Practical machine learning: Prediction assignment, Week 4 Project

3rd July, 2017

J.Li

==

Introduction

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, our 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).

The training data are available as follows: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv

The test data are available as follows: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

```r
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.4.1
```

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

```
## Loading required package: ggplot2
```

```
## Warning: package 'ggplot2' was built under R version 3.4.1
```

```r
library(rpart)
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 3.4.1
```

```
## 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
```

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

```
## Warning: package 'rpart.plot' was built under R version 3.4.1
```

Loading Data

We load the training and testing data and replace the missing values by "NA".

```
urlTraining <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
urlTesting <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
training <- read.csv(url(urlTraining), na.strings = c("NA", "#DIV0!" , ""))
testing <- read.csv(url(urlTesting), na.strings = c("NA", "#DIV0!", ""))
```

Let's define the same columns

```
sameColumns <- colnames(training) == colnames(testing)
colnames(training)[sameColumns==FALSE]
```

```
## [1] "classe"
```

Note that "classe" is not included in the testing data.

Cleaning data

```
training<-training[,colSums(is.na(training)) == 0]
testing <-testing[,colSums(is.na(testing)) == 0]
```

We deleted the first 7 variables which are not related to prediction.

```
training <- training[,8:dim(training)[2]]
testing <- testing[,8:dim(testing)[2]]
```

Cross Validation

We devide training dataset into three portions for different purposes: trainning 60%, testing 20%, and validation 20%.

```
set.seed(12345)
dataset1 <- createDataPartition(y = training$classe, p = 0.8, list = F)
dataset2 <- training[dataset1,]
validation <- training[-dataset1,]
trainingdata1 <- createDataPartition(y = dataset2$classe, p = 0.75, list = F)
trainingdata2 <- dataset2[trainingdata1,]
testingdata <- dataset2[-trainingdata1,]
```

Random forest model

```
forestModel <- randomForest(classe ~ ., data=trainingdata2, method="class")
predictionForest <- predict(forestModel, testingdata, type="class")
randomForestModel <- confusionMatrix(predictionForest, testingdata$classe)
randomForestModel
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1114    4    0    0    0
##          B    1  754    5    0    0
##          C    1    1  676   10    0
##          D    0    0    3  632    3
##          E    0    0    0    1  718
##
## Overall Statistics
##
##                Accuracy : 0.9926
##                  95% CI : (0.9894, 0.995)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9906
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9982   0.9934   0.9883   0.9829   0.9958
## Specificity            0.9986   0.9981   0.9963   0.9982   0.9997
## Pos Pred Value         0.9964   0.9921   0.9826   0.9906   0.9986
## Neg Pred Value         0.9993   0.9984   0.9975   0.9967   0.9991
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2840   0.1922   0.1723   0.1611   0.1830
## Detection Prevalence   0.2850   0.1937   0.1754   0.1626   0.1833
## Balanced Accuracy      0.9984   0.9958   0.9923   0.9905   0.9978
```

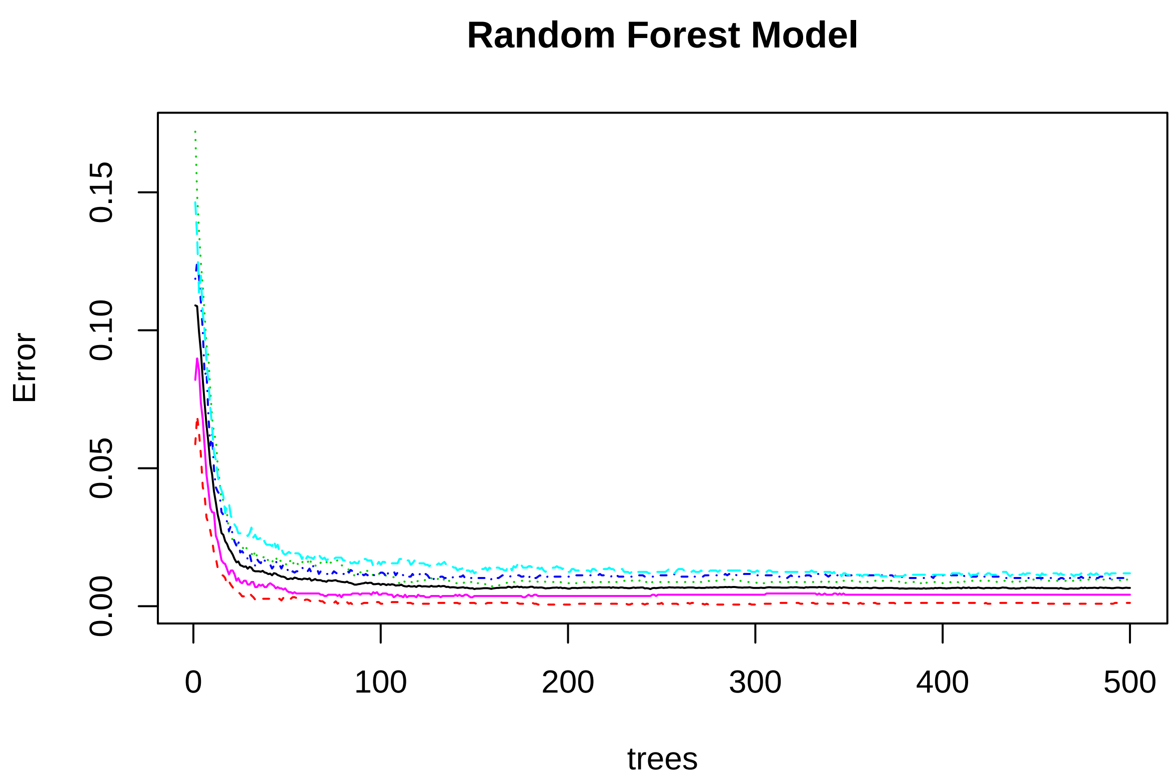
```
accuracy1 <- round(randomForestModel$overall['Accuracy'] * 100, 2)
error1 <- round(1 - randomForestModel$overall['Accuracy'],2)
accuracy1
```

```
## Accuracy
##    99.26
```

```
error1
```

```
## Accuracy
##     0.01
```

```r
plot(forestModel, main = "Random Forest Model")
```



Decision Tree Model

```r
modelTree <- rpart(classe ~ ., data=trainingdata2, method="class")
predictionTree <- predict(modelTree, testingdata, type="class")
decisionTree <- confusionMatrix(predictionTree, testingdata$classe)
decisionTree
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   A   B   C   D   E
##          A 981 148  12  48  37
##          B  37 437  99  33  79
##          C  35  70 517 115  95
##          D  47  69  39 412  56
##          E  16  35  17  35 454
##
## Overall Statistics
##
##                Accuracy : 0.714
##                  95% CI : (0.6996, 0.7281)
```

```
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6371
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8790   0.5758   0.7558   0.6407   0.6297
## Specificity            0.9127   0.9216   0.9027   0.9357   0.9678
## Pos Pred Value         0.8002   0.6380   0.6214   0.6613   0.8151
## Neg Pred Value         0.9499   0.9006   0.9460   0.9300   0.9207
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2501   0.1114   0.1318   0.1050   0.1157
## Detection Prevalence   0.3125   0.1746   0.2121   0.1588   0.1420
## Balanced Accuracy      0.8959   0.7487   0.8293   0.7882   0.7988
```

```r
accuracy2 <- round(decisionTree$overall['Accuracy'] * 100, 2)
error2 <- round(1 - decisionTree$overall['Accuracy'],2)
accuracy2
```

```
## Accuracy
##     71.4
```

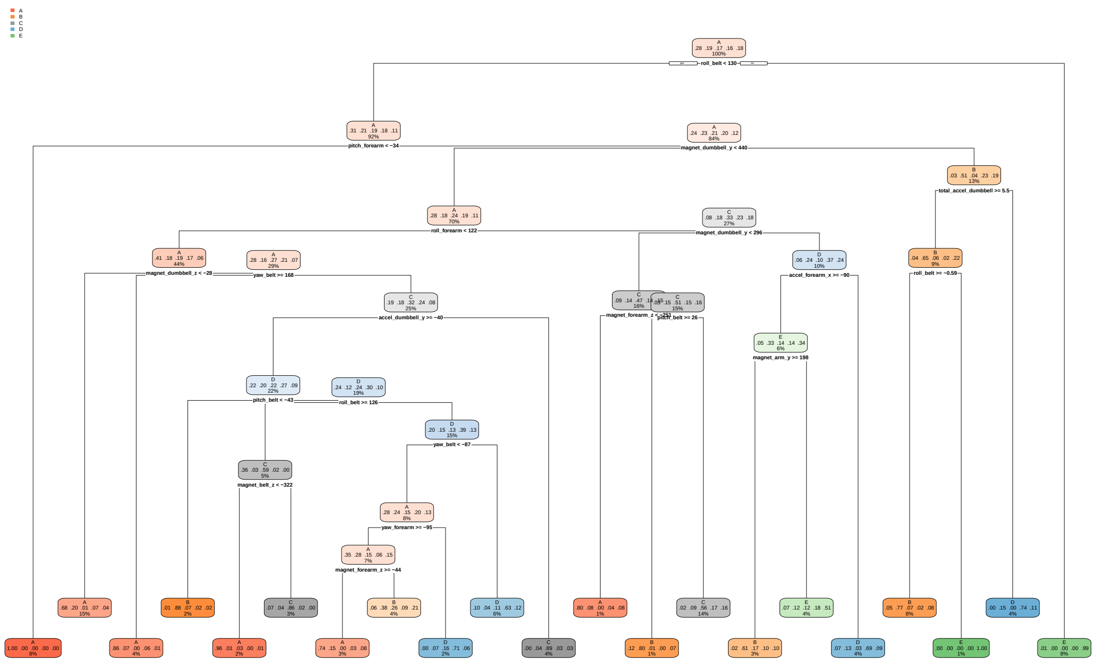
```r
error2
```

```
## Accuracy
##     0.29
```

```r
rpart.plot(modelTree, main = "Decision Tree Model")
```

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

Decision Tree Model



Summary

In this prediction project, we processed and analyzed both traning and testing datasets in order to establish prediction models and find out their relevent accuracy. We found that the Random Forest produces better accuracy (99.26%) than the Decision Tree does (71.4%).