Prediction Assignment Writeup

Johnnery Aldana

15/3/2020

Introduction

The instructions for this project have been the following:

"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://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

The data have been the following:

The training data for this project are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv

The test data are available here:

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

The data for this project come from this source: http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har.

Step 1: Loading libraries

```
library(knitr)
library(caret)
library(rpart)
library(rpart.plot)
library(randomForest)
library(corrplot)
library(rattle)
```

Step 2: Loading and cleaning Data

Set the URL for the download

```
UrlTrain <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
UrlTest  <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
```

Download the datasets

```
training <- read.csv(url(UrlTrain))
testing  <- read.csv(url(UrlTest))
dim(training)
```

```
## [1] 19622   160
```

```
dim(testing)
```

```
## [1]  20 160
```

Removing Variables which are having Nearly Zero Variance.

```
nzv <- nearZeroVar(training)
training_data <- training[,-nzv]
testing_data <- testing[,-nzv]
dim(training_data)
```

```
## [1] 19622   100
```

```
dim(testing_data)
```

```
## [1]  20 100
```

Removing NA Values of Variables.

```
allNA <- sapply(training_data, function(x) mean(is.na(x))) > 0.95
training_data <- training_data[,allNA == FALSE]
testing_data <- testing_data[,allNA == FALSE]
dim(training_data)
```

```
## [1] 19622    59
```

```
dim(testing_data)
```

```
## [1] 20 59
```

Remove identification only variables (columns 1 to 6)

```
training_data<- training_data[, 7:59]
testing_data<- testing_data[, 7:59]
dim(training_data)
```

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

```
dim(testing_data)
```

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

Step 3: Prediction Model Building

I will use three model:

- Random Forest Model
- Decision trees with CART (rpart)

Data Partioning:

```
inTrain<- createDataPartition(training_data$classe, p=0.7, list=FALSE)
training<- training_data[inTrain,]
testing<- training_data[-inTrain,]
dim(training)
```

```
## [1] 13737    53
```

```
dim(testing)
```

```
## [1] 5885   53
```

Random Forest Model and Prediction:

```
set.seed(11111)

RF_model<- train(classe ~. , data=training, method= "rf", ntree=100)

RF_prediction<- predict(RF_model, testing)
RF_cm<-confusionMatrix(RF_prediction, testing$classe)
RF_cm
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1673   12    0    0    0
##          B    1 1126    7    1    0
##          C    0    1 1016   13    0
```

```
##          D    0    0    3  950    5
##          E    0    0    0    0 1077
##
## Overall Statistics
##
##                Accuracy : 0.9927
##                  95% CI : (0.9902, 0.9947)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9908
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9994   0.9886   0.9903   0.9855   0.9954
## Specificity            0.9972   0.9981   0.9971   0.9984   1.0000
## Pos Pred Value         0.9929   0.9921   0.9864   0.9916   1.0000
## Neg Pred Value         0.9998   0.9973   0.9979   0.9972   0.9990
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2843   0.1913   0.1726   0.1614   0.1830
## Detection Prevalence   0.2863   0.1929   0.1750   0.1628   0.1830
## Balanced Accuracy      0.9983   0.9933   0.9937   0.9919   0.9977
```

From the Random Forest Model we see the prediction accuracy is 99%.

Decision Tree Model and Prediction

```
#Fit the model and plot
set.seed(3333)
DT_model<- train(classe ~. , data=training, method= "rpart")

#Prediction
DT_prediction<- predict(DT_model, testing)
confusionMatrix(DT_prediction, testing$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   A   B   C   D   E
##          A 988 160  31  46  14
##          B  25 379  34 182 152
##          C 339 283 861 296 214
##          D 317 317 100 440 208
##          E   5   0   0   0 494
##
## Overall Statistics
##
##                Accuracy : 0.5373
```

```
##                  95% CI : (0.5245, 0.5501)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4228
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.5902   0.3327   0.8392  0.45643  0.45656
## Specificity            0.9404   0.9172   0.7670  0.80858  0.99896
## Pos Pred Value         0.7974   0.4909   0.4320  0.31838  0.98998
## Neg Pred Value         0.8523   0.8514   0.9576  0.88363  0.89083
## Prevalence             0.2845   0.1935   0.1743  0.16381  0.18386
## Detection Rate         0.1679   0.0644   0.1463  0.07477  0.08394
## Detection Prevalence   0.2105   0.1312   0.3387  0.23483  0.08479
## Balanced Accuracy      0.7653   0.6250   0.8031  0.63250  0.72776
```

From the Decision Tree Model we see the prediction accuracy is 50%

Step 4: Conclusion

We conclude that, Random Forest is more accurate than Decision Tree Model.

Step 5: Applying the Selected Model to the Test Data

```
prediction_test<- predict(RF_model, testing_data)
prediction_test
```

```
##  [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
```