Machine Learning

Javier de la Vega

22 de agosto de 2015

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)

The data for this project come from this source: http://groupware.les.inf.puc-rio.br/har (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.

What you should submit

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.

1. Your submission should consist of a link to a Github repo with your R markdown and compiled HTML file describing your analysis. Please constrain the text of the writeup to < 2000 words and the number of figures to be less than 5. It will make it easier for the graders if you submit a repo with a gh-pages branch so the HTML page can be viewed online (and you always want to make it easy on graders :-).

2. You should also apply your machine learning algorithm to the 20 test cases available in the test data above. Please submit your predictions in appropriate format to the programming assignment for automated grading. See the programming assignment for additional details.

Report

Librerías utilizadas

```
library(ggplot2)
library(lattice)
library(caret)

library(randomForest)
```

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

```
library(rpart)
library(kernlab)
```

```
## Warning: package 'kernlab' was built under R version 3.2.2
```

```
## Warning in .recacheSubclasses(def@className, def, doSubclasses, env):
## undefined subclass "externalRefMethod" of class "kfunction"; definition not
## updated
```

```
training = read.csv("pml-training.csv", na.strings = c("NA","", "#DIV/0!" ), header = T)
testing = read.csv("pml-testing.csv", na.strings = c("NA","", "#DIV/0!" ), header = T)
```

The number of null values that has the file is verified

```
table(is.na(training))
```

```
##
##   FALSE    TRUE
## 1214418 1925102
```

```
table(is.na(testing))
```

```
##
## FALSE  TRUE
##  1200  2000
```

Because records have too many null values that do not help the model to predict accurately, proceed to clean the columns with null values and the first 7 columns, which are not required for the process is also removed.

```
countOfNA<-sapply(training, function(x) sum(is.na(x)))
training <- training[,which(countOfNA == 0)]
training = training[,-c(1:7)]
countOfNA<-sapply(testing, function(x) sum(is.na(x)))
testing <- testing[,which(countOfNA == 0)]
testing = testing[,-c(1:7)]
```

Subsets are created from training set and having distributions in both subsets classes are presented.

```
inTrain = createDataPartition(y=training$classe, p=0.7, list=FALSE)
subTraining = training[inTrain, ]; subTesting = training[-inTrain,]
dim(subTraining); dim(subTesting)
```

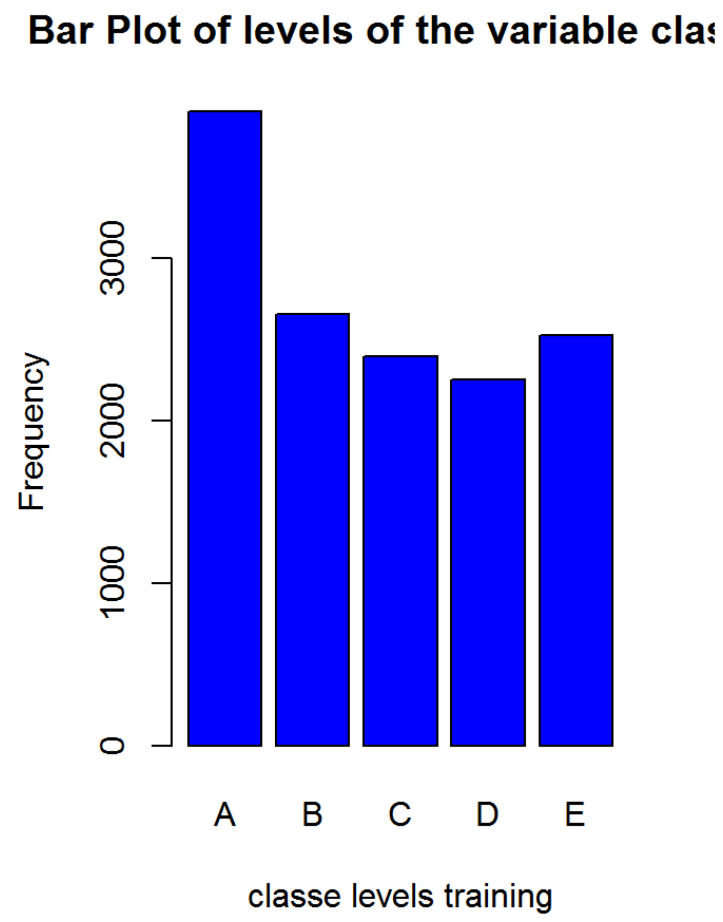
```
## [1] 13737    53
```

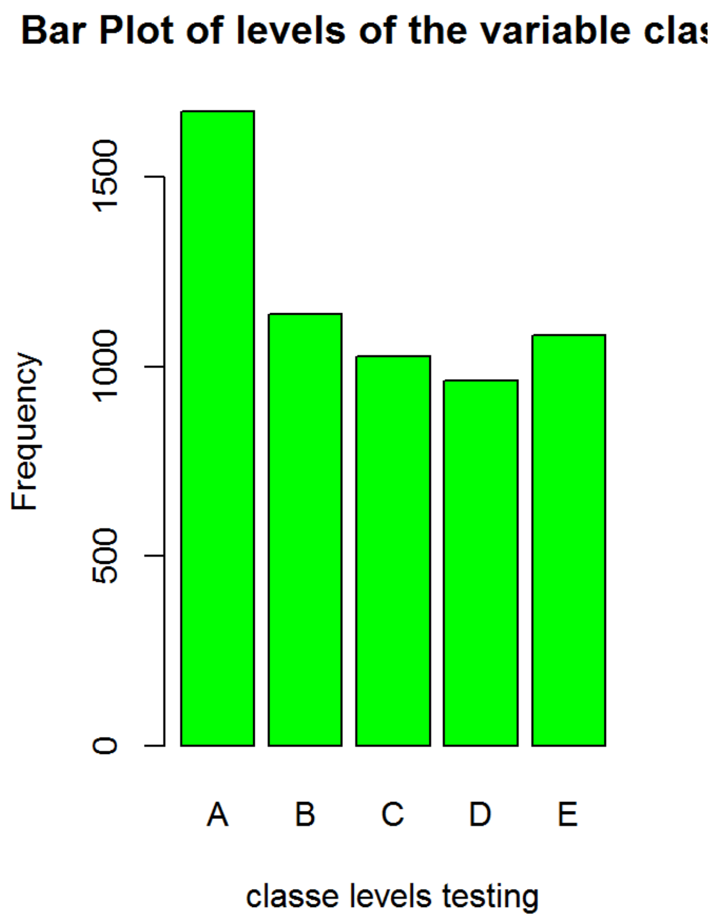
```
## [1] 5885   53
```

```
par(mfrow = c(1, 2))

plot(subTraining$classe, col="blue", main="Bar Plot of levels of the variable classe",
     xlab="classe levels training", ylab="Frequency")

plot(subTesting$classe, col="green", main="Bar Plot of levels of the variable classe",
     xlab="classe levels testing", ylab="Frequency")
```





```
par(mfrow = c(1, 1))
```

3 different models are implemented to select the model with better accuracy.

```
################################################################################

ptm = proc.time()

tc <- trainControl(method = "cv", number = 7, verboseIter=FALSE , preProcOptions="pca", a
llowParallel=TRUE)

# Random Forest
model_rf = train(classe ~., method="rf", data=subTraining, trControl = tc)
print(model_rf)
```

```
## Random Forest
##
## 13737 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (7 fold)
##
## Summary of sample sizes: 11773, 11775, 11775, 11775, 11773, 11776, ...
##
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa      Accuracy SD  Kappa SD
##    2    0.9916275  0.9894080  0.002852635  0.003609123
##   27    0.9904631  0.9879346  0.003083359  0.003902106
##   52    0.9832560  0.9788171  0.003636014  0.004602893
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 2.
```

```
max(model_rf$results$Accuracy)
```

```
## [1] 0.9916275
```

```
# SVM Linear
model_svm = train(classe ~., method="svmLinear", data=subTraining, trControl = tc)
print(model_svm)
```

```
## Support Vector Machines with Linear Kernel
##
## 13737 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (7 fold)
##
## Summary of sample sizes: 11775, 11775, 11775, 11775, 11773, 11775, ...
##
## Resampling results
##
##   Accuracy   Kappa      Accuracy SD  Kappa SD
##   0.7792824  0.7193979  0.00960305   0.0122804
##
## Tuning parameter 'C' was held constant at a value of 1
##
```

```
max(model_svm$results$Accuracy)
```

```
## [1] 0.7792824
```

```
# Decision tree
model_rpart = train(classe ~., method="rpart", data=subTraining, trControl = tc)
print(model_rpart)
```

```
## CART
##
## 13737 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (7 fold)
##
## Summary of sample sizes: 11775, 11775, 11774, 11774, 11774, 11775, ...
##
## Resampling results across tuning parameters:
##
##   cp          Accuracy   Kappa       Accuracy SD  Kappa SD
##   0.03712745  0.5037481  0.35118364  0.02276950   0.03023063
##   0.06011596  0.4380930  0.24627528  0.06070330   0.10315612
##   0.11616316  0.3411960  0.08705215  0.03892263   0.05959275
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was cp = 0.03712745.
```

```
max(model_rpart$results$Accuracy)
```

```
## [1] 0.5037481
```

```
proc.time() - ptm
```

```
##    user  system elapsed
## 1690.04    8.08 1716.68
```

Because the Random Forest model is the one with better accuracy, it is the one that is selected to predict the results with test subsets.

First is predicted on the subset of Training subsequently are predicted on the subset of testing finally is predicted over the original file of testing and obtain the results on the 20 values.

```
pred_train = predict(model_rf, subTraining)
table(pred_train, subTraining$classe)
```

```
##
## pred_train    A    B    C    D    E
##          A 3906    0    0    0    0
##          B    0 2658    0    0    0
##          C    0    0 2396    0    0
##          D    0    0    0 2252    0
##          E    0    0    0    0 2525
```

```
pred_test = predict(model_rf, subTesting)
table(pred_test, subTesting$classe)
```

```
##
## pred_test    A    B    C    D    E
##         A 1674    7    0    0    0
##         B    0 1132    3    0    0
##         C    0    0 1021   19    0
##         D    0    0    2  941    3
##         E    0    0    0    4 1079
```

```
pred_fin = predict(model_rf, testing, type = "raw")
pred_fin
```

```
##  [1] 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
```

```
proc.time() - ptm
```

```
##    user  system elapsed
## 1691.88    8.09 1718.54
```