Machine Learning Class Project

Author: John Taylor

Date: July 27, 2014

Executive Summary

This analysis develops a machine learning prediction algorithm using data from research done by Velloso, et al., for their paper "Qualitative Activity Recognition of Weight Lifting Exercises". Their work is published on the web here. The algorithm classifies data from an exercise movement and catagorizes it as either correct or in one of four incorrect categories.

The Data

The following quotation from Velloso, et al.'s paper describes the data in the data set: "For feature extraction we used a sliding window approach with different lengths from 0.5 second to 2.5 seconds, with 0.5 second overlap. In each step of the sliding window approach we calculated features on the Euler angles (roll, pitch and yaw), as well as the raw accelerometer, gyroscope and magnetometer readings. For the Euler angles of each of the four sensors we calculated eight features: mean, variance, standard deviation, max, min, amplitude, kurtosis and skewness, generating in total 96 derived feature sets."

```
# Set the working directory and read in the data.
setwd("D:/My Documents/John's Stuff/Coursera/Machine Learning")
allTrainingData <- read.csv("pml-training.csv")
allTestingData <- read.csv("pml-testing.csv")
```

Looking at the data reveals a pattern of individual data points likely taken at 45Hz as the authors describe followed by a row summarizing the preceeding data. *Unfortunately, the dataset has compilation errors.* The min and max columns do not match the labels for the raw data. For instance, `max_roll_belt` appears to point to the data in the yaw belt column. Another error is that the row summaries do not yield the results for the rows indicated by `num_window`. For instance `stddev_roll_arm` does not provide the value of of the standard deviation for roll_arm in window 13. The value given is 0.4686, while the actual values are 0.497347 and 0.488386 for sample and population data, respectively. These summary errors will not be of consequence given the test data, as explained below.

In reviewing the testing data, none of the columns of summary information have values. Thus, if I use these data to test for the out of sample errors, I will only have the raw data to work with. Given this fact, I have deleted all the columns of summary information (avg., min, max, etc.), along with the user and timestamp columns. I initially keep the window column as this separates the data into time-based chunks.

```
# Remove unnecessary columns from the dataset.
goodColsTrainingData <- allTrainingData[,-c(2:5, 7, 12:36, 50:59, 69:83, 87:101, 103:112, 125:139, 141:
goodColsTestingData <- allTestingData[,-c(2:5, 7, 12:36, 50:59, 69:83, 87:101, 103:112, 125:139, 141:150
```

The test data set only has a single row from each window precluding the use of any averaging statistics as Velloso, et al. used. To train the model for this test set, I remove all the rows per window, save the one marked as a new window. While the test set data specifically does not do this, I can find no other rationale for choosing any other data point in the window.

```
# Remove all rows except for the new window rows
TrainingData <- goodColsTrainingData[goodColsTrainingData$new_window == "yes",]
# Remove the first two columns--the record number and the new window column aren't helpful data.
TrainingData <- TrainingData[,-c(1:2)]
```

I next split the training data into training and probing data sets to do model testing.

```
library(caret); library(lattice); library(ggplot2)
```

```
## Warning: package 'caret' was built under R version 3.1.1
```

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

```
inTrain <- createDataPartition(y=TrainingData$classe, p=0.75, list=FALSE)
training <- TrainingData[inTrain,]
probing <- TrainingData[-inTrain,]
```

Next, I begin the process of choosing a model. I tried a number of different models, the two that gave me the best results were Random Forests and Boosting. I show the results of the Random Forest model creation below.

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.1.1
```

```
## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug fixes.
```

```
modFit <- train(classe ~ ., data = training, method = "rf", preProcess = c("center", "scale"))
```

```
## Warning: package 'e1071' was built under R version 3.1.1
```

```
finMod <- modFit$finalModel
confusionMatrix(probing$classe, predict(modFit, probing))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  A  B  C  D  E
##          A 21  2  2  2  0
##          B  1 12  4  2  0
##          C  2  1 12  2  0
##          D  1  1  1 14  0
##          E  0  2  2  2 13
##
## Overall Statistics
##
##                Accuracy : 0.727
```

```
##                  95% CI : (0.629, 0.812)
##     No Information Rate : 0.253
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.658
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity             0.840    0.667    0.571    0.636    1.000
## Specificity             0.919    0.914    0.936    0.961    0.930
## Pos Pred Value          0.778    0.632    0.706    0.824    0.684
## Neg Pred Value          0.944    0.925    0.890    0.902    1.000
## Prevalence              0.253    0.182    0.212    0.222    0.131
## Detection Rate          0.212    0.121    0.121    0.141    0.131
## Detection Prevalence    0.273    0.192    0.172    0.172    0.192
## Balanced Accuracy       0.879    0.790    0.754    0.799    0.965
```

As you can see the accuracy was .7071, or 71% accurate. The other models I tested achieved the following results: gbm: 71% nb: 65% preprocessing with PCA and Random Forests: 64% ld: 54%

Conclusion

A result of 71% accuracy is not encouraging. I prefer to use summary statistics of the raw data as the authors did, but these data are not available in the Testing data set.