Correct Weight Lifting through Machine Learning

Phillip Chin

Tuesday, August 18, 2015

Based on exercise sensor data, we would like to determine if someone else performing an exercise correctly. In this case, the data that will be used is from 6 participants doing Unilateral Dumbbell Biceps Curls while wearing various sensors. Each repetition was categorized into one of the following classes:

- A - exactly according to the specification
- B - throwing the elbows to the front
- C - lifting the dumbbell only halfway
- D - lowering the dumbbell only halfway
- E - throwing the hips to the front

Using this data, we will use random forest as the model for predicting the classes for future input data.

Exploratory Data Analysis/Loading and Cleaning Data

There are two files:

- pml-training.csv - training data set
- pml-testing.csv - test data set

Some data cleaning was necessary. All columns with only NA's were removed. All columns with a near zero variance were also removed. In the training data set, there are some extra non-sensor measurement data (i.e. user_name, raw_timestamp_part_1, etc.) and calculated data (i.e. total_..., kurtosis_..., max_..., etc.). Those fields were removed to further reduce the training set and were not used in the training.

The training data set were normalized(center/scale). NA's were replaced with imputted values(knnImpute).

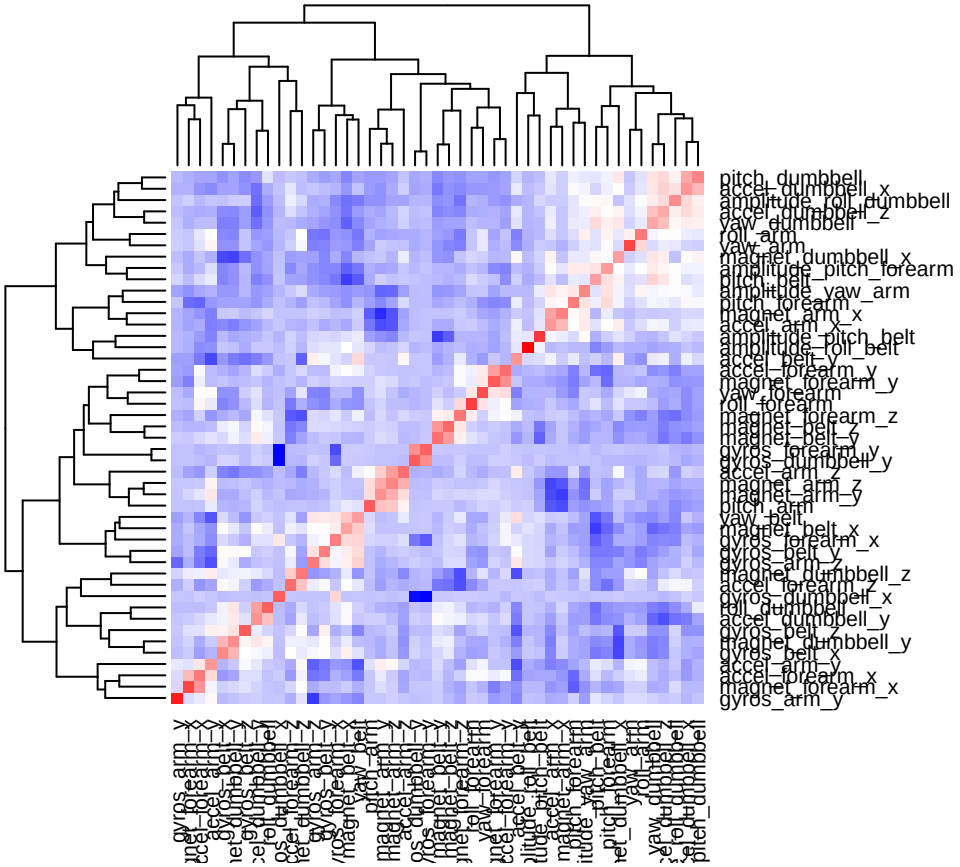
25% of the training data set was used for the actual training. The rest was used as our probe data set for checking the accuracy of our model through cross validation. A parred down training set was use to try to reduce the training time.

Based on the new training set, remove columns with a correlation of 0.9 or greater. Only the following columns were used:

```
##  [1] "roll_belt"               "pitch_belt"
##  [3] "yaw_belt"                "amplitude_roll_belt"
##  [5] "amplitude_pitch_belt"    "gyros_belt_x"
##  [7] "gyros_belt_y"            "gyros_belt_z"
##  [9] "accel_belt_x"            "accel_belt_y"
## [11] "accel_belt_z"            "magnet_belt_x"
## [13] "magnet_belt_y"           "magnet_belt_z"
## [15] "roll_arm"                "pitch_arm"
## [17] "yaw_arm"                 "gyros_arm_x"
## [19] "gyros_arm_y"             "gyros_arm_z"
## [21] "accel_arm_x"             "accel_arm_y"
## [23] "accel_arm_z"             "magnet_arm_x"
## [25] "magnet_arm_y"            "magnet_arm_z"
```

```
## [27] "amplitude_yaw_arm"         "roll_dumbbell"
## [29] "pitch_dumbbell"            "yaw_dumbbell"
## [31] "amplitude_roll_dumbbell"   "amplitude_pitch_dumbbell"
## [33] "gyros_dumbbell_x"          "gyros_dumbbell_y"
## [35] "gyros_dumbbell_z"          "accel_dumbbell_x"
## [37] "accel_dumbbell_y"          "accel_dumbbell_z"
## [39] "magnet_dumbbell_x"         "magnet_dumbbell_y"
## [41] "magnet_dumbbell_z"         "roll_forearm"
## [43] "pitch_forearm"             "yaw_forearm"
## [45] "amplitude_pitch_forearm"   "gyros_forearm_x"
## [47] "gyros_forearm_y"           "gyros_forearm_z"
## [49] "accel_forearm_x"           "accel_forearm_y"
## [51] "accel_forearm_z"           "magnet_forearm_x"
## [53] "magnet_forearm_y"          "magnet_forearm_z"
## [55] "classe"
```

Here is a heat map for our training data. The red regions are mostly on the diagonal as expected. There are not that many red hot spots elsewhere suggesting that none of the other values are highly correlated.



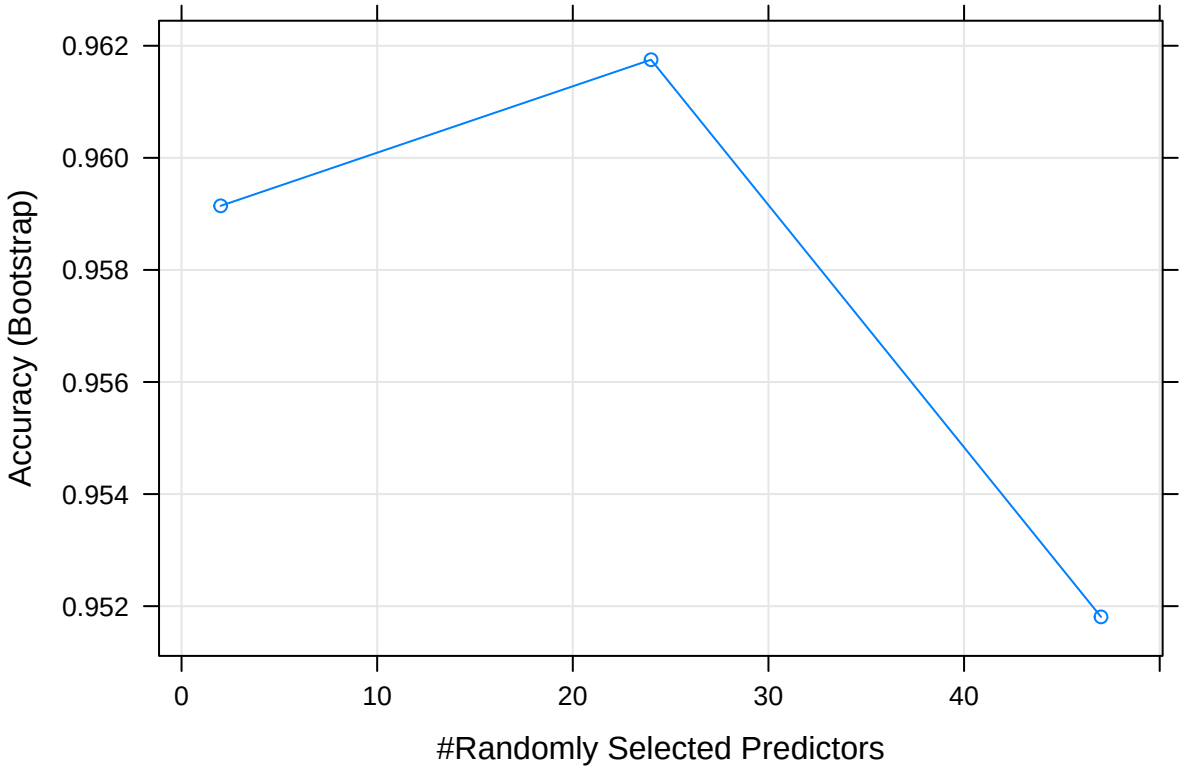
For the test data set, all of the unnecessary columns were removed. Only the columns that were kept in the training data set were saved. All NA's were replaced with 0's. They were not imputted.

Model Training

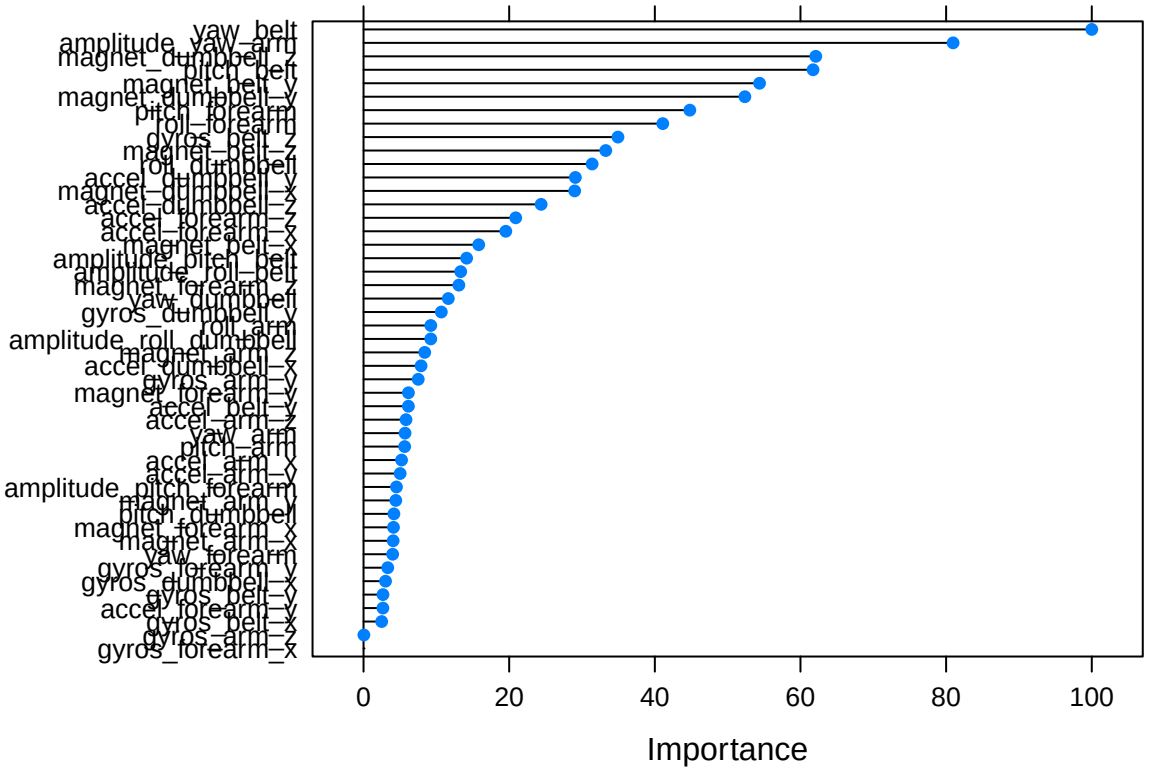
The model was trained using the Random Forest method.

Variable Importance

The model ended up using 24 out of the 47 variables from the trainning set as Selected Predictors.



Here are all of the variables ranked bases on importance.



Cross Validation

The probe data set was fed back into our model for cross validation.

Confusion Matrix This is confusion matrix using the probe data set. Most of the predictions lie on the diagonal so the model is accurately predicting the correct class in most cases.

```
##           Reference
## Prediction    A    B    C    D    E
##          A 4142   22   11    7    3
##          B   64 2733   47    3    0
##          C    5   42 2492   27    0
##          D    6    1   49 2345   11
##          E    4    1    6   17 2677
```

Going back to the training set, all predictions are along the diagonal.

```
##           Reference
## Prediction    A    B    C    D    E
##          A 1395    0    0    0    0
##          B    0  950    0    0    0
##          C    0    0  856    0    0
##          D    0    0    0  804    0
##          E    0    0    0    0  902
```

Accuracy Based on the probe data set, the accuracy is 0.9778457 and the **out of sample error** is 0.0221543.

Going back to the test data set, the accuracy is 1 and the **in sample error** is 0.

Key Quantities Here are the key quantities of the model based on the probe data set. They are all above 95%.

```
##          Sensitivity Specificity Pos Pred Value Neg Pred Value
## Class: A   0.9812841   0.9959024      0.9897252      0.9924976
## Class: B   0.9764202   0.9904330      0.9599579      0.9944388
## Class: C   0.9566219   0.9938893      0.9711613      0.9906988
## Class: D   0.9774906   0.9945599      0.9722222      0.9956108
## Class: E   0.9947975   0.9976713      0.9896488      0.9988343
##          Balanced Accuracy
## Class: A         0.9885932
## Class: B         0.9834266
## Class: C         0.9752556
## Class: D         0.9860253
## Class: E         0.9962344
```

Answers

Results from applying the test data set to the model:

```
##  [1] 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
```