Prediction Assignment Writeup

Adrián

1/9/2020

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).

```r
#We load the training set and test set
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
dataset <- read_csv("pml-training.csv")
```

```
## Warning: Missing column names filled in: 'X1' [1]
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   user_name = col_character(),
##   cvtd_timestamp = col_character(),
##   new_window = col_character(),
##   kurtosis_roll_belt = col_character(),
```

```
##   kurtosis_picth_belt = col_character(),
##   kurtosis_yaw_belt = col_character(),
##   skewness_roll_belt = col_character(),
##   skewness_roll_belt.1 = col_character(),
##   skewness_yaw_belt = col_character(),
##   max_yaw_belt = col_character(),
##   min_yaw_belt = col_character(),
##   amplitude_yaw_belt = col_character(),
##   kurtosis_picth_arm = col_character(),
##   kurtosis_yaw_arm = col_character(),
##   skewness_pitch_arm = col_character(),
##   skewness_yaw_arm = col_character(),
##   kurtosis_yaw_dumbbell = col_character(),
##   skewness_yaw_dumbbell = col_character(),
##   kurtosis_roll_forearm = col_character(),
##   kurtosis_picth_forearm = col_character()
##   # ... with 8 more columns
## )

## See spec(...) for full column specifications.

## Warning: 182 parsing failures.
##  row               col expected  actual               file
## 2231 kurtosis_roll_arm a double #DIV/0! 'pml-training.csv'
## 2231 skewness_roll_arm a double #DIV/0! 'pml-training.csv'
## 2255 kurtosis_roll_arm a double #DIV/0! 'pml-training.csv'
## 2255 skewness_roll_arm a double #DIV/0! 'pml-training.csv'
## 2282 kurtosis_roll_arm a double #DIV/0! 'pml-training.csv'
## .... ................. ........ ....... ..................
## See problems(...) for more details.
```

```
testdata <- read_csv("pml-testing.csv")
```

```
## Warning: Missing column names filled in: 'X1' [1]

## Parsed with column specification:
## cols(
##   .default = col_logical(),
##   X1 = col_double(),
##   user_name = col_character(),
##   raw_timestamp_part_1 = col_double(),
##   raw_timestamp_part_2 = col_double(),
##   cvtd_timestamp = col_character(),
##   new_window = col_character(),
##   num_window = col_double(),
##   roll_belt = col_double(),
##   pitch_belt = col_double(),
##   yaw_belt = col_double(),
##   total_accel_belt = col_double(),
##   gyros_belt_x = col_double(),
##   gyros_belt_y = col_double(),
##   gyros_belt_z = col_double(),
##   accel_belt_x = col_double(),
```

```
##   accel_belt_y = col_double(),
##   accel_belt_z = col_double(),
##   magnet_belt_x = col_double(),
##   magnet_belt_y = col_double(),
##   magnet_belt_z = col_double()
##   # ... with 40 more columns
## )
## See spec(...) for full column specifications.
```

```
#We delete the variables that we do not use
comps <- complete.cases(t(dataset)) & complete.cases(t(dataset))

training_set <- dataset[,comps]
training_set <- training_set[,-c(1,3,4,5,6,7)]

testing_set <- testdata[,comps]
testing_set <- testing_set[,-c(1,3,4,5,6,7,60)]

# Convert variable to predict into factor
training_set$classe <- factor(training_set$classe)
```

The training set is divided to learn the random forest model in 70% and avoid overfitting

```
# Split data into training set
library(caTools)
set.seed(123)
split <- sample.split(training_set$classe, SplitRatio = 0.70)
training_set_1 <- subset(training_set, split == TRUE)
training_set_2 <- subset(training_set, split == FALSE)
```

```
# Adjust the Random Forest with the training set with 10 trees.
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:dplyr':
##
##     combine

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

```
classifier <- randomForest(x = training_set_1[,2:53],
                           y = training_set_1$classe,
                           ntree = 10)
```

```r
# Prediction of the results with the training set
y_pred <- predict(classifier, newdata = training_set_2[,-54],
                  type = "class")
```

```r
# Create the confusion matrix
cm <- table(training_set_2$classe, y_pred)
porcentaje <- ((cm[1,1] + cm[2,2] + cm[3,3] + cm[4,4] + cm[5,5]) / sum(cm)) * 100
porcentaje
```

```
## [1] 98.62409
```

It is observed that the model had an accuracy of just over 98%

Prediction of test data

```r
# Predicting the results with the test set
y_pred_2 <- predict(classifier, newdata = testing_set,
                  type = "class")
print("Classifications on the test set:"); y_pred_2
```

```
## [1] "Classifications on the test set:"
```

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

Conclusion

It can be concluded that because it is a very powerful algorithm and that it learned almost 100%, it is assumed that the new results will be effective in its classifications.