Machine Learning Algorithm on Human-Activity Data

Sufia Khatun

August 19, 2016

Synopsis

This human activity recognition research has traditionally focused on discriminating between different activities, i.e. to predict "which" activity was performed at a specific point in time. The Weight Lifting Exercises dataset was used to investigate "how (well)" an activity was performed by the wearer.

Six young health participants were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in five different fashions: 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).

Read more: http://groupware.les.inf.puc-rio.br/har#ixzz4Hmnx71jD

The goal of this project is to predict the manner in which they did the exercise. This is the "classe" variable in the training set.

```
library(caret)
library(randomForest)
library(rpart)
library(ggplot2)
library(doParallel)
source("http://peterhaschke.com/Code/multiplot.R") # for multiplot function
```

Loding Dataset

The training dataset was loaded from the following:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv

The test dataset was loaded from the following:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

During reading the dataset, the missing values were replaced by NA

```
# Replaced missing values with NA
data1 <- read.csv("pml-training.csv", na.strings=c("NA","#DIV/0!", ""))
data2 <- read.csv("pml-testing.csv", na.strings=c("NA","#DIV/0!", ""))
dim(data1)
```

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

```
dim(data2)
```

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

Data Processing

```
# Deleting columns with missing values.
newdata1 <- data1[,colSums(is.na(data1)) == 0]
newdata2 <- data2[,colSums(is.na(data2)) == 0] # test dataset
dim(newdata1); dim(newdata2)
```

```
## [1] 19622    60
```

```
## [1] 20 60
```

```
names(newdata1)
```

```
##  [1] "X"                    "user_name"            "raw_timestamp_part_1"
##  [4] "raw_timestamp_part_2" "cvtd_timestamp"       "new_window"
##  [7] "num_window"           "roll_belt"            "pitch_belt"
## [10] "yaw_belt"             "total_accel_belt"     "gyros_belt_x"
## [13] "gyros_belt_y"         "gyros_belt_z"         "accel_belt_x"
## [16] "accel_belt_y"         "accel_belt_z"         "magnet_belt_x"
## [19] "magnet_belt_y"        "magnet_belt_z"        "roll_arm"
## [22] "pitch_arm"            "yaw_arm"              "total_accel_arm"
## [25] "gyros_arm_x"          "gyros_arm_y"          "gyros_arm_z"
## [28] "accel_arm_x"          "accel_arm_y"          "accel_arm_z"
## [31] "magnet_arm_x"         "magnet_arm_y"         "magnet_arm_z"
## [34] "roll_dumbbell"        "pitch_dumbbell"       "yaw_dumbbell"
## [37] "total_accel_dumbbell" "gyros_dumbbell_x"     "gyros_dumbbell_y"
## [40] "gyros_dumbbell_z"     "accel_dumbbell_x"     "accel_dumbbell_y"
## [43] "accel_dumbbell_z"     "magnet_dumbbell_x"    "magnet_dumbbell_y"
## [46] "magnet_dumbbell_z"    "roll_forearm"         "pitch_forearm"
## [49] "yaw_forearm"          "total_accel_forearm"  "gyros_forearm_x"
## [52] "gyros_forearm_y"      "gyros_forearm_z"      "accel_forearm_x"
## [55] "accel_forearm_y"      "accel_forearm_z"      "magnet_forearm_x"
## [58] "magnet_forearm_y"     "magnet_forearm_z"     "classe"
```

```
# Now remove the variables that is not needed for this analysis
newdata1 <- newdata1[, -c(1:7)]
newdata2 <- newdata2[,-c(1:7)]
```

Cross-Validation

Since the test dataset was given, the training dataset was split into training and testing dataset.

```
set.seed(12300)
inTrain <- createDataPartition(y=newdata1$classe, p=0.75, list=FALSE)
training <- newdata1[inTrain, ]
testing <- newdata1[-inTrain, ]
```

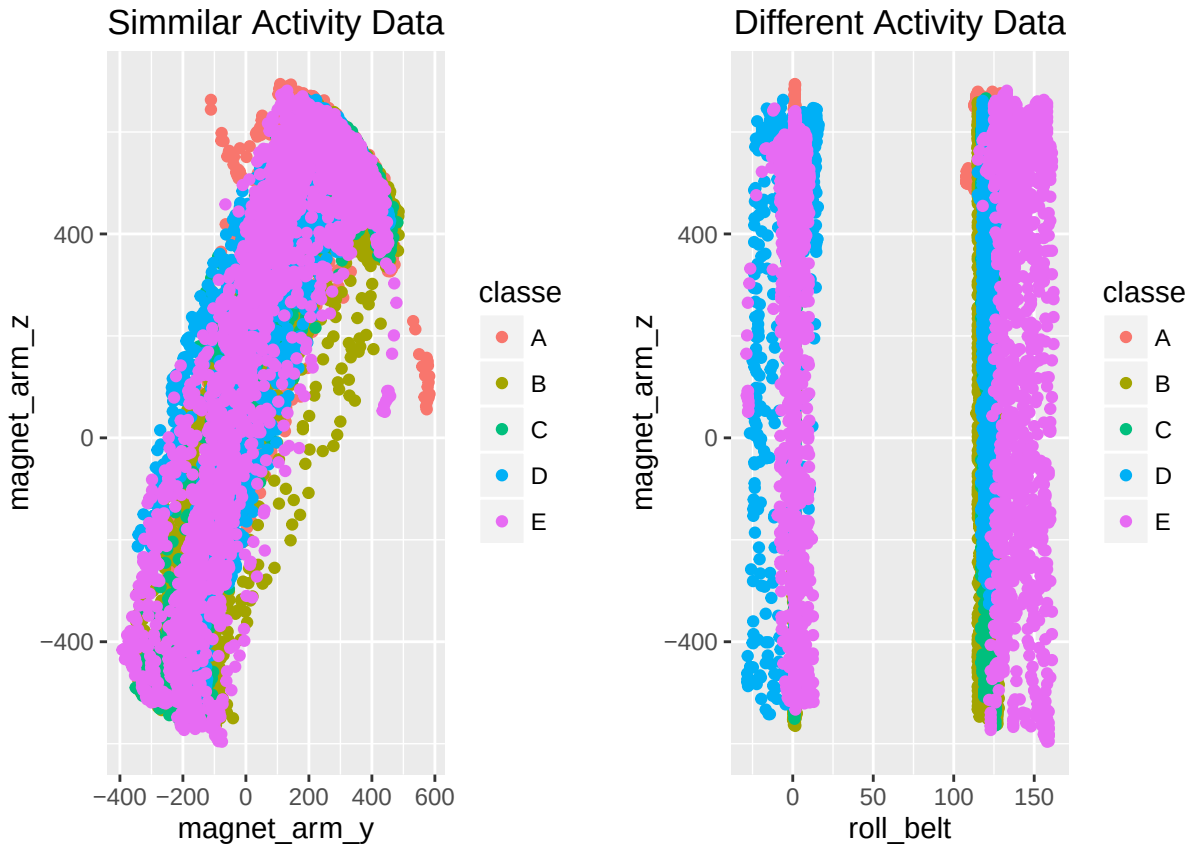
Visualize the relation of two variables for different classe and their correlation

```
g1 <- ggplot(data = training, aes(x=magnet_arm_y, magnet_arm_z, col =classe))+
  geom_point() + ggtitle("Simmilar Activity Data")
```

```
g2 <- ggplot(data = training, aes(x=roll_belt, magnet_arm_z, col = classe)) +
  geom_point() + ggtitle("Different Activity Data")

multiplot(g1, g2, cols = 2) # two plot in a single row
```

```
## Loading required package: grid
```




```
# Correlation between the Var
cor(training$magnet_arm_y, training$magnet_arm_z)
```

```
## [1] 0.8127497
```

```
cor(training$roll_belt, training$magnet_arm_z)
```

```
## [1] 0.03271678
```

Model Selection

This study was a multiclass classification problem. Also, from the above figure and correlation values, it was positive that there were some variables that were highly correlated. So, there was a possibility to get high variance or overfitting problem if linear or logistic regression were chosen. Random forest reduce the probability of high variance, handle large number of variables and can estimates of what variables are important in the classification. These were the reasons for chossing Random Forest algorithm for this study. For final decision, this model were compared with Decision Tree model.

Model1: Random Forest

```
model1 <- randomForest(classe~.,data=training, importance = TRUE, ntrees = 10)
pred1 <- predict(model1, training)
confusionMatrix(pred1, training$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 4185    0    0    0    0
##          B    0 2848    0    0    0
##          C    0    0 2567    0    0
##          D    0    0    0 2412    0
##          E    0    0    0    0 2706
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (0.9997, 1)
##     No Information Rate : 0.2843
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   1.0000   1.0000   1.0000   1.0000
## Specificity            1.0000   1.0000   1.0000   1.0000   1.0000
## Pos Pred Value         1.0000   1.0000   1.0000   1.0000   1.0000
## Neg Pred Value         1.0000   1.0000   1.0000   1.0000   1.0000
## Prevalence             0.2843   0.1935   0.1744   0.1639   0.1839
## Detection Rate         0.2843   0.1935   0.1744   0.1639   0.1839
## Detection Prevalence   0.2843   0.1935   0.1744   0.1639   0.1839
## Balanced Accuracy      1.0000   1.0000   1.0000   1.0000   1.0000
```

Model2: Decision Tree

```
model2 <- rpart(classe ~ ., data=training, method="class")
pred2 <- predict(model2, training, type = "class")
confusionMatrix(pred2, training$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 3734  451   82  154  134
##          B   86 1478  128  179   70
##          C   74  398 1741  190  264
##          D  256  324  540 1741  321
```

```
##          E   35  197   76  148 1917
##
## Overall Statistics
##
##                Accuracy : 0.721
##                  95% CI : (0.7136, 0.7282)
##     No Information Rate : 0.2843
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6467
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8922   0.5190   0.6782   0.7218   0.7084
## Specificity            0.9221   0.9610   0.9238   0.8829   0.9620
## Pos Pred Value         0.8198   0.7615   0.6528   0.5471   0.8078
## Neg Pred Value         0.9556   0.8928   0.9315   0.9418   0.9361
## Prevalence             0.2843   0.1935   0.1744   0.1639   0.1839
## Detection Rate         0.2537   0.1004   0.1183   0.1183   0.1302
## Detection Prevalence   0.3095   0.1319   0.1812   0.2162   0.1612
## Balanced Accuracy      0.9071   0.7400   0.8010   0.8024   0.8352
```

From observing the accuracy and prediction values of these two model, it can be said that Random Forest is better model for this dataset. Now, this model can be applied to the testing dataset before applying to the test dataset.

Model1 on testing dataset.

```
# predict outcome for test data set using the random forest model
pred3 <- predict(model1,testing)
confusionMatrix(pred3, testing$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1395    4    0    0    0
##          B    0  945    3    0    0
##          C    0    0  852    5    0
##          D    0    0    0  799    3
##          E    0    0    0    0  898
##
## Overall Statistics
##
##                Accuracy : 0.9969
##                  95% CI : (0.995, 0.9983)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9961
##  Mcnemar's Test P-Value : NA
```

```
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9958   0.9965   0.9938   0.9967
## Specificity            0.9989   0.9992   0.9988   0.9993   1.0000
## Pos Pred Value         0.9971   0.9968   0.9942   0.9963   1.0000
## Neg Pred Value         1.0000   0.9990   0.9993   0.9988   0.9993
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2845   0.1927   0.1737   0.1629   0.1831
## Detection Prevalence   0.2853   0.1933   0.1748   0.1635   0.1831
## Balanced Accuracy      0.9994   0.9975   0.9976   0.9965   0.9983
```

Model1 on test dataset.

```
pred4 <- predict(model1, newdata2) # Applying model1 on test dataset for quiz
pred4
```

```
##  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
```

In and Out of sample error

In of sample error:

```
(1-.9969)*100
```

```
## [1] 0.31
```

Out of sample error:

```
SA <- sum(pred3 == testing$classe)/length(pred3)

# Out of sample error
(1-SA)*100
```

```
## [1] 0.3058728
```

Conclusion:

After study the two models, it can be said that Random Forest model is appropriate for this dataset. The accuracy of random forest model is 99% for both training and test dataset where the accuracy of decision tree model is 72% for the training dataset. Also, this model predicted the test dataset appropriately.