ML Project Karishma Yadav

Karishma Yadav

5/29/2020

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. 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. This dataset consists of data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants for Human Activity Recognition(HAR) research which focuses on discriminating between different Weight lifting exercises. The exercises were performed by six male participants aged between 20-28 years, with little weight lifting experience. They were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in five different fashions:

1. Exactly according to the specification (Class A)
2. Throwing the elbows to the front (Class B)
3. Lifting the dumbbell only halfway (Class C)
4. Lowering the dumbbell only halfway (Class D)
5. Throwing the hips to the front (Class E)

Class A corresponds to the specified execution of the exercise, while the other 4 classes correspond to common mistakes. Participants were supervised by an experienced weight lifter to make sure the execution complied to the manner they were supposed to simulate.

DATA SETS: The training data for this project:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv (https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv)

The test data:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv (https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv)

LOADING AND EXPLORING THE DATA SETS:

```
library(caret);library(rpart);library(rpart.plot);library(RColorBrewer);library(rattle);library(e1071);library(ra
ndomForest); library(gbm)
```

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

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

```
## Loading required package: tibble
```

```
## Loading required package: bitops
```

```
## Rattle: A free graphical interface for data science with R.
## Version 5.4.0 Copyright (c) 2006-2020 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:rattle':
##
##     importance
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
## Loaded gbm 2.1.5
```

```
set.seed(111)
trainset <- read.csv("pml-training.csv", na.strings=c("NA","#DIV/0!", ""))
testset <- read.csv("pml-testing.csv", na.strings=c("NA","#DIV/0!", ""))
dim(trainset); dim(testset)
```

```
## [1] 19622   160
## [1]  20 160
```

Checking for NA's :

```
sum(is.na(trainset))
```

```
## [1] 1925102
```

```
sum(is.na(testset))
```

```
## [1] 2000
```

Removing NA's and unnecessary columns:

```
trainset <- trainset[,colSums(is.na(trainset))==0]
testset <- testset[, colSums(is.na(testset))==0]
trainset <- trainset[,-c(1:7)]
testset <- testset[,-c(1:7)]
dim(trainset); dim(testset)
```

```
## [1] 19622    53
```

```
## [1] 20 53
```

```
sum(is.na(trainset))
```

```
## [1] 0
```

```
sum(is.na(testset))
```

```
## [1] 0
```

Now that we have gotten rid of all the necessary columns, we can start cross validation of the data sets as follows: Partitioning the trainset dataset into sub samples of training (75%) and testing (25%) data sets. The out of sample error should be higher than the in sample error because the the model is based on the training set and will therefor most likely have a slightly worst performance on the testset. This will be shown further in the project.

```
sub_sample <- createDataPartition(y=trainset$classe, p=0.75, list=FALSE)
subtrain <- trainset[sub_sample,]
subtest <- trainset[-sub_sample,]
```

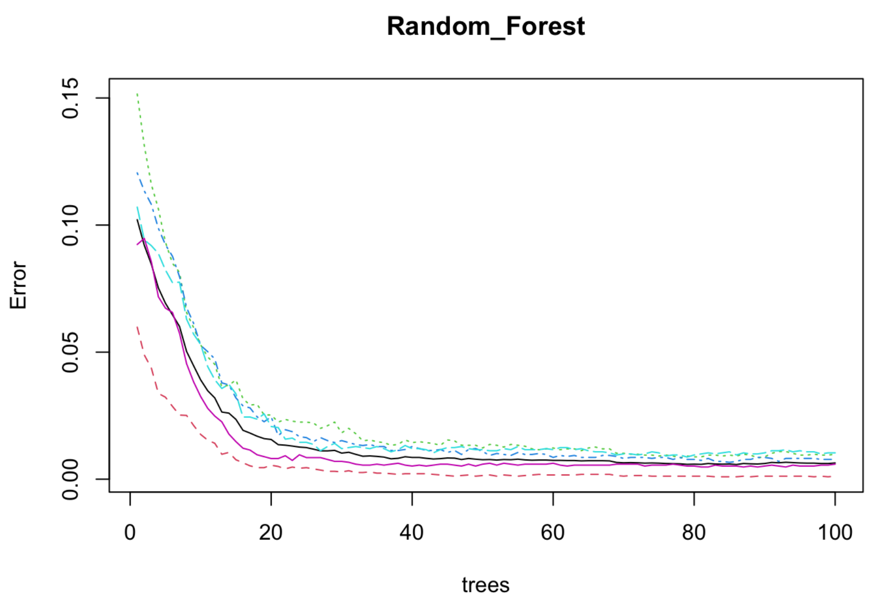
Modelling the data sets:

RANDOM FOREST MODEL:

```
Random_Forest <- randomForest(factor(classe) ~ .,
                             data = subtrain,
                             ntree=100
                             , metric = "Accuracy"
                             , preProcess=c("center", "scale")
                             , trControl=trainControl(method = "cv"
                                                     , number = 4
                                                     , allowParallel = TRUE))
Random_Forest
```

```
##
## Call:
##  randomForest(formula = factor(classe) ~ ., data = subtrain, ntree = 100,      metric = "Accuracy", preProcess
= c("center", "scale"), trControl = trainControl(method = "cv",          number = 4, allowParallel = TRUE))
##                Type of random forest: classification
##                      Number of trees: 100
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 0.64%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 4180    4    0    0    1 0.001194743
## B   20 2820    8    0    0 0.009831461
## C    0   15 2547    5    0 0.007791196
## D    0    0   22 2387    3 0.010364842
## E    0    0    3   13 2690 0.005912786
```

```
plot(Random_Forest)
```



```
trainresults <- predict(Random_Forest, subtest)
trainacc <- sum(trainresults==subtest$classe)/length(trainresults)

paste("Accuracy on training set =",trainacc)
```

```
## [1] "Accuracy on training set = 0.995513866231648"
```

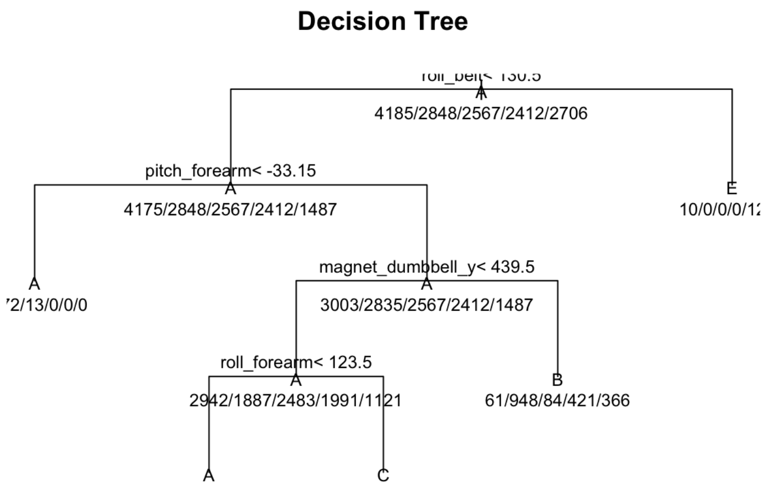
Results The accuracy of this model is 0.994 and the Overall Out-of-Sample error is 0.0051.

DECISION TREE VISUALIZATION:

```
treeModel <- train(classe~.,method="rpart",data=subtrain)
print(treeModel$finalModel)
```

```
## n= 14718
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 14718 10533 A (0.28 0.19 0.17 0.16 0.18)
##    2) roll_belt< 130.5 13489  9314 A (0.31 0.21 0.19 0.18 0.11)
##      4) pitch_forearm< -33.15 1185    13 A (0.99 0.011 0 0 0) *
##      5) pitch_forearm>=-33.15 12304  9301 A (0.24 0.23 0.21 0.2 0.12)
##       10) magnet_dumbbell_y< 439.5 10424  7482 A (0.28 0.18 0.24 0.19 0.11)
##         20) roll_forearm< 123.5 6495  3859 A (0.41 0.18 0.18 0.17 0.058) *
##         21) roll_forearm>=123.5 3929  2646 C (0.078 0.18 0.33 0.23 0.19) *
##       11) magnet_dumbbell_y>=439.5 1880   932 B (0.032 0.5 0.045 0.22 0.19) *
##    3) roll_belt>=130.5 1229    10 E (0.0081 0 0 0 0.99) *
```

```
plot(treeModel$finalModel,uniform=TRUE,main="Decision Tree")
text(treeModel$finalModel, use.n=TRUE,all=TRUE, cex=.8)
```


Decision Tree
roll_belt< 130.5
A
4185/2848/2567/2412/2706
pitch_forearm< -33.15
A
4175/2848/2567/2412/1487
E
10/0/0/0/12
A
'2/13/0/0/0
magnet_dumbbell_y< 439.5
A
3003/2835/2567/2412/1487
roll_forearm< 123.5
A
2942/1887/2483/1991/1121
B
61/948/84/421/366
A
C