ML Project

Raghuvir

2018-12-17

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. In this project, our goal is 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://groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

Objective

The goal of this project is to predict the manner in which they did the exercise. This is the "classe" variable in the training set. We will use the other variables to predict with.

Data import

The training data for this project are available here: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv

The test data are available here: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

The data for this project come from this source: http://groupware.les.inf.puc-rio.br/har.

```
rm(list=ls())
setwd("~/Desktop/Coursera/8. Project")
```

As the data has a lot of NA values, we are defining na.strings upon import

```
training<-read.csv(file="pml-training.csv", na.strings = c("NA","#DIV/0!", ""))
testing<-read.csv(file="pml-testing.csv", na.strings = c("NA","#DIV/0!", ""))

dim(training);dim(testing)
```

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

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

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

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

The training data has 19622 rows and testing data has 20 rows. There are 160 variables including the the classe, the result. In testing data classe, the result is replaced by problem ID. Our objective is to prepare a model based on training data, apply it to test data and derive the calsse and answer quiz questions.

Remove redundant columns/variables

```
#find data with missing values
NAvalues<-sapply(training, function(x) sum(is.na(x)))
summary(NAvalues[NAvalues>0])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   19216   19216   19216   19251   19225   19622
```

```
NAcolumns<-names(NAvalues[NAvalues>0])
length(NAcolumns)
```

```
## [1] 100
```

All columns that have NAs are irrelevant as number of rows with NAs are from 19216 to 19622, whereas total rows are 19622. Also columns 1:7 are not required as they don't represent useful variables but simply the data about the record.

```
training<-training[,!names(training) %in% NAcolumns]
training<-training[,-c(1:7)]
```

cross validation

As we want to measure accuracy of model before applying it to test data, we will split the training data in to 2 parts (75:25) to do cross validation.

```
set.seed(123)
library(caret)
```

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

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

```
intrain<-createDataPartition(y=training$classe,p=.75,list=FALSE)
trainingA<-training[intrain,]
trainingB<-training[-intrain,]
dim(trainingA)
```

```
## [1] 14718    53
```

```
dim(trainingB)
```

```
## [1] 4904   53
```

configure parrallel processing

These steps are taken from the following site where Len Greski explains how to improve system performance when making rf model. #https://github.com/lgreski/datasciencectacontent/blob/master/markdown/pml-randomForestPerformance.md

```
#install.packages("parallel")
library(parallel)
#install.packages("doParallel")
library(doParallel)
```

```
## Loading required package: foreach
```

```
## Loading required package: iterators
```

```
#cluster <- makeCluster(detectCores() - 1) # convention to leave 1 core for OS
#registerDoParallel(cluster)
```

The most critical arguments for the trainControl function are the resampling method method, the number that specifies the quantity of folds for k-fold cross-validation, and allowParallel which tells caret to use the cluster that we've registered in the previous step.

```
fitControl <- trainControl(method = "cv", number = 5,allowParallel = TRUE)
```

Develop training models

We use caret::train() to train the model, using the trainControl() object that we just created. We will use 2 models - random forests and Linear Discriminant Analysis.

```
x<-trainingA[,-53]
y<-trainingA[,"classe"]
modRF <- train(x, y, method="rf",data=trainingA,trControl = fitControl)
modLDA<-train(x,y, data=trainingA,method="lda", verbose=FALSE, trControl=fitControl)
```

Now that models are developed, we will de-register parallel processing cluster

```
#stopCluster(cluster)
#registerDoSEQ()
```

Prediction and out of sample errors

First we will do prediction off trainingB data using 2 models developed above

```
predRF<-predict(modRF,trainingB)
predLDA<-predict(modLDA,trainingB)
```

```
cfmRF <- confusionMatrix(trainingB$classe, predRF)
cfmLDA <- confusionMatrix(trainingB$classe, predLDA)
```

```
cfmRF$overall["Accuracy"]
```

```
## Accuracy
## 0.992863
```

```
cfmLDA$overall["Accuracy"]
```

```
##  Accuracy
## 0.7012643
```

The accuracy parameter above shows that the out of sample error is less than .01% for random forest method. Therefore, we will select this model and do prediction on testing data.

```
predRF<-predict(modRF,testing)
```