Prediction Assignment Writeup

Shail

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, 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 (http://groupware.les.inf.puc-rio.br/har) (see the section on the Weight Lifting Exercise Dataset).

Load and clean the data

Load the data and remove unneeded variable with user information, time and undefined

```
library(knitr)
library(caret)
```

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

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

```
library(e1071)
library(randomForest)
```

```
## randomForest 4.6-12
```

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

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

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

```
pmlTrain <- read.csv("pml-training.csv", na.strings = c("NA", ""))
pmlTest <- read.csv("pml-testing.csv", na.strings = c("NA", ""))

pmlTrain <- pmlTrain[, colSums(is.na(pmlTrain)) == 0]
pmlTest <- pmlTest[, colSums(is.na(pmlTest)) == 0]

pmlTrain <- pmlTrain[, -c(1:7)]
dim(pmlTrain)
```

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

```
pmlTest <- pmlTest[, -c(1:7)]
dim(pmlTest)
```

```
## [1] 20 53
```

Partition the data

```
inTrain<-createDataPartition(y=pmlTrain$classe, p=0.75,list=F)
training<-pmlTrain[inTrain,]
test<-pmlTrain[-inTrain,]
dim(training)
```

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

```
dim(test)
```

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

Prediction Process

Random forests is used for prediction process

```
control <- trainControl(method = "cv", number = 5)
fit_rf <- train(classe ~ ., data=training, method = "rf", trControl = control)
print(fit_rf, digits = 4)
```

```
## Random Forest
##
## 14718 samples
##    52 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 11776, 11774, 11775, 11773, 11774
## Resampling results across tuning parameters:
##
##   mtry  Accuracy  Kappa   Accuracy SD  Kappa SD
##    2    0.9908    0.9883  0.0021778    0.002756
##   27    0.9921    0.9899  0.0018363    0.002323
##   52    0.9882    0.9850  0.0009998    0.001266
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 27.
```

```
#Predict using test data
predict_rf <- predict(fit_rf, test)
conf_rf <-confusionMatrix(predict_rf, test$classe)

(accuracy_rf <- conf_rf$overall[1])
```

```
## Accuracy
##  0.99531
```

Result and Test Prediction

Predict the outcome variable classe for the testing set

```
(predict(fit_rf, pmlTest))
```

```
##  [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
```

Conclusion:

Random forests with accurancy rate of 0.994 is good model