Final Project_Prediction Assignment Writeup

Antonio Gálvez

3/12/2019

Getting and cleaning data

The information regarding to models developed in this document are available in the follow website: http://groupware.les.inf.puc-rio.br/har On the another hand, both sets of data used as a training data and test data are available from the links below: - Training data https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv - Test data https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

The first following lines download the data from the links shown above. And the last lines load the data cleaned into the memory.

```
url_train <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
url_testin <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"

download.file(url_train, file.path(getwd(), "pml-training.csv"))
download.file(url_testin, file.path(getwd(), "pml-testing.csv"))



training1 = read.csv("pml-training.csv", na.strings = c("NA", "#DIV/0!" , ""))
testing1 = read.csv("pml-testing.csv", na.strings = c("NA", "#DIV/0!" , ""))

training12 <- training1[,colSums(is.na(training1)) == 0]
testing12 <- testing1[,colSums(is.na(testing1)) == 0]
training1 <- training12[,-c(1:7)]
testing1 <- testing12[,-c(1:7)]
dim(training1); dim(testing1)
```

```
## [1] 19622    53
```

```
## [1] 20 53
```

Partitioning the training dataset

These few lines of code make the partition of the training data set into two datasets. The new trainig dataset contains a 70% of the data, and the rest is in saved unto the new testing dataset.

```
set.seed(1991)
inTrain <- caret::createDataPartition(y = training1$classe, p = 0.7, list = FALSE)
training <- training1[inTrain, ]
testing <- training1[-inTrain, ]
dim(training); dim(testing)
```

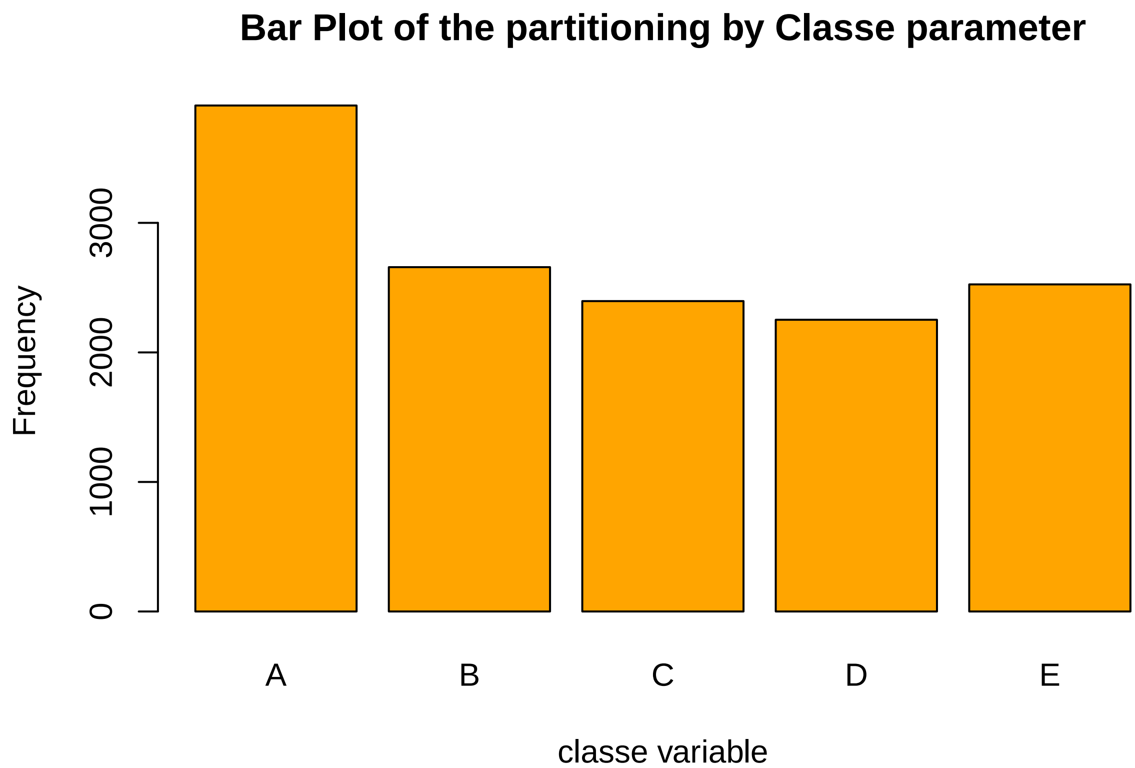
```
## [1] 13737    53
```

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

Plotting the classe

The following line plots the classe paramenter using the new dataset defined for training.

```
plot(training$classe, col = "orange", main="Bar Plot of the partitioning by Classe parameter", xlab="cla
```



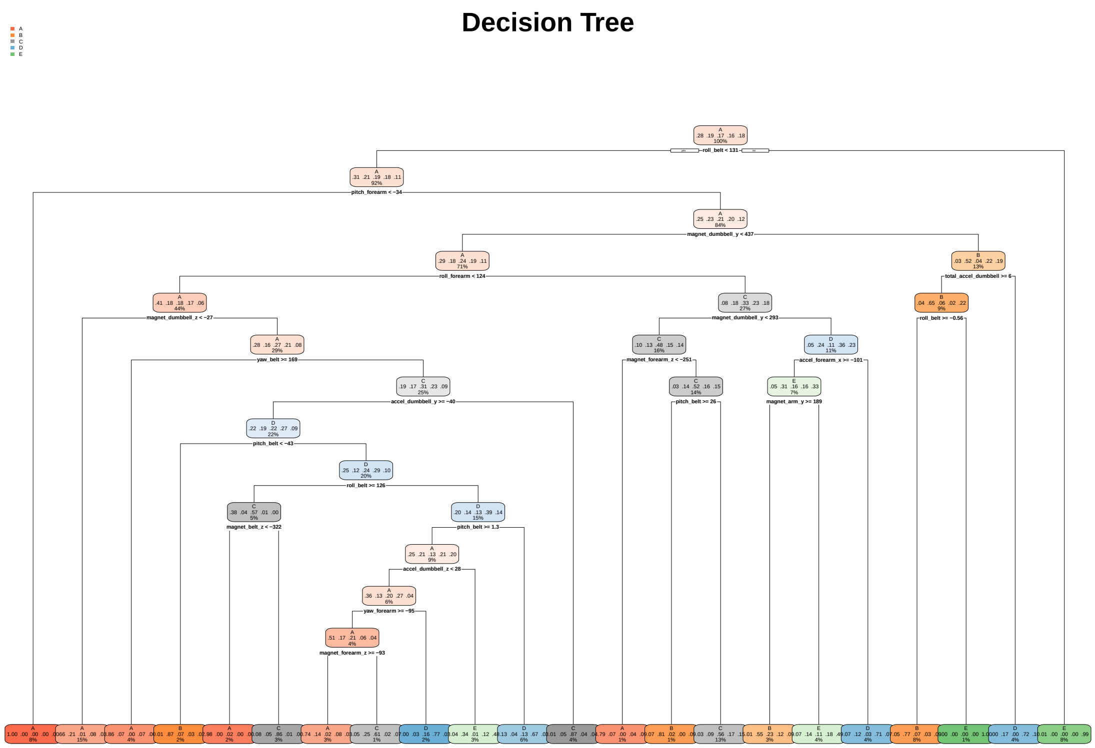
Prediction model: Decision tree

First model and its prediction.

```
Modelo1 <- rpart::rpart(classe ~ ., data= training, method="class")
pred_Mod1 <- predict(Modelo1, testing, type = "class")

rpart.plot::rpart.plot(Modelo1, main="Decision Tree", extra="auto", faclen=0)
```

```
## Warning: labs do not fit even at cex 0.15, there may be some overplotting
```




```r
caret::confusionMatrix(pred_Mod1, testing$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1501  220   17  105   34
##          B   54  606   85   26   72
##          C   38  128  840  159  143
##          D   55   71   60  612   59
##          E   26  114   24   62  774
##
## Overall Statistics
##
##                Accuracy : 0.7363
##                  95% CI : (0.7248, 0.7475)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6652
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
```

```
## Sensitivity            0.8967   0.5320   0.8187   0.6349   0.7153
## Specificity            0.9107   0.9501   0.9037   0.9502   0.9529
## Pos Pred Value         0.7997   0.7189   0.6422   0.7141   0.7740
## Neg Pred Value         0.9568   0.8943   0.9594   0.9300   0.9369
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2551   0.1030   0.1427   0.1040   0.1315
## Detection Prevalence   0.3189   0.1432   0.2223   0.1456   0.1699
## Balanced Accuracy      0.9037   0.7411   0.8612   0.7925   0.8341
```

Prediction model: Random Forest Algorithm

Second model and its prediction.

```r
Modelo2 <- randomForest::randomForest(classe ~., training, method = "class")
pred_Mod2 <- predict(Modelo2, testing, type = "class")

caret::confusionMatrix(pred_Mod2, testing$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1673    5    0    0    0
##          B    1 1130    4    0    0
##          C    0    4 1020    5    1
##          D    0    0    2  958    1
##          E    0    0    0    1 1080
##
## Overall Statistics
##
##                Accuracy : 0.9959
##                  95% CI : (0.9939, 0.9974)
##     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            0.9994   0.9921   0.9942   0.9938   0.9982
## Specificity            0.9988   0.9989   0.9979   0.9994   0.9998
## Pos Pred Value         0.9970   0.9956   0.9903   0.9969   0.9991
## Neg Pred Value         0.9998   0.9981   0.9988   0.9988   0.9996
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2843   0.1920   0.1733   0.1628   0.1835
## Detection Prevalence   0.2851   0.1929   0.1750   0.1633   0.1837
## Balanced Accuracy      0.9991   0.9955   0.9960   0.9966   0.9990
```

Comparison and decision

Random Forest algorithm achieve better results than Decision Trees. Accuracy for Random Forest model was 0.9959 (95% CI: (0.9939, 0.9974)) compared to model base on Decision Tree, it achieves an Accuracy of 0.7363 (95% CI : (0.7248, 0.7475) for Decision Tree model. For the reason mentioned abobe the random Forest model is choosen.

Submission

The following line makes a prediction using the Random Forest Model defined above and the dataset given in the problem description.

```r
predict(Modelo2, testing1, type = "class")
```

```
##  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
```