Machine Learning Project

HalilovicAj

19 Juli 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, a group of 6 people were asked to perform barbell lifts correctly and incorrectly in 5 different ways, and our goal will be to use the data gathered from accelerometers on the belt, forearm, arm, and dumbell, and predict the manner in which they did the exercise.

Importing and cleaning data

```
library(caret)
library(randomForest)
library(rattle)
library(rpart)
library(party)
library(rpart.plot)

training_and_validation <- read.csv("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv");
dim(training_and_validation)
```

```
## [1] 19622   160
```

```
testing <- read.csv("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"); dim(testing)
```

```
## [1]  20 160
```

```
set.seed(78863439)
inTrain <- sample(2, nrow(training_and_validation), replace = TRUE, prob = c(0.7,0.3))

training   <- training_and_validation[inTrain==1,]
validation <- training_and_validation[inTrain==2,]
dim(training); dim(validation)
```

```
## [1] 13791   160
```

```
## [1] 5831  160
```

```
training <- subset(training, select=-c(X, user_name, raw_timestamp_part_1, raw_timestamp_part_2, cvtd_timest
amp, new_window, num_window, kurtosis_roll_belt, kurtosis_picth_belt,kurtosis_yaw_belt, skewness_roll_belt,
skewness_roll_belt.1, skewness_yaw_belt, max_yaw_belt, min_yaw_belt, amplitude_yaw_belt,kurtosis_roll_arm, k
urtosis_picth_arm, kurtosis_yaw_arm, skewness_roll_arm, skewness_pitch_arm, skewness_yaw_arm, kurtosis_roll_
dumbbell, kurtosis_picth_dumbbell, kurtosis_yaw_dumbbell, skewness_roll_dumbbell, skewness_pitch_dumbbell, s
kewness_yaw_dumbbell, max_yaw_dumbbell,min_yaw_dumbbell, amplitude_yaw_dumbbell, kurtosis_roll_forearm, kurt
osis_picth_forearm, kurtosis_yaw_forearm, skewness_roll_forearm,skewness_pitch_forearm, skewness_yaw_forearm
, max_yaw_forearm, min_yaw_forearm, amplitude_yaw_forearm,max_roll_belt,
max_picth_belt, min_roll_belt, min_pitch_belt, amplitude_roll_belt,amplitude_pitch_belt,var_total_accel_belt
, avg_roll_belt, stddev_roll_belt, var_roll_belt, avg_pitch_belt, stddev_pitch_belt,var_pitch_belt,avg_yaw_b
elt,stddev_yaw_belt,var_yaw_belt,var_accel_arm,avg_roll_arm,stddev_roll_arm, var_roll_arm, avg_pitch_arm,std
dev_pitch_arm,var_pitch_arm, avg_yaw_arm, stddev_yaw_arm, var_yaw_arm,max_roll_arm,max_picth_arm, max_yaw_ar
m,min_roll_arm,min_pitch_arm,min_yaw_arm,amplitude_roll_arm,amplitude_pitch_arm,amplitude_yaw_arm,max_roll_d
umbbell,max_picth_dumbbell, min_roll_dumbbell, min_pitch_dumbbell, amplitude_roll_dumbbell, amplitude_pitch_
dumbbell,var_accel_dumbbell,avg_roll_dumbbell,stddev_roll_dumbbell, var_roll_dumbbell, avg_pitch_dumbbell, s
tddev_pitch_dumbbell,stddev_pitch_dumbbell,var_pitch_dumbbell,avg_yaw_dumbbell, stddev_yaw_dumbbell, var_yaw
_dumbbell, max_roll_forearm,max_picth_forearm, min_roll_forearm, min_pitch_forearm, amplitude_roll_forearm,
amplitude_pitch_forearm,
var_accel_forearm, avg_roll_forearm, stddev_roll_forearm,var_roll_forearm,avg_pitch_forearm, stddev_pitch_fo
rearm,var_pitch_forearm, avg_yaw_forearm, stddev_yaw_forearm, var_yaw_forearm))
```

We will build our predictive model using trees. For this purpose we will be comparing the R functions rpart (CART algorithm) and ctree (CHAID algorithm). As you can see above, the original training sample was split into training and validation, so that we can validate the performance of the models.

We have excluded any irrelevant predictors or predictors with too many missing values.

Model selection

Both, for rpart and ctree, we have decided to set \(minbucket=100\) - in order to have more stability out-of-sample).

We begin by using the ctree function. We calculate the accuracy both, on the training sample and on the validation sample:

```
modFit1 <- ctree(classe ~ .,
                 data=training, controls = ctree_control(mincriterion = 0.95,minbucket = 100))
pred   <- predict(modFit1,newdata=training, type="response")
mean(pred==training$classe)
```

```
## [1] 0.7159017
```

```
pred   <- predict(modFit1,newdata=validation, type="response")
mean(pred==validation$classe)
```

```
## [1] 0.695078
```

Next we use the rpart function which has a built-in cross validation procedure:

```
modFit2 <- rpart(classe~.,data=training, control = rpart.control(minbucket = 100,cp=-1))
pred   <- predict(modFit2,newdata=training, type="class")
mean(pred==training$classe)
```

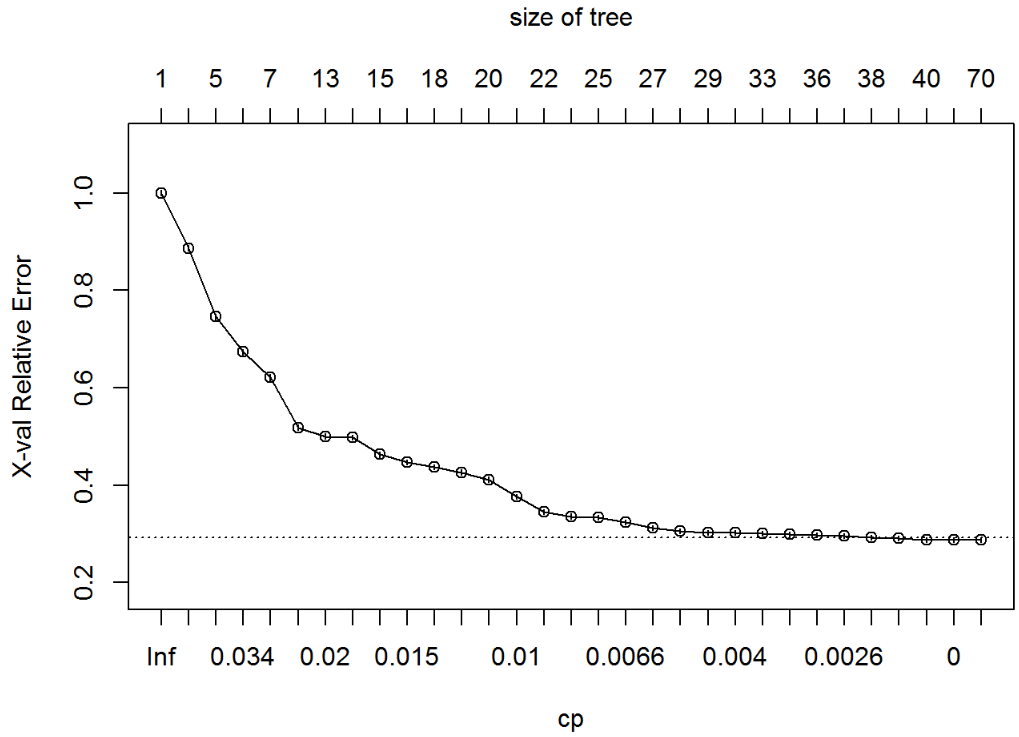
```
## [1] 0.7988543
```

```
pred   <- predict(modFit2,newdata=validation, type="class")
mean(pred==validation$classe)
```

```
## [1] 0.786143
```

The model modFit2 needs to be pruned, since we have selected \(cp=-1\). In order to decide where to prune, we visualize the relative cv-errors:

```
plotcp(modFit2)
```



We choose to prune at \(cp=0.0052\) and hence reduce the number of nodes from 70 to 27. We obtain

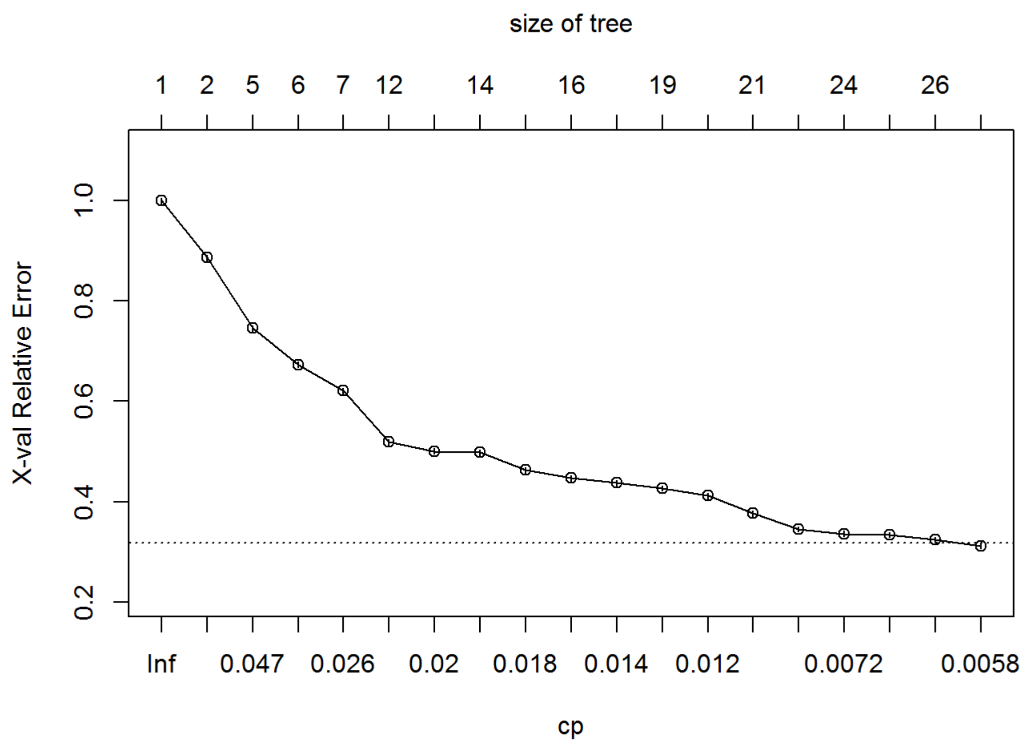
```
modFit2_pruned <- prune.rpart(modFit2,cp=0.0052)
pred   <- predict(modFit2_pruned,newdata=training, type="class")
mean(pred==training$classe)
```

```
## [1] 0.7668044
```

```
pred   <- predict(modFit2_pruned,newdata=validation, type="class")
mean(pred==validation$classe)
```

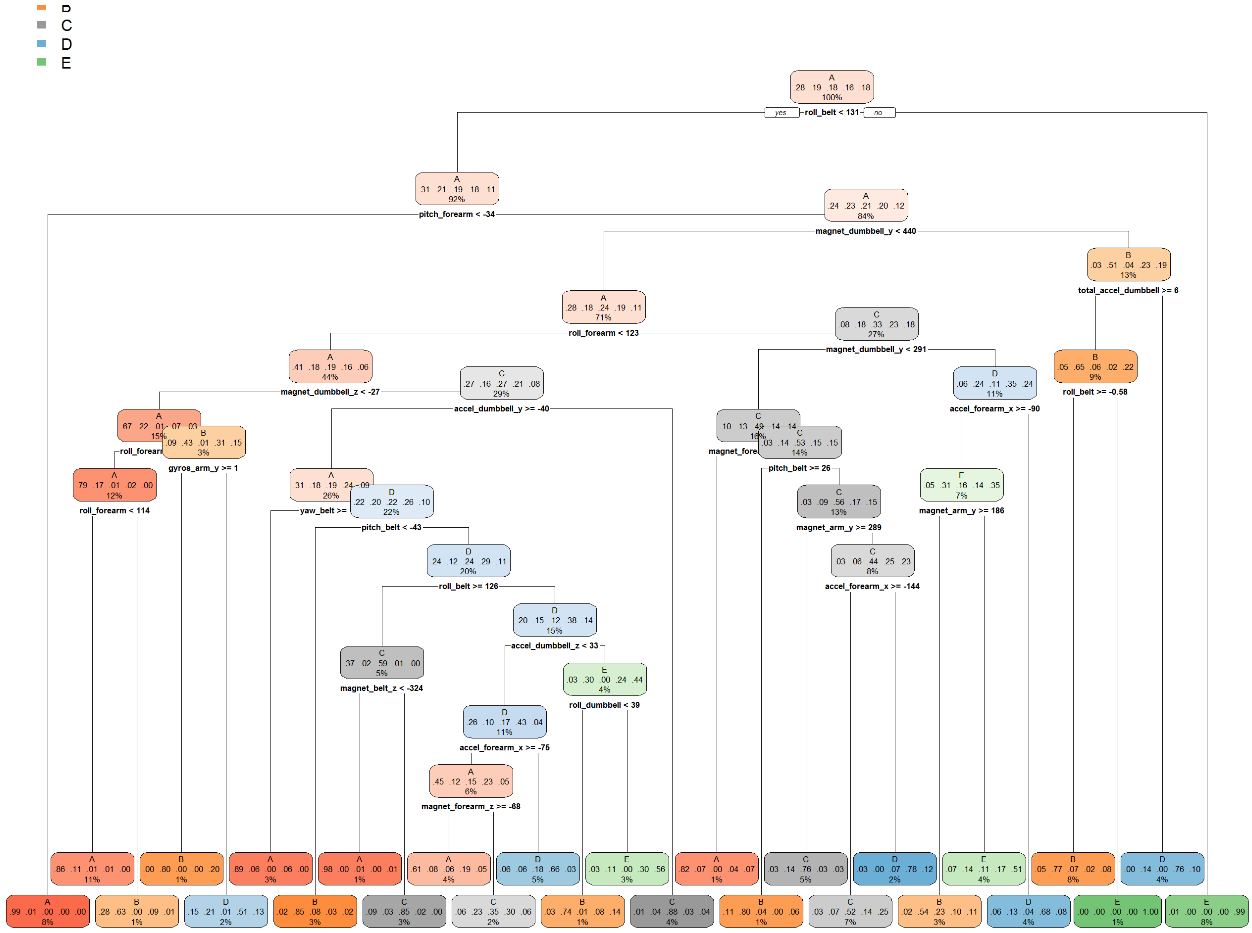
```
## [1] 0.7506431
```

```
plotcp(modFit2_pruned)
```



We choose modFit2_pruned to be our final model.

```
#fancyRpartPlot(modFit2_pruned, cex=0.15)
rpart.plot(modFit2_pruned,tweak=1.7)
```



The predictions on the testing sample are

```
testing$magnet_forearm_z <- as.numeric(testing$magnet_forearm_z)
predict(modFit2_pruned,newdata=testing, type="class")
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  B  A  B  D  A  C  D  D  A  A  C  B  C  A  E  E  A  D  D  B
## Levels: A B C D E
```

The expected out-of-sample error should be close to the error of the validation set (ca. 0.25).