Practical Machine Learning - Final Project

Erika Eden-Wynter

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 comes 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.

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.

Setting Up and Cleaning Data

```r
library(rpart)
library(rpart.plot)
library(RColorBrewer)
library(rattle)
```

```
## Rattle: A free graphical interface for data mining with R.
## Version 4.1.0 Copyright (c) 2006-2015 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```r
library(caret)
```

```
## Loading required package: lattice

## Loading required package: ggplot2
```

```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(e1071)

setwd("C:/Users/Erika/Desktop/Machine Learning")
```

Values "NA", "#DIV/0!", and " " are set to missing values:

```r
trainingset.raw <- read.csv("C:/Users/Erika/Desktop/Machine Learning/pml-training.csv", na.strings=c("N
testingset.raw <- read.csv("C:/Users/Erika/Desktop/Machine Learning/pml-testing.csv", na.strings=c("NA"
```

Columns with said missing values are removed. Also, we notice that the first 6 variables do not show data relevant to our analysis:

```r
set.seed(123)

trainingset <- trainingset.raw[,-c(1:6)]
testingset <- testingset.raw[,-c(1:6)]

dim(trainingset)
```

```
## [1] 19622   154
```

```r
nzvcol <- nearZeroVar(trainingset.raw)
trainingset <- trainingset[, -nzvcol]

dim(trainingset)
```

```
## [1] 19622   118
```

```r
trainingset <- trainingset[,colSums(is.na(trainingset)) == 0]
testingset <- testingset[,colSums(is.na(testingset)) == 0]

dim(trainingset)
```

```
## [1] 19622    39
```

Creating data partition to split the training set into a training and validation sets (80% and 20%)

```r
inTrain <- createDataPartition(y = trainingset$classe, p=0.8, list=FALSE)

subTraining <- trainingset[inTrain, ]
subTesting <- trainingset[-inTrain, ]

dim(subTraining)
```

```
## [1] 15699    39
```

We use a 10-fold cross validation to prevent overfitting

```r
tc <- trainControl(method='cv', number=10)
```

MODEL 1: DECISION TREE

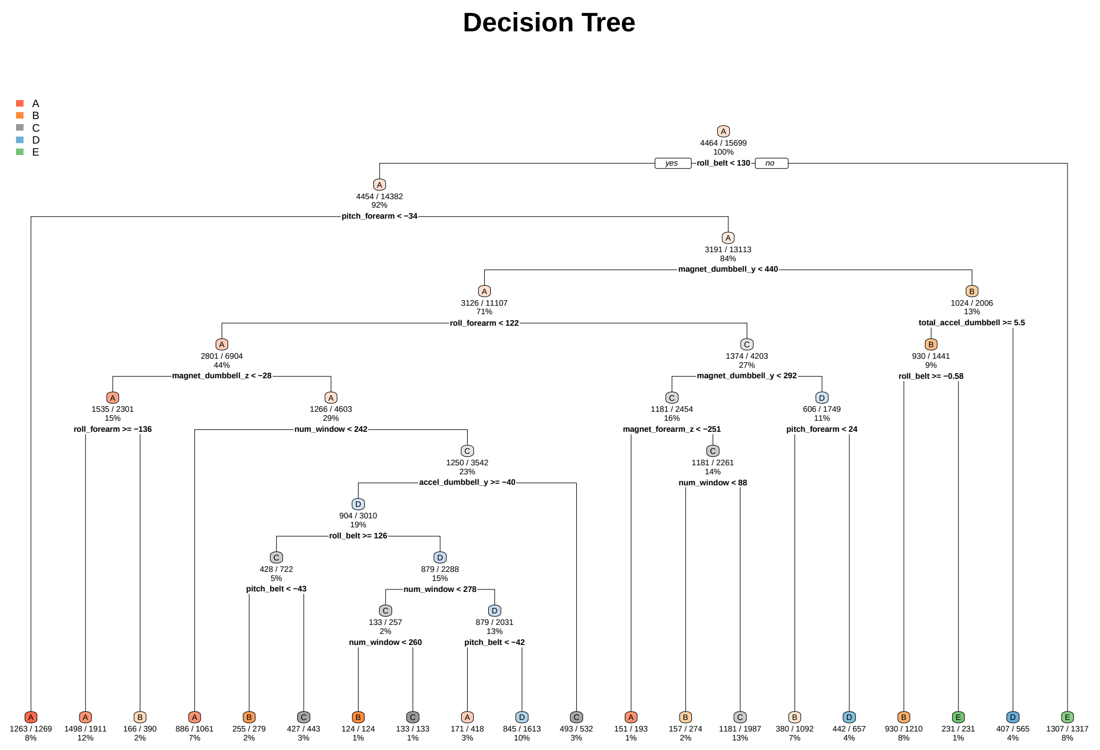
```r
DTmodel <- rpart(classe ~ ., data=subTraining, method="class")
```

Predicting:

```r
DTprediction <- predict(DTmodel, subTesting, type = "class")
```

Plot of the Decision Tree

```r
rpart.plot(DTmodel, main="Decision Tree", extra=102, under=TRUE, faclen=0)
```



Test results on our subTesting data set:

```
confusionMatrix(DTprediction, subTesting$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   A   B   C   D   E
##          A 998 127  31  39  35
##          B  52 488  73  85 159
##          C  14  57 543  95  61
##          D  48  87  37 424  91
##          E   4   0   0   0 375
##
## Overall Statistics
##
##                Accuracy : 0.7209
##                  95% CI : (0.7066, 0.7349)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6457
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8943   0.6430   0.7939   0.6594  0.52011
## Specificity            0.9173   0.8834   0.9299   0.9198  0.99875
## Pos Pred Value         0.8114   0.5694   0.7052   0.6172  0.98945
## Neg Pred Value         0.9562   0.9116   0.9553   0.9323  0.90237
## Prevalence             0.2845   0.1935   0.1744   0.1639  0.18379
## Detection Rate         0.2544   0.1244   0.1384   0.1081  0.09559
## Detection Prevalence   0.3135   0.2185   0.1963   0.1751  0.09661
## Balanced Accuracy      0.9058   0.7632   0.8619   0.7896  0.75943
```

MODEL 2: RANDOM FOREST

```
RFmodel <- randomForest(classe ~. , data=subTraining, method="class")
```

Predicting:

```
RFprediction <- predict(RFmodel, subTesting, type = "class")
```

Test results on subTesting data set:

```
confusionMatrix(RFprediction, subTesting$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1116    1    0    0    0
##          B    0  757    0    0    0
##          C    0    1  684    0    0
##          D    0    0    0  643    1
##          E    0    0    0    0  720
##
## Overall Statistics
##
##                Accuracy : 0.9992
##                  95% CI : (0.9978, 0.9998)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.999
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9974   1.0000   1.0000   0.9986
## Specificity            0.9996   1.0000   0.9997   0.9997   1.0000
## Pos Pred Value         0.9991   1.0000   0.9985   0.9984   1.0000
## Neg Pred Value         1.0000   0.9994   1.0000   1.0000   0.9997
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2845   0.1930   0.1744   0.1639   0.1835
## Detection Prevalence   0.2847   0.1930   0.1746   0.1642   0.1835
## Balanced Accuracy      0.9998   0.9987   0.9998   0.9998   0.9993
```

Comparisson of Decision Tree V.S. Random Forest models

We notice that accuracy for RF is higher with 0.9985 (99.85%) compared to that of the DT, which is at 0.8274 (82.74%)

Now that we have chosen our model, we predict the outcomes:

```r
finalPrediction <- predict(RFmodel, testingset, type = "class")
print(finalPrediction)
```

```
##  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
```