Exercise Analysis

Bradley Hof

Sunday, October 26, 2014

Summary

In this analysis, we are going to analyze the factors that predict how "well" a workout exercise was performed using data from sensors attached to the participants body. In this analysis we built a random forest model using cross-validation, and achieved a 99.6% out-of-sample accuracy and an OOB error rate of 0.44%.

Training Data Set

The training data set contains 160 variables from body sensors. The sensors tracked the body movements and acceleration while the participant performed the excercise. A professional fitness intructor tracked each excercise and noted if the workout was performed correctly. The "classe" variable designates whether the exercise was performed correctly or not. A classe of "A" designates the exercise was correct. Otherwise, it was not performed correctly. Our goal is to predict the instructor's grade of the excercise ("A","B","C","D", or "E")

```
set.seed(12321)
trainurl <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
training <- read.csv(trainurl, stringsAsFactors=F, na.strings=c("","NA"))
```

Variable Selection

The data set contains many aggregate variables (average, minimum, maximum, etc) for each sensor. We are going to remove the aggregate variables because most of the variables are missing.

```
colPattern <- "^max_|skewness_|max_|min_|avg_|stddev_|amplitude_|var_|kurtosis_|raw_time|cvtd_|X|user_na
training <- training[,!grepl(colPattern, colnames(training))]
training$classe <- as.factor(training$classe)

str(training)
```

```
## 'data.frame':    19622 obs. of  53 variables:
##  $ roll_belt           : num  1.41 1.41 1.42 1.48 1.48 1.45 1.42 1.42 1.43 1.45 ...
##  $ pitch_belt          : num  8.07 8.07 8.07 8.05 8.07 8.06 8.09 8.13 8.16 8.17 ...
##  $ 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 0.02 0.02 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 -0.02 0 ...
##  $ accel_belt_x        : int  -21 -22 -20 -22 -21 -21 -22 -22 -20 -21 ...
##  $ accel_belt_y        : int  4 4 5 3 2 4 3 4 2 4 ...
##  $ accel_belt_z        : int  22 22 23 21 24 21 21 21 24 22 ...
##  $ magnet_belt_x       : int  -3 -7 -2 -6 -6 0 -4 -2 1 -3 ...
##  $ magnet_belt_y       : int  599 608 600 604 600 603 599 603 602 609 ...
##  $ magnet_belt_z       : int  -313 -311 -305 -310 -302 -312 -311 -313 -312 -308 ...
```

```
##  $ 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 21.8 21.7 21.6 ...
##  $ 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 -0.02 -0.03 -0.03 ...
##  $ gyros_arm_z         : num  -0.02 -0.02 -0.02 0.02 0 0 0 0 -0.02 -0.02 ...
##  $ accel_arm_x         : int  -288 -290 -289 -289 -289 -289 -289 -289 -288 -288 ...
##  $ accel_arm_y         : int  109 110 110 111 111 111 111 111 109 110 ...
##  $ accel_arm_z         : int  -123 -125 -126 -123 -123 -122 -125 -124 -122 -124 ...
##  $ magnet_arm_x        : int  -368 -369 -368 -372 -374 -369 -373 -372 -369 -376 ...
##  $ magnet_arm_y        : int  337 337 344 344 337 342 336 338 341 334 ...
##  $ magnet_arm_z        : int  516 513 513 512 506 513 509 510 518 516 ...
##  $ 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 -234 -232 -235 ...
##  $ accel_dumbbell_y    : int  47 47 46 48 48 48 47 46 47 48 ...
##  $ accel_dumbbell_z    : int  -271 -269 -270 -269 -270 -269 -270 -272 -269 -270 ...
##  $ magnet_dumbbell_x   : int  -559 -555 -561 -552 -554 -558 -551 -555 -549 -558 ...
##  $ magnet_dumbbell_y   : int  293 296 298 303 292 294 295 300 292 291 ...
##  $ magnet_dumbbell_z   : num  -65 -64 -63 -60 -68 -66 -70 -74 -65 -69 ...
##  $ roll_forearm        : num  28.4 28.3 28.3 28.1 28 27.9 27.9 27.8 27.7 27.7 ...
##  $ pitch_forearm       : num  -63.9 -63.9 -63.9 -63.9 -63.9 -63.9 -63.9 -63.8 -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 0.02 0.03 0.02 ...
##  $ gyros_forearm_y     : num  0 0 -0.02 -0.02 0 -0.02 0 -0.02 0 0 ...
##  $ gyros_forearm_z     : num  -0.02 -0.02 0 0 -0.02 -0.03 -0.02 0 -0.02 -0.02 ...
##  $ accel_forearm_x     : int  192 192 196 189 189 193 195 193 193 190 ...
##  $ accel_forearm_y     : int  203 203 204 206 206 203 205 205 204 205 ...
##  $ accel_forearm_z     : int  -215 -216 -213 -214 -214 -215 -215 -213 -214 -215 ...
##  $ magnet_forearm_x    : int  -17 -18 -18 -16 -17 -9 -18 -9 -16 -22 ...
##  $ magnet_forearm_y    : num  654 661 658 658 655 660 659 660 653 656 ...
##  $ magnet_forearm_z    : num  476 473 469 469 473 478 470 474 476 473 ...
##  $ classe              : Factor w/ 5 levels "A","B","C","D",..: 1 1 1 1 1 1 1 1 1 1 ...
```

Cross Validation data set

Since we have a large data set (19622 rows), We split the training data set into a smaller training data set and a cross-validation testing set with a 70/30 split.

```r
inTrain <- createDataPartition(y=training$classe, p=.7, list=F)
train <- training[inTrain,]
test <- training[-inTrain,]
```

Training: 13737 rows
Cross-Validation: 5885 rows

Random Forest

We used a random forest to model the predictors on the training set. As you can see, the model provides an OOB error rate of 0.44%.

```
fit <- randomForest(classe~., data=train)
fit
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = train)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 0.44%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 3903    2    0    0    1    0.000768
## B    9 2645    4    0    0    0.004891
## C    0   10 2382    4    0    0.005843
## D    0    0   22 2227    3    0.011101
## E    0    0    1    5 2519    0.002376
```

Cross Valiadation

We used the cross-validation test set to test the accuacy of our prediction on an out-of-sample data set and achieved a 99.6% accuracy on the prediction.

```
testpreds <- predict(fit, test)
confusionMatrix(testpreds, test$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    5    0    0    0
##          B    0 1132    6    0    0
##          C    0    2 1019    8    0
##          D    0    0    1  956    3
##          E    0    0    0    0 1079
##
## Overall Statistics
##
##                Accuracy : 0.996
##                  95% CI : (0.994, 0.997)
##     No Information Rate : 0.284
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.995
##  Mcnemar's Test P-Value : NA
##
```

```
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity             1.000    0.994    0.993    0.992    0.997
## Specificity             0.999    0.999    0.998    0.999    1.000
## Pos Pred Value          0.997    0.995    0.990    0.996    1.000
## Neg Pred Value          1.000    0.999    0.999    0.998    0.999
## Prevalence              0.284    0.194    0.174    0.164    0.184
## Detection Rate          0.284    0.192    0.173    0.162    0.183
## Detection Prevalence    0.285    0.193    0.175    0.163    0.183
## Balanced Accuracy       0.999    0.996    0.996    0.995    0.999
```

Apply to Test set

We have 20 observations in which to predict. We need to prepare out dataset in a similar way to the training set by removing variables. The prediction for each sample is noted below.

```r
testurl <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
dsf <- read.csv(testurl, stringsAsFactors=F, na.strings=c("","NA"))
dsf <- dsf[,!grepl(colPattern,colnames(dsf))]
preds <- predict(fit, dsf)
preds
```

```
##  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
```