Course Project

Omar Juarez

3 de febrero de 2020

## Abstract

In this document we are going to analyse the data provided by Human Activity Recognition. Our goal is to determine if the exercises of the participants in this test were done properly. In this particular case what we want to predict the “classe” variable based on the regressors. This variable can take 5 labels(from A to E) so we will compare the results of two methods that we have learned in this course: Decision Trees and Random Forest and the we will select the best model according the accuracy of each one.

## Data Loading

library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

library(lattice)  
library(ggplot2)  
library(caret)  
library(kernlab)

##   
## Attaching package: 'kernlab'

## The following object is masked from 'package:ggplot2':  
##   
## alpha

library(rpart)  
library(rpart.plot)  
library(corrplot)

## corrplot 0.84 loaded

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:ggplot2':  
##   
## margin

## The following object is masked from 'package:dplyr':  
##   
## combine

library(e1071)  
library(rattle)

## Rattle: A free graphical interface for data science with R.  
## Versión 5.3.0 Copyright (c) 2006-2018 Togaware Pty Ltd.  
## Escriba 'rattle()' para agitar, sacudir y rotar sus datos.

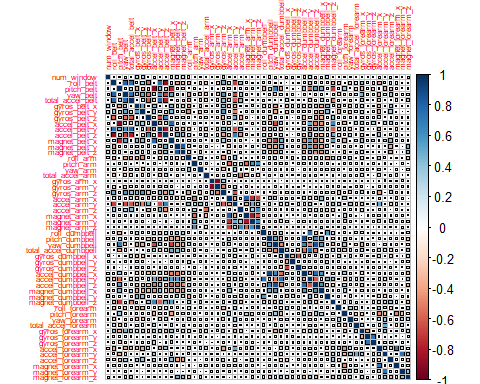
##   
## Attaching package: 'rattle'

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

set.seed(12345)  
  
setwd("D:/02 Coursera/02 R/01 Johns Hopkings-Coursera/08 Machine Learning/Data")  
  
trainUrl = "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"  
  
testUrl = "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"  
  
TrainFile = "./data/pml-training.csv"  
  
TestFile = "./data/pml-testing.csv"  
  
if(!file.exists("./data")){  
 dir.create(path = "./data")  
}  
  
if(!file.exists(TrainFile)){  
 download.file(trainUrl,destfile = TrainFile, method = "curl")  
}  
  
if(!file.exists(TestFile)){  
 download.file(testUrl,destfile = TestFile,method = "curl")  
}  
  
Flag\_Training = read.csv(TrainFile)  
Flag\_Test = read.csv(TestFile)

## Cleaning data

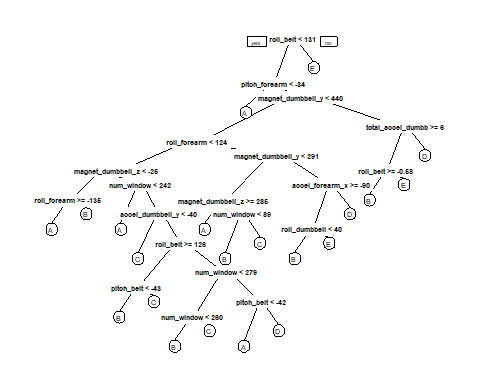
1. We can remove variables that do not change between observations. In this case, since the variables are not binary, we can remove them without biased our model.
2. We can also remove the variables that contain missing values (i.e NA’s)

We can create a Correlation Matrix to have hint about which variables might influence others. 

## Creating Data Partition

Once we have done the cleaning and an overlook about the correlations, we now can split the data into the training set and the testing (or validation) set. Just like the course we are going to split the data into 70% for the training and 30% for the validation dataset. This partition has to be done with the trainig dataset not with test daset otherwise our prediction would be biased.

## Decision Tree: Analysis

As we can see in the results this kind of analysis give us an accuracy of 73.42% with a confidence interval between 72.28% and 74.55% and the out of sample error is 32.32%. The sensitivity and specificity values are high so we can say that the false positive and false negative are undercontrol. But perhaps this could be improved by using an slightly different approach. 

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 1502 58 4 90 20  
## B 201 660 66 148 64  
## C 59 37 815 54 61  
## D 66 64 129 648 57  
## E 74 114 72 126 696  
##   
## Overall Statistics  
##   
## Accuracy : 0.7342   
## 95% CI : (0.7228, 0.7455)  
## No Information Rate : 0.3232   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.6625   
##   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Statistics by Class:  
##   
## Class: A Class: B Class: C Class: D Class: E  
## Sensitivity 0.7897 0.7074 0.7505 0.6079 0.7751  
## Specificity 0.9568 0.9033 0.9560 0.9344 0.9226  
## Pos Pred Value 0.8973 0.5795 0.7943 0.6722 0.6433  
## Neg Pred Value 0.9050 0.9425 0.9442 0.9151 0.9579  
## Prevalence 0.3232 0.1585 0.1845 0.1811 0.1526  
## Detection Rate 0.2552 0.1121 0.1385 0.1101 0.1183  
## Detection Prevalence 0.2845 0.1935 0.1743 0.1638 0.1839  
## Balanced Accuracy 0.8733 0.8053 0.8532 0.7712 0.8488

## Ramdom Forest

In this case, using 5 folds, we can see that the results improved by using this approach. The accuracy of the model has risen from 73.42% to nearly 1 (i.e 0.99%) while the out of sample error also improved to almost 0(0.0084%). So in general terms this model is way better than just the decision tree. We have to test those results with the validation dataset

## Random Forest   
##   
## 13737 samples  
## 54 predictor  
## 5 classes: 'A', 'B', 'C', 'D', 'E'   
##   
## No pre-processing  
## Resampling: Cross-Validated (5 fold)   
## Summary of sample sizes: 10988, 10990, 10990, 10990, 10990   
## Resampling results across tuning parameters:  
##   
## mtry Accuracy Kappa   
## 2 0.9925746 0.9906063  
## 30 0.9970880 0.9963167  
## 58 0.9934484 0.9917126  
##   
## Accuracy was used to select the optimal model using the largest value.  
## The final value used for the model was mtry = 30.

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 1674 0 0 0 0  
## B 1 1138 0 0 0  
## C 0 1 1025 0 0  
## D 0 0 2 962 0  
## E 0 0 0 1 1081  
##   
## Overall Statistics  
##   
## Accuracy : 0.9992   
## 95% CI : (0.998, 0.9997)  
## No Information Rate : 0.2846   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.9989   
##   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: A Class: B Class: C Class: D Class: E  
## Sensitivity 0.9994 0.9991 0.9981 0.9990 1.0000  
## Specificity 1.0000 0.9998 0.9998 0.9996 0.9998  
## Pos Pred Value 1.0000 0.9991 0.9990 0.9979 0.9991  
## Neg Pred Value 0.9998 0.9998 0.9996 0.9998 1.0000  
## Prevalence 0.2846 0.1935 0.1745 0.1636 0.1837  
## Detection Rate 0.2845 0.1934 0.1742 0.1635 0.1837  
## Detection Prevalence 0.2845 0.1935 0.1743 0.1638 0.1839  
## Balanced Accuracy 0.9997 0.9995 0.9989 0.9993 0.9999

## [1] 0.0008496177

## Conclusions

If we use random forest the accuracy increases a lot, so based on the results this is the model we can use. However we could do so more analysis via Area Under the Receiver Operating Characteristics(AUROC), this will tell us the how the model is capable of distinguish between classes, but this is a more advanced topic in the data science field. Another thing to consider is to make a logistic regression and compare it’s result with the previous model.