Regression models - course project

Jerez

3/23/2021

Practical Machine Learning

Background

from coursera project

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 (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
- 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.

Project goal

The goal of your project is to predict the manner in which they did the exercise. This is the "classe" variable in the training set. You may use any of the other variables to predict with. You should create a report describing how you built your model, how you used cross validation, what you think the expected out of sample error is, and why you made the choices you did. You will also use your prediction model to predict 20 different test cases.

How to do

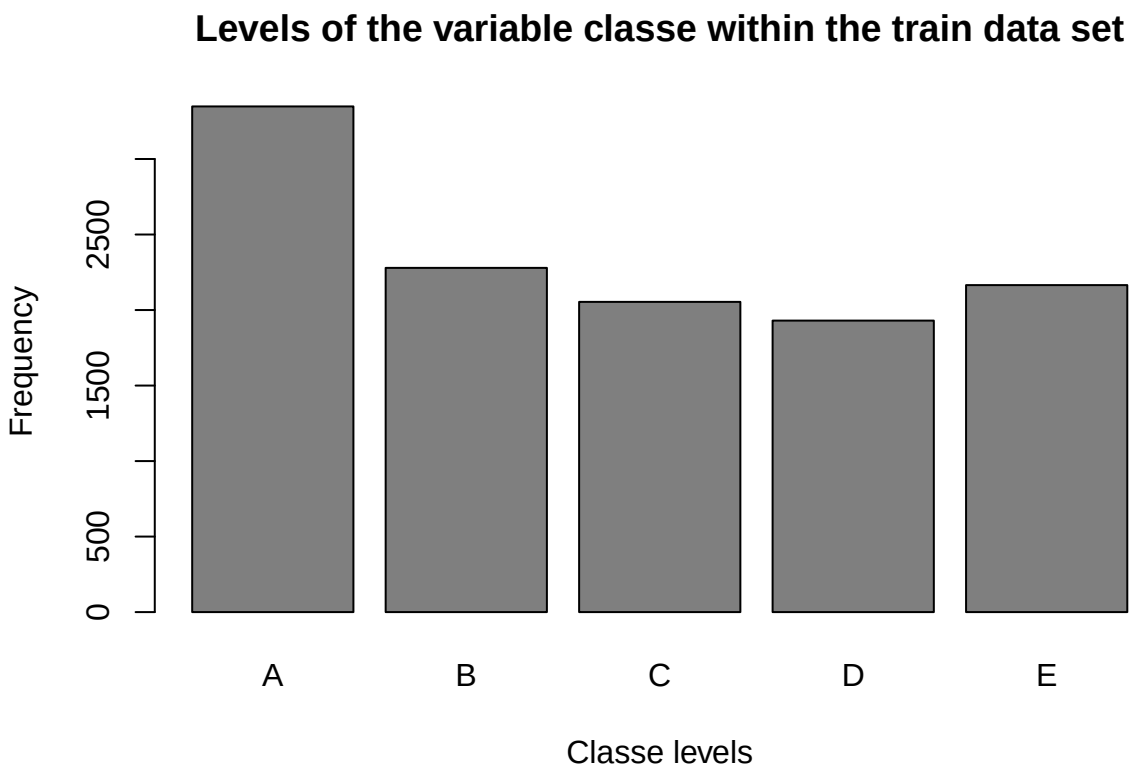
Getting the data

```r
set.seed(3)

# Library
library(caret)
library(tidyverse)
library(randomForest)
library(rpart)
library(rpart.plot)


# Url's

url.train <-"https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
url.test <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"

# Preprocessing and cleaning

# Remove redundant variables
training <- read.csv(url(url.train), na.strings=c("NA","#DIV/0!",""))
testing  <- read.csv(url(url.test), na.strings=c("NA","#DIV/0!",""))

# Delete missing values
training <- training[,colSums(is.na(training)) == 0]
testing  <- testing[,colSums(is.na(testing)) == 0]

# Some variables are irrelevant for the project:
# user_name, raw_timestamp_part_1, raw_timestamp_part_,2 cvtd_timestamp,
# new_window, and  num_window (columns 1 to 7).
# We can delete these variables.
training <- training[,-c(1:7)]
testing  <- testing[,-c(1:7)]

# Partioning the training set into two
# 60% for data.train, 40% for data.test

inTrain    <- createDataPartition(y=training$classe, p=0.6, list=FALSE)
data.train <- training[inTrain,  ]
data.test  <- training[-inTrain, ]

# Dimensions
dim(data.train); dim(data.test)
```

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

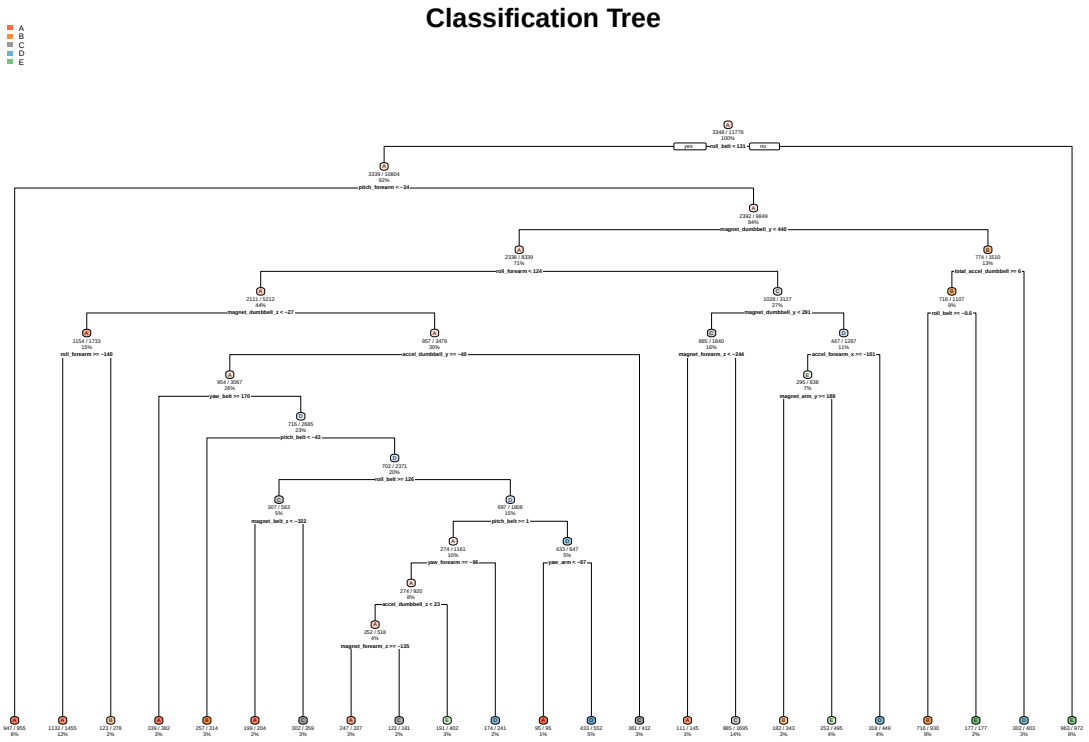
```
## [1] 7846   53
```

```r
# The variable "classe" contains 5 levels: A, B, C, D and E.
# A plot of the outcome variable will allow us to see the frequency of each levels
# in the train data set and compare one another.

plot(data.train$classe, col="grey50", main="Levels of the variable classe within the train data set", xl
```



Prediction model I: Decision tree

```
# Model
tree.model <- rpart(classe ~ ., data = data.train, method="class")

# Predicting
tree.pred  <- predict(tree.model, data.test, type = "class")

# Plot of the Decision Tree
rpart.plot(tree.model, main="Classification Tree", extra=102, under=TRUE, faclen=0)
```




```
# Test results on the test data set:
confusionMatrix(tree.pred, data.test$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2034  232   18   97   22
##          B   59  822  119  100  120
##          C   60  200 1113  181  174
##          D   37  112   79  827   84
##          E   42  152   39   81 1042
##
## Overall Statistics
##
##                Accuracy : 0.7441
##                  95% CI : (0.7343, 0.7537)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6756
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9113   0.5415   0.8136   0.6431   0.7226
## Specificity            0.9343   0.9371   0.9051   0.9524   0.9510
## Pos Pred Value         0.8464   0.6738   0.6441   0.7261   0.7684
```

```
## Neg Pred Value         0.9636   0.8950   0.9583   0.9316   0.9384
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2592   0.1048   0.1419   0.1054   0.1328
## Detection Prevalence   0.3063   0.1555   0.2202   0.1452   0.1728
## Balanced Accuracy      0.9228   0.7393   0.8593   0.7978   0.8368
```

Prediction model II: Random Forest

```
# Model
rand.model <- randomForest(classe ~. , data = data.train, method = "class")

# Predicting
rand.pred  <- predict(rand.model, data.test, type = "class")

# Test results on the test data set:
confusionMatrix(rand.pred, data.test$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2228    7    0    0    0
##          B    1 1510   13    0    0
##          C    3    1 1353   14    1
##          D    0    0    2 1271    3
##          E    0    0    0    1 1438
##
## 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            0.9982   0.9947   0.9890   0.9883   0.9972
## Specificity            0.9988   0.9978   0.9971   0.9992   0.9998
## Pos Pred Value         0.9969   0.9908   0.9862   0.9961   0.9993
## Neg Pred Value         0.9993   0.9987   0.9977   0.9977   0.9994
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2840   0.1925   0.1724   0.1620   0.1833
## Detection Prevalence   0.2849   0.1942   0.1749   0.1626   0.1834
## Balanced Accuracy      0.9985   0.9963   0.9931   0.9938   0.9985
```

Random Forests yielded better Results.

Submission

```
# predict outcome levels on the original Testing data set using Random Forest algorithm
predict.final <- predict(rand.model, testing, type="class")
predict.final
```

```
##  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
```

```
# Files

# Function to generate files with predictions to submit for assignment
pml_write_files = function(x){
  n = length(x)
  for(i in 1:n){
    filename = paste0("problem_id_",i,".txt")
    write.table(x[i],file=filename,quote=FALSE,row.names=FALSE,col.names=FALSE)
  }
}

pml_write_files(predict.final)
```