Assignment: Prediction Assignment Writeup

Najiha

March 29, 2016

Introduction

Objective

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, how you used cross validation, what you think the expected out of sample error is, and why you made the choices you did. You will also use your prediction model to predict 20 different test cases.

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

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://groupware.les.inf.puc-rio.br/har (http://groupware.les.inf.puc-rio.br/har). 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.

Load Data

```
library(caret)
```

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

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

```
library(randomForest)
```

```
## randomForest 4.6-12
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
library(rpart) # Regressive Partitioning and Regression trees
library(rpart.plot) # Decision Tree plot
library(RColorBrewer)

set.seed(1234)

training = data.frame(read.csv("/Users/najiha/Desktop/mypersonal/cousera/quiz4-module8/pml-training.csv"))
testing = data.frame(read.csv("/Users/najiha/Desktop/mypersonal/cousera/quiz4-module8/pml-testing.csv"))
```

Partioning the training set into two (split data)

```
inTrain <- createDataPartition(y=training$classe, p=0.6, list=FALSE)
myTraining <- training[inTrain, ];
myTesting <- training[-inTrain, ]

dim(myTraining); dim(myTesting)
```

```
## [1] 11776   160
```

```
## [1] 7846  160
```

Cleaning the data

```
# Remove value using nearZeroVar().
nzv <- nearZeroVar(myTraining, saveMetrics=TRUE)
myTraining <- myTraining[,nzv$nzv==FALSE]

nzv<- nearZeroVar(myTesting,saveMetrics=TRUE)
myTesting <- myTesting[,nzv$nzv==FALSE]

# Remove NA value
myTraining <- myTraining[ , colSums(is.na(myTraining)) == 0]
myTesting <- myTesting[ , colSums(is.na(myTesting)) == 0]

# Remove first ID of variable
myTraining <- myTraining[c(-1)]
myTesting <- myTesting[c(-1)]
```

Random Forest

```
RFmodel  <- randomForest(classe ~. , data=myTraining, method="class")

# Predicting:
prediction1 <- predict(RFmodel, myTesting, type = "class")

# Test results on subTesting data set:
confusionMatrix(prediction1, myTesting$classe)
```

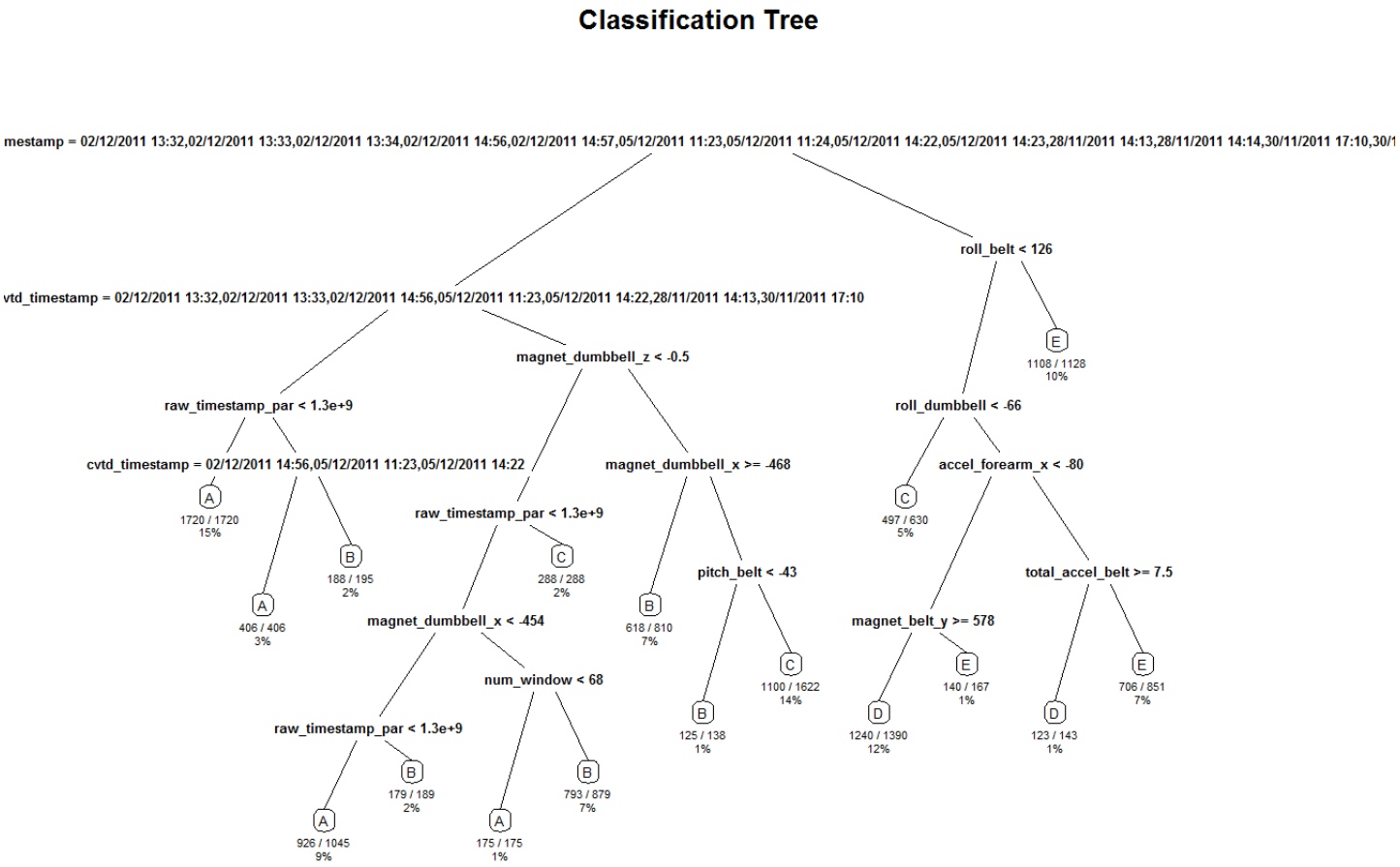
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2232    2    0    0    0
##          B    0 1516    5    0    0
##          C    0    0 1362    5    0
##          D    0    0    1 1280    1
##          E    0    0    0    1 1441
##
## Overall Statistics
##
##                Accuracy : 0.9981
##                  95% CI : (0.9968, 0.9989)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9976
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9987   0.9956   0.9953   0.9993
## Specificity            0.9996   0.9992   0.9992   0.9997   0.9998
## Pos Pred Value         0.9991   0.9967   0.9963   0.9984   0.9993
## Neg Pred Value         1.0000   0.9997   0.9991   0.9991   0.9998
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2845   0.1932   0.1736   0.1631   0.1837
## Detection Prevalence   0.2847   0.1939   0.1742   0.1634   0.1838
## Balanced Accuracy      0.9998   0.9989   0.9974   0.9975   0.9996
```

Decision Tree

```
# Fit model
DTmodel <- rpart(classe ~ ., data=myTraining, method="class")

# Perform prediction
prediction2 <- predict(DTmodel, myTesting, type = "class")

# Plot result
rpart.plot(DTmodel, main="Classification Tree", extra=102, under=TRUE, faclen=0)
```



```
confusionMatrix(prediction2, myTesting$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2161   61    5    3    0
##          B   50 1271   95   64    0
##          C   21  177 1242  203   65
##          D    0    9   19  899   92
##          E    0    0    7  117 1285
##
## Overall Statistics
##
##                Accuracy : 0.8741
##                  95% CI : (0.8665, 0.8813)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8407
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9682   0.8373   0.9079   0.6991   0.8911
## Specificity            0.9877   0.9670   0.9281   0.9817   0.9806
## Pos Pred Value         0.9691   0.8588   0.7272   0.8822   0.9120
## Neg Pred Value         0.9874   0.9612   0.9795   0.9433   0.9756
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2754   0.1620   0.1583   0.1146   0.1638
## Detection Prevalence   0.2842   0.1886   0.2177   0.1299   0.1796
## Balanced Accuracy      0.9779   0.9021   0.9180   0.8404   0.9359
```

Conclusion

Random Forest algorithm give better result than Decision Trees. Accuracy for Random Forest model was 95% CI : (0.9968, 0.9989) compared to 95% CI : (0.8665, 0.8813) for Decision Tree model. The random Forest model is choosen. The accuracy of the model is 0.997.