -2 -6 -6 -4 1 -3 -5 -3 ...
##  $ magnet_belt_y       : int  599 608 600 604 600 599 602 609 596 606 ...
##  $ magnet_belt_z       : int  -313 -311 -305 -310 -302 -311 -312 -308 -317 -309 ...
##  $ 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 21.9 21.7 21.6 21.5 21.4 ...
##  $ 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 0.02 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 -0.02 ...
##  $ gyros_arm_z         : num  -0.02 -0.02 -0.02 0.02 0 0 -0.02 -0.02 0 -0.02 ...
##  $ accel_arm_x         : int  -288 -290 -289 -289 -289 -289 -288 -288 -290 -287 ...
##  $ accel_arm_y         : int  109 110 110 111 111 111 109 110 110 111 ...
##  $ accel_arm_z         : int  -123 -125 -126 -123 -123 -125 -122 -124 -123 -124 ...
##  $ magnet_arm_x        : int  -368 -369 -368 -372 -374 -373 -369 -376 -366 -372 ...
##  $ magnet_arm_y        : int  337 337 344 344 337 336 341 334 339 338 ...
##  $ magnet_arm_z        : int  516 513 513 512 506 509 518 516 509 509 ...
##  $ 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 -0.02 ...
##  $ accel_dumbbell_x    : int  -234 -233 -232 -232 -233 -232 -232 -235 -233 -234 ...
##  $ accel_dumbbell_y    : int  47 47 46 48 48 47 47 48 47 48 ...
##  $ accel_dumbbell_z    : int  -271 -269 -270 -269 -270 -270 -269 -270 -269 -269 ...
##  $ magnet_dumbbell_x   : int  -559 -555 -561 -552 -554 -551 -549 -558 -564 -552 ...
##  $ magnet_dumbbell_y   : int  293 296 298 303 292 295 292 291 299 302 ...
##  $ magnet_dumbbell_z   : num  -65 -64 -63 -60 -68 -70 -65 -69 -64 -69 ...
##  $ roll_forearm        : num  28.4 28.3 28.3 28.1 28 27.9 27.7 27.7 27.6 27.2 ...
##  $ pitch_forearm       : num  -63.9 -63.9 -63.9 -63.9 -63.9 -63.9 -63.8 -63.8 -63.8 -63.9 ...
##  $ yaw_forearm         : num  -153 -153 -152 -152 -152 -152 -152 -152 -152 -151 ...
##  $ 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 0.03 0.02 0.02 0 ...
##  $ gyros_forearm_y     : num  0 0 -0.02 -0.02 0 0 0 0 -0.02 0 ...
##  $ gyros_forearm_z     : num  -0.02 -0.02 0 0 -0.02 -0.02 -0.02 -0.02 -0.02 -0.03 ...
##  $ accel_forearm_x     : int  192 192 196 189 189 195 193 190 193 193 ...
##  $ accel_forearm_y     : int  203 203 204 206 206 205 204 205 205 205 ...
##  $ accel_forearm_z     : int  -215 -216 -213 -214 -214 -215 -214 -215 -214 -215 ...
##  $ magnet_forearm_x    : int  -17 -18 -18 -16 -17 -18 -16 -22 -17 -15 ...
##  $ magnet_forearm_y    : num  654 661 658 658 655 659 653 656 657 655 ...
##  $ magnet_forearm_z    : num  476 473 469 469 473 470 476 473 465 472 ...
##  $ classe              : Factor w/ 5 levels "A","B","C","D",..: 1 1 1 1 1 1 1 1 1 1 ...
```

Building the Model

Enable parellel processing:

```
library(doSNOW)
cluster <- makeCluster(2)
registerDoSNOW(cluster)
```

Set the trControl parameter of the train function:

```
fitControl <- trainControl(method = "cv", allowParallel = TRUE, number = 2, repeats = 1)
```

Using random forest:

```
modelFit <- train(training$classe ~ ., data = training, method = "rf", preProcess = "pca", trControl = 
```

ConfustionMatrix:

```
confusionMatrix(testing$classe, predict(modelFit, testing))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1390    1    3    1    0
##          B    8  929   12    0    0
##          C    0   11  836    8    0
##          D    0    0   39  762    3
##          E    0    2    3    2  894
##
## Overall Statistics
##
##                Accuracy : 0.981
##                  95% CI : (0.977, 0.985)
##     No Information Rate : 0.285
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.976
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity             0.994    0.985    0.936    0.986    0.997
## Specificity             0.999    0.995    0.995    0.990    0.998
## Pos Pred Value          0.996    0.979    0.978    0.948    0.992
## Neg Pred Value          0.998    0.996    0.986    0.997    0.999
## Prevalence              0.285    0.192    0.182    0.158    0.183
## Detection Rate          0.283    0.189    0.170    0.155    0.182
## Detection Prevalence    0.284    0.194    0.174    0.164    0.184
## Balanced Accuracy       0.996    0.990    0.966    0.988    0.997
```

Disable parellel processing:

```
stopCluster(cluster)
```

Predict the 20 from the Final Testing Set

```
finalTesting <- finalTesting[, c(7:11,37:49, 60:68, 84:86, 102, 113:124, 140, 151:160)]
finalPred <- predict(modelFit, finalTesting)
finalPred
```

```
##  [1] B A C A A E D B A A B C B A E E A B B B
## Levels: A B C D E
```