Practical Machine Learning

Anita Lin

Sunday, December 14, 2014

Background

Using devices such as Jawbone Up, Nike FuelBand, and Fitbit it 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, your goal will be to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants. They were asked to perform barbell lifts correctly and incorrectly in 5 different ways. More information is available from the website here: [http://groupware.les.inf.puc-rio.br/har (http://groupware.les.inf.puc-rio.br/har)] (see the section on the Weight Lifting Exercise Dataset).

Data

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

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

Start to analysis

1.Prepare the train and test sets.

Load the training dataset

```
library(caret)
```

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

```
library(ggplot2)
library(lattice)
setwd("E:/Course/Johns hopkins_Data Science/Practical Machine Learning/Project")
fileUrl<-c("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv",
           "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv")
fileName<-c("pml-training.csv","pml-testing.csv")
##Download the files
#for (i in 1 : length(fileUrl)){
#  download.file(fileUrl[i],fileName[i])
#}
trainingDB<-read.csv(fileName[1])
```

Spilt training dataset

Spilt te training dataset into the train for building the model and test databases for verifying the model

```
inTrain <- createDataPartition(y = trainingDB$classe, p = 0.7, list = FALSE)
train <- trainingDB[inTrain, ]
test <- trainingDB[-inTrain, ]
```

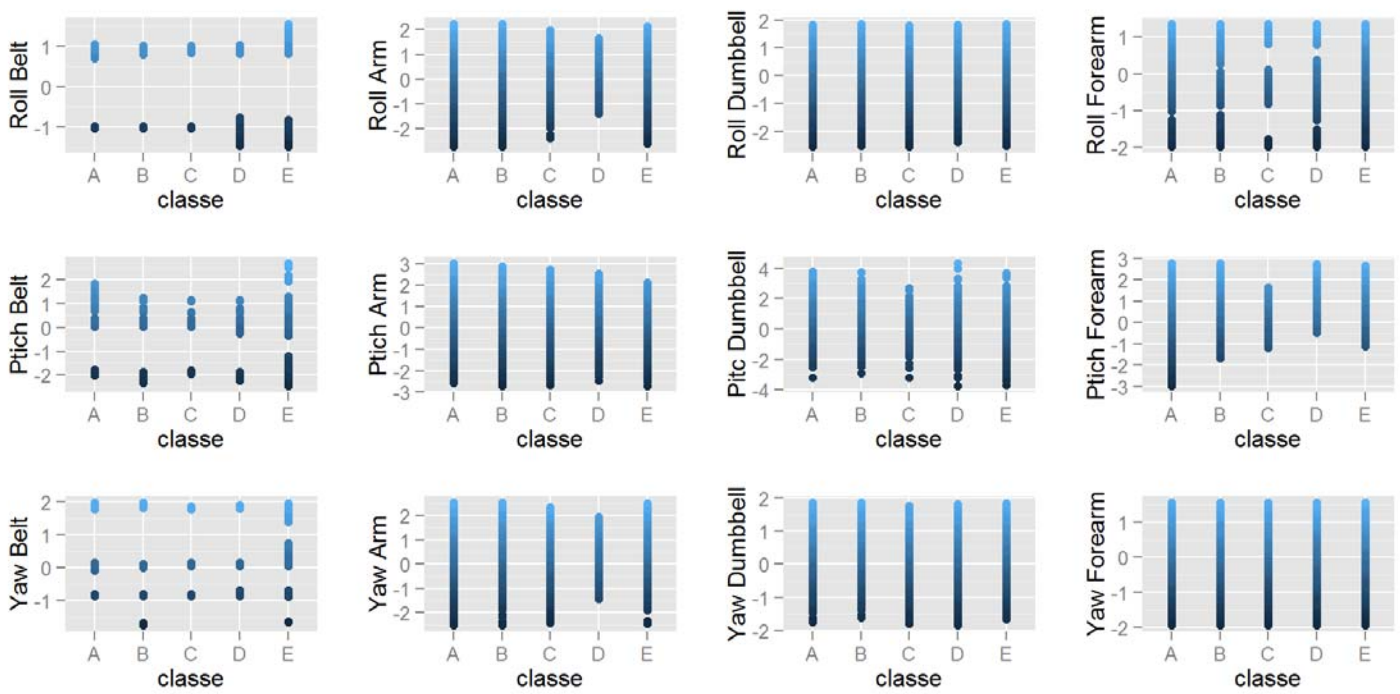
Choose the features for the model

There are too many variables on the training dataset and the range of values bween different four accelerometers are big. Due to the speed of selectling the fitting model, it might be a better way to standard the values of traget variables per each user. Ex: stardard value = (value-mean)/stard deviation.

```
##  [1] "classe"                "feature_roll_belt"
##  [3] "feature_roll_arm"      "feature_roll_dumbbell"
##  [5] "feature_roll_forearm"  "feature_pitch_belt"
##  [7] "feature_pitch_arm"     "feature_pitch_dumbbell"
##  [9] "feature_pitch_forearm" "feature_yaw_belt"
## [11] "feature_yaw_arm"       "feature_yaw_dumbbell"
## [13] "feature_yaw_forearm"
```

Plot figures for the relationshops

the relationshops between accelerometers with different body locations and classification



Covariates

If there are zero covarated, the varialbe will be removed. However, in the below list, no zero is in the freqRatio column. So, all variables will be accounted for the model.

```
nearZeroVar(train,saveMetric=TRUE)
```

```
##                         freqRatio percentUnique zeroVar   nzv
## classe                   1.469526    0.03639805   FALSE FALSE
## feature_roll_belt        1.075949    7.97117275   FALSE FALSE
## feature_roll_arm        47.680000   17.57297809   FALSE FALSE
## feature_roll_dumbbell    1.086022   86.51816263   FALSE FALSE
## feature_roll_forearm    11.500000   13.53279464   FALSE FALSE
## feature_pitch_belt       1.146341   12.30982019   FALSE FALSE
## feature_pitch_arm       95.360000   20.33922982   FALSE FALSE
## feature_pitch_dumbbell   2.287129   84.41435539   FALSE FALSE
## feature_pitch_forearm   72.605263   18.97794278   FALSE FALSE
## feature_yaw_belt         1.026954   13.05962000   FALSE FALSE
## feature_yaw_arm         31.368421   19.02162044   FALSE FALSE
## feature_yaw_dumbbell     1.134831   85.92851423   FALSE FALSE
## feature_yaw_forearm     15.581921   12.89218898   FALSE FALSE
```

2.Fit a model to predict the classe using 12 features as predictors

```
library(randomForest)
```

```
## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug fixes.
```

```
model <- randomForest(classe ~ ., data = train,method='rf',prox=TRUE)
model
```

```
## 
## Call:
##  randomForest(formula = classe ~ ., data = train, method = "rf",      prox = TRUE) 
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
## 
##         OOB estimate of  error rate: 1.29%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 3890   11    3    2    0 0.004096262
## B   20 2591   39    7    1 0.025206922
## C    0   19 2358   17    2 0.015859766
## D    1    4   19 2225    3 0.011989343
## E    1    5    9   14 2496 0.011485149
```

As the above te table, the OOB estimate of error rate is low, so this model might be high accuracy for predicaiton.

3.Runing the test dataset for vaerifying the model

Firstly, before starting to run the model, test datatset should be ready. The remaining 30% of data are assigned to test dataset. All vaiarbles in test dataset should be transfered to the standard values following by the process of trian dataset.

The first block is Roll Belt, the second block is Pitch Belt and the last one is Yaw Belt.

Model validation

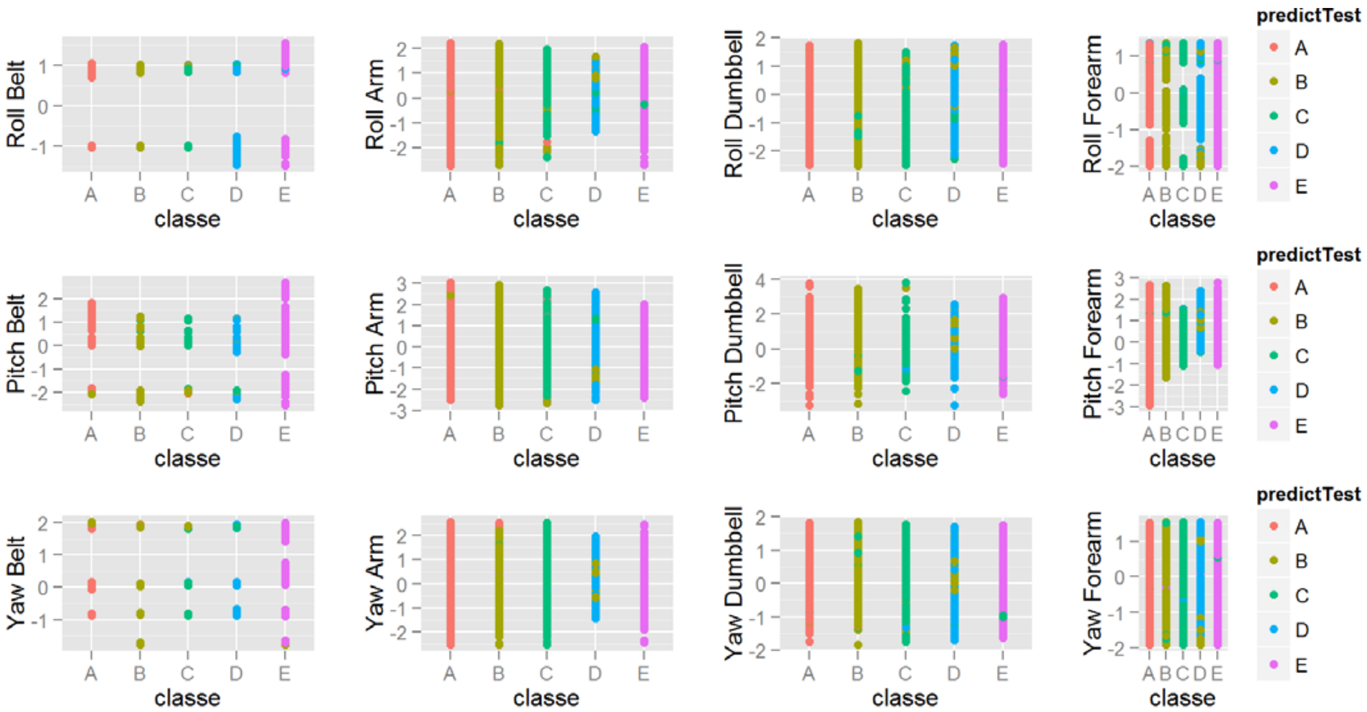
```
test$predictTest<- predict(model, test)
con<-confusionMatrix(test$classe, test$predictTest)
con
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1636   26    5    5    2
##          B   14 1072   22   13   18
##          C   18   57  871   71    9
##          D   11   94   64  786    9
##          E    0    5   40   12 1025
##
## Overall Statistics
##
##                Accuracy : 0.9159
##                  95% CI : (0.9085, 0.9229)
##     No Information Rate : 0.2853
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8935
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9744   0.8549   0.8693   0.8861   0.9643
## Specificity            0.9910   0.9855   0.9683   0.9644   0.9882
## Pos Pred Value         0.9773   0.9412   0.8489   0.8154   0.9473
## Neg Pred Value         0.9898   0.9617   0.9730   0.9795   0.9921
## Prevalence             0.2853   0.2131   0.1703   0.1507   0.1806
## Detection Rate         0.2780   0.1822   0.1480   0.1336   0.1742
## Detection Prevalence   0.2845   0.1935   0.1743   0.1638   0.1839
## Balanced Accuracy      0.9827   0.9202   0.9188   0.9253   0.9762
```

As the above the talbe, the accuracy is 0.9158879.

Plot figures for the predicted outcomes

In the bleow figure, it shows that the predicted outcomes between accelerometers with different body locations and classification. As you can see, there are a few missclassification; however, most calssifications are corrected.



The first block is Roll Belt, the second block is Pitch Belt and the last one is Yaw Belt.

4.Predication

This is the final setp for ruing the dataset, which are waiting for assigning the classification. Before prediction, data run the process of standardized values.

Predication of Model

```
predictTest <- predict(model, testingDB)
predictTest
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  E  A  A  E  A  E  E  E  A  E  B  A  D  A  E  D  E  E  E  D
## Levels: A B C D E
```

Conclusion

Using this model with multiple measuring points is possibly predicting the accurate classification.