Practical Machine Learning Peer Assessment - Prediction Assignment Writeup

Summary

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, your goal will be 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. The goal of this project is to predict the manner in which they did the execise.

The dataset used in this project can be found https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv.

Preparing

The datasets have been downloaded to local machine, and saved in the working directory. Below codes loaded the datasets and the necessary library.

```
library(caret)
```

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

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

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

Data Exploring

```
dim(training)
```

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

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

```
##
```

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

There are 19622 observation in traning dataset, including 160 variables. The last column is the target variable *classe*. The most abundant class is **A**.

There are some variables having a lot of missing values, for simplicity, I have removed all the variables containing **NA** values. And also, several variables are not direcly related to the target variable **classe**, I also removed those varialbes, those variables are “x”, “user_name”, and all the time related variables, such as “raw_timestamp_part_1” etc.

```
NA_Count = sapply(1:dim(training)[2],function(x)sum(is.na(training[,x])))
NA_list = which(NA_Count>0)
colnames(training[,c(1:7)])
## [1] "X"                    "user_name"            "raw_timestamp_part_1"
## [4] "raw_timestamp_part_2" "cvtd_timestamp"       "new_window"
## [7] "num_window"
training = training[,-NA_list]
training = training[,-c(1:7)]
training$classe = factor(training$classe)
```

The testing dataset has been processed in the same way

```
testing = testing[,-NA_list]
testing = testing[,-c(1:7)]
```

Modeling with Cross Validation

The problem presenting here is a classification problem, I tried to use the classification method in **caret** package: classification tree algorithm and random force. I also carried out 3-fold validation using the trainControl function.

```
set.seed(1234)
cv3 = trainControl(method="cv",number=3,allowParallel=TRUE,verboseIter=TRUE)
modrf = train(classe~., data=training, method="rf",trControl=cv3)
## Loading required package: randomForest
## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug fixes.
## + Fold1: mtry= 2
## - Fold1: mtry= 2
## + Fold1: mtry=27
## - Fold1: mtry=27
## + Fold1: mtry=52
## - Fold1: mtry=52
## + Fold2: mtry= 2
## - Fold2: mtry= 2
## + Fold2: mtry=27
## - Fold2: mtry=27
## + Fold2: mtry=52
## - Fold2: mtry=52
## + Fold3: mtry= 2
## - Fold3: mtry= 2
## + Fold3: mtry=27
## - Fold3: mtry=27
## + Fold3: mtry=52
## - Fold3: mtry=52
## Aggregating results
## Selecting tuning parameters
## Fitting mtry = 27 on full training set
modtree = train(classe~.,data=training,method="rpart",trControl=cv3)
## Loading required package: rpart
## + Fold1: cp=0.03568
## - Fold1: cp=0.03568
## + Fold2: cp=0.03568
## - Fold2: cp=0.03568
## + Fold3: cp=0.03568
## - Fold3: cp=0.03568
## Aggregating results
## Selecting tuning parameters
## Fitting cp = 0.0357 on full training set
```

We can check the performance of these two model on the testing dataset

```
prf=predict(modrf,training)
ptree=predict(modtree,training)
```

```
table(prf,training$classe)
##
## prf    A    B    C    D    E
##   A 5580    0    0    0    0
##   B    0 3797    0    0    0
##   C    0    0 3422    0    0
##   D    0    0    0 3216    0
##   E    0    0    0    0 3607
table(ptree,training$classe)
##
## ptree    A    B    C    D    E
##     A 5080 1581 1587 1449  524
##     B   81 1286  108  568  486
##     C  405  930 1727 1199  966
##     D    0    0    0    0    0
##     E   14    0    0    0 1631
```

For the testing dataset.

```
prf=predict(modrf,testing)
ptree=predict(modtree,testing)
table(prf,ptree)
##    ptree
## prf A B C D E
##   A 7 0 0 0 0
##   B 3 0 5 0 0
##   C 0 0 1 0 0
##   D 0 0 1 0 0
##   E 1 0 2 0 0
```

From the results, it appears that the random forest model has the best accuracy for testing dataset.

Conclusion

Finally, I chosed the random forest model to the testing dataset for submission result.

```
answers=predict(modrf,testing)
```

```
pml_write_files = function(x){
  n = length(x)
  for(i in 1:n){
    filename = paste0("problem_id_",i,".txt")
    write.table(x[i],file=filename,quote=FALSE,row.names=FALSE,col.names=FALSE)
  }
}
answers
```

```
##  [1] 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
```

```
pml_write_files(answers)
```

The predicted classes for the 20 tests are: B A B A A E D B A A B C B A E E A B B B.