Practical Machine Learning Final Project

Peggy Lu
February 19, 2017

Executive Summary

With devices such as Jawbone Up, Nike FuelBand, and Fitbit it is possible to collect a larger amount of data about personal activity. People who wear these type of devices can quantity how much of a particular activity they do, but they rarely quantity how well they do it. These participants were asked to performed one set of ten repetitions of the unilateral dumbbell biceps curl in five ways - 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) and throwing the hips to the front (Class E).

The training data for this project are available at https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv
The test data are available at https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

The goal of this project is to predict the manner in which they did the exercise based the data collected from accelerometers on the belt, forearm, arm, and dumbbell of six participants.

Data Exploratory

Include necessary libraries and download data from source website

```
library(caret)
```

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

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

```
library(gbm)
```

```
## Loading required package: survival
```

```
##
## Attaching package: 'survival'
```

```
## The following object is masked from 'package:caret':
##
##     cluster
```

```
## Loading required package: splines
```

```
## Loading required package: parallel
```

```
## Loaded gbm 2.1.1
```

```
library(rattle)
```

```
## Rattle: A free graphical interface for data mining with R.
## Version 4.1.0 Copyright (c) 2006-2015 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
trainCSV <- 'pml-training.csv'
validationCSV <- 'pml-testing.csv'
if (!file.exists(trainCSV) | (!file.exists(validationCSV)))
```

```
{
  trainURL <- 'https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv'
  validationURL <- 'https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv'
  download.file(trainURL, trainCSV)
  download.file(validationURL, validationCSV)
}
```

Create two datasets
One is for training and one for validation purpose. Remove any blank or null values.

```
trainRaw<-read.csv(trainCSV, na.strings = c("", "NA"))
validationRaw<-read.csv(validationCSV, na.strings = c("", "NA"))
```

For training dataset, there are total of 19622 observations with 160 variables.

```
dim(trainRaw)
```

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

For Validation dataset, there are total of 20 observations with 160 variables.

```
dim(validationRaw)
```

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

Covariate Creation

First 7 columns will be excluded from the model since they are irrelevant.

```
colnames(trainRaw)[1:7]
```

```
## [1] "X"                    "user_name"            "raw_timestamp_part_1"
## [4] "raw_timestamp_part_2" "cvtd_timestamp"       "new_window"
## [7] "num_window"
```

```
colnames(validationRaw)[1:7]
```

```
## [1] "X"                    "user_name"            "raw_timestamp_part_1"
## [4] "raw_timestamp_part_2" "cvtd_timestamp"       "new_window"
## [7] "num_window"
```

```
trainInput<-trainRaw[,-c(1:7)]
validationInput<-validationRaw[,-c(1:7)]
```

Remove Near Zero Variables

```
trainNZV<-nearZeroVar(trainInput,saveMetrics = TRUE)
trainInput<-trainInput[,trainNZV[,4]==FALSE]
validationNZV<-nearZeroVar(validationInput,saveMetrics = TRUE)
validationInput<-validationInput[,validationNZV[,4]==FALSE]
```

Since there are lots of variables in both datasets, the intersection of the 2 datasets will yield a common list of variables to be included in the models. See Figure 1 of the appendix for a common set of variables which exist in the training and validation datasets.

```
trainColumns<-colnames(trainInput, do.NULL = TRUE, prefix = "col")
validationColumns<-colnames(validationInput, do.NULL = TRUE, prefix = "col")
columns<-intersect(validationColumns, trainColumns)
trainInput<-trainInput[, c('classe', columns)]
```

Create Training and Test Sets
Split the original training dataset into training and test sets based on a 60/40 allocation.

```
set.seed(7)
inTrain<- createDataPartition(trainInput$classe, p=0.6, list=FALSE)
training<-trainInput[inTrain,]
testing<-trainInput[-inTrain,]
```

Set up the 5-fold cross validation to improve model runtime performance

```
fitControl <- trainControl(method="cv", number=5)
```

Model Selections

The following Models are selected for comparison purpose

Predict with Random Forest

```
modRF <- train(classe~ .,data=training,method="rf", trControl=fitControl, verbose=FALSE)
```

```
## Loading required package: randomForest
## randomForest 4.6-12
## Type rfNews() to see new features/changes/bug fixes.
##
## Attaching package: 'randomForest'
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
predRF<-predict(modRF, testing)
confusionMatrix(predRF, testing$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2229   13    0    0    0
##          B    3 1505    3    2    1
##          C    0    0 1359   10    2
##          D    0    0    6 1271    6
##          E    0    0    0    3 1433
##
## Overall Statistics
##
##                Accuracy : 0.9938
##                  95% CI : (0.9918, 0.9954)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9921
##  Mcnemar's Test P-Value : NA
##
```

```
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9987   0.9914   0.9934   0.9883   0.9938
## Specificity            0.9977   0.9986   0.9981   0.9982   0.9995
## Pos Pred Value         0.9942   0.9941   0.9912   0.9906   0.9979
## Neg Pred Value         0.9995   0.9979   0.9986   0.9977   0.9986
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2841   0.1918   0.1732   0.1620   0.1826
## Detection Prevalence   0.2858   0.1930   0.1747   0.1635   0.1830
## Balanced Accuracy      0.9982   0.9950   0.9958   0.9933   0.9966
```

Predict with GBM

```
modGBM <- train(classe~ .,data=training,method="gbm", trControl=fitControl, verbose=FALSE)
```

```
## Loading required package: plyr
```

```
predGBM<-predict(modGBM, testing)
confusionMatrix(predGBM, testing$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2205   40    0    1    2
##          B   16 1434   43    1   23
##          C    3   42 1299   37   15
##          D    7    2   21 1239   26
##          E    1    0    5    8 1376
##
## Overall Statistics
##
##                Accuracy : 0.9627
##                  95% CI : (0.9582, 0.9667)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9528
##  Mcnemar's Test P-Value : 3.035e-09
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9879   0.9447   0.9496   0.9635   0.9542
## Specificity            0.9923   0.9869   0.9850   0.9915   0.9978
## Pos Pred Value         0.9809   0.9453   0.9305   0.9568   0.9899
## Neg Pred Value         0.9952   0.9867   0.9893   0.9928   0.9898
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2810   0.1828   0.1656   0.1579   0.1754
## Detection Prevalence   0.2865   0.1933   0.1779   0.1651   0.1772
## Balanced Accuracy      0.9901   0.9658   0.9673   0.9775   0.9760
```

Predict with Classification Tree

```
modTree <- train(classe~ .,data=training,method="rpart", trControl=fitControl)
```

```
## Loading required package: rpart
```

```
predTree<-predict(modTree, testing)
confusionMatrix(predTree, testing$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2002  612  644  585  198
##          B   36  513   40  217  174
##          C  160  393  684  484  401
##          D    0    0    0    0    0
##          E   34    0    0    0  669
##
## Overall Statistics
##
##                Accuracy : 0.493
##                  95% CI : (0.4819, 0.5041)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3378
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8970  0.33794  0.50000   0.0000  0.46394
## Specificity            0.6368  0.92620  0.77802   1.0000  0.99469
## Pos Pred Value         0.4954  0.52347  0.32234      NaN  0.95164
## Neg Pred Value         0.9396  0.85363  0.88050   0.8361  0.89178
## Prevalence             0.2845  0.19347  0.17436   0.1639  0.18379
## Detection Rate         0.2552  0.06538  0.08718   0.0000  0.08527
## Detection Prevalence   0.5150  0.12490  0.27046   0.0000  0.08960
## Balanced Accuracy      0.7669  0.63207  0.63901   0.5000  0.72931
```

See Figure 2 for the Classification Tree plot

Conclusion

When comparing the results from all three models above, the accuracy from the Random Forest model is 0.9938, the accuracy of GBM is 0.9627 and the accuracy of the Classification Tree model is 0.493. Since the Random Forest model yields the best accuracy, it is used for predicting the validation dataset.

```
pred<-predict(modRF, validationInput)
print(pred)
```

```
##  [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
```

Out of Sample Error
Since the accuracy of the random forest is 0.9937548, therefore the out of sample error is 0.00624522(1-0.9937548)

```
1-confusionMatrix(predRF, testing$classe)$overall[1]
```

```
##   Accuracy
## 0.00624522
```

Appendix

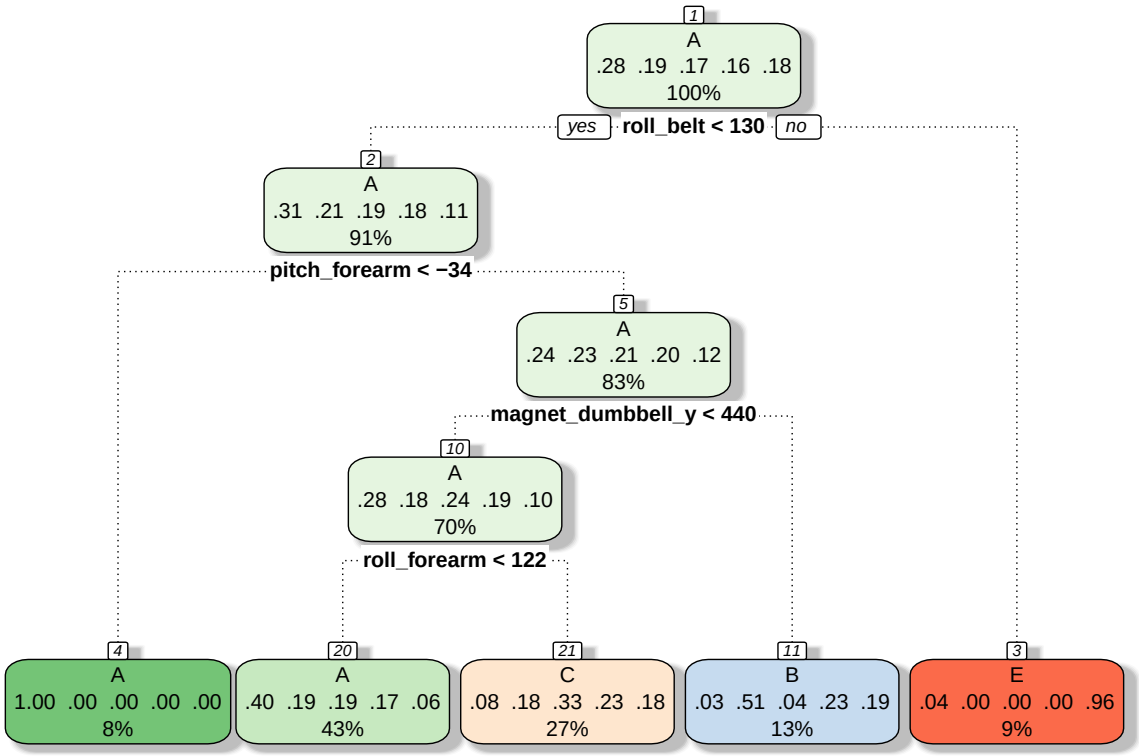
Figure 1 - Common variables in training and validation datasets

```
print(columns)
```

```
##  [1] "roll_belt"            "pitch_belt"           "yaw_belt"
##  [4] "total_accel_belt"     "gyros_belt_x"         "gyros_belt_y"
##  [7] "gyros_belt_z"         "accel_belt_x"         "accel_belt_y"
## [10] "accel_belt_z"         "magnet_belt_x"        "magnet_belt_y"
## [13] "magnet_belt_z"        "roll_arm"             "pitch_arm"
## [16] "yaw_arm"              "total_accel_arm"      "gyros_arm_x"
## [19] "gyros_arm_y"          "gyros_arm_z"          "accel_arm_x"
## [22] "accel_arm_y"          "accel_arm_z"          "magnet_arm_x"
## [25] "magnet_arm_y"         "magnet_arm_z"         "roll_dumbbell"
## [28] "pitch_dumbbell"       "yaw_dumbbell"         "total_accel_dumbbell"
## [31] "gyros_dumbbell_x"     "gyros_dumbbell_y"     "gyros_dumbbell_z"
## [34] "accel_dumbbell_x"     "accel_dumbbell_y"     "accel_dumbbell_z"
## [37] "magnet_dumbbell_x"    "magnet_dumbbell_y"    "magnet_dumbbell_z"
## [40] "roll_forearm"         "pitch_forearm"        "yaw_forearm"
## [43] "total_accel_forearm"  "gyros_forearm_x"      "gyros_forearm_y"
## [46] "gyros_forearm_z"      "accel_forearm_x"      "accel_forearm_y"
## [49] "accel_forearm_z"      "magnet_forearm_x"     "magnet_forearm_y"
## [52] "magnet_forearm_z"
```

Figure 2 - Plot for the Classification Tree

```
fancyRpartPlot(modTree$finalModel)
```



Rattle 2017−Feb−19 14:13:13 lu