Machine learning Prediction Assignment Writeup

Nicolas Gonzalez Rojas

25/11/2020

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 (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.

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.

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 :-). 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.

Preliminary Work

Reproduceability

An overall pseudo-random number generator seed was set at 1234 for all code. In order to reproduce the results below, the same seed should be used. Different packages were downloaded and installed, such as caret and randomForest. These should also be installed in order to reproduce the results below (please see code below for ways and syntax to do so).

Cross-validation

Cross-validation will be performed by subsampling our training data set randomly without replacement into 2 subsamples: subTraining data (75% of the original Training data set) and subTesting data (25%). Our models will be fitted on the subTraining data set, and tested on the subTesting data. Once the most accurate model is choosen, it will be tested on the original Testing data set.

Expected out-of-sample error

The expected out-of-sample error will correspond to the quantity: 1-accuracy in the cross-validation data. Accuracy is the proportion of correct classified observation over the total sample in the subTesting data set. Expected accuracy is the expected accuracy in the out-of-sample data set (i.e. original testing data set). Thus, the expected value of the out-of-sample error will correspond to the expected number of missclassified observations/total observations in the Test data set, which is the quantity: 1-accuracy found from the cross-validation data set.

Our outcome variable "classe" is an unordered factor variable. Thus, we can choose our error type as 1-accuracy. We have a large sample size with N= 19622 in the Training data set. This allow us to divide our Training sample into subTraining and subTesting to allow cross-validation. Features with all missing values will be discarded as well as features that are irrelevant. All other features will be kept as relevant variables. Decision tree and random forest algorithms are known for their ability of detecting the features that are important for classification [2].

##Packages, Libraries and Seed Installing packages, loading libraries, and setting the seed for reproduceability:

```
library(caret); library(randomForest); library(rpart); library(RColorBrewer); library(rattle)
```

```
## Loading required package: lattice

## Loading required package: ggplot2

## randomForest 4.6-14

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

##
## Attaching package: 'randomForest'

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

```
## Loading required package: tibble
```

```
## Loading required package: bitops
```

```
## Rattle: A free graphical interface for data science with R.
## Versión 5.4.0 Copyright (c) 2006-2020 Togaware Pty Ltd.
## Escriba 'rattle()' para agitar, sacudir y  rotar sus datos.
```

```
##
## Attaching package: 'rattle'
```

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

Getting and cleaning data

```r
trUrl <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
teUrl <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
training <- read.csv(url(trUrl), na.strings=c("NA","#DIV/0!",""))
testing <- read.csv(url(teUrl), na.strings=c("NA","#DIV/0!",""))
training <- training[,colSums(is.na(training)) == 0]
testing <- testing[,colSums(is.na(testing)) == 0]
training <- training[,-c(1:7)]
testing <- testing[,-c(1:7)]
```

Partioning the training set into two

Partioning Training data set into two data sets, 75% for myTraining, 25% for myTesting:

```r
set.seed(1234)
inTrain <- createDataPartition(y=training$classe, p=0.75, list=FALSE)
mytraining <- training[inTrain, ]
mytesting <- training[-inTrain, ]
```

A look at the Data

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 subTraining data set and compare one another.

```r
plot(mytraining$classe, col="lightgreen", main = "Plot of levels of variable classe within the TrainTrai
```

Plot of levels of variable classe within the TrainTrainingSet data set
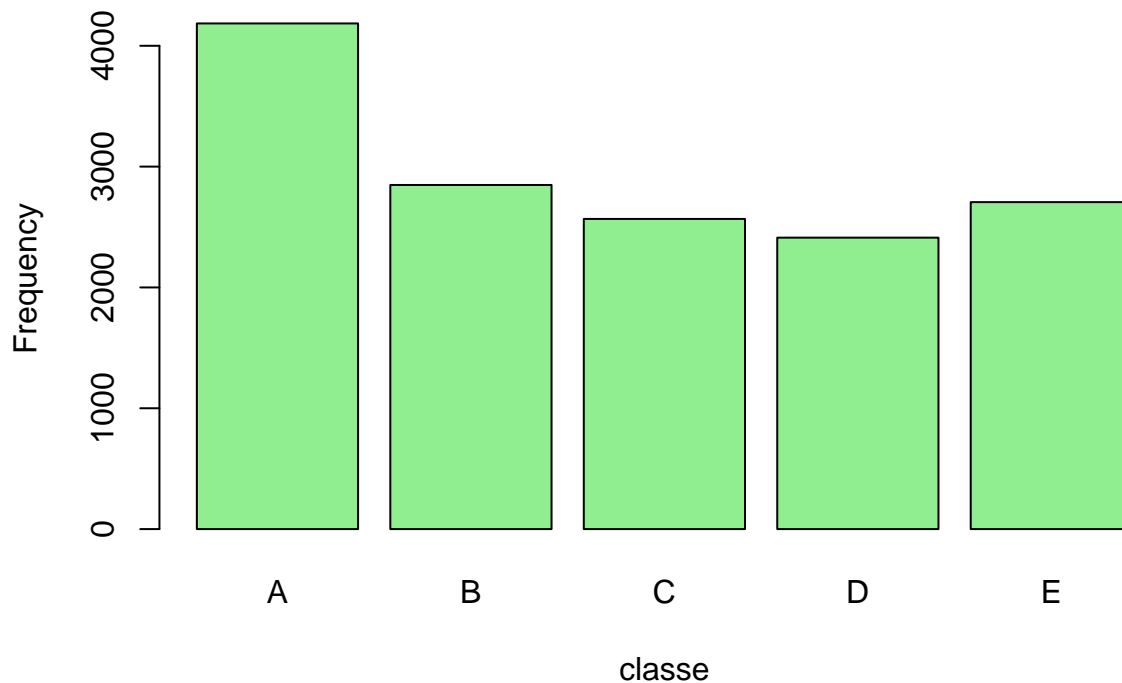

From the graph above, we can see that each level frequency is within the same order of magnitude of each other. Level A is the most frequent with more than 4000 occurrences while level D is the least frequent with about 2500 occurrences.

First prediction model: Using Decision Tree

```
model <- rpart(classe ~ ., data=mytraining, method="class")
prediction1 <- predict(model, mytesting, type = "class")

fancyRpartPlot(model)
```

```
## Warning: labs do not fit even at cex 0.15, there may be some overplotting
```
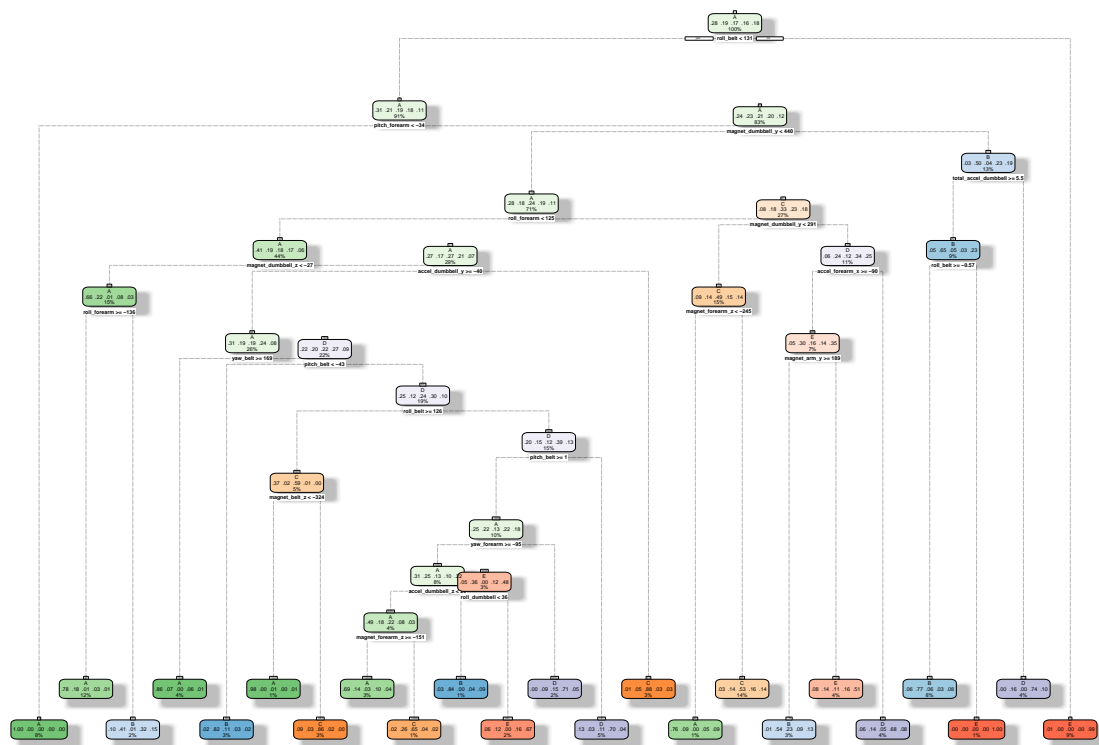
Rattle 2020–nov.–25 15:59:48 EQUIPO

```
confusionMatrix(prediction1, mytesting$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1251  149   15   61   17
##          B   38  572   75   60   75
##          C   39  117  696  117  122
##          D   49   58   51  508   58
##          E   18   53   18   58  629
##
## Overall Statistics
##
##                Accuracy : 0.7455
##                  95% CI : (0.7331, 0.7577)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6774
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
```

```
##                  Class: A Class: B Class: C Class: D Class: E
## Sensitivity         0.8968   0.6027   0.8140   0.6318   0.6981
## Specificity         0.9310   0.9373   0.9024   0.9473   0.9633
## Pos Pred Value      0.8379   0.6976   0.6379   0.7017   0.8106
## Neg Pred Value      0.9578   0.9077   0.9583   0.9292   0.9341
## Prevalence          0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate      0.2551   0.1166   0.1419   0.1036   0.1283
## Detection Prevalence 0.3044  0.1672   0.2225   0.1476   0.1582
## Balanced Accuracy   0.9139   0.7700   0.8582   0.7896   0.8307
```

Prediction model 2: Random Forest

```r
model2 <- randomForest(classe ~. , data=mytraining, method="class")
prediction2 <- predict(model2, mytesting, type = "class")
confusionMatrix(prediction2, mytesting$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1395    4    0    0    0
##          B    0  944    8    0    0
##          C    0    1  847    6    0
##          D    0    0    0  798    1
##          E    0    0    0    0  900
##
## Overall Statistics
##
##                Accuracy : 0.9959
##                  95% CI : (0.9937, 0.9975)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9948
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                  Class: A Class: B Class: C Class: D Class: E
## Sensitivity        1.0000   0.9947   0.9906   0.9925   0.9989
## Specificity        0.9989   0.9980   0.9983   0.9998   1.0000
## Pos Pred Value     0.9971   0.9916   0.9918   0.9987   1.0000
## Neg Pred Value     1.0000   0.9987   0.9980   0.9985   0.9998
## Prevalence         0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate     0.2845   0.1925   0.1727   0.1627   0.1835
## Detection Prevalence 0.2853 0.1941   0.1741   0.1629   0.1835
## Balanced Accuracy  0.9994   0.9964   0.9945   0.9961   0.9994
```

Decision

As expected, Random Forest algorithm performed better than Decision Trees. Accuracy for Random Forest model was 0.995 (95% CI: (0.993, 0.997)) compared to 0.739 (95% CI: (0.727, 0.752)) for Decision Tree model. The random Forest model is choosen. The accuracy of the model is 0.995. The expected out-of-sample error is estimated at 0.005, or 0.5%. The expected out-of-sample error is calculated as 1 - accuracy for predictions made against the cross-validation set. Our Test data set comprises 20 cases. With an accuracy above 99% on our cross-validation data, we can expect that very few, or none, of the test samples will be missclassified.

Submission

```r
finalpredict <- predict(model2, testing, type="class")
finalpredict
```

```
##  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
```

```r
# Write files for submission

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(finalpredict)
```