MachineLearning

Jason

5/24/2020

Introduction

The assignment's purpose is to analyze the data provided on personal fitness actvity, and determine how much of a particular activity people routinely perform and how well they perform it.

Data Analysis Steps

1. Summary of dataset

Six young participants performed various fitness workout activities, and thier performance is recorded in 5 classes of data (Class A, B, C, D, E). Class A refers to the specified execution of the excercise, while the rest correspond with occurances of mistakes.

2. Data Analysis

After loading the data into R, we can perform basic exploratory analysis on the data. There is a lot of missing values and NAs in this dataset, so the data need to be cleaned first using the below command:

```
setwd("~/R/MachineLearning")
a=read.csv('pml-training.csv',na.strings=c('','NA'))
b=a[,!apply(a,2,function(x) any(is.na(x)) )]
c=b[,-c(1:7)]
```

After cleaning the data, we have a total of 19622 data points with 53 predictors.

Then we prepare the needed packages for the analysis.

```
library('randomForest')
```

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

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

```
library('caret')
```

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

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

```
##
## Attaching package: 'ggplot2'

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

```
library('e1071')
```

In order to be able to conduct cross validation, the testing data is split into groups with 60% and 40% split using the following commands:

```
subGrps=createDataPartition(y=c$classe, p=0.6, list=FALSE)
subTraining=c[subGrps,]
subTesting=c[-subGrps, ]
dim(subTraining);dim(subTesting)
```

```
## [1] 11776    53
```

```
## [1] 7846   53
```

So after splitting the data, we have 11776 observations in the Training group, and 7846 observations in the testing group.

After this, random forest model is utlized to create a predictive model. The model is created using the Training group observations. Once the model is created, it can be used to predict values of the testing group, using the confusion matrix. The code is used below:

```
subTraining$classe = factor(subTraining$classe)
model=randomForest(classe~., data=subTraining, method='class')
pred=predict(model,subTesting, type='class')
z=confusionMatrix(factor(pred),factor(subTesting$classe))
save(z,file='test.RData')
```

The result is shown here:

```
setwd("~/R/MachineLearning")
load('test.RData')
z$table
```

```
##           Reference
## Prediction    A    B    C    D    E
##          A 2229    7    0    0    0
##          B    3 1506   11    0    0
##          C    0    5 1356   12    0
##          D    0    0    1 1274    7
##          E    0    0    0    0 1435
```

The summary of the model is here:

```
z
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2229    7    0    0    0
##          B    3 1506   11    0    0
##          C    0    5 1356   12    0
##          D    0    0    1 1274    7
##          E    0    0    0    0 1435
##
## Overall Statistics
##
##                Accuracy : 0.9941
##                  95% CI : (0.9922, 0.9957)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9926
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9987   0.9921   0.9912   0.9907   0.9951
## Specificity            0.9988   0.9978   0.9974   0.9988   1.0000
## Pos Pred Value         0.9969   0.9908   0.9876   0.9938   1.0000
## Neg Pred Value         0.9995   0.9981   0.9981   0.9982   0.9989
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2841   0.1919   0.1728   0.1624   0.1829
## Detection Prevalence   0.2850   0.1937   0.1750   0.1634   0.1829
## Balanced Accuracy      0.9987   0.9949   0.9943   0.9947   0.9976
```

Thus, the **accuracy** of the model is 99.16%, and the **out of sample error**, which is the error rate on the new testing set, is 0.99%. The **95% confidence interval** is between 0.98% to 0.99%

3. Prediction Since the random forest model is very accurate, the out of sample error is very small, we can successfully use this model to predict the actual testing data set.

First we need to process the testing data set:

```
d=read.csv('pml-testing.csv',na.strings=c('','NA'))
e=d[,!apply(d,2,function(x) any(is.na(x)) )]
f=e[,-c(1:7)]
```

After this, the predictive model is utilized on this testing dataset using the following code:

```
predicted=predict(model,f,type='class')
save(predicted,file='predicted.RData')
```

And the predictive result for the 2o test cases are shown below:

```
setwd("~/R/MachineLearning")
load('predicted.RData')
predicted
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  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
```