Weight Lifting Exercise Prediction

Kiattisak Chaisomboon

26/6/2020

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, 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 (see the section on the Weight Lifting Exercise Dataset).

Approach

Our goal is to predict a type of incorrect manner in the weight lifting excercise. There are 5 different types such as
- Class A : exactly according to the specification
- Class B : throwing the elbows to the front
- Class C : lifting the dumbbell only halfway
- Class D : lowering the dumbbell only halfway
- Class E : throwing the hips to the front

We use a decision tree model and random forest model to test the accuracy. We'll choose a model with the highest accuracy to predict the incorrect manner in the test cases.

Data Preparation

Load and clean data

```
library(caret)
```

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

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

```
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:ggplot2':
```

```
##
##     margin
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

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

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
download.file('https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv', 'pml-training.csv
download.file('https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv', 'pml-testing.csv')

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

pml.training <- mutate(pml.training, classe=factor(classe))
dim(pml.training)
```

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

There are lots of missing data in many variables. We remove variables which Contain missing data more than 50%.

```
na.portion <- sapply(pml.training, function(x) sum(is.na(x))/nrow(pml.training))
select.feature <- pml.training[!(na.portion > 0.5)]
dim(select.feature)
```

```
## [1] 19622    60
```

We remove non-related variables from the dataset.

```
select.feature <- select(select.feature, -c('X', 'user_name', 'cvtd_timestamp', 'raw_timestamp_part_1',
dim(select.feature)
```

```
## [1] 19622    53
```

We split the dataset into training set and test set.

```
set.seed(33933)
idx.train <- createDataPartition(select.feature$classe, p=0.75, list=FALSE)
training <- select.feature[idx.train, ]
testing <- select.feature[-idx.train, ]
dim(training)
```

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

```
dim(testing)
```

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

Create 10-fold cross validation object to validate models.

```
train.control <- trainControl(method = "cv", number = 10)
```

Model 1 : Boosting

Create a boosting model and check a confusion matrix with test set.

```
set.seed(33933)
#gbm.modFit <- train(classe ~ ., data=training, method='gbm', trControl = train.control)
```

```
#confusionMatrix(testing$classe, predict(gbm.modFit, testing))
```

Model 2 : Random Forests

Create a random forest model and check a confusion matrix with test set.

```
set.seed(33933)
#rf.modFit <- randomForest(classe ~ ., data=training, trControl = train.control)
```

```
#confusionMatrix(testing$classe, predict(rf.modFit, testing))
```

Predict 20 types of incorrect manner

We'll choose the random forest model to predict 20 types of incorrect manner

```
#predict(rf.modFit, pml.testing)
```