Mini Project: A Prediction of Exercise Manners

Backgrounds

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, the goal will be to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants.

This report is going to:

- Find the pattern of these exercise behaviours in the dataset, using **random forest** algorithms.
- Examine the fitted model.
- Predict the 20 testing cases.

This report is written with these methods:

- Random Forest Model
- Cross Validation

System Initialization

Before we start, I need to go over some initialization process.

```
library(caret)    # Load the caret package for training data
```

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

```
setwd('E:/Rdir/predmachlearn')
set.seed(35443)   # Setting the random seed to 35443, in order to make the research reproducible
```

Getting and Cleaning Data

- Download the dataset from the given website.

```
dat <- read.table("./pml-training.csv", header = TRUE, sep = ",",  na.strings = c("","NA"))
```

- Subsetting the data into the cross validation sets. 70% observations will be set into the training set.

```
inTrain <- createDataPartition(dat$classe, p = 0.70, list = FALSE)
training <- dat[inTrain, ]
testing <- dat[-inTrain, ]
```

- In order to guarantee the training process is flexible, we gonna eliminate the variables containing NAs.

```
training <- training[, !colSums(is.na(training))]
testing <- testing[, !colSums(is.na(testing))]
```

- Testing the identity of the training and test set.

```
identical(names(training), names(testing))
```

```
## [1] TRUE
```

- Getting the dimension of the clean dataset

```
dim(training)
```

```
## [1] 13737    60
```

Model Fitting

This is the most important part of this report. We will train the dataset (training set of `pml_training.csv`, **13737 obs. of 60 variables**).
As you see, the training dataset is pretty large and the performance of my computer is pretty limited. It took me over an hour to train this model.
In order to make the research more efficient, I saved the fitted model into a file in the working directory (using `save` function). So in the next time I can directly read the file and load the model into my current environment, without the need for waiting that long.

```
if (sum(dir() == "modFit_rf.RData")) {    # To check if there's a file containing the model
        load(file = "./modFit_rf.RData")  # What if the file exists in my working dir?
        } else {
        modFit_rf <- train(classe ~ .,    # What if the file doesn't exist?
                           data = training[, -c(1,2)], method = "rf")
        save(list = c("modFit_rf"), file = "./modFit_rf.RData")
                                          # And save the model into file, make the research efficient.
}
```

The model training is accomplished, let's take a look at the model.

```
print(modFit_rf)
```

```
## Random Forest
##
## 13737 samples
##    57 predictors
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
##
## Summary of sample sizes: 13737, 13737, 13737, 13737, 13737, 13737, ...
##
```

```
## Resampling results across tuning parameters:
##
##   mtry  Accuracy  Kappa  Accuracy SD  Kappa SD
##   2     1         1      0.002        0.002
##   40    1         1      8e-04        0.001
##   80    1         1      0.001        0.001
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 38.
```

Examining into the Model

In this part, we will use the cross validation set to examine the model.

```
pred_rf <- predict(modFit_rf, testing)        # Predict the testing output with the fitted model.
```

```
## Loading required package: randomForest
## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug fixes.
```

```
confusionMatrix(pred_rf, testing$classe)      # To get the accuracy, sensitibility and specificity.
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    0    0    0    0
##          B    0 1139    1    0    0
##          C    0    0 1025    0    0
##          D    0    0    0  964    3
##          E    0    0    0    0 1079
##
## Overall Statistics
##
##                Accuracy : 0.999
##                  95% CI : (0.998, 1)
##     No Information Rate : 0.284
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.999
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity             1.000    1.000    0.999    1.000    0.997
## Specificity             1.000    1.000    1.000    0.999    1.000
## Pos Pred Value          1.000    0.999    1.000    0.997    1.000
## Neg Pred Value          1.000    1.000    1.000    1.000    0.999
## Prevalence              0.284    0.194    0.174    0.164    0.184
## Detection Rate          0.284    0.194    0.174    0.164    0.183
## Detection Prevalence    0.284    0.194    0.174    0.164    0.183
## Balanced Accuracy       1.000    1.000    1.000    1.000    0.999
```

We get the accuracy of this model is `0.993`, quite good. But is it likely to overfit the data? Let's examine the model with the 20 testing cases.

Predicting with 20 Cases

In this part, we will use the model to predict the 20 testing cases. Hopefully it'll be all right.

```
testing_cases_20 <- read.csv("./pml-testing.csv", header = TRUE, sep = ",",  na.strings = c("","NA"),st
answers <- predict(modFit_rf , testing_cases_20)
print(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
```

And next we will save these 20 answer into seperated file and submit it.

```
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)
        }
}
pml_write_files(answers)
```

The submit process is through the web.

References

[1] *Weight Lifting Exercise Dataset, Human Activity Recognition*, http://groupware.les.inf.puc-rio.br/har