Practical Machine Learning Course Project

Executive Summary

Using devices such as Jawbone Up, Nike FuelBand, and Fitbit 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, the goal is to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants to predict the manner in which they did the exercise, which is the "classe" variable in the training set.

From the anlysis in below, we conclude that using the Random Forest model, we reached accuracy of 99.44% to predict how well a person is preforming an excercise with .

Data Processing

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

The test data are available here https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

Downloading Data

```
TrainingRaw <- "./pml-trainingraw.csv"
url <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
download.file(url, destfile=TrainingRaw)
trainingraw <- read.csv(TrainingRaw)

TestingRaw <- "./pml-testingraw.csv"
url2 <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
download.file(url2, destfile=TestingRaw)
testingraw <- read.csv(TestingRaw)
```

Cleaning up Data

First we will collect all the data from columns whose name containing belt, forearm, arm, and dumbbell.

There are many columns with empty and NA values, remove those columns and only keep the columns with valid numbers.

```
idxTraining <- grepl("belt|forearm|arm|dumbbell", names(trainingraw), ignore.case=TRUE)
isMissing <- sapply(trainingraw, function (x) any(is.na(x) | x == ""))
cleanedTraining <- trainingraw[, idxTraining & !isMissing==TRUE]

# include the last column "classe" to the new training data set and give a name to it in new data set
cleanedTraining <- cbind(trainingraw$classe, cleanedTraining)
colnames(cleanedTraining)[1] <- "classe"
```

```r
# set the factor for the first column
cleanedTraining$classe <- factor(cleanedTraining$classe)

# Check the new cleaned data set
names(cleanedTraining)
```

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

```r
# Perform the same clean up for testing raw data
idxTesting <- grepl("belt|forearm|arm|dumbbell", names(testingraw), ignore.case=TRUE)

# Find the columns with "NA" or empty values
isMissingTest <- sapply(testingraw, function (x) any(is.na(x) | x == ""))

# Generate clean test data
cleanedTesting <- testingraw[, idxTesting & !isMissingTest==TRUE]
cleanedTesting <- cbind(testingraw$problem_id, cleanedTesting)
colnames(cleanedTesting)[1] <- "classe"
```

After the cleanup, besides of the outcome column "classe", the new datasets (both cleanedTraining and cleanedTesting) contain 52 predictor variables (we had 159 predictors in orignal datasets downloaded from web).

Building Models Using Training Data

First split the dataset into a typical 60% training and 40% testing dataset.

```r
library(caret)
library(randomForest)
set.seed(666666)
inTrain <- createDataPartition(y=cleanedTraining$classe, p=0.60, list=FALSE)
TrainingSet <- cleanedTraining[ inTrain,]
TestingSet  <- cleanedTraining[-inTrain,]
```

Next, use "rpart" (Recursive Partitioning and Regression Trees) method and "lda" (Linear Discriminant Analysis) to build 2 models and check the model accuracy using Confusion Matrix on the remaining 40% of test data from training set.

```
# Model #1 - Using rpart: Recursive Partitioning and Regression Trees
library(rpart)
modFit1 <- train(classe~., data=TrainingSet, method="rpart")
pred1 <- predict(modFit1, newdata=TestingSet)
c1 <- confusionMatrix(pred1, TestingSet$classe)$overall

# Model #2 - Using lda: Linear Discriminant Analysis.
library(MASS)
modFit2 <- train(classe~., data=TrainingSet, method="lda")
pred2 <- predict(modFit2, newdata=TestingSet)
c2 <- confusionMatrix(pred2, TestingSet$classe)$overall

# Compare the model accuracy
accuracyrate <- cbind(c1[1], c2[1])
colnames(accuracyrate) <- c("rpart", "lda")
accuracyrate
```

```
##              rpart       lda
## Accuracy 0.4910783 0.7100433
```

The model accuracy of "rpart" is ~54.78% and the model accuracy of lda is ~70.39%, both are not high.

Random Forest is one of the two top performing algorithms in predictions contests. Although it is difficult to interpret, it is often very accurate, We will create the third model using Random Forest and compare accuracy with other 2 models.

```
modFit3 <- randomForest(TrainingSet$classe ~ ., data = TrainingSet)
modFit3
```

```
##
## Call:
##  randomForest(formula = TrainingSet$classe ~ ., data = TrainingSet)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 0.59%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 3347    1    0    0    0 0.0002986858
## B   15 2261    3    0    0 0.0078982010
## C    0   15 2032    7    0 0.0107108082
## D    0    0   25 1905    0 0.0129533679
## E    0    0    1    2 2162 0.0013856813
```

```
pred3 <- predict(modFit3, newdata=TestingSet)
c3 <- confusionMatrix(pred3, TestingSet$classe)$overall
accuracyrate <- cbind(c1[1], c2[1], c3[1])
colnames(accuracyrate) <- c("rpart", "lda", "  RandomForest")
accuracyrate
```

```
##              rpart       lda   RandomForest
## Accuracy 0.4910783 0.7100433      0.9914606
```

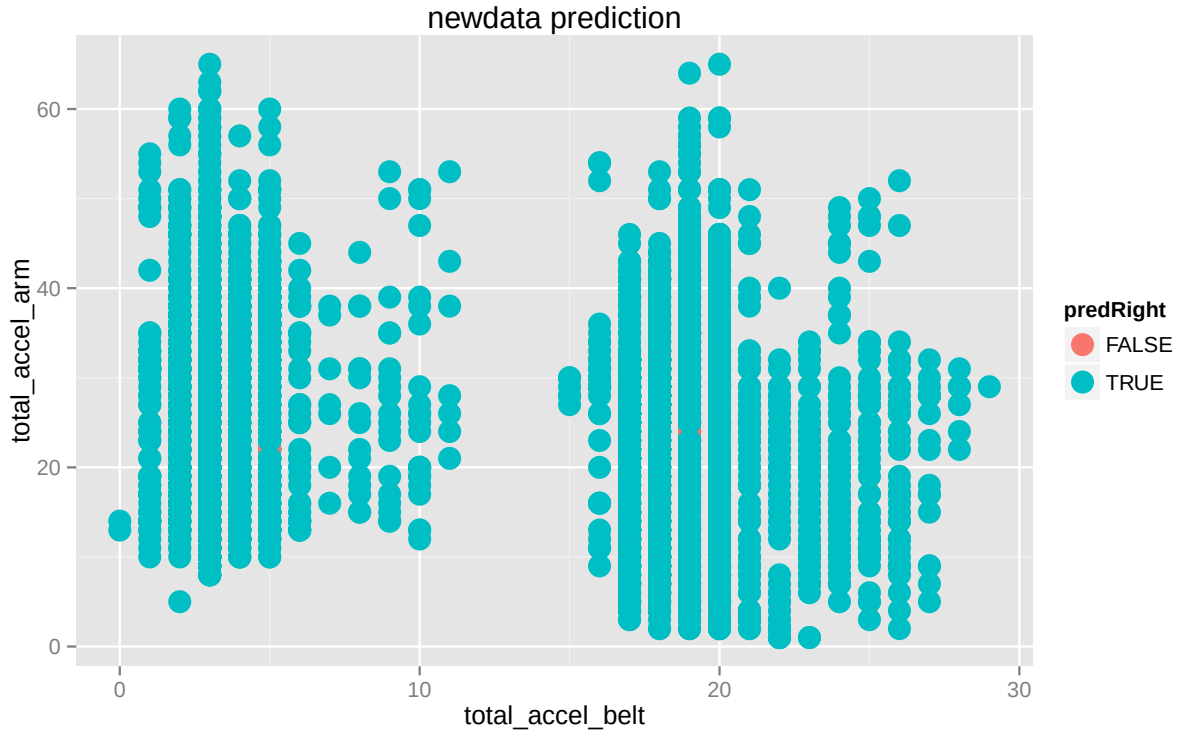
Cross-Validation

The Random Forest model is used to classify the remaining 40% of the data. A Confusion Matrix is created by passing the predictions from the model and the actual classifications, which determines the accuracy of the model.

```
confusionMatrix(pred3, TestingSet$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2225   17    0    0    0
##          B    5 1493   10    0    0
##          C    1    8 1356   17    0
##          D    0    0    2 1266    3
##          E    1    0    0    3 1439
##
## Overall Statistics
##
##                Accuracy : 0.9915
##                  95% CI : (0.9892, 0.9934)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9892
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9969   0.9835   0.9912   0.9844   0.9979
## Specificity            0.9970   0.9976   0.9960   0.9992   0.9994
## Pos Pred Value         0.9924   0.9901   0.9812   0.9961   0.9972
## Neg Pred Value         0.9988   0.9961   0.9981   0.9970   0.9995
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2836   0.1903   0.1728   0.1614   0.1834
## Detection Prevalence   0.2858   0.1922   0.1761   0.1620   0.1839
## Balanced Accuracy      0.9969   0.9906   0.9936   0.9918   0.9986
```

```
library(ggplot2)
TestingSet$predRight <- pred3==TestingSet$classe
qplot(total_accel_belt, total_accel_arm, colour=predRight, data=TestingSet, main="newdata prediction", s
```



The accuracy of the above model is 99.44% with 0.6% of out-of-sample error. By comparing with "rpart" and "lda", it turns out Random Forest is a great model to fit the given training dataset.

Using Models to Predict 20 Test Cases

Using random forest model to predict the classifications of the 20 results in new testing data (cleanedTesting).

```
# Apply the prediction to the cleaned test data
predictTest <- predict(modFit3, newdata=cleanedTesting)
predictTest
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  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
```