Practical Machine Learning Course Project

Yuan C

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

Description for the Steps

We need to build a model to predict "classe" variable, which is a factor variable with 5 levels. We have a large size of training dataset (19622) and over 150 variable candidates. It allows us to use the cross-validation and pick the model with the highest accuracy. Considering the large size of candidates, we will drop all features with missing values and features which are irrelevant. All other features will be considered as relevant features. We will test both decision tree and random forest models and pick the best models by prediction accuracy. Both models are known for their ability to perform well for feature selection.

In order to perform cross validation for the model selection, we will break the traing set into sub-training set and sub-testing set with 75% and 25% of the total training set. We will use both model to fit the sub-training set and test on the sub-testing set and pick the model with the bigger accuracy.

The define our accuracy as the quotation between the correctly predicted "classe" variable in the sub-testing dataset and the total sample in the sub-testing dataset. Then we expect the real testing dataset to have the same accuracy. Hence, we define the expected out of sample error to be

```
1 - (correctly predicted classe in testing set)/(total size in testing set)
```

R Code and Output

Loading data and delete missing values

```r
#set seed in order to reproduce the result
set.seed(888)

#loading training and testing datasets and set all missing values to NA
training_data = read.csv("pml-training.csv",na.strings = c("NA","#DIV/0!",""))
testing_data = read.csv("pml-testing.csv",na.strings = c("NA","#DIV/0!",""))

#delete colums with missing values and irrelevant to the prediction variable
training_data = training_data[,colSums(is.na(training_data))==0]
testing_data = testing_data[,colSums(is.na(testing_data))==0]
training_data = training_data[,-c(1:7)]
testing_data = testing_data[,-c(1:7)]
```

Partition the training data into sub-training and sub-testing data set in order to do the cross validation In order to perform cross validation for the model selectio, we will break the traing set into sub-training set and sub-testing set with 75% and 25% of the total training data set.

```
#load the rpart library
library(caret)
```

```
## Loading required package: lattice

## Loading required package: ggplot2
```

```
#divide the training data into sub_training and sub_testing in order to do the cross validation
sub_data_partition <- createDataPartition(y=training_data$classe, p=0.75, list=FALSE)
sub_training <- training_data[sub_data_partition,]
sub_testing <- training_data[-sub_data_partition,]
```

Next we will test two methods, decision tree and random forest, on the sub_training and sub_testing data in order to make decision. The first prediction model is Decision Tree:

```
library(rpart)

#model fitting
model_dt <- rpart(classe ~ ., data = sub_training, method = "class")

#predicting on the sub_testing data
pred_dt <- predict(model_dt, sub_testing, type = "class")

#calculate a cross-tabulation of observed and predicted data with associated statistics on sub_testing
confusionMatrix(pred_dt, sub_testing$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1256  213   27   96   45
##          B   32  547   64   36   69
##          C   27   83  690  112   84
##          D   53   62   44  509   55
##          E   27   44   30   51  648
##
## Overall Statistics
##
##                Accuracy : 0.7443
##                  95% CI : (0.7318, 0.7565)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6747
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9004   0.5764   0.8070   0.6331   0.7192
## Specificity            0.8914   0.9492   0.9244   0.9478   0.9620
## Pos Pred Value         0.7673   0.7313   0.6928   0.7040   0.8100
## Neg Pred Value         0.9575   0.9033   0.9578   0.9294   0.9384
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2561   0.1115   0.1407   0.1038   0.1321
## Detection Prevalence   0.3338   0.1525   0.2031   0.1474   0.1631
## Balanced Accuracy      0.8959   0.7628   0.8657   0.7904   0.8406
```

We conclude the accuracy in this model is 0.739 with confidence interval (0.7318, 0.7565) The second prediction model is Random Forest

```
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
```

```
#model fitting
model_rf <- randomForest(classe ~ .,data = sub_training, method = "class")

#predicting
pred_rf <- predict(model_rf, sub_testing, type = "class")

#calculate a cross-tabulation of observed and predicted data with associated statistics on sub_testing
confusionMatrix(pred_rf, sub_testing$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1394    7    0    0    0
##          B    1  941    3    0    0
##          C    0    1  852    2    2
##          D    0    0    0  800    1
##          E    0    0    0    2  898
##
## Overall Statistics
##
##                Accuracy : 0.9961
##                  95% CI : (0.994, 0.9977)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9951
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9993   0.9916   0.9965   0.9950   0.9967
## Specificity            0.9980   0.9990   0.9988   0.9998   0.9995
## Pos Pred Value         0.9950   0.9958   0.9942   0.9988   0.9978
## Neg Pred Value         0.9997   0.9980   0.9993   0.9990   0.9993
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2843   0.1919   0.1737   0.1631   0.1831
## Detection Prevalence   0.2857   0.1927   0.1748   0.1633   0.1835
## Balanced Accuracy      0.9986   0.9953   0.9976   0.9974   0.9981
```

We conclude the accuracy in this model is 0.9961 with confidence interval (0.994, 0.9977) Hence, we choose the Random Forest as our final model as it's accuracy 99.61% is much better than the Decision Tree 73.9%. The expected out of sample error should be just 1-99.61% = 0.39%. **Finally we make prediction on our 20 testing dataset using the calibrated Random Forest model**

```r
#predict on the original testing dataset
pred_testing <- predict(model_rf, testing_data, type = "class")

#print out the predictions for the manner in which they do the exercise
pred_testing
```

```
##  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
```