Practical Machine Learning Assignment

Jae Kwon

September 16, 2018

Overview

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://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

Data Sources

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

The data for this project come from this source: http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har. If you use the document you create for this class for any purpose please cite them as they have been very generous in allowing their data to be used for this kind of assignment.

Dataset

```r
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.3.3

## Loading required package: lattice

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 3.3.2
```

```r
library(rpart)
library(rpart.plot)
library(RColorBrewer)
```

```
## Warning: package 'RColorBrewer' was built under R version 3.3.2
```

```r
library(rattle)
```

```
## Rattle: A free graphical interface for data science with R.
## Version 5.2.0 Copyright (c) 2006-2018 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```r
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.3.3
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:rattle':
##
##     importance
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```r
# Download the training data
download.file(url = "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv",
              destfile = "./pml-training.csv", method = "curl")

# Load the training dataset
dt_training <- read.csv("./pml-training.csv", na.strings=c("NA","#DIV/0!",""))

# Download the testing data
download.file(url = "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv",
              destfile = "./pml-testing.csv", method = "curl")

# Load the testing dataset
dt_testing <- read.csv("./pml-testing.csv", na.strings=c("NA","#DIV/0!",""))
```

Data Cleaning

```r
features <- names(dt_testing[,colSums(is.na(dt_testing)) == 0])[8:59]

# Only use features used in testing cases.
dt_training <- dt_training[,c(features,"classe")]
dt_testing <- dt_testing[,c(features,"problem_id")]

dim(dt_training); dim(dt_testing);
```

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

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

Partitioning the Dataset

```
set.seed(12345)

inTrain <- createDataPartition(dt_training$classe, p=0.6, list=FALSE)
training <- dt_training[inTrain,]
testing <- dt_training[-inTrain,]

dim(training); dim(testing);
```

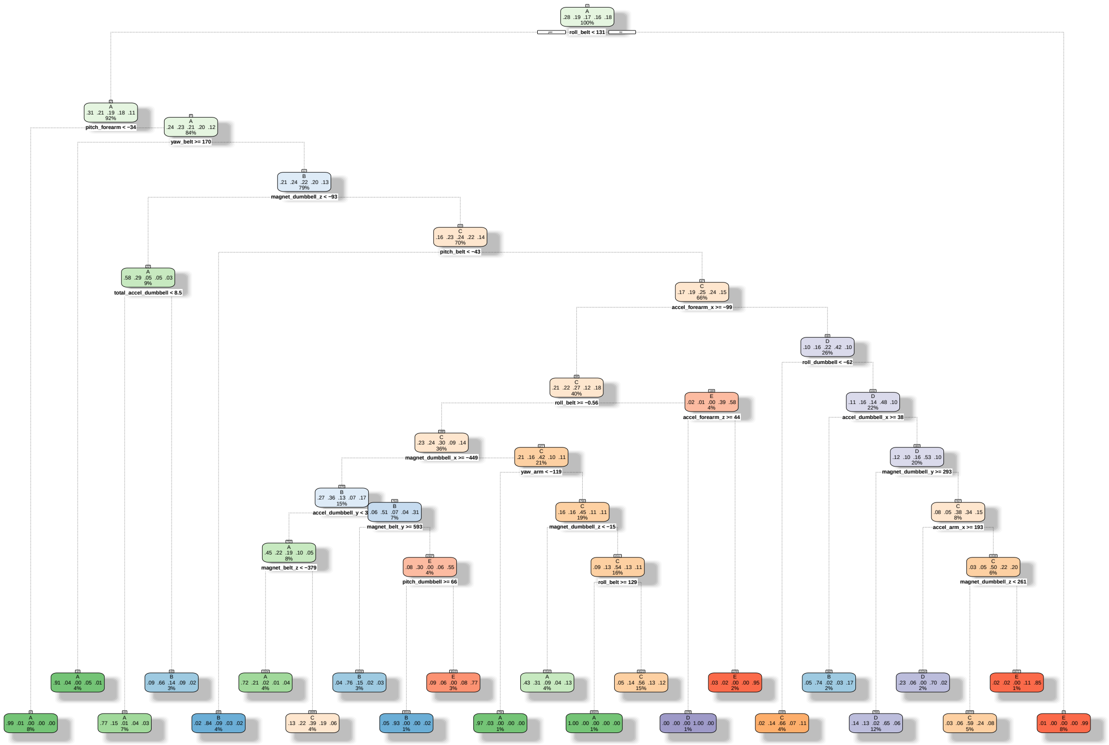
```
## [1] 11776    53
```

```
## [1] 7846   53
```

Decision Tree Model

```
# Building
modFitDT <- rpart(classe ~ ., data = training, method="class")
fancyRpartPlot(modFitDT)
```

```
## Warning: labs do not fit even at cex 0.15, there may be some overplotting
```



Rattle 2018−Sep−16 15:32:03 Jae

```
# Prediction
set.seed(12345)
```

```
prediction <- predict(modFitDT, testing, type = "class")
confusionMatrix(prediction, testing$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1879  260   30   69   66
##          B   56  759   88   34   54
##          C  105  340 1226  354  234
##          D  155  132   23  807   57
##          E   37   27    1   22 1031
##
## Overall Statistics
##
##                Accuracy : 0.7267
##                  95% CI : (0.7167, 0.7366)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6546
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8418  0.50000   0.8962   0.6275   0.7150
## Specificity            0.9243  0.96334   0.8405   0.9441   0.9864
## Pos Pred Value         0.8155  0.76589   0.5427   0.6874   0.9222
## Neg Pred Value         0.9363  0.88928   0.9746   0.9282   0.9389
## Prevalence             0.2845  0.19347   0.1744   0.1639   0.1838
## Detection Rate         0.2395  0.09674   0.1563   0.1029   0.1314
## Detection Prevalence   0.2937  0.12631   0.2879   0.1496   0.1425
## Balanced Accuracy      0.8831  0.73167   0.8684   0.7858   0.8507
```

Random Forest Model

```
# Building
set.seed(12345)
modFitRF <- randomForest(classe ~ ., data = training, ntree = 1000)

# Prediction
prediction <- predict(modFitRF, testing, type = "class")
confusionMatrix(prediction, testing$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2230    9    0    0    0
##          B    2 1505    7    0    0
##          C    0    4 1361   16    2
##          D    0    0    0 1268    4
##          E    0    0    0    2 1436
##
## Overall Statistics
##
##                Accuracy : 0.9941
##                  95% CI : (0.9922, 0.9957)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9926
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9991   0.9914   0.9949   0.9860   0.9958
## Specificity            0.9984   0.9986   0.9966   0.9994   0.9997
## Pos Pred Value         0.9960   0.9941   0.9841   0.9969   0.9986
## Neg Pred Value         0.9996   0.9979   0.9989   0.9973   0.9991
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2842   0.1918   0.1735   0.1616   0.1830
## Detection Prevalence   0.2854   0.1930   0.1763   0.1621   0.1833
## Balanced Accuracy      0.9988   0.9950   0.9957   0.9927   0.9978
```

Prediction using Testing Data

```r
# Decision Tree Model
predictionDT <- predict(modFitDT, dt_testing, type = "class")
predictionDT
```

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

```r
# Random Forest Model
predictionRF <- predict(modFitRF, dt_testing, type = "class")
predictionRF
```

```
##  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
```