Practical Machine Learning Project

Vinay Kowshik A

05/03/2017

Executive Summary

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, the 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).

Preliminary Setup

Loading the following libraries that will be using throughout this exercise

```
suppressWarnings(library(kernlab))
suppressWarnings(library(caret))
suppressWarnings(library(randomForest))
suppressWarnings(library(rpart))
suppressWarnings(library(rpart.plot))
suppressWarnings(library(rattle))
set.seed(123)
```

Data Load & Processing

The training data for this project is available here:

```
trainLink <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"
```

The testing data for this project is available here:

```
testLink <- "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"
```

Downloading the files and reading them

```
if(file.exists('Data/pml-training.csv')){
    trainingDF <- read.csv(file = 'Data/pml-training.csv',na.strings = c("NA","#DIV/0!",""))
}else {
    download.file(url = trainLink,destfile = 'Data/pml-training.csv',method = 'curl')
    trainingDF <- read.csv(file = 'Data/pml-training.csv',na.strings = c("NA","#DIV/0!",""))
}
if(file.exists('Data/pml-testing.csv')){
    testingDF <- read.csv(file = 'Data/pml-testing.csv',na.strings = c("NA","#DIV/0!",""))
}else {
    download.file(url = testLink,destfile = 'Data/pml-testing.csv',method = 'curl')
    testingDF <- read.csv(file = 'Data/pml-testing.csv',na.strings = c("NA","#DIV/0!",""))
}
```

Data Cleaning

Removing the 1st seven columns from the training data set as these will not be used in the prediction model.

```
trainingDF <- trainingDF[, -(1:7)]
```

Then removing all the fields that have a near zero variance.

```
nearzero <- nearZeroVar(trainingDF, saveMetrics = TRUE)
trainingDF <- trainingDF[, !nearzero$nzv]
```

Variables that have more than 60% missing values will be removed next.

```
myfunc <- function(x){
  if(sum(is.na(trainingDF[,x])) > 0.60*nrow(trainingDF)) {
      return (TRUE)
      } else
          {return (FALSE)
              }
}
missingVal <- sapply(colnames(trainingDF),myfunc)
trainingDF <- trainingDF[,!missingVal]

#These will be the variables that are selected for model building
names(trainingDF)
```

```
##  [1] "roll_belt"            "pitch_belt"           "yaw_belt"
##  [4] "total_accel_belt"     "gyros_belt_x"         "gyros_belt_y"
##  [7] "gyros_belt_z"         "accel_belt_x"         "accel_belt_y"
## [10] "accel_belt_z"         "magnet_belt_x"        "magnet_belt_y"
## [13] "magnet_belt_z"        "roll_arm"             "pitch_arm"
## [16] "yaw_arm"              "total_accel_arm"      "gyros_arm_x"
## [19] "gyros_arm_y"          "gyros_arm_z"          "accel_arm_x"
## [22] "accel_arm_y"          "accel_arm_z"          "magnet_arm_x"
## [25] "magnet_arm_y"         "magnet_arm_z"         "roll_dumbbell"
## [28] "pitch_dumbbell"       "yaw_dumbbell"         "total_accel_dumbbell"
## [31] "gyros_dumbbell_x"     "gyros_dumbbell_y"     "gyros_dumbbell_z"
## [34] "accel_dumbbell_x"     "accel_dumbbell_y"     "accel_dumbbell_z"
## [37] "magnet_dumbbell_x"    "magnet_dumbbell_y"    "magnet_dumbbell_z"
## [40] "roll_forearm"         "pitch_forearm"        "yaw_forearm"
## [43] "total_accel_forearm"  "gyros_forearm_x"      "gyros_forearm_y"
## [46] "gyros_forearm_z"      "accel_forearm_x"      "accel_forearm_y"
## [49] "accel_forearm_z"      "magnet_forearm_x"     "magnet_forearm_y"
## [52] "magnet_forearm_z"     "classe"
```

Prepare the testing data similar to the training set

```
testingDF <- testingDF[,-(1:7)]
testingDF <- testingDF[, !nearzero$nzv]
testingDF <- testingDF[,!missingVal]
```

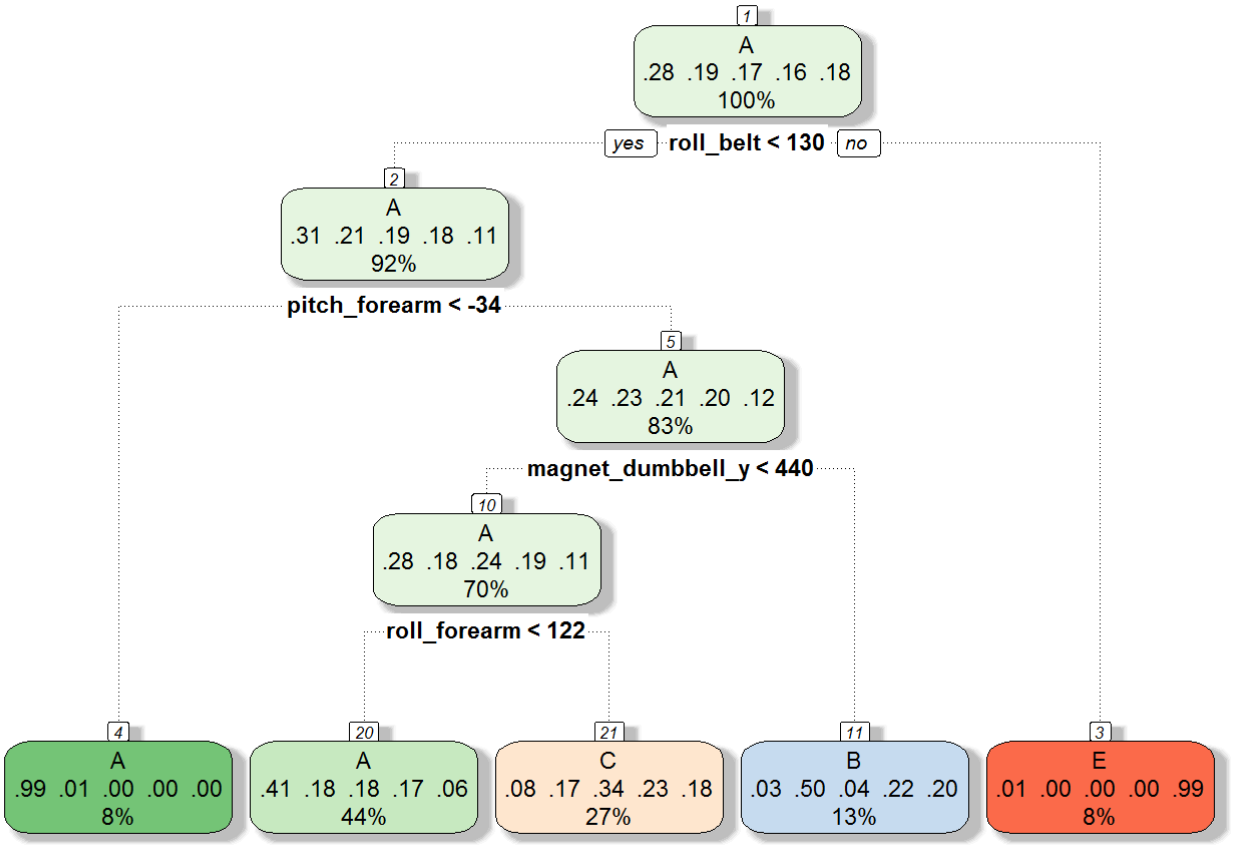
Splitting the training set into training and validation sets

```
inTrain <- createDataPartition(y=trainingDF$classe, p = 0.75,list = FALSE)
mytraining <- trainingDF[inTrain,]
myvalidate <- trainingDF[-inTrain,]
```

Prediction Model 1: Decision Tree

```
#Building the model on porting of the training set that was separated
model1 <- train(classe ~., data = mytraining,method = "rpart")

#Predicting the outcome on the validation set to test for accuracy
predict1 <- predict(model1,myvalidate)
fancyRpartPlot(model1$finalModel)
```



Rattle 2017-May-03 22:36:42 vadisheshan

```
#Looking at the confusion matrix informs that the prediction has a very high out of sample error
 rate
confusionMatrix(predict1,myvalidate$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1262  378  418  356  144
##          B   20  312   26  141  101
##          C  107  259  411  307  248
##          D    0    0    0    0    0
##          E    6    0    0    0  408
##
## Overall Statistics
##
##                Accuracy : 0.488
##                  95% CI : (0.4739, 0.5021)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3307
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9047  0.32877  0.48070   0.0000  0.45283
## Specificity            0.6307  0.92718  0.77254   1.0000  0.99850
## Pos Pred Value         0.4934  0.52000  0.30856      NaN  0.98551
## Neg Pred Value         0.9433  0.85200  0.87570   0.8361  0.89020
## Prevalence             0.2845  0.19352  0.17435   0.1639  0.18373
## Detection Rate         0.2573  0.06362  0.08381   0.0000  0.08320
## Detection Prevalence   0.5216  0.12235  0.27162   0.0000  0.08442
## Balanced Accuracy      0.7677  0.62797  0.62662   0.5000  0.72567
```

Prediction Model 2: Random Forest

```
#Building the model on porting of the training set that was separated
model2 <- randomForest(classe~.,data = mytraining)

#Predicting the outcome on the validation set to test for accuracy
predict2 <- predict(model2,myvalidate)

#Looking at the confusion matrix informs that the prediction has a very low out of sample error
 rate
confusionMatrix(predict2,myvalidate$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1394    1    0    0    0
##          B    1  945    8    0    0
##          C    0    3  846    9    0
##          D    0    0    1  793    1
##          E    0    0    0    2  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   0.9958   0.9895   0.9863   0.9989
## Specificity            0.9997   0.9977   0.9970   0.9995   0.9995
## Pos Pred Value         0.9993   0.9906   0.9860   0.9975   0.9978
## Neg Pred Value         0.9997   0.9990   0.9978   0.9973   0.9998
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2843   0.1927   0.1725   0.1617   0.1835
## Detection Prevalence   0.2845   0.1945   0.1750   0.1621   0.1839
## Balanced Accuracy      0.9995   0.9968   0.9933   0.9929   0.9992
```

Decision: Which prediction Model to use in this exercise

Random Forest algorithm performed better than Decision Trees. Accuracy for Random Forest model was 0.9947 (95% CI: (0.9922, 0.9965)) compared to Decision Tree model with 0.488 (95% CI: (0.4739, 0.5021)). The Random Forests model is choosen. The expected out-of-sample error is estimated at 0.0053, or 0.53%.

Final Submission

Below is the final outcome based on the Random Forest (model2) model applied to the Test data set:

```
predictFinal <- predict(model2,testingDF)
predictFinal
```

```
##  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
```