Coursera_Practical ML

durvesh

5/15/2021

#Assignment

#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. The aim of this project is to predict the manner in which participants perform a barbell lift. The data comes from http://groupware.les.inf.puc-rio.br/har (http://groupware.les.inf.puc-rio.br/har) wherein 6 participants were asked to perform the same set of exercises correctly and incorrectly with accelerometers placed on the belt, forearm, arm, and dumbell.

#For the purpose of this project, the following steps would be followed:

#1.Import the Data #2.Removing Columns that are not relevant #3.Using Random Forest #4.Predicting Test Set Output for 20 cases

```
library(caret)
```

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

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

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v tibble  3.1.1     v dplyr   1.0.5
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1
## v purrr   0.3.4
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x purrr::lift()   masks caret::lift()
```

```
library(future)
```

```
##
## Attaching package: 'future'
## The following object is masked from 'package:caret':
##
##     cluster
```

```
library(randomForest)
```

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

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

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

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

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

```
## Loading the training and the test data from the given url

train<-read.csv(url("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"))
test<-read.csv(url("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"))
```

```
# We check the size of the training and test sets

object.size(train)
```

```
## 22799176 bytes
```

```
object.size(test)
```

```
## 45056 bytes
```

```
# For the classification problem we find number of classes to predict
unique(train$classe)
```

```
## [1] "A" "B" "C" "D" "E"
```

```
train<-train[,-c(1:6)] # removing the id columns that are not relevant train
test<-test[,-c(1:6)]   # removing the id columns that are not relevant test

var<-nearZeroVar(train) # find the variables with zero variance
var<-var[-26] # retaining the classe variable
t1<-train %>% select(-all_of(var))
t2<-select_if(t1,function(x) sum(is.na(x)) <=1) #dropping columns that contain nas
```

```
plan(multisession)  # parallel processing to speed up
t2$classe <-as.factor(t2$classe) # converting the classification var to factor
model=randomForest(classe~., data=t2, method='class')
```

```
model # printing the results of the model
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = t2, method = "class")
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 0.14%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 5579    0    0    0    1 0.0001792115
## B    3 3793    1    0    0 0.0010534633
## C    0    5 3417    0    0 0.0014611338
## D    0    0   13 3202    1 0.0043532338
## E    0    0    0    4 3603 0.0011089548
```

```
vec_names<-names(t2)
vec_names<-vec_names[-length(vec_names)] # applying the same subset from train to test set colum
ns  removing the classe columns
test_1<-test %>% select(all_of(vec_names))
```

```
base_predict<-predict(model,test_1) # creating a prediction based on model
```

```
base_predict # printing out the results of prediction
```

```
##  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
```