Activity Background

Parth Samant

August 24, 2020

Prediction-Assignment-Writeup

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.

#Loading Packages

```r
library(lattice)
```

```
## Warning: package 'lattice' was built under R version 3.6.3
```

```r
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.6.3
```

```r
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.6.3
```

```r
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.6.3

## 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
```

```
library(rpart)
library(rpart.plot);
```

```
## Warning: package 'rpart.plot' was built under R version 3.6.3
```

#Data Pre-Processing

```
set.seed(4096)
#Load the data
trainUrl <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
testUrl <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
```

#Cleaning data

```
#Delete missing values
training <- read.csv(url(trainUrl), na.strings=c("NA","#DIV/0!",""))
testing <- read.csv(url(testUrl), na.strings=c("NA","#DIV/0!",""))

#Remove variables with near zero variance
training<-training[,colSums(is.na(training)) == 0]
testing <-testing[,colSums(is.na(testing)) == 0]

#Remove columns that are not predictors, which are the the seven first columns
training   <-training[,-c(1:7)]
testing <-testing[,-c(1:7)]

#The data after cleaning
dim(training)
```

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

#Cross-validation

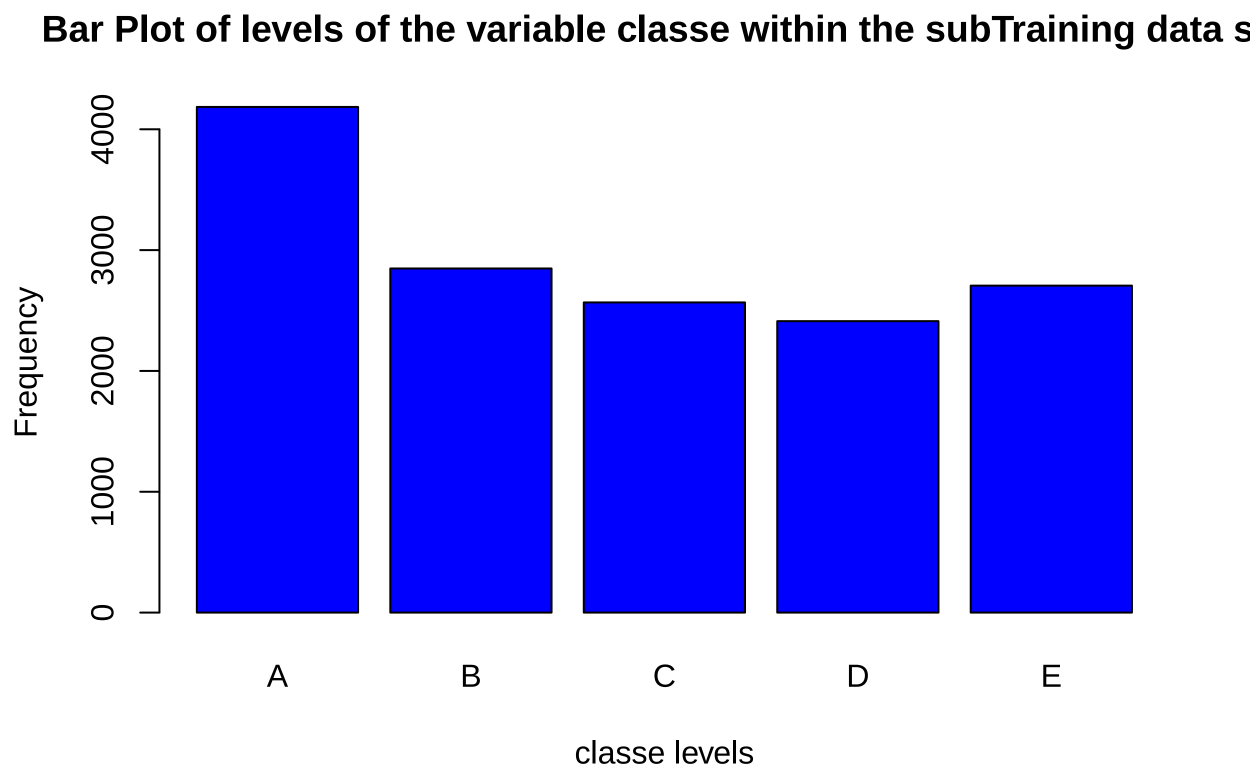
```
#In order to get out-of-sample errors, split the training data in training (75%) and testing (25%) data
inTrain <- createDataPartition(y=training$classe, p=0.75, list=FALSE)
NEOTraining <- training[inTrain, ]
NEOTesting <- training[-inTrain, ]
dim(NEOTraining)
```

```
## [1] 14718    53
```

```
dim(NEOTesting)
```

```
## [1] 4904   53
```

```
plot(NEOTraining$classe, col="blue", main="Bar Plot of levels of the variable classe within the subTrai
```

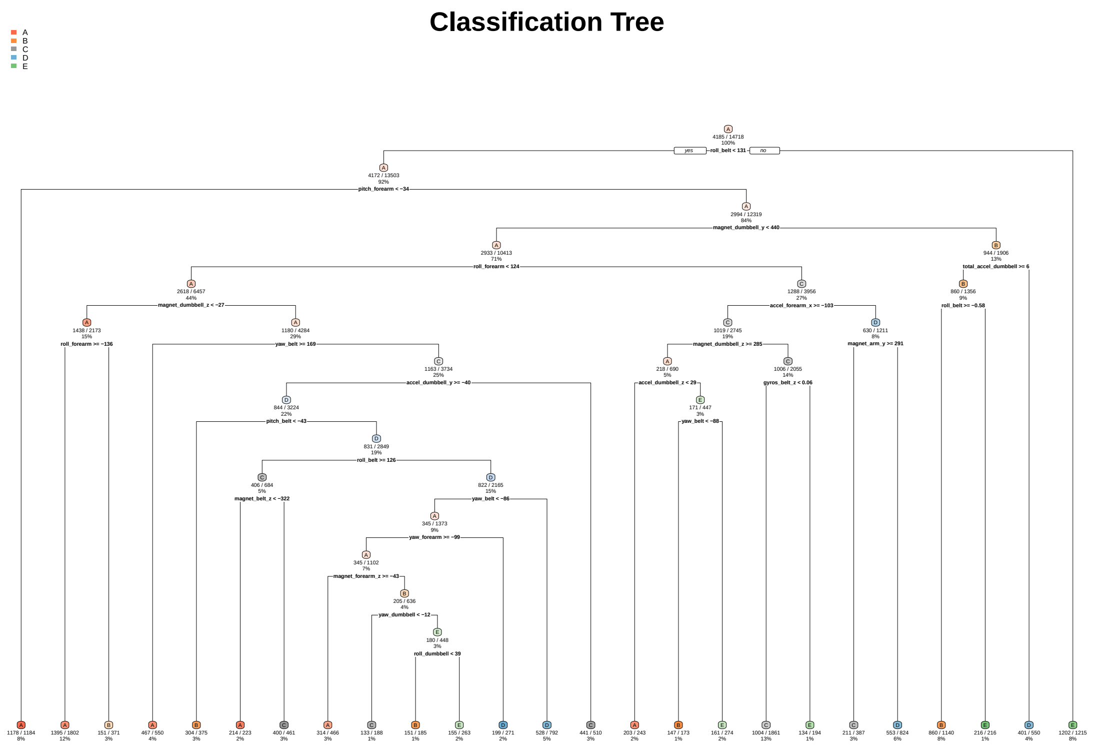


#Prediction Models

```r
#DECISION TREE
#Fit model on NEOTraining data
fitDT <- rpart(classe ~ ., data=NEOTraining, method="class")

#Use model to predict class in validation set (NEOTesting)
predictionDT <- predict(fitDT, NEOTesting, type = "class")

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



```
#Estimate the errors of the prediction algorithm in the Decision Tree model
confusionMatrix(NEOTesting$classe, predictionDT)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1274   41   29   41   10
##          B  148  564  155   63   19
##          C   14   39  739   57    6
##          D   46   46   66  557   89
##          E   29   60  113   57  642
##
## Overall Statistics
##
##                Accuracy : 0.77
##                  95% CI : (0.7579, 0.7817)
##     No Information Rate : 0.3081
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7085
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
```

```
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8432   0.7520   0.6706   0.7187   0.8381
## Specificity            0.9643   0.9073   0.9695   0.9402   0.9374
## Pos Pred Value         0.9133   0.5943   0.8643   0.6928   0.7125
## Neg Pred Value         0.9325   0.9530   0.9103   0.9468   0.9690
## Prevalence             0.3081   0.1529   0.2247   0.1580   0.1562
## Detection Rate         0.2598   0.1150   0.1507   0.1136   0.1309
## Detection Prevalence   0.2845   0.1935   0.1743   0.1639   0.1837
## Balanced Accuracy      0.9037   0.8297   0.8200   0.8294   0.8878
```

```
#RANDOM FOREST
#Fit model on NEOTraining data
fitRF <- randomForest(classe ~ ., data=NEOTraining, method="class")

#Use model to predict class in validation set (NEOTesting)
predictionRF <- predict(fitRF, NEOTesting, type = "class")

#Estimate the errors of the prediction algorithm in the Random Forest
confusionMatrix(NEOTesting$classe, predictionRF)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1395    0    0    0    0
##          B    2  947    0    0    0
##          C    0    4  851    0    0
##          D    0    0    5  799    0
##          E    0    0    0    4  897
##
## Overall Statistics
##
##                Accuracy : 0.9969
##                  95% CI : (0.995, 0.9983)
##     No Information Rate : 0.2849
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9961
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9986   0.9958   0.9942   0.9950   1.0000
## Specificity            1.0000   0.9995   0.9990   0.9988   0.9990
## Pos Pred Value         1.0000   0.9979   0.9953   0.9938   0.9956
## Neg Pred Value         0.9994   0.9990   0.9988   0.9990   1.0000
## Prevalence             0.2849   0.1939   0.1746   0.1637   0.1829
## Detection Rate         0.2845   0.1931   0.1735   0.1629   0.1829
## Detection Prevalence   0.2845   0.1935   0.1743   0.1639   0.1837
## Balanced Accuracy      0.9993   0.9976   0.9966   0.9969   0.9995
```

#Submission

```r
# Perform prediction
predictSubmission <- predict(fitRF, testing, type="class")
predictSubmission
```

```
##  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
```