Practical Machine Learning. Prediction assignment

V. Demydov

Sunday, May 24, 2015

Step 1. Loading data

```
setwd("D:/stud/courcera/pml")
trainData <- read.csv('pml-training.csv')
testData <- read.csv('pml-testing.csv')
```

Outcome field is "Classe", so lets study it

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

So we have classification task. There are 5 different classes. Anyway our test data is not really data for test, so we need divide train data to really train set and test set. Size of data is rather big so we could divide data 70% (train set) to 30% (test set)

```
set.seed(5765)
library(caret)
```

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

```
testIndex = createDataPartition(trainData$classe, p = 0.30,list=FALSE)
training = trainData[-testIndex,]
testing = trainData[testIndex,]
dim(training);
```

```
## [1] 13733   160
```

```
dim(testing)
```

```
## [1] 5889  160
```

There are so mny columns. Lets ignore columns with many NAs or empty data

```
trainSh <- training[,colSums(is.na(trainData[,]))<10000]
testSh <- testing[,colSums(is.na(trainData[,]))<10000]
toCalcSh <- testData[,colSums(is.na(trainData[,]))<10000]

trainSh1<-trainSh[,colSums(trainSh[,]=='')<10000]
testSh1<-testSh[,colSums(trainSh[,]=='')<10000]
toCalcSh1<-toCalcSh[,colSums(trainSh[,]=='')<10000]
```

Maybe timestamp is not very useful, testing examples could be done much later

```
trainSh2<-trainSh1[,-c(3:5)]
testSh2<-testSh1[,-c(3:5)]
toCalcSh2<-toCalcSh1[,-c(3:5)]
```

Ok, lets try to build tree

```
mod1 <- train(classe~.-X, method='rpart2', data=trainSh2)
```

```
## Loading required package: rpart
```

```
print(mod1$finalModel)
```

```
## n= 13733
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##   1) root 13733 9827 A (0.28 0.19 0.17 0.16 0.18)
##     2) roll_belt< 130.5 12597 8702 A (0.31 0.21 0.19 0.18 0.11)
##       4) pitch_forearm< -33.95 1104    8 A (0.99 0.0072 0 0 0) *
##       5) pitch_forearm>=-33.95 11493 8694 A (0.24 0.23 0.21 0.2 0.12)
##        10) num_window>=45.5 10978 8179 A (0.25 0.24 0.22 0.2 0.09)
##          20) magnet_dumbbell_y< 439.5 9393 6646 A (0.29 0.19 0.25 0.19 0.086)
##            40) roll_forearm< 122.5 5858 3396 A (0.42 0.18 0.19 0.16 0.046)
##              80) num_window< 241.5 1426  304 A (0.79 0.12 0.0014 0.06 0.032) *
##              81) num_window>=241.5 4432 3092 A (0.3 0.2 0.25 0.19 0.051)
##               162) magnet_dumbbell_z< -28.5 1310  312 A (0.76 0.18 0.016 0.037 0.0038) *
##               163) magnet_dumbbell_z>=-28.5 3122 2037 C (0.11 0.21 0.35 0.26 0.07) *
##            41) roll_forearm>=122.5 3535 2327 C (0.081 0.19 0.34 0.23 0.15)
##              82) magnet_dumbbell_y< 292.5 2122 1075 C (0.094 0.14 0.49 0.16 0.11)
##               164) num_window< 88.5 163   25 B (0.15 0.85 0 0 0) *
##               165) num_window>=88.5 1959  912 C (0.089 0.083 0.53 0.18 0.12) *
##              83) magnet_dumbbell_y>=292.5 1413  936 D (0.06 0.27 0.11 0.34 0.22)
##               166) accel_forearm_x>=-102.5 920  608 B (0.052 0.34 0.16 0.15 0.3) *
##               167) accel_forearm_x< -102.5 493  152 D (0.075 0.14 0.028 0.69 0.067) *
##          21) magnet_dumbbell_y>=439.5 1585  695 B (0.033 0.56 0.05 0.24 0.11)
##            42) total_accel_dumbbell>=5.5 1109  287 B (0.047 0.74 0.07 0.025 0.12) *
##            43) total_accel_dumbbell< 5.5 476  118 D (0 0.14 0.0021 0.75 0.1) *
##        11) num_window< 45.5 515  104 E (0 0 0 0.2 0.8) *
##     3) roll_belt>=130.5 1136   11 E (0.0097 0 0 0 0.99) *
```

lets see prediction tables for train and test datasets:

```
table(predict(mod1,trainSh2),trainSh2$classe)
```

```
##
##        A    B    C    D    E
##   A 3216  418   23  133   50
##   B  125 1272  225  164  406
##   C  517  831 2132 1151  450
##   D   37  136   15  699   82
##   E   11    0    0  104 1536
```

```
table(predict(mod1,testSh2),testSh2$classe)
```

```
##
##        A    B    C    D    E
##   A 1391  187   12   76   23
##   B   63  531   97   76  180
##   C  201  353  908  467  170
##   D   16   69   10  304   42
##   E    3    0    0   42  668
```

So accuracy on train and test datasets is

```
sum(predict(mod1,trainSh2)==trainSh2$classe)/length(trainSh2$classe)
```

```
## [1] 0.6447972
```

```
sum(predict(mod1,testSh2)==testSh2$classe)/length(testSh2$classe)
```

```
## [1] 0.6456105
```

And predicts for 20 test examples:

```
predict(mod1,toCalcSh2)
```

```
##  [1] B A B A A C D C A A C B C A B C A A A B
## Levels: A B C D E
```