Practical Machine Learning: Prediction Assignment Writeup

Masaaki Inaba

January 17, 2016

Contents

1. Executive Summary

1.1. 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).

1.2. Objectives

The goal of your project is to predict the manner in which they did the exercise. This is the "classe" variable in the training set. You may use any of the other variables to predict with. You should create a report describing:

- how you built your model, (Chapter 3. Model building)
- how you used cross validation, (Chapter 4.1. Create a confusion matrix)

- what you think the expected out of sample error is, and why you made the choices you did. (Chapter 4.2. Out of Sample Error)

You will also use your prediction model to predict 20 different test cases. (Chapter 5. Test)

2. Prepare library and data

```r
library(caret)
```

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

```r
library(randomForest)
```

```
## randomForest 4.6-12
## Type rfNews() to see new features/changes/bug fixes.
```

```r
# Read CSV Data
if (!exists('data.training')) {
  data.training = read.csv('./pml-training.csv', na.strings = c("#DIV/0!", "NA"))
}
if (!exists('data.testing')) {
  data.testing = read.csv('./pml-testing.csv', na.strings = c("#DIV/0!", "NA"))
}

# Omit NA column
effectiveColumns = sapply(data.training, function (vector) {
  sum(is.na(vector)) / length(vector) < 0.95
})
data.training.effective = data.training[, effectiveColumns]

# Omit irrelevant column
data.training.effective = data.training.effective[, -c(1:7)]

set.seed(178)
index.train = createDataPartition(y = data.training.effective$classe, p = 0.75, list = F)
training = data.training.effective[index.train,]
testing = data.training.effective[-index.train,]
```

3. Model Building

```r
ctrl_param = trainControl(method = "cv", number = 5, allowParallel = T, verbose = T)
model_rf = train(classe ~ ., data = training, method = "rf", trControl = ctrl_param, verbose = F)
```

```
## + Fold1: mtry= 2
## - Fold1: mtry= 2
## + Fold1: mtry=27
## - Fold1: mtry=27
## + Fold1: mtry=52
## - Fold1: mtry=52
## + Fold2: mtry= 2
## - Fold2: mtry= 2
## + Fold2: mtry=27
## - Fold2: mtry=27
## + Fold2: mtry=52
## - Fold2: mtry=52
## + Fold3: mtry= 2
## - Fold3: mtry= 2
## + Fold3: mtry=27
## - Fold3: mtry=27
## + Fold3: mtry=52
## - Fold3: mtry=52
## + Fold4: mtry= 2
## - Fold4: mtry= 2
## + Fold4: mtry=27
## - Fold4: mtry=27
## + Fold4: mtry=52
## - Fold4: mtry=52
## + Fold5: mtry= 2
## - Fold5: mtry= 2
## + Fold5: mtry=27
## - Fold5: mtry=27
## + Fold5: mtry=52
## - Fold5: mtry=52
## Aggregating results
## Selecting tuning parameters
## Fitting mtry = 27 on full training set
```

```
model_rf
```

```
## 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, 11772, 11774, 11775, 11775
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa      Accuracy SD   Kappa SD
##    2    0.9918467  0.9896851  0.0028431312  0.003598213
##   27    0.9930017  0.9911470  0.0008870087  0.001122904
##   52    0.9848485  0.9808316  0.0030260587  0.003830675
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 27.
```

4. Cross Validation

4.1. Confusion Matrix

```
predict_rf = predict(model_rf, newdata = testing)
confusion_matrix = confusionMatrix(predict_rf, testing$classe)
confusion_matrix
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1394    6    0    0    0
##          B    1  941    4    0    1
##          C    0    2  847   22    4
##          D    0    0    4  782    2
##          E    0    0    0    0  894
##
## Overall Statistics
##
##                Accuracy : 0.9906
##                  95% CI : (0.9875, 0.9931)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9881
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9993   0.9916   0.9906   0.9726   0.9922
## Specificity            0.9983   0.9985   0.9931   0.9985   1.0000
## Pos Pred Value         0.9957   0.9937   0.9680   0.9924   1.0000
## Neg Pred Value         0.9997   0.9980   0.9980   0.9947   0.9983
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2843   0.1919   0.1727   0.1595   0.1823
## Detection Prevalence   0.2855   0.1931   0.1784   0.1607   0.1823
## Balanced Accuracy      0.9988   0.9950   0.9919   0.9856   0.9961
```

4.2. Out of Sample Error

```
1 - confusion_matrix$overall[1]
```

```
##    Accuracy
## 0.009380098
```

5. Test

```r
predict(model_rf, data.testing)
```

```
##  [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
```