Prediction Assignment Writeup

Angelina

11/19/2017

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://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har (http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har) (see the section on the Weight Lifting Exercise Dataset).

Data

The training data for this project are available here: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv (https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv)

The test data are available here: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv (https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv)

The data for this project come from this source: http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har (http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har). Thanks for their generoucity in allowing their data to be used for this kind of assignment.

Set libraries

This project is started with setting the required libraries.

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

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

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

```
library(rpart)
library(rpart.plot)
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
```

```
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
set.seed(12345)
```

Download datasets

The datasets are downloaded and read as training and testing data.

```
TrainData <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
TestData <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
training <- read.csv(url(TrainData))
testing <- read.csv(url(TestData))
```

Create training and validation set

In this step, the training data was split into two part: 70% for training and 30% for validation.

```
trainPart <- createDataPartition(training$classe, p =0.7, list=FALSE)
trainSet <- training[trainPart, ]
valSet <- training[-trainPart, ]
```

Cleaning data

Cleaning data includes removing near zero values, removing mostly NA variables, and remove identification columns.

```
#remove near zero data
nearZero <- nearZeroVar(trainSet)
trainSet <- trainSet[, -nearZero]
valSet <- valSet[, -nearZero]
dim(trainSet)
```

```
## [1] 13737   106
```

```
dim(valSet)
```

```
## [1] 5885  106
```

```
##remove mostly NA variables
allNA <- sapply(trainSet, function(x) mean(is.na(x))) > 0.95
trainSet <- trainSet[, allNA==FALSE]
valSet <- valSet[, allNA==FALSE]
dim(trainSet)
```

```
## [1] 13737    59
```

```
dim(valSet)
```

```
## [1] 5885   59
```

```
#remove column 1-5 containing identifications
trainSet <- trainSet[, -(1:5)]
valSet <- valSet[, -(1:5)]
dim(trainSet)
```

```
## [1] 13737    54
```

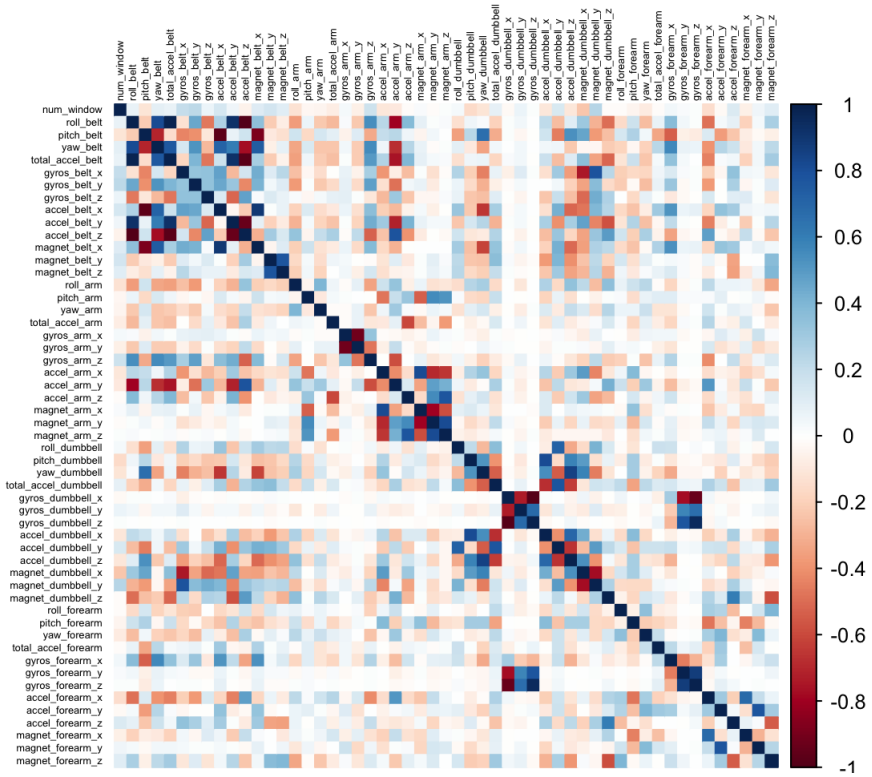
```
dim(valSet)
```

```
## [1] 5885   54
```

The dimensions on the final training and test set are reduced into 54 dimensions.

Correlation analysis

```
corMatrix <- cor(trainSet[, -54])
corrplot(corMatrix,method="color",tl.cex=0.4, tl.col="black")
```



Create prediction model with RandomForest

The prediction model is built with Random Forest with 3-fold cross-validation.

```
set.seed(12345)
controlRF <- trainControl(method="cv", number=3, verboseIter = FALSE)
randForest <- train(classe ~ ., data=trainSet, method="rf", trControl = controlRF)
randForest$finalModel
```

```
##
## Call:
##  randomForest(x = x, y = y, mtry = param$mtry)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 27
##
##         OOB estimate of  error rate: 0.2%
## Confusion matrix:
##      A    B    C    D    E  class.error
## A 3904    1    0    0    1 0.0005120328
## B    5 2652    1    0    0 0.0022573363
## C    0    5 2390    1    0 0.0025041736
## D    0    0    7 2245    0 0.0031083481
## E    0    1    0    5 2519 0.0023762376
```

Apply prediction on validation data

The prediction model is applied to the validation data to validate the accuracy of the model.

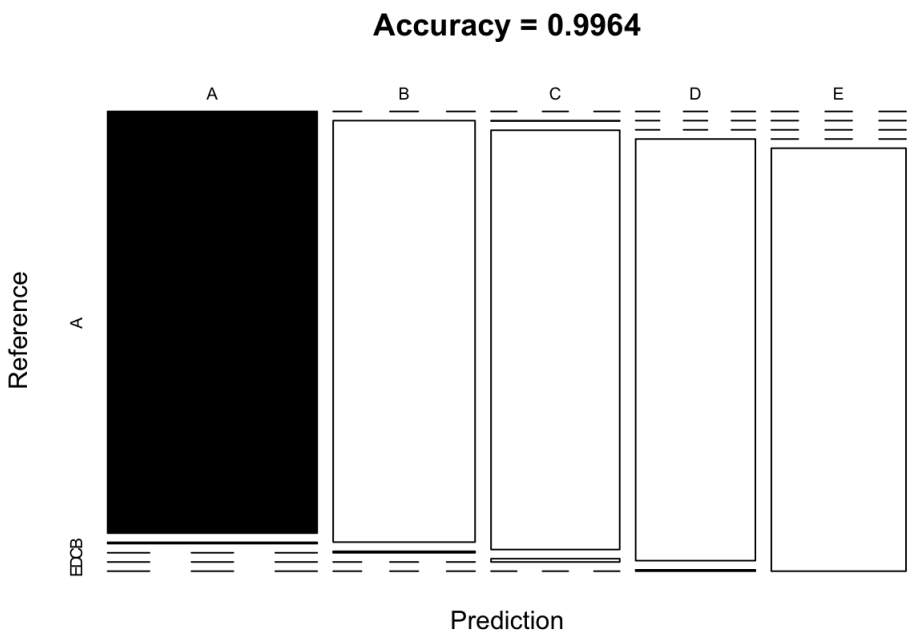
```
predictVal <- predict(randForest, newdata=valSet)
confMatrix <- confusionMatrix(predictVal, valSet$classe)
confMatrix
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1674    5    0    0    0
##          B    0 1133    4    0    0
##          C    0    1 1022    8    0
##          D    0    0    0  956    3
##          E    0    0    0    0 1079
##
## Overall Statistics
##
##                Accuracy : 0.9964
##                  95% CI : (0.9946, 0.9978)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9955
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9947   0.9961   0.9917   0.9972
## Specificity            0.9988   0.9992   0.9981   0.9994   1.0000
## Pos Pred Value         0.9970   0.9965   0.9913   0.9969   1.0000
## Neg Pred Value         1.0000   0.9987   0.9992   0.9984   0.9994
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2845   0.1925   0.1737   0.1624   0.1833
## Detection Prevalence   0.2853   0.1932   0.1752   0.1630   0.1833
## Balanced Accuracy      0.9994   0.9969   0.9971   0.9955   0.9986
```

As the accuracy is very high (0.9964) with 95% CI (0.9946, 0.9978), the model is representative and valid to be used for prediction purpose.

Plot matrix results

```
plot(confMatrix$table, col=confMatrix$byClass, main=paste("Accuracy =",
round(confMatrix$overall['Accuracy'], 4)))
```



Apply model to predict test data

```
predictTest <- predict(randForest, newdata=testing)
predictTest
```

```
##  [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 forest can be used to predict the class of exercises with a very high accuracy on the validation set. Creating validation set from a 30% split of the training set was helpful to validate the resulted model, before it is applied to test the test data.