Final Project on Machine Learning Course

Hari Venkatesh

April 08, 2020

Introduction

The main obejective of this project is to predict the manner in which people did exercise. 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.

Prerequisite

```
# load R - packages for ML algorithms
library(randomForest)
library(tidyverse)
library(caret)
```

Data Source

For the prediction analysis I collected two datasets from the below sources:

1) The training dataset: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv
2) The test dataset: https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

Data Loading and Shape

```
training = read.csv("./pml-training.csv",na.strings=c("NA","#DIV/0!",""))
testing = read.csv("./pml-testing.csv",na.strings=c("NA","#DIV/0!",""))

# Data shape
dim(training)
```

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

```
dim(testing)
```

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

```
table(training$classe)
```

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

```
# Take a look at the dataset
head(training[0:5])
```

```
##   X user_name raw_timestamp_part_1 raw_timestamp_part_2   cvtd_timestamp
## 1 1  carlitos           1323084231               788290 05/12/2011 11:23
## 2 2  carlitos           1323084231               808298 05/12/2011 11:23
## 3 3  carlitos           1323084231               820366 05/12/2011 11:23
## 4 4  carlitos           1323084232               120339 05/12/2011 11:23
## 5 5  carlitos           1323084232               196328 05/12/2011 11:23
## 6 6  carlitos           1323084232               304277 05/12/2011 11:23
```

```
#head(testing[0:5])
```

Predictors Selection

The slection of specific predictors is based on cleaning the near zero variance predictors and missing observations.

```
# remove NA columns for the training and testing data
comps <- complete.cases(t(training)) & complete.cases(t(testing))
traindata <- training[,comps]
testdata  <- testing[,comps]

# remove columns with data that isn't useful
traindata <- traindata[,-c(1,3,4,5,6,7)]
testdata <- testdata[,-c(1,3,4,5,6,7)]
```

Cross-Validation Analysis

Now, I perform the cross-validation by spliting the data into train and test set:

1) The train dataset consist of 70% of observations and
2) The test datset includes the rest 30% of observations. The evaulation of our trained models will testing using this dataset.

```
# data splitting
set.seed(12345)
inTrain <- createDataPartition(traindata$classe, p=0.7, list=FALSE)
traindata2 <- traindata[inTrain,]
testing.set <- traindata[-inTrain,]
```

Prediction Algorithms

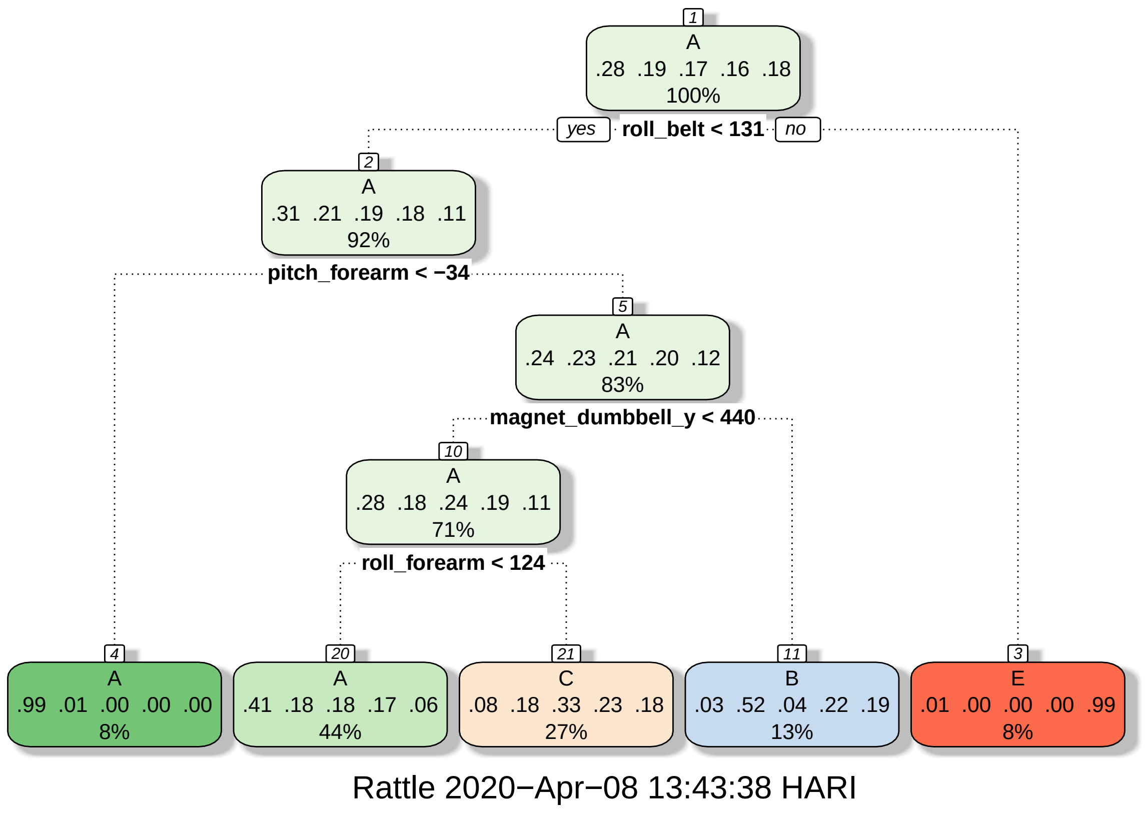
In this project I use two machine learning algorithms to predict the excercise.

1) Decision Treee and
2) Random Forest

1. Prediction with Decision Trees

```
# Build model
library(rpart)
set.seed(12345)
tree.fit = train(y = traindata2$classe,
                 x = traindata2[,-ncol(traindata2)],
                 method = "rpart")
```

```r
# Plot classification tree
rattle::fancyRpartPlot(
  tree.fit$finalModel
)
```



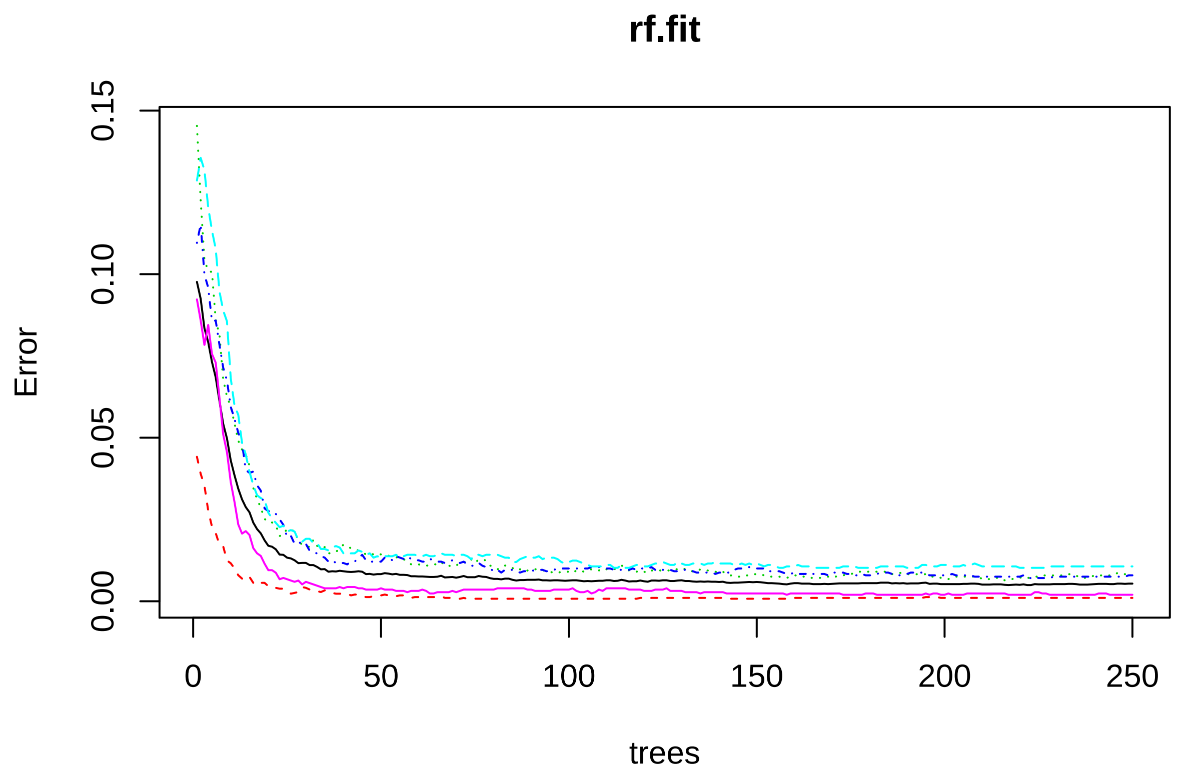
```r
# Predictions with rpart model
pred.tree = predict(tree.fit, testing.set[,-ncol(testing.set)])

# Get results (Accuracy, etc.)
confusionMatrix(pred.tree, testing.set$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1525  484  499  423  153
##          B   29  385   37  187  159
##          C  116  270  490  354  289
##          D    0    0    0    0    0
##          E    4    0    0    0  481
##
## Overall Statistics
##
##                Accuracy : 0.4895
##                  95% CI : (0.4767, 0.5024)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3324
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9110  0.33802  0.47758   0.0000  0.44455
## Specificity            0.6298  0.91319  0.78823   1.0000  0.99917
## Pos Pred Value         0.4945  0.48306  0.32258      NaN  0.99175
## Neg Pred Value         0.9468  0.85181  0.87723   0.8362  0.88870
## Prevalence             0.2845  0.19354  0.17434   0.1638  0.18386
## Detection Rate         0.2591  0.06542  0.08326   0.0000  0.08173
## Detection Prevalence   0.5240  0.13543  0.25811   0.0000  0.08241
## Balanced Accuracy      0.7704  0.62560  0.63291   0.5000  0.72186
```

2. Prediction with Random forest

Now I will use the random forest model.

```r
# Build model
set.seed(12345)
rf.fit = randomForest(
  classe ~ .,
  data = traindata2,
  ntree = 250)
# Plot the Random Forests model
plot(rf.fit)
```



```
# Predict with random forest model
pred2 = predict(
  rf.fit,
  testing.set[,-ncol(testing.set)]
)

# Get results (Accuracy, etc.)
confusionMatrix(pred2, testing.set$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1673    3    0    0    0
##          B    1 1136    3    0    0
##          C    0    0 1023   12    1
##          D    0    0    0  952    1
##          E    0    0    0    0 1080
##
## Overall Statistics
##
##                Accuracy : 0.9964
##                  95% CI : (0.9946, 0.9978)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
```

```
##                  Kappa : 0.9955
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9994   0.9974   0.9971   0.9876   0.9982
## Specificity            0.9993   0.9992   0.9973   0.9998   1.0000
## Pos Pred Value         0.9982   0.9965   0.9875   0.9990   1.0000
## Neg Pred Value         0.9998   0.9994   0.9994   0.9976   0.9996
## Prevalence             0.2845   0.1935   0.1743   0.1638   0.1839
## Detection Rate         0.2843   0.1930   0.1738   0.1618   0.1835
## Detection Prevalence   0.2848   0.1937   0.1760   0.1619   0.1835
## Balanced Accuracy      0.9993   0.9983   0.9972   0.9937   0.9991
```

Results Comparision

As we expected the random forest model is performed better than the decision tree. Moreover, the accuracy of the former model is better than latter. Therefore, we selected the random forest model to find the choice of 20 observations in original testing dataset.

```
# find the predictions of 20 observations from the original testing dataset

pred.validation = predict(rf.fit, testing)
pred.validation
```

```
##  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
```

Save the Prediction Results

```
testing$pred.classe = pred.validation

write.table(
  testing,
  file = "testing_with_predictions",
  quote = F
)
```

Reference

Ugulino, W.; Cardador, D.; Vega, K.; Velloso, E.; Milidiu, R.; Fuks, H. Wearable Computing: Accelerometers' Data Classification of Body Postures and Movements. Proceedings of 21st Brazilian Symposium on Artificial Intelligence. Advances in Artificial Intelligence - SBIA 2012. In: Lecture Notes in Computer Science. , pp. 52-61. Curitiba, PR: Springer Berlin / Heidelberg, 2012. ISBN 978-3-642-34458-9. DOI: 10.1007/978-3-642-34459-6_6.