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

```
table(training$classe)
```

```
##
##    A    B    C    D    E
## 4185 2848 2567 2412 2706
```

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
```

Results

I chose to use a random forest model because I needed to predict categorical variables. I used cross validation by setting the parameter in the train function. The accuracy of the model is 98% and the in sample and out of sample error rates are 2%.