Weight Lifting Exercises Prediction

Course project of "Practical Machine Learning"

Background

A large amount of data about personal activity can be collected using inexpensive device such as Jawbone Up, Nike FuelBand, etc. In this program, we did take measurements using accelerators which were put on the belt, forearm, arm and dumbbell of 6 participant athletes. They were asked independently from each other to perform barbell lifts in correct way (labeled as "A") or in mistaken ways to different degrees which got labeled manually from "B" to "E". For more information, see at http://groupware.les.inf.puc-rio.br/har.

Motivation

In this work, it is going to develop and train a predictive model such that given a data set of measurements it can predict the "correctness" class as outcome.

Data Preparation

Load original data for both training and test

```
library(randomForest)
```

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

```
library(caret)
```

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

```
training <- read.csv("pml-training.csv", header = TRUE, sep = ",")
test <- read.csv("pml-testing.csv", header = TRUE, sep = ",")

## remove those variables that have (almost) one unqiue value
nsv <- nearZeroVar(training)
training <- training[-nsv]
test <- test[-nsv]

## replace the NA data with 0
training[is.na(training)] <- 0
test[is.na(test)] <- 0
```

And in this work only use those variables with numeric values

```
num_features_idx <- which(lapply(training, class) %in% c("numeric"))
training <- cbind(training$classe, training[, num_features_idx])
test <- test[, num_features_idx]
names(training)[1] <- "classe"
```

Train the predictive model

We choose random forests model on training data for following reasons:

1. It uses multiple models for better performance than any single tree model.
2. It can provide importance about the predictor variables. That is useful to reduce the data dimensions. In this work, each observation has 160 variables and hopefully only small part of them could contribute to the prediction.

```
ptm <- proc.time()

fit.rf <- randomForest(classe ~ .,
                       data=training,
                       ntree = 500,
                       importance=T)

## How much time spent to train this model
proc.time() - ptm
```

```
##    user  system elapsed
##  100.27    0.14  100.59
```

```
imp <- importance(fit.rf, type = 1) # only the "mean decrease in accuracy"
impvar <- rownames(imp)[order(-imp[, 1])] # variable names
```

Base on their importance factor, rebuild the model with only the most 10 important variables as predictors

```
var10 <- paste( impvar[1:10], collapse = " + ")
f10 <- paste( "classe", var10, sep = " ~ " ) # formular in string
```

The compact formular for rebuild is

```
## [1] "classe ~ roll_belt + yaw_belt + pitch_belt + magnet_dumbbell_z + pitch_forearm + roll_dumbbell +
```

```
ptm <- proc.time()

# training new model with only the most 10 important predictor variables
fit.rf10 <- randomForest( as.formula(f10),
                       data=training,
                       ntree = 500,
                       importance=T)

## How much time spent to train this model
proc.time() - ptm
```

```
##    user  system elapsed
##   18.42    0.23   18.72
```

The training of new model takes much less time for its short predictor list.

Evaluation with test data

Now evaluate the model with loaded test data

```
test_answers <- predict(fit.rf10, test)
test_answers
```

```
##  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
```

The results have been verified as correct in the "Submission" section.

Conclusion

To the end, a predictive model with both fast speed and very good accuracy has been developed.