Predictive analysis of activity

Elena Schnell

10 November 2018

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, 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 processing

```
library(caret)
library(rpart)
library(rpart.plot)
library(rattle)
set.seed(1290)
```

First we load the data and replace missing, NA and #DIV/0! values with NA.

```
trainURL <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
testURL <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
training <- read.csv(url(trainURL),na.strings=c('#DIV/0!', '', 'NA'))
testing <- read.csv(url(testURL),na.strings=c('#DIV/0!', '', 'NA'))
```

Then we clean the data by removing columns with NA entries.

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

We further remove Near-Zero Variance variables.

```
nzv <- nearZeroVar(training)
training<-training[,-nzv]
```

We look at the data and remove not needed columns

```
head(training)
training<-training[,-c(1:6)]
```

Then we check the outcome variable

```
table(training$classe)
```

```
##
##    A    B    C    D    E
## 5580 3797 3422 3216 3607
```

```
prop.table(table(training$classe))
```

```
##
##         A         B         C         D         E
## 0.2843747 0.1935073 0.1743961 0.1638977 0.1838243
```

Cross Validation

Now that we have reduced the number of variables, we create a training (60%) and test set (40%) for cross validation:

```
inTrain=createDataPartition(y=training$classe, p=0.6, list=FALSE)
    subtrain <-training[inTrain,]
    subtest <- training[-inTrain,]
```

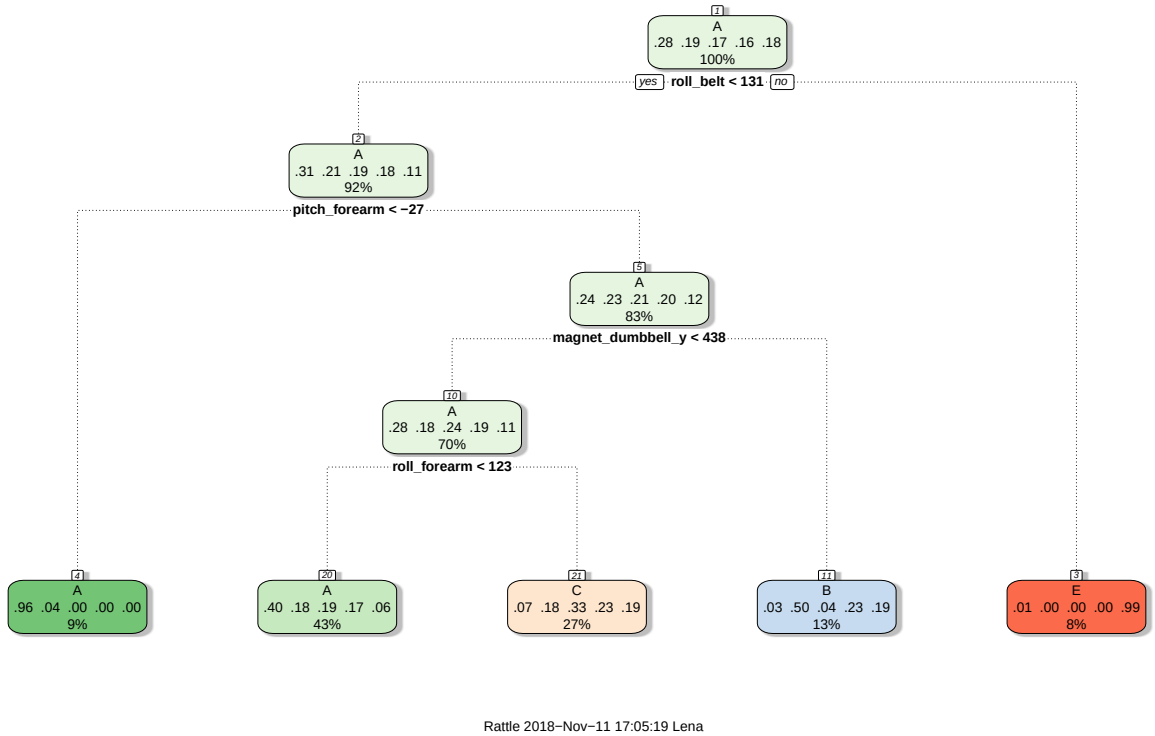
Model selection

We compare a decision tree and random forest method.

Decision Tree

We build the decision tree and use the function fancyRpartPlot() to plot the classification tree as a dendogram. Then we validate the model, which was build on the subtrain data, on the subtest data.

```
modfit1 <- train(classe ~ .,method='rpart',data=subtrain)
    fancyRpartPlot(modfit1$finalModel)
```




```
     pred=predict(modfit1,newdata=subtest)
    z=confusionMatrix(pred,subtest$classe)
    z$table
```

```
##           Reference
## Prediction    A    B    C    D    E
##          A 2028  655  638  561  211
##          B   36  515   44  229  201
##          C  160  348  686  496  372
##          D    0    0    0    0    0
##          E    8    0    0    0  658
```

```
    z$overall[1]
```

```
##  Accuracy
## 0.4954117
```

The accuracy of the model is only 0.49, which would mean a big out-of-sample error of 0.51.

Random Forest

We first determine the model and then again validate it on the subtest data.

```
modfit2 <- train(classe ~ .,method='rf',data=subtrain)
     pred2=predict(modfit2,newdata=subtest)
    z2=confusionMatrix(pred2,subtest$classe)
    z2$table
```

```
##           Reference
## Prediction    A    B    C    D    E
##          A 2229   15    0    0    0
##          B    0 1499    9    0    2
##          C    1    4 1355   14    3
##          D    0    0    4 1269    7
##          E    2    0    0    3 1430
```

```
    z2$overall[1]
```

```
## Accuracy
## 0.991843
```

As the random forest has a accuracy of 0.99 compared to an accuracy of 0.49 of the decision tree, we will use the random forest method.The out-of-sample error is then only 0.0028.

Predicting

We then use our model to predict the 20 cases in the test data, which was kept unchanged.

```
predicted=predict(modfit2,newdata=testing)
predicted
```

```
##  [1] 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
```