Practical Machine Learning - Class Project 1 - Human Activity Recogninition - Machine Learning

Sunday, June 21, 2015

Executive Summary

```
With the advent of wearables and other smart devices with the capability of
```

human activity monitoring, an increasing number of programs associated with Human Activity Recognition has emerged. The details of such programs and research can be found at: http://groupware.les.inf.puc-rio.br/har{[}http://groupware.les.inf.pu c-rio.br/har]. The wearables such as Jawbone, Nike, UP, Fuelband and Fitbit can be used to collect human wellness realted data and used to develop useful suggestions and programs for fitness. In this paper, we are taking a set of data that is available from the above cited web source and developing predictions based on the machine learning models. The data is collected from different activity monitors and the participants were asked to lift the barbell in different ways. The prediction models are used to predict the ways the exercise is performed. We engaged Decision Tree and Random Forest machine learning models to predict and found that Random Forest model is best fitting.

Data Processing

```
The data is taken from the HAR web source mentioned above and has excersise
```

infromation with various information. In the 'Weight Lifting Data set', six young participants were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in five ways and some of them are the correct way. These methods are captured in the variable 'Classe' in the data set. The 'Classe' variable has five values: 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).

```
First include all the required R packages to process the data and do the
```

prediction models and the exploratory data analysis.

```r
library(caret)
library(randomForest)
library(rpart)
library(rpart.plot)
library(ggplot2)
library(e1071)
```

```
For reproducibility the random seed is set to 1234.  Also the data is read in
```

from the downloaded data sets in the .CSV format. Also the the data is cleaned by excluding the NA variables and the irrelevant data variables from the data set.

```
set.seed(1234)
setwd("C:/Bahul/Coursera/JHU-DataScience/ML/ML-CProj1")
TrainingSet <- read.csv("pml-training.csv", na.strings=c("NA","#DIV/0!", ""))
TestingSet <- read.csv("pml-testing.csv", na.strings=c("NA","#DIV/0!", ""))
TrainingSet<-TrainingSet[,colSums(is.na(TrainingSet)) == 0]
TestingSet <-TestingSet[,colSums(is.na(TestingSet)) == 0]
TrainingSet   <-TrainingSet[,-c(1:7)]
TestingSet <-TestingSet[,-c(1:7)]
```

Cross Validation

The training data set has 19622 obsevrations and 53 variables, and the testing data set has 20 observations and 53 variables.

```
dim(TrainingSet)
```

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

```
dim(TestingSet)
```

```
## [1] 20 53
```

To do the cross validation the data is partitioned into two with 75% and 25% from the training and testing data sets respectively.

```
SubSamples <- createDataPartition(y=TrainingSet$classe, p=0.75, list=FALSE)
SubTraining <- TrainingSet[SubSamples, ]
SubTesting <- TrainingSet[-SubSamples, ]
```

Exploratory Data Analysis

As stated in the data processing section, the Classe variable has five values. The plot 1 in Appendix A, gives the plot of these values with corresponding frequencies. From the plot we can see the excercise is performed correctly (classe value A) most of the times with roughly 20% (That is 4000/19622). The rest of the values are incorrect methods as defined in the data processing section and they are closely with 12% to 13% or 2500 occurences.

Machine Learning Prediction Models

```
The prediction models used are decision tree and the random forest.
```

How the Prediction Model is bulit

```
The data is cleaned and processes as illustrated in the data processing
```

section; and then partitioned within the Training and Testing data sets. The the decision tree model and Random forest models were engaged. The best prediction model is chosen with the help of the confusion matrix test on the machine learning algorithms.

Decision Tree

```
The classe variable in the subTraining data set is used to model the decision tree as below.
```

```
Model1 <- rpart(classe ~ ., data=SubTraining, method="class")
Prediction1 <- predict(Model1, SubTesting, type = "class")
```

```
The Plot 2 of the Appendix A is the plot of the decision tree. The machine
```

learning algorithm is tested using the confusion matrix as shown in the Test 1 of Appendix A.

Random Forest model

```
The Random Forest prediction model is used to do the second model as below.
```

```
Model2 <- randomForest(classe ~. , data=SubTraining, method="class")
Prediction2 <- predict(Model2, SubTesting, type = "class")
```

```
The machine learning model is tested using the confusion matrix as shown in
```

the Test 2 of Appendix A.

Results and Conclusion

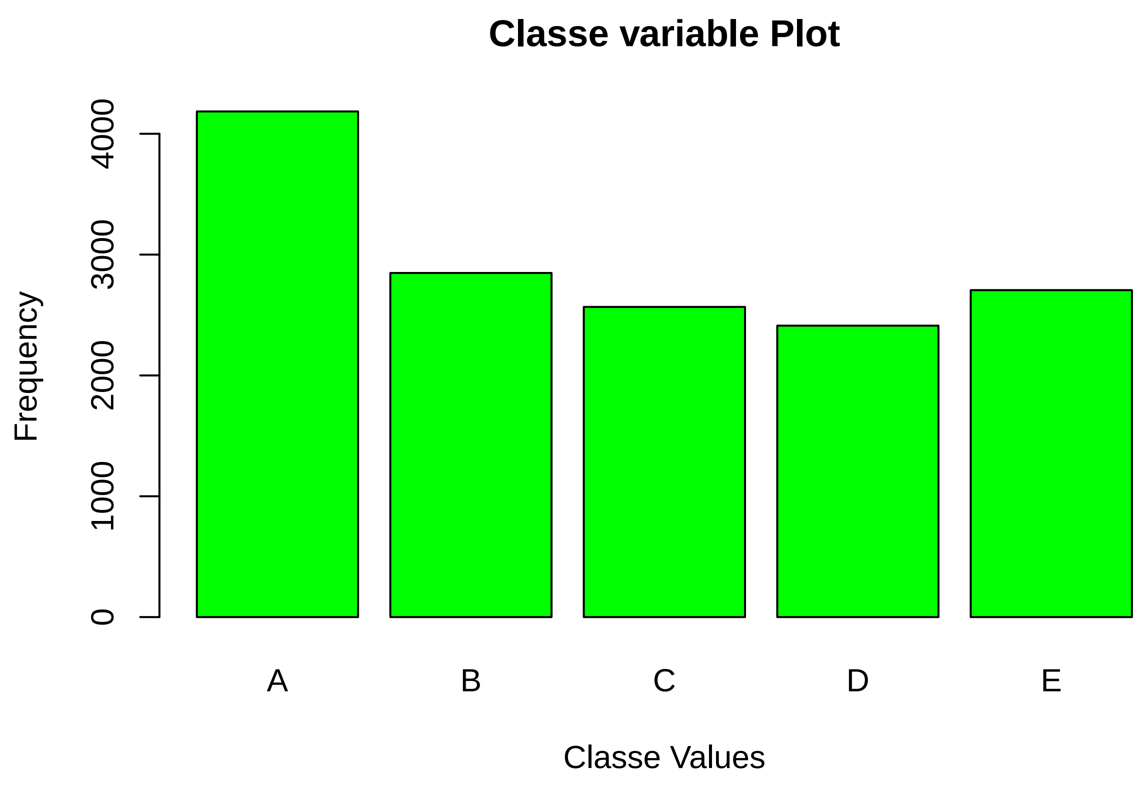
The Random Forest model has the better prediction compared to the Decision Tree model. The Randome Forest model has 0.9951 accuracy and the Decision Tree model has 0.7394 as illustrated by the Confusion Matrix Test.

Appendix A

The section of Appendix A holds the required plots and tests.

Plot 1: Plot of Classe variable

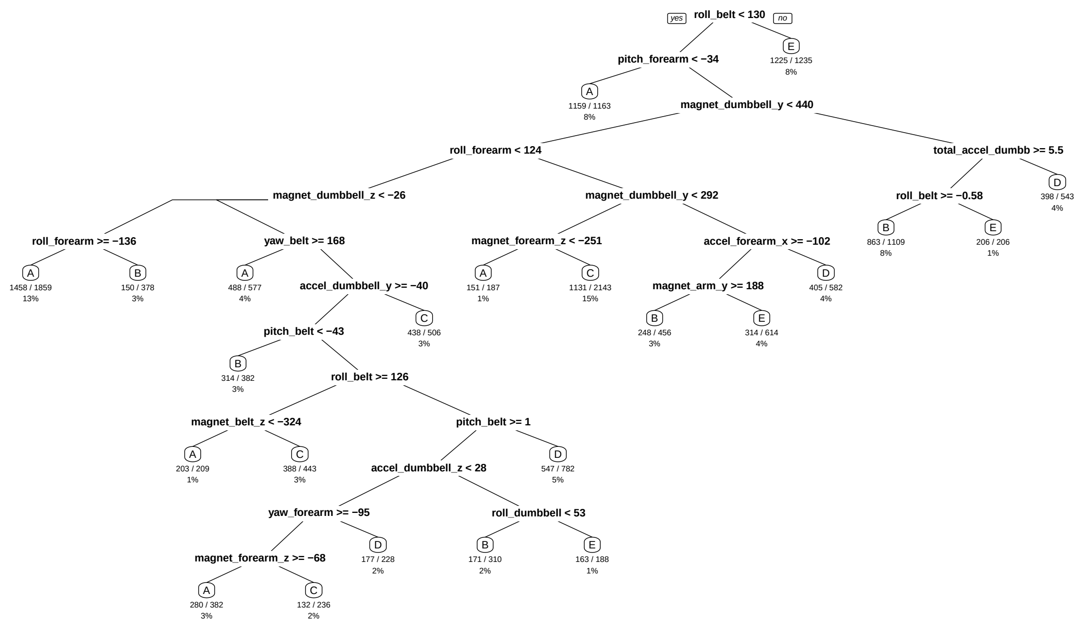
```
plot(SubTraining$classe, col="green", main="Classe variable Plot", xlab="Classe Values", ylab="Frequenc
```



Plot 2: Decision Tree on Classe Variable

```
rpart.plot(Model1, main="Decision Tree", extra=102, under=TRUE, faclen=0)
```

Decision Tree



Test 1: Confusion Matrix Test on Decision Tree Model

```
confusionMatrix(Prediction1, SubTesting$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1235  157   16   50   20
##          B   55  568   73   80  102
##          C   44  125  690  118  116
##          D   41   64   50  508   38
##          E   20   35   26   48  625
##
## Overall Statistics
##
##                Accuracy : 0.7394
##                  95% CI : (0.7269, 0.7516)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6697
##  Mcnemar's Test P-Value : < 2.2e-16
##
```

```
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8853   0.5985   0.8070   0.6318   0.6937
## Specificity            0.9307   0.9216   0.9005   0.9529   0.9678
## Pos Pred Value         0.8356   0.6469   0.6313   0.7247   0.8289
## Neg Pred Value         0.9533   0.9054   0.9567   0.9296   0.9335
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2518   0.1158   0.1407   0.1036   0.1274
## Detection Prevalence   0.3014   0.1790   0.2229   0.1429   0.1538
## Balanced Accuracy      0.9080   0.7601   0.8537   0.7924   0.8307
```

Test 2: Confusion Matrix Test on Random Forest Model

```
confusionMatrix(Prediction2, SubTesting$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1394    3    0    0    0
##          B    1  944   10    0    0
##          C    0    2  843    6    0
##          D    0    0    2  798    0
##          E    0    0    0    0  901
##
## Overall Statistics
##
##                Accuracy : 0.9951
##                  95% CI : (0.9927, 0.9969)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9938
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9993   0.9947   0.9860   0.9925   1.0000
## Specificity            0.9991   0.9972   0.9980   0.9995   1.0000
## Pos Pred Value         0.9979   0.9885   0.9906   0.9975   1.0000
## Neg Pred Value         0.9997   0.9987   0.9970   0.9985   1.0000
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2843   0.1925   0.1719   0.1627   0.1837
## Detection Prevalence   0.2849   0.1947   0.1735   0.1631   0.1837
## Balanced Accuracy      0.9992   0.9960   0.9920   0.9960   1.0000
```
