Practical Machine Learning Assignment

finbark

30 September 2018

Overview

This report outlines predictions for the manner in which six participants performed a particular type of exercise as based on data from a fitness tracker. 20 differnet test cases are predicted using the prediction model.

Getting the data

```
set.seed(12345)
trainUrl <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
predictionUrl <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
```

Load and tidy data

Load the data and standardise how the data represents NA values.

```
trainingData <- read.csv(url(trainUrl), na.strings = c("NA", "#DIV/0!", ""))
predictionData <- read.csv(url(predictionUrl), na.strings = c("NA", "#DIV/0!", ""))
```

Remove variables one to six as they are not relevant.

```
trainingData <- trainingData[, -(1:6)]
predictionData <- predictionData[, -(1:6)]
```

Remove variables that are more than 70% NAs.

```
NAs <- sapply(trainingData, function(x) mean(is.na(x))) > 0.7
trainingData <- trainingData[, NAs == FALSE]
predictionData <- predictionData[, NAs == FALSE]
```

Partition into training and testing sets.

```
inTraining <- createDataPartition(trainingData$classe, p = 0.6, list = FALSE)
trainingSet <- trainingData[inTraining, ]
testSet <- trainingData[-inTraining, ]
```

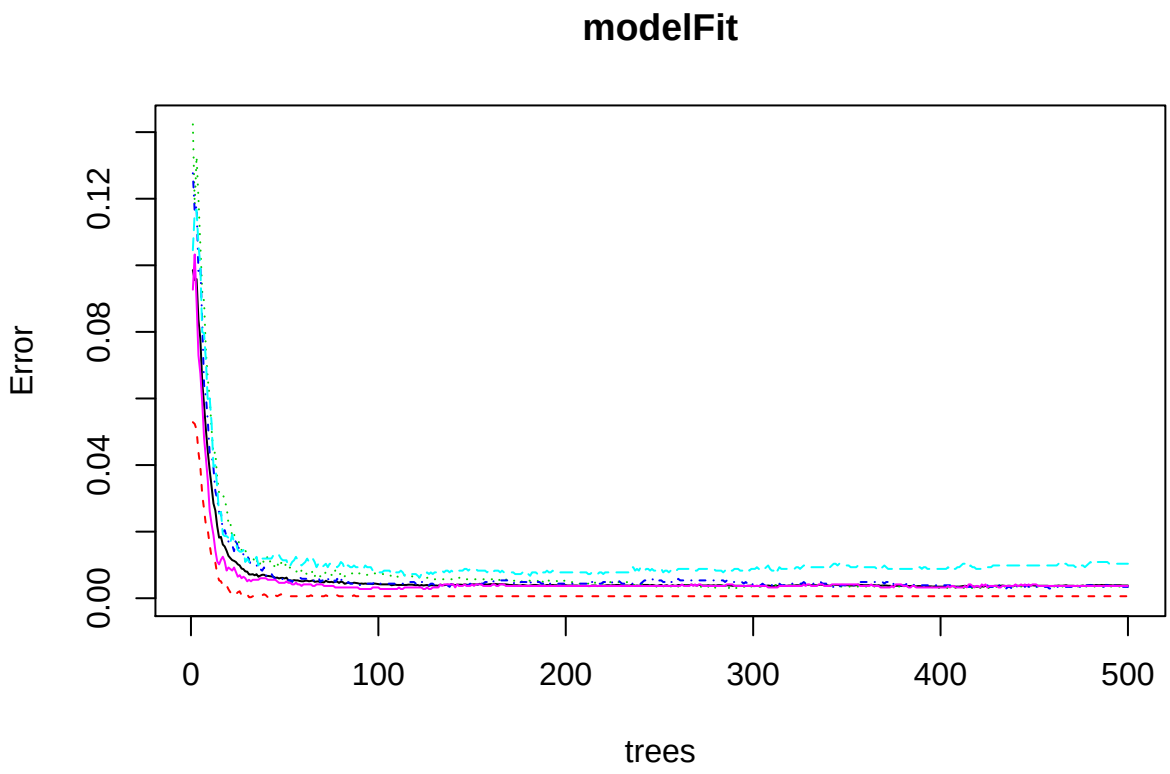
Predicting with random forests

We shall first try to predict with random forests as they are know to be quite effective.

```
modelFit <- randomForest(classe ~ ., data = trainingSet)
prediction <- predict(modelFit, testSet, type = "class")
matrix <- confusionMatrix(prediction, testSet$classe)
matrix
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2232    3    0    0    0
##          B    0 1513    3    0    0
##          C    0    2 1365   13    0
##          D    0    0    0 1272    3
##          E    0    0    0    1 1439
##
## Overall Statistics
##
##                Accuracy : 0.9968
##                  95% CI : (0.9953, 0.9979)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.996
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9967   0.9978   0.9891   0.9979
## Specificity            0.9995   0.9995   0.9977   0.9995   0.9998
## Pos Pred Value         0.9987   0.9980   0.9891   0.9976   0.9993
## Neg Pred Value         1.0000   0.9992   0.9995   0.9979   0.9995
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2845   0.1928   0.1740   0.1621   0.1834
## Detection Prevalence   0.2849   0.1932   0.1759   0.1625   0.1835
## Balanced Accuracy      0.9997   0.9981   0.9977   0.9943   0.9989
```

```
plot(modelFit)
```



The above matrix and plot show that the random forest does a good job at predicting the classe variable as it has a high accuracy score of 0.9975.

Predicting class on test data

```
prediction <- predict(modelFit, predictionData, type = "class")
prediction
```

```
##  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
```