Practical Machine Learning Project

John J. Como, MD, MPH

October 25, 2015

Background

Using devices such as Jawbone Up, Nike FuelBand, and Fitbit it is now possible to collect a large amount of data about personal activity relatively inexpensively. These type of devices are part of the quantified self movement - a group of enthusiasts who take measurements about themselves regularly to improve their health, to find patterns in their behavior, or because they are tech geeks. One thing that people regularly do is quantify how much of a particular activity they do, but they rarely quantify how well they do it.

In this project, my goal was to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants. They were asked to perform barbell lifts correctly and incorrectly in 5 different ways.

More information is available from the website here: http://groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

Data

The training data for this project were available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv

The test data were available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

Both of these files were downloaded into the working directory, generating the two files: 1) pml-training.csv 2) pml-testing.csv

The data for this project came from this source: http://groupware.les.inf.puc-rio.br/har.

First, I read in the training and the testing data from the respective csv files. I changed all missing data to type "NA".

```
training<-read.csv("pml-training.csv", na.strings=c("#DIV/0!", "", "NA"))
testing<-read.csv("pml-testing.csv", na.strings=c("#DIV/0!", "", "NA"))
dim(training)
```

```
## [1] 19622   160
```

```
dim(testing)
```

```
## [1]  20 160
```

You can see that the original training file has 19622 obervations and the testing file has 20 observations. Both originally have 160 variables.

The relevant data are located in columns 8 through 159. The 160th column of the training data contains the outcome variable "classe". The 160th column of the 20 test cases contained the values 1 through 20, and it was deleted. Also, columns with overwhelminly missing data were also deleted - see variable "omit" below.

```
training<-training[, 8:160]
testing<-testing[, 8:160]
omit<-which(colSums(is.na(training))>19200)
training<-training[, -omit]
testing<-testing[, -omit]
testing<-testing[, -dim(testing)[2]]
dim(training)
```

```
## [1] 19622    53
```

```
dim(testing)
```

```
## [1] 20 52
```

Now the training file has 53 variables (includes the dependent variable "classe"), and the testing file has 52.

In cross-validation, we use the training set, we split it into training/test sets, we build a model on the training set, and we evaluate this model on the test set.

Three-fourths of the data is set to be training data ("train"), and one-fourth is testing data ("test"). This latter "test" data, which contains 4904 observations, is not to be confused with the 20 test cases referred to above. The "train" data contains 14718 observations.

The seed is set for reproducibility.

```
set.seed(125)
library(caret)
```

```
## Loading required package: lattice
## Loading required package: ggplot2
```

```
inTrain<-createDataPartition(training$classe, p=3/4, list=FALSE)
train<-training[inTrain,]
test<-training[-inTrain,]
```

A random forest predictor was fit relating the variable "classe" to the remaining variables in the training set. A confusion matrix was generated.

```
library(randomForest)
```

```
## randomForest 4.6-12
## Type rfNews() to see new features/changes/bug fixes.
```

```
modFit<-randomForest(classe~., data=train, ntree=500)
modFit
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = train, ntree = 500)
##                Type of random forest: classification
##                      Number of trees: 500
```

```
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 0.53%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 4183    1    0    0    1 0.0004778973
## B   16 2826    6    0    0 0.0077247191
## C    0   16 2546    5    0 0.0081807557
## D    0    0   23 2387    2 0.0103648425
## E    0    0    1    7 2698 0.0029563932
```

The OOB (out of bag) error estimate is very low at 0.53%. This indicates that the prediction model is accurate on the training set.

The random forest predictor was then used on the "test" data (4904 variables). Another confusion matrix was generated. We do not expect this error to be as small as the error on the training set.

```
predRF<-predict(modFit, test)
confusionMatrix(predRF, test$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1395    5    0    0    0
##          B    0  943    8    0    0
##          C    0    1  847    8    0
##          D    0    0    0  796    4
##          E    0    0    0    0  897
##
## Overall Statistics
##
##                Accuracy : 0.9947
##                  95% CI : (0.9922, 0.9965)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9933
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9937   0.9906   0.9900   0.9956
## Specificity            0.9986   0.9980   0.9978   0.9990   1.0000
## Pos Pred Value         0.9964   0.9916   0.9895   0.9950   1.0000
## Neg Pred Value         1.0000   0.9985   0.9980   0.9981   0.9990
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2845   0.1923   0.1727   0.1623   0.1829
## Detection Prevalence   0.2855   0.1939   0.1746   0.1631   0.1829
## Balanced Accuracy      0.9993   0.9958   0.9942   0.9945   0.9978
```

The out of sample error is the error you get on a new data set. This will always be greater than in sample error. The accuracy of the model is 0.993, indicate=ing that the prediction model is also very accurate on the testing set.

The algorithm was then applied to the 20 test cases. Files containing the predictions are written to the working directory.

```
predictionAssignment<-as.character(predict(modFit, testing))

pml_write_files = function(x){
  n = length(x)
  for(i in 1:n){
    filename = paste0("problem_id_",i,".txt")
    write.table(x[i],file=filename,quote=FALSE,row.names=FALSE,col.names=FALSE)
  }
}

predictionAssignment
```

```
##  [1] "B" "A" "B" "A" "A" "E" "D" "B" "A" "A" "B" "C" "B" "A" "E" "E" "A"
## [18] "B" "B" "B"
```

```
pml_write_files(predictionAssignment)
```