Prediction Assignment Writeup

Ignas

19/04/2020

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.The goal is to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants who were asked to perform barbell lifts correctly and incorrectly in 5 different ways.

Task

The goal of this project is to predict the manner in which they did the exercise and build prediction model to predict 20 different test cases.

Data preparation

Let's load necessary libraries and import the data.

```
library(rpart)
library(rattle)
library(caret)
library(randomForest)
set.seed(12345)

# Training and Testing data
TrainURL <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
TestURL <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"

# Download and clean the datasets
trainingdata <- read.csv(url(TrainURL), na.strings=c("NA","#DIV/0!",""))
testingdata <- read.csv(url(TestURL), na.strings=c("NA","#DIV/0!",""))
```

Dimensions of downloaded data

```
dim(trainingdata); dim(testingdata)
```

```
[1] 19622   160
```

```
[1]  20 160
```

Summary of the data (as the data is quite large, summary results are not provided).

```
str(trainingdata); str(testingdata)
```

Let's delete columns with NA values and delete unnecessary columns:

```
trainingdata <-trainingdata[,colSums(is.na(trainingdata)) == 0]
testingdata <-testingdata[,colSums(is.na(testingdata)) == 0]
```

```
trainingdata <-trainingdata[,-c(1:7)]
testingdata <-testingdata[,-c(1:7)]

dim(trainingdata); dim(testingdata)
```

```
[1] 19622    53
```

```
[1] 20 53
```

Data partition

In order to be able to apply cross-validation, data will be splitted into 70% of Training data and 30 percent of Testing data.

```
datapart <- createDataPartition(trainingdata$classe, p=0.7, list=FALSE)
trainingset <- trainingdata[datapart, ]
testingset <- trainingdata[-datapart, ]
dim(trainingset);dim(testingset)
```

```
## [1] 13737    53
```

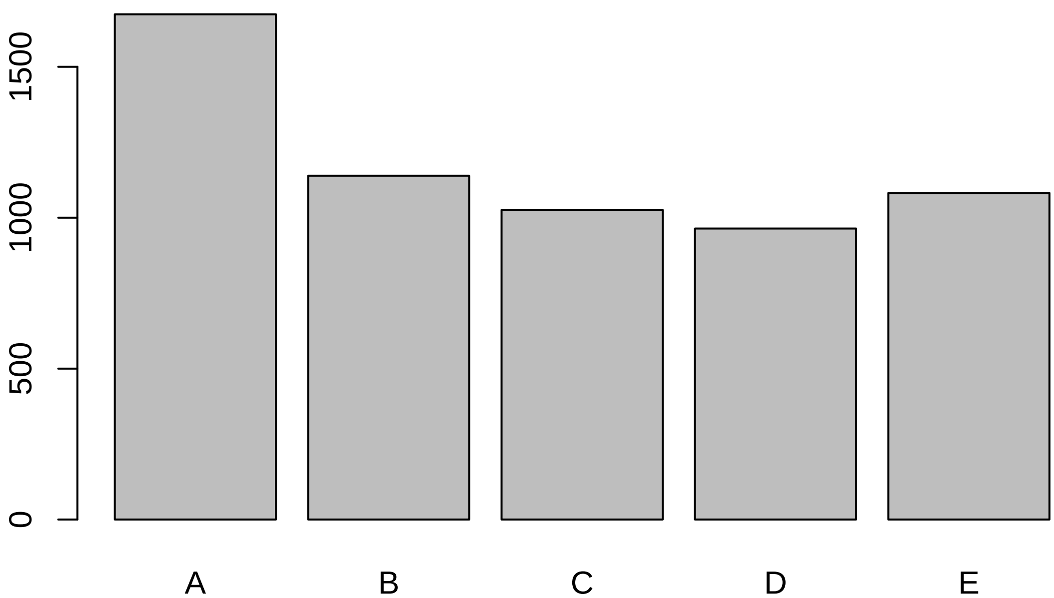
```
## [1] 5885   53
```

Now we can explore variable `classe` of the training data set:

```
summary(testingset$classe)
```

```
##    A    B    C    D    E
## 1674 1139 1026  964 1082
```

```
plot(testingset$classe)
```



We see that there are 5 classes where A is the most frequent and D is at least frequent. However, all of the 5 classes are distributed more or less the same.

Prediction

In order to generate predictions, we will use decision tree and random forest models.

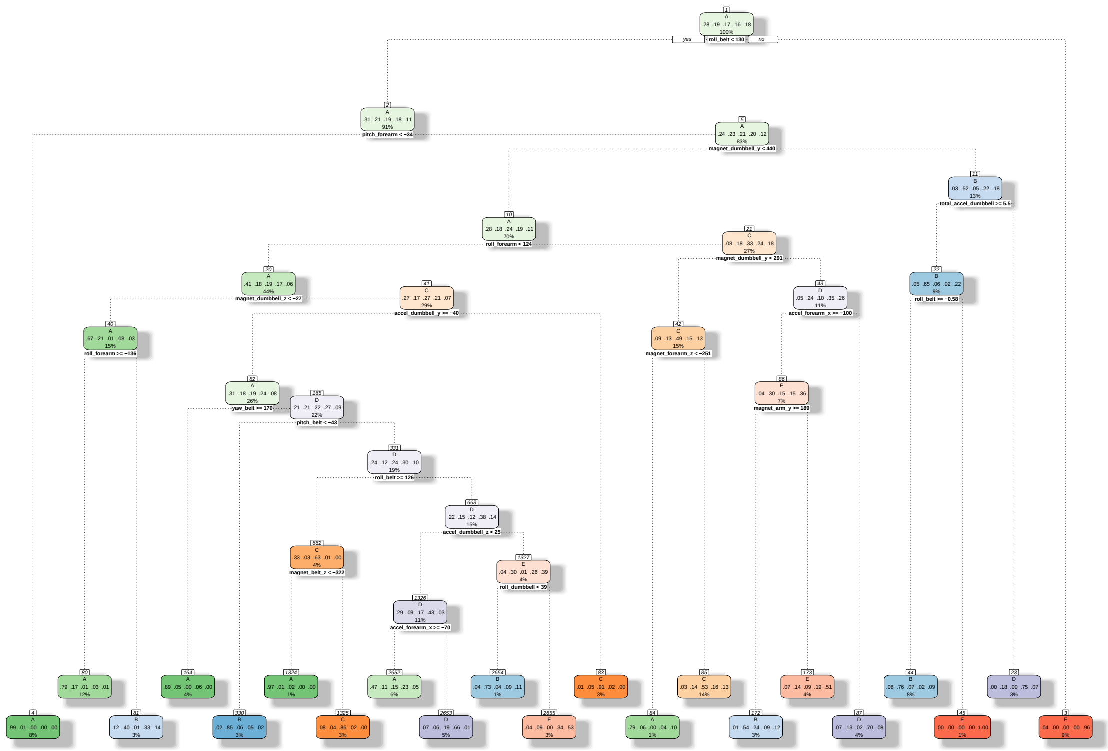
1. Decision tree model

```r
dtmod <- rpart(classe ~ ., data = trainingset, method = "class")
dtpred <- predict(dtmod, testingset, type = "class")
confusionMatrix(dtpred, testingset$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1498  196   69  106   25
##          B   42  669   85   86   92
##          C   43  136  739  129  131
##          D   33   85   98  553   44
##          E   58   53   35   90  790
##
## Overall Statistics
##
##                Accuracy : 0.722
##                  95% CI : (0.7104, 0.7334)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6467
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.8949   0.5874   0.7203  0.57365   0.7301
## Specificity            0.9060   0.9357   0.9097  0.94717   0.9509
## Pos Pred Value         0.7909   0.6869   0.6273  0.68020   0.7700
## Neg Pred Value         0.9559   0.9043   0.9390  0.91897   0.9399
## Prevalence             0.2845   0.1935   0.1743  0.16381   0.1839
## Detection Rate         0.2545   0.1137   0.1256  0.09397   0.1342
## Detection Prevalence   0.3218   0.1655   0.2002  0.13815   0.1743
## Balanced Accuracy      0.9004   0.7615   0.8150  0.76041   0.8405
```

Decision tree:

```r
fancyRpartPlot(dtmod)
```



Rattle 2020−Apr−19 15:10:06 Ignas

2. Random forest model

```r
rfmod <- randomForest(classe ~., data=trainingset, method="class")
rfpred <- predict(rfmod, testingset, Type="class")
confusionMatrix(rfpred, testingset$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1673   10    0    0    0
##          B    1 1127   11    0    0
##          C    0    2 1015   13    0
##          D    0    0    0  951    4
##          E    0    0    0    0 1078
##
## Overall Statistics
##
##                Accuracy : 0.993
##                  95% CI : (0.9906, 0.995)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9912
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
```

```
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9994   0.9895   0.9893   0.9865   0.9963
## Specificity            0.9976   0.9975   0.9969   0.9992   1.0000
## Pos Pred Value         0.9941   0.9895   0.9854   0.9958   1.0000
## Neg Pred Value         0.9998   0.9975   0.9977   0.9974   0.9992
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2843   0.1915   0.1725   0.1616   0.1832
## Detection Prevalence   0.2860   0.1935   0.1750   0.1623   0.1832
## Balanced Accuracy      0.9985   0.9935   0.9931   0.9929   0.9982
```

Conclusion

Accuracy of decision tree model is 72.2% while using random forest is 99.3%. Out-of-sample error is 0.7% (calculated as 1-ACCURACY).

The results are as we have expected, i.e. random forest model performs better than decision tree.

Original test set prediction

Now the results will be applied on original test date (20 observations)

```
ftest <- predict(rfmod, testingdata, type = "class")
ftest
```

```
##  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
```

```
write.csv(ftest, "final_prediction.csv")
```