Practical Machine Learning

Sujitha P

December 20, 2018

## Introduction

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).

## Source Of Data

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://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.

## Data Processing

library(caret)

## Loading required package: lattice

## Loading required package: ggplot2

library(rpart)  
library(rpart.plot)  
library(RColorBrewer)  
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.

library(randomForest)

## 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

# Load the training dataset  
dt\_training <- read.csv("pml-training.csv", na.strings=c("NA",""), strip.white=T)  
  
# Load the testing dataset  
dt\_testing <- read.csv("pml-testing.csv", na.strings=c("NA",""), strip.white=T)

## Data Cleaning

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)

## [1] 19622 53

dim(dt\_testing)

## [1] 20 53

## Partitioning the Dataset

set.seed(1234567)  
  
inTrain <- createDataPartition(dt\_training$classe, p=0.6, list=FALSE)  
training <- dt\_training[inTrain,]  
testing <- dt\_training[-inTrain,]  
  
dim(training)

## [1] 11776 53

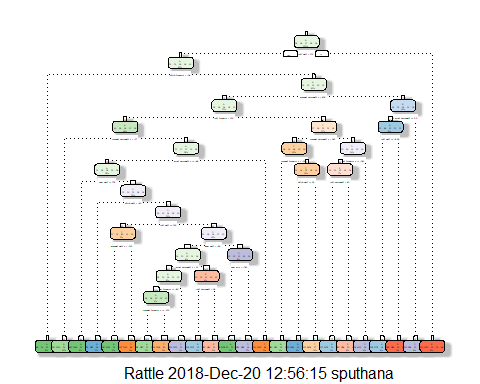
dim(testing)

## [1] 7846 53

## Decision Tree Model

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

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



## Predicting with the Decision Tree Model

set.seed(1234567)  
prediction <- predict(modFitDT, testing, type = "class")  
confusionMatrix(prediction, testing$class)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 2038 312 24 117 47  
## B 60 842 83 42 98  
## C 67 156 1105 193 181  
## D 30 105 76 800 76  
## E 37 103 80 134 1040  
##   
## Overall Statistics  
##   
## Accuracy : 0.7424   
## 95% CI : (0.7326, 0.7521)  
## No Information Rate : 0.2845   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.6727   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Statistics by Class:  
##   
## Class: A Class: B Class: C Class: D Class: E  
## Sensitivity 0.9131 0.5547 0.8077 0.6221 0.7212  
## Specificity 0.9109 0.9553 0.9078 0.9563 0.9447  
## Pos Pred Value 0.8030 0.7484 0.6492 0.7360 0.7461  
## Neg Pred Value 0.9635 0.8994 0.9572 0.9281 0.9377  
## Prevalence 0.2845 0.1935 0.1744 0.1639 0.1838  
## Detection Rate 0.2598 0.1073 0.1408 0.1020 0.1326  
## Detection Prevalence 0.3235 0.1434 0.2169 0.1385 0.1777  
## Balanced Accuracy 0.9120 0.7550 0.8578 0.7892 0.8330

## Building the Random Forest Model

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

## Predicting on the Testing Data

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   
## B A E A A C D A A A C E C A E E A A A B   
## Levels: A B C D E

## Random Forest Prediction

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

## Conclusion

Accury is 99% for the test cases from the matrix of Random Forest Model.