Peer-graded Assignment: Prediction Assignment Writeup (R)

Mark

March 4th 2018

Goal

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.

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://web.archive.org/web/20161224072740/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

1. Load packages and data

```r
library(ggplot2)
library(caret)
```

```
## Loading required package: lattice
```

```r
library(randomForest)
```

```
## randomForest 4.6-12
```

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

```
##
## Attaching package: 'randomForest'

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

```r
library(RCurl)
```

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

```r
library(rpart)
library(rpart.plot)
library(e1071)

set.seed(112358)

train_url <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
test_url <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
training_data <- read.csv(text=getURL(train_url), na.strings=c("", "NA"))
testing_data <- read.csv(text=getURL(test_url), na.strings=c("", "NA"))

samples <- createDataPartition(y=training_data$classe, p=0.75, list=FALSE)
train_data <- training_data[samples, ]
test_data <- training_data[-samples, ]
```

2. Exploratory data analysis (preparation and basic summary of the data)

```r
# Current datasets:
dim(train_data)
```

```
## [1] 14718   160
```

```r
dim(test_data)
```

```
## [1] 4904  160
```

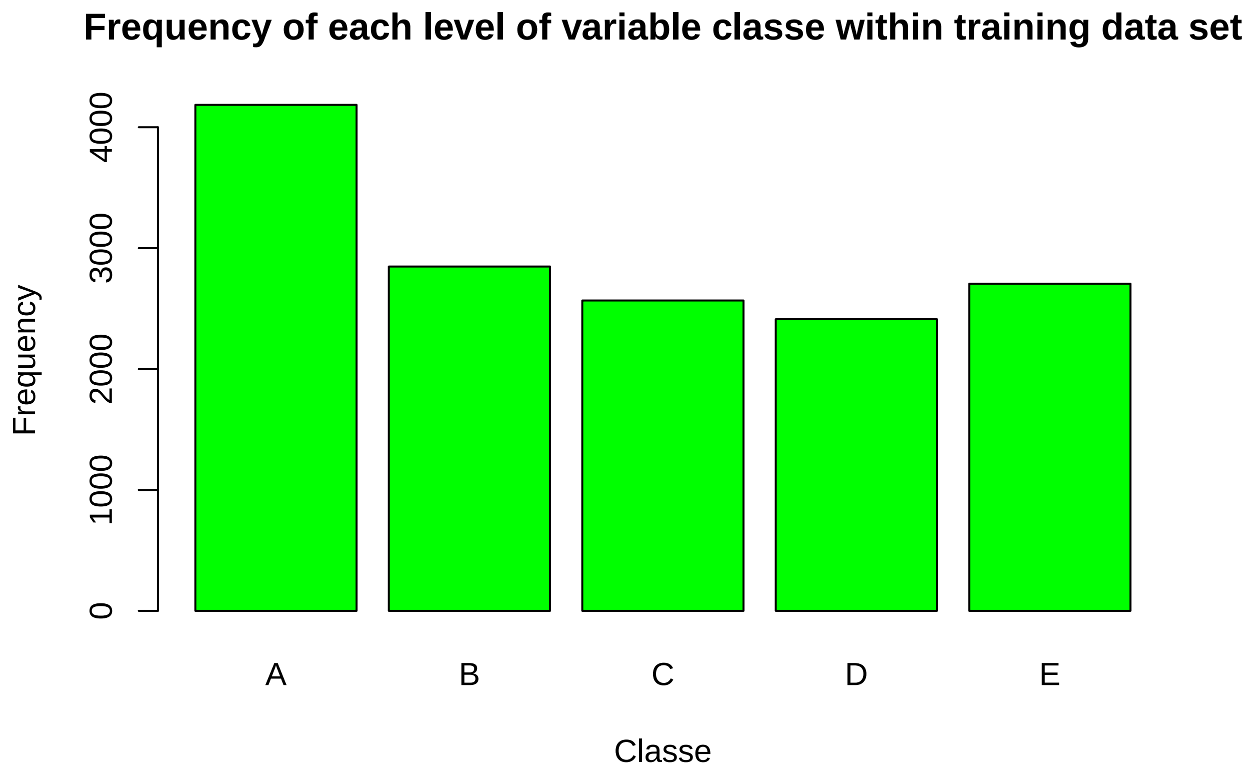
```r
# Delete columns with missing values:
training <- train_data [,colSums(is.na(train_data)) == 0]
testing <- test_data [,colSums(is.na(test_data)) == 0]

# Delete irrelevant variables: user_name, raw_timestamp_part_1, raw_timestamp_part_,2 cvtd_timestamp, n
training2 <- training[,-c(1:7)]
testing2 <- testing[,-c(1:7)]

# New datasets:
dim(training2)
```

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

```r
dim(testing2)
```

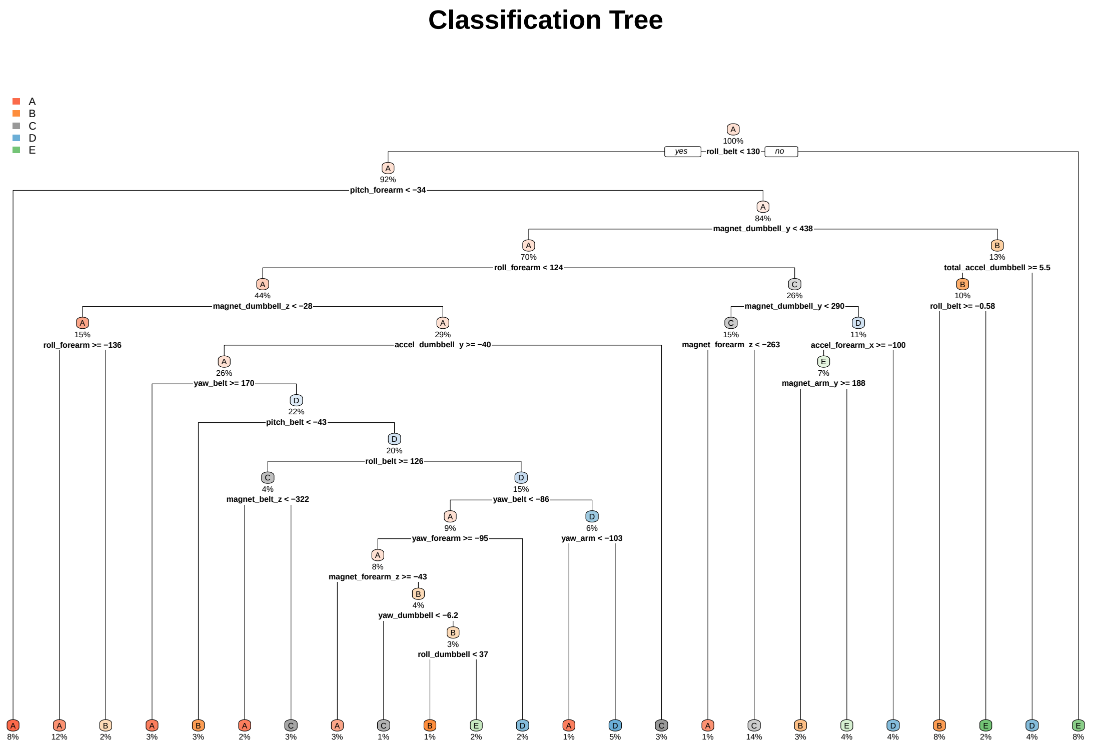
```
## [1] 4904   53
```

```r
#Plot of the outcome variable, frequency of each level in the training data set:
plot(training2$classe, col="green", main="Frequency of each level of variable classe within training da
```



3. Building a Classification Tree Model

```r
# Building the Classification Tree:
classtree <- rpart(classe ~ ., data=training2, method="class")

# Predict using the test set:
prediction_classtree <- predict(classtree, testing2, type = "class")

# Plot of the Decision Tree:
rpart.plot(classtree, main="Classification Tree", extra=100, under=TRUE, faclen=0)
```



```
# Determine the accuracy of prediction:
confusionMatrix(prediction_classtree, testing2$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1253  151   18   32   35
##          B   45  538   73   69   74
##          C   47  138  683  133  109
##          D   19   79   50  513   50
##          E   31   43   31   57  633
##
## Overall Statistics
##
##                Accuracy : 0.7382
##                  95% CI : (0.7256, 0.7504)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6682
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8982   0.5669   0.7988   0.6381   0.7026
## Specificity            0.9327   0.9340   0.8945   0.9517   0.9595
## Pos Pred Value         0.8415   0.6733   0.6153   0.7215   0.7962
## Neg Pred Value         0.9584   0.8999   0.9547   0.9306   0.9348
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2555   0.1097   0.1393   0.1046   0.1291
## Detection Prevalence   0.3036   0.1629   0.2263   0.1450   0.1621
## Balanced Accuracy      0.9155   0.7505   0.8467   0.7949   0.8310
```

4. Building a Random Forest Model

```
# Building the model:
model.rf <- randomForest(classe ~ ., data=training2)

# Predict using the test set:
predict.rf <- predict(model.rf, testing2)

# Determine the accuracy of prediction:
confusionMatrix(predict.rf, testing2$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1393    2    0    0    0
##          B    1  944    3    0    0
##          C    0    3  850    8    0
##          D    1    0    2  796    2
##          E    0    0    0    0  899
##
## Overall Statistics
##
##                Accuracy : 0.9955
##                  95% CI : (0.9932, 0.9972)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9943
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9986   0.9947   0.9942   0.9900   0.9978
## Specificity            0.9994   0.9990   0.9973   0.9988   1.0000
## Pos Pred Value         0.9986   0.9958   0.9872   0.9938   1.0000
## Neg Pred Value         0.9994   0.9987   0.9988   0.9981   0.9995
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2841   0.1925   0.1733   0.1623   0.1833
## Detection Prevalence   0.2845   0.1933   0.1756   0.1633   0.1833
## Balanced Accuracy      0.9990   0.9969   0.9957   0.9944   0.9989
```

5. Testing the Final Model

The Random Forest Model (Accuracy : 0.9955) performed better than the Classification Tree Model (Accuracy : 0.7382). We therefore choose the Random Forest Model (95% CI : (0.9932, 0.9972)). The expected out-of-sample error is 0.005, or 0.5%.

This Random Forest Model will be used for the predicting the Training Data.

```r
# Testing the Random Forest Model:

predict_training <- predict(model.rf, testing_data, type="class")

predict_training
```

```
##  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
```