JH_8_PA(Course Project)

Takashi Sendo

2017/4/7

Coursera Practical Machine Learning Project

Introduction

The analysis uses the Weight Lifting Exercises dataset to investigate　predictions model(s) on “how (well)” an activity was performed by using data from belt, forearm, arm, and dumbbell monitors. There are five classifications of this exercise, one method is the correct form of the exercise while the other four are common mistakes: exactly according to the specification (Class A), throwing the elbows to the front (Class B), lifting the dumbbell only halfway (Class C), lowering the dumbbell only halfway (Class D) and throwing the hips to the front (Class E). Random forest and decision tree models will be tried and one with better accuracy will be selected to predict classes for 20 test cases.

prepartion and Dat Loading

packaging

```
packages <- c("caret","randomForest","rpart")
```

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.3.2
```

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

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

```
## Warning: package 'ggplot2' was built under R version 3.3.2
```

```
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(rpart)
library(e1071)
```

```
## Warning: package 'e1071' was built under R version 3.3.2
```

```
library(MASS)
```

Data Loading

```
## from local
training<-read.table("./pml-training.csv", header=TRUE, sep=",")
testing<-read.table("./pml-testing.csv",header=TRUE, sep=",")
```

Data Processing

```
#Delete columns with NA in testing_data
training_data <- training[, colSums(is.na(testing)) == 0]
testing_data  <- testing[, colSums(is.na(testing)) == 0]
#Delete some irrelevant variables: user_name, raw_timestamp_part_1, raw_timestamp_
part_,2 cvtd_timestamp, new_window, num_window
training_data  <- training_data[, -c(1:7)]
testing_data   <- testing_data[, -c(1:7)]
dim(training_data)
```

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

```
dim(testing_data)
```

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

Data Partition for Validation

Training data thus created are divided 3 to 1 between traning and validating of the model. Validating data will be used to select a better models (thus validating model) based upon training accuracy.

```
part_ind<- createDataPartition(y = training_data$classe, p = 0.75, list = FALSE,)
sub_training_data  <- training_data[part_ind,]
sub_validating_data   <- training_data[-part_ind,]
table(training_data$classe)
```

```
##
##    A    B    C    D    E
## 5580 3797 3422 3216 3607
```

Building Predictive Models

Random Forest, Decision Tree, SVM and lda models are first trained by the training data set, then accuracy will be compared using validating data. One with the best accuracy will be selected for predicting the 20 test cases. #### Random Forest Model

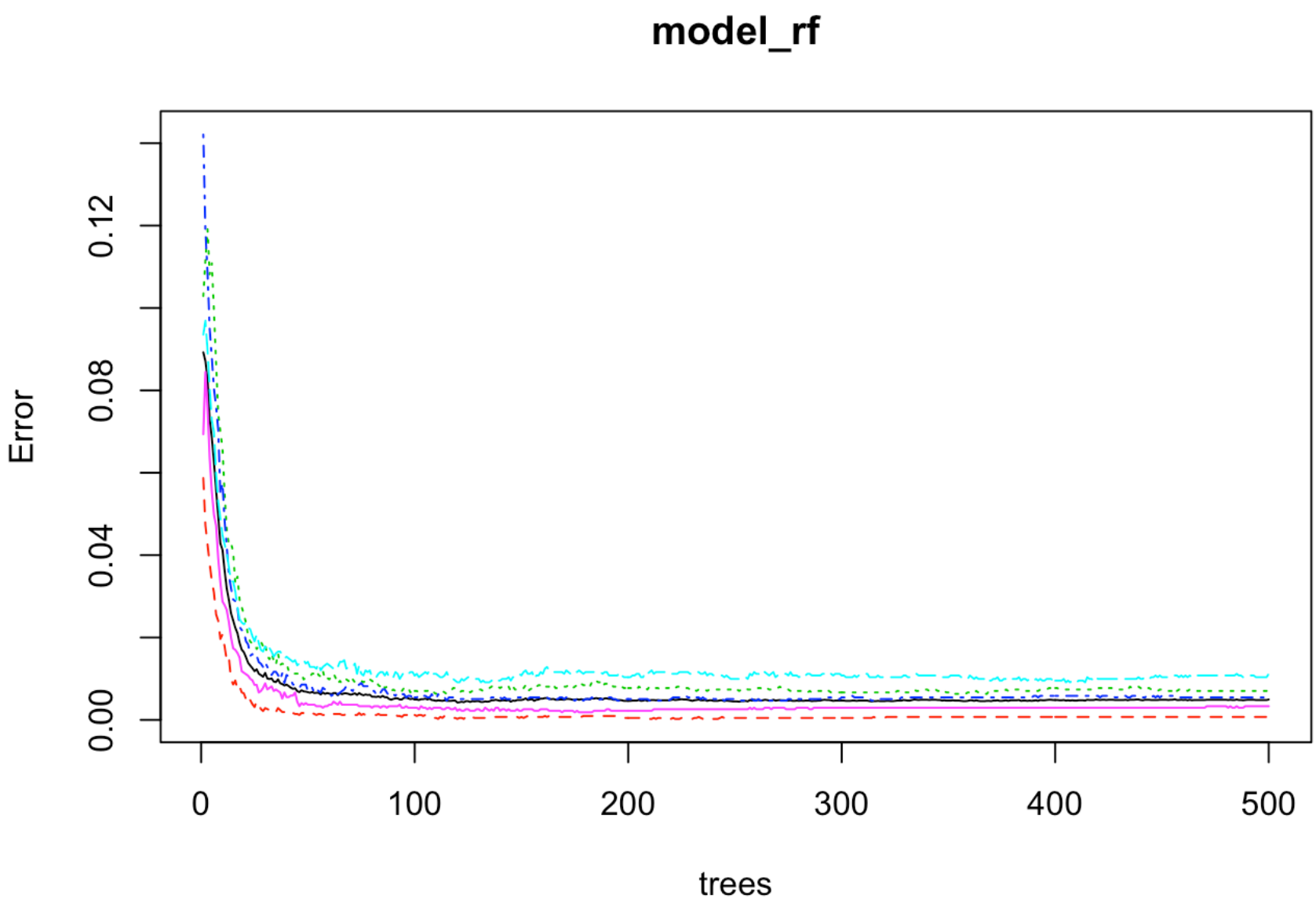
```
#Random Forest Model
model_rf <- randomForest(classe ~. , data = sub_training_data)
pred_rf  <- predict(model_rf, sub_validating_data)
res_rf    <- confusionMatrix(pred_rf, sub_validating_data$classe)
res_rf
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1394    3    0    1    0
##          B    0  946    8    0    0
##          C    0    0  843    8    0
##          D    0    0    4  795    1
##          E    1    0    0    0  900
##
## Overall Statistics
##
##                Accuracy : 0.9947
##                  95% CI : (0.9922, 0.9965)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9933
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9993   0.9968   0.9860   0.9888   0.9989
## Specificity            0.9989   0.9980   0.9980   0.9988   0.9998
## Pos Pred Value         0.9971   0.9916   0.9906   0.9938   0.9989
## Neg Pred Value         0.9997   0.9992   0.9970   0.9978   0.9998
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2843   0.1929   0.1719   0.1621   0.1835
## Detection Prevalence   0.2851   0.1945   0.1735   0.1631   0.1837
## Balanced Accuracy      0.9991   0.9974   0.9920   0.9938   0.9993
```

```
res_rf$overall[1]
```

```
##  Accuracy
## 0.9946982
```

```
plot(model_rf)
```



Decision Tree Model

```
#Decision Tree Model
model_dt  <- rpart(classe ~ ., data = sub_training_data,method = "class")
pred_dt   <- predict(model_dt, sub_validating_data,type = "class")
## (converting prediciton to the best fitted class)
res_dt   <- confusionMatrix(pred_dt, sub_validating_data$classe)
res_dt
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1257  167   46   87   30
##          B   54  607   78   68   85
##          C   36   68  666  134  110
##          D   20   63   55  437   32
##          E   28   44   10   78  644
##
## Overall Statistics
##
##                Accuracy : 0.7363
##                  95% CI : (0.7238, 0.7486)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6647
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9011   0.6396   0.7789  0.54353   0.7148
## Specificity            0.9060   0.9279   0.9141  0.95854   0.9600
## Pos Pred Value         0.7921   0.6805   0.6568  0.71993   0.8010
## Neg Pred Value         0.9584   0.9148   0.9514  0.91459   0.9373
## Prevalence             0.2845   0.1935   0.1743  0.16395   0.1837
## Detection Rate         0.2563   0.1238   0.1358  0.08911   0.1313
## Detection Prevalence   0.3236   0.1819   0.2068  0.12378   0.1639
## Balanced Accuracy      0.9035   0.7838   0.8465  0.75103   0.8374
```

```
res_dt$overall[1]
```

```
##  Accuracy
## 0.7363377
```

Support Vector Machine (uses library(e1071))

```
model_svm <-svm(classe ~ ., data = sub_training_data)
pred_svm <- predict(model_svm, sub_validating_data)
res_svm<-confusionMatrix(pred_svm, sub_validating_data$classe)
res_svm$overall[1]
```

```
##  Accuracy
## 0.9490212
```

Linear Discriminate Analysis (lda) Model (using library(MASS))

```
model_lda<-train(classe ~ ., data = sub_training_data,method="lda")
pred_lda <- predict(model_lda, sub_validating_data)
res_lda<-confusionMatrix(pred_lda, sub_validating_data$classe)
res_lda$overall[1]
```

```
##  Accuracy
## 0.6980016
```

Model selections

Random Forest gives the best accuracy among the models tested so that Random Forest is selected. ### Prediction Predicting on the test case using the Random Forest Model

```
final_pred <- predict(model_rf, testing_data)
final_pred
```

```
##  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
```