Practical Machine Learning Course Project Report

Jiahua Jiang

Feb 28, 2016

Script have been produced and tested on R 3.2.3 and OSX El Capitan. This Document is the final project for Coursera “Practical Machine Learning” from Johns Hopkins University. Assignement Instructions, Background and Data sections are directly copied from course’s assignement page.

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

Data

You can also embed plots, for example: 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 testing data are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv (https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv)

If you use the document you create for this class for any purpose please cite them as they have been very generous in allowing their data to be used for this kind of assignment.

Preliminary Work

Reproduceability

An overall random number generator seed was set at 12345 for all code. Required packages such as caret, randomForest and rpart were downloaded and installed.

How to bulid the model

Our outcome variable is classe, a factor variable with 5 levels. For this data set, participants were asked to perform one set of 10 reprtitions of the Unilateral Dumbbell Biceps Curl in 5 different fashions:

Class A: exactly according to the specification

Class B: throwing the elbows to the front

Class C: lifting the dumbbell only halfway

Class D: lowering the dumbbell only halfway

Class E: throwing the hips to the front

Prediction evaluations are supposed to be based on maximizing the accuracy and minimizing the out-of-sample error. Two models will be tested using decision tree and random forest algorithms. The model with higher accuracy with will be chosen as our final model.

Cross-validation

In order to get more convincing result, cross-validation will be performed by subsampling our training data set randomly: subTraining(75%) and subTesting data(25%). The proposed two models will be fitted on the subTraining data set and tested on the subTesting data set. The more accurate one will be chosen to be tested on the original Testing data set.

Code and Results

Preprocessing

```
install.packages("caret")
install.packages("randomForest")
install.packages("rpart")
install.packages("rpart.plot")
```

Loading libraries:

```
library(caret)
library(randomForest)
library(rpart)
library(rpart.plot)
```

Setting the seed for reproduceability:

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

Loading data sets and preliminary cleaning:

```
# Loading the training data set into R replacing all missing with "NA":
trainingset <- read.csv("Downloads/pml-training.csv",na.strings=c("NA","#DIV/0!",
""))

# Loading the testing data set into R replacing all missing with "NA":
testingset <- read.csv("Downloads/pml-testing.csv",na.strings=c("NA","#DIV/0!",
""))

#Check dimensions for variables and observations:
dim(trainingset)
[1] 19622   160
dim(testingset)
[1]  20 160

# Delete columns with all missing values
trainingset<-trainingset[,colSums(is.na(trainingset)) == 0]
testingset <-testingset[,colSums(is.na(testingset)) == 0]

# Delete irrelevant variables:
names(trainingset)
names(testingset)
trainingset   <-trainingset[,-c(1:7)]
testingset <-testingset[,-c(1:7)]

# Check new data sets(long result will be hidden):
dim(trainingset)
[1] 19622   153
dim(testingset)
[1]  20 153
names(trainingset)
names(testingset)
```

Partitioning the training data set

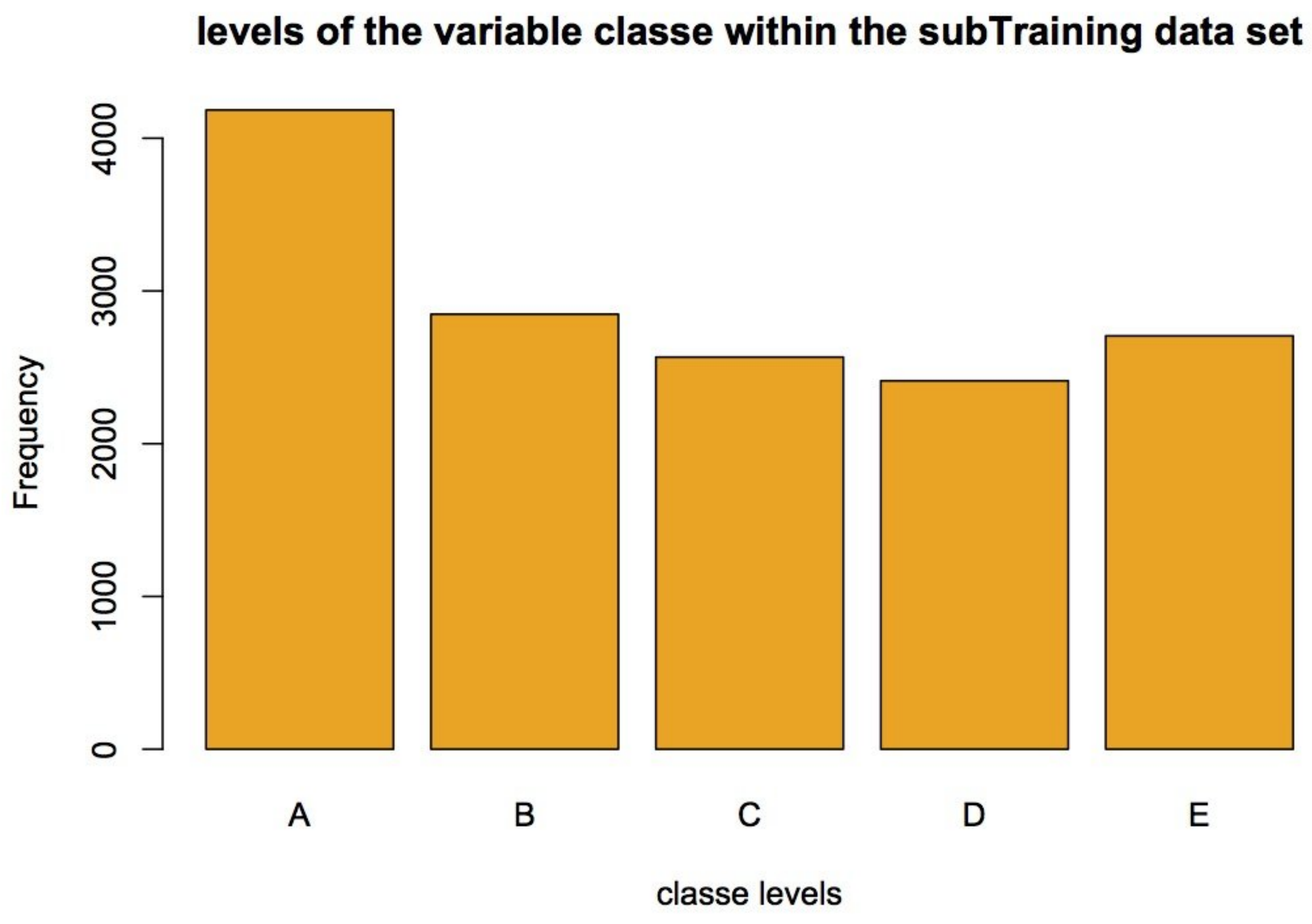
```
subsamples <- createDataPartition(y=trainingset$classe, p=0.75, list=FALSE)
subTraining <- trainingset[subsamples, ]
subTesting <- trainingset[-subsamples, ]

dim(subTraining)
[1] 14718   153
dim(subTesting)
[1] 4904  153
```

Plot of the data

The variable “classe” contains 5 levels mentioned above: A,B,C,D and E. We draw the picture of the data to see the frequency of each level in subTraining data set.

```
plot(subTraining$classe, col="orange", main="levels of the variable classe within
the subTraining data set", xlab="classe levels", ylab="Frequency")
```



From the graph above, we can see that level A is the most frequent while level D is the least frequent.

Model train and Validation

At first we will build models: decision tree and random forest based on the subTraining set and then test them on subTesting set. Finally, we will choose the model with better performance and implement it on Training set.

Model 1: Decision Tree

```
model1 <- rpart(classe ~ ., data=subTraining, method="class")

# Predicting:
pred1 <- predict(model1, subTesting, type = "class")

# Validation on subTesting data set:
confusionMatrix(pred1, subTesting$classe)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    A    B    C    D    E
         A 1260  156   33   40   23
         B   52  555   73   52   52
         C   24  136  575   83   95
         D   40   33  150  513   89
         E   19   69   24  116  642

Overall Statistics

               Accuracy : 0.7229
                 95% CI : (0.7101, 0.7354)
    No Information Rate : 0.2845
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.6486
 Mcnemar's Test P-Value : < 2.2e-16

Statistics by Class:

                     Class: A Class: B Class: C Class: D Class: E
Sensitivity            0.9032   0.5848   0.6725   0.6381   0.7125
Specificity            0.9282   0.9421   0.9165   0.9239   0.9430
Pos Pred Value         0.8333   0.7079   0.6298   0.6218   0.7379
Neg Pred Value         0.9602   0.9044   0.9298   0.9287   0.9358
Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
Detection Rate         0.2569   0.1132   0.1173   0.1046   0.1309
Detection Prevalence   0.3083   0.1599   0.1862   0.1682   0.1774
Balanced Accuracy      0.9157   0.7635   0.7945   0.7810   0.8278
```

Model 2: Random Forest

```
model2 <- randomForest(classe ~. , data=subTraining, method="class")

# Predicting:
pred2 <- predict(model2, subTesting, type = "class")


# Validation on subTesting data set:
confusionMatrix(pred2, subTesting$classe)
```

```
Confusion Matrix and Statistics

          Reference
Prediction    A    B    C    D    E
         A 1395    6    0    0    0
         B    0  939    2    0    0
         C    0    4  852    7    1
         D    0    0    1  797    4
         E    0    0    0    0  896

Overall Statistics

               Accuracy : 0.9949
                 95% CI : (0.9925, 0.9967)
    No Information Rate : 0.2845
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9936
 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: A Class: B Class: C Class: D Class: E
Sensitivity            1.0000   0.9895   0.9965   0.9913   0.9945
Specificity            0.9983   0.9995   0.9970   0.9988   1.0000
Pos Pred Value         0.9957   0.9979   0.9861   0.9938   1.0000
Neg Pred Value         1.0000   0.9975   0.9993   0.9983   0.9988
Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
Detection Rate         0.2845   0.1915   0.1737   0.1625   0.1827
Detection Prevalence   0.2857   0.1919   0.1762   0.1635   0.1827
Balanced Accuracy      0.9991   0.9945   0.9968   0.9950   0.9972
```

From the table of validation set accuracy, random forest algorithm(0.9949 accuracy) performed better than decision tree(0.7229 accuracy). Therefore, **random forest model is choosen**.

Test set prediction

```
predfinal <- predict(model2, testingset, type="class")
predfinal
```

```
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
```

```
answers <- as.vector(predfinal)

pml_write_files = function(x) {
    n = length(x)
    for (i in 1:n) {
        filename = paste0("problem_id_", i, ".txt")
        write.table(x[i], file = filename, quote = FALSE, row.names = FALSE,
            col.names = FALSE)
    }
}

pml_write_files(answers)
```