Practical Machine Learning

Robert Deng

7/11/2017

```
library(caret)
library(randomForest)
library(rattle)
```

```
## Warning: Failed to load RGtk2 dynamic library, attempting to install it.
```

```
library(rpart)
library(rpart.plot)
```

Background

Wearable technology (Fitbit, Nike FuelBand, and Jawbone Up) has made it possible to collect data about personal activity. People are measuring their own volume of activity, but rarely are they measuring quality. Training data was collected by participants to perform barbell lifts correctly and incorrectly in 5 different ways. Using data from accelerometers, the reseach question of interest is:

> *Can you learn and properly classify correct and incorrect workout form?*

More information is available from the website here: http://groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

Data Agenda:

-Load Data

-Clean Data

-Learn

-Predict

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

Most of the NA / #DIV/0 cleaning can be done within the read.csv funtion

```
training <- read.csv('https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv',
                  na.strings=c('#DIV/0!', '', 'NA') , stringsAsFactors = T)

testing <- read.csv('https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv',
                  na.strings=c('#DIV/0!', '', 'NA') ,stringsAsFactors = T)
```

Cross validation used to create a standard 60% / 40% split on the training data set. The testing data doesn't have a classe column to compare results with and is used in the final model evaluation phase.

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

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

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

Data cleaning

Near zero variance removes the columns with very little variance and columns with more than 70% NAs are removed. This is to keep the columns with more meaningful data. Also the X, id column 1 is removed.

```r
#Near Zero Variance
nzv <- nearZeroVar(myTraining, saveMetrics=TRUE)
myTraining <- myTraining[nzv$nzv==FALSE]
myTesting <- myTesting[nzv$nzv==FALSE]
myTraining <- myTraining[-1]
myTesting <- myTesting[-1]
testing <- testing[-1]

#More than 70% NAs
myTraining <- myTraining[, -which(colMeans(is.na(myTraining)) > 0.7)]
myTesting <- myTesting[, -which(colMeans(is.na(myTesting)) > 0.7)]
```

The myTraining and final testing datasets need the same data classes and features coerced together for the RandomForest model. I have a hard time figuring out why, but the RF model works on an appended testing set with 1 row from myTraining. Something about the familiarity with the training data.

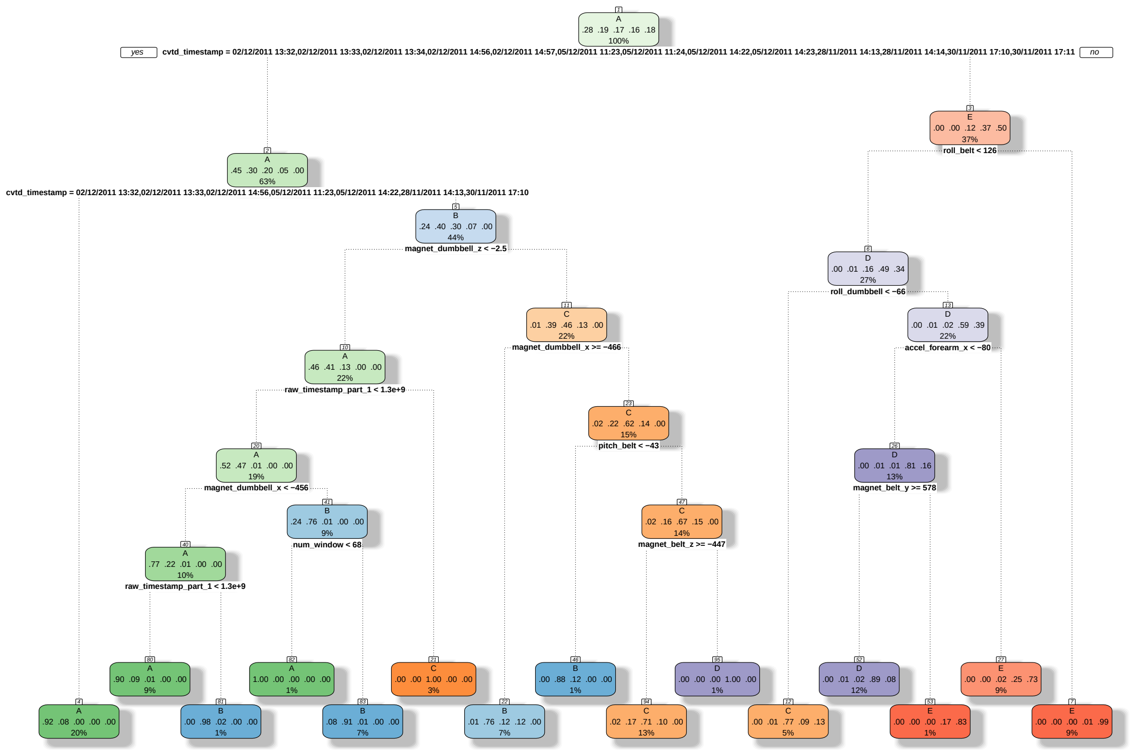
```r
#Match columns between myTraining and testing
col_match <- colnames(myTraining)[-58]
testing2 <- testing[col_match]

#Coerce the same datatypes for both testing and myTraining datasets for randomForest
for (i in 1:length(testing2) ) {
  for(j in 1:length(myTraining)) {
    if( length( grep(names(myTraining[i]), names(testing2)[j]) ) ==1)  {
      class(testing2[j]) <- class(myTraining[i])
    }
  }
}

#Rbind 1 from of myTraining to testing2
testing2 <- rbind(myTraining[2, -58] , testing2)
```

Now the fun part. First train for the rpart model using myTraining, then predict using myTesting.

```r
set.seed(98765)
model1 <- rpart(classe ~ ., data=myTraining, method="class")
fancyRpartPlot(model1)
```


yes
cvtd_timestamp = 02/12/2011 13:32,02/12/2011 13:33,02/12/2011 13:34,02/12/2011 14:56,02/12/2011 14:57,05/12/2011 11:23,05/12/2011 11:24,05/12/2011 14:22,05/12/2011 14:23,28/11/2011 14:13,28/11/2011 14:14,30/11/2011 17:10,30/11/2011 17:11
no
roll_belt < 126
cvtd_timestamp = 02/12/2011 13:32,02/12/2011 13:33,02/12/2011 14:56,05/12/2011 11:23,05/12/2011 14:22,28/11/2011 14:13,30/11/2011 17:10
magnet_dumbbell_z < -2.5
roll_dumbbell < -66
magnet_dumbbell_x >= -466
accel_forearm_x < -80
raw_timestamp_part_1 < 1.3e+9
pitch_belt < -43
magnet_belt_y >= 578
magnet_dumbbell_x < -456
num_window < 68
magnet_belt_z >= -447
raw_timestamp_part_1 < 1.3e+9

Rattle 2017−Jul−11 16:39:35 robo

Out of sample error is ~12% with most of the misclassifications on D.

```r
predResults1 <- predict(model1, myTesting, type = "class")
confusionMatrix(predResults1, myTesting$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2151  213   10    4    0
##          B   56 1116   83   56    0
##          C   25  180 1257  148   55
##          D    0    9   11  880   90
##          E    0    0    7  198 1297
##
## Overall Statistics
##
##                Accuracy : 0.8541
##                  95% CI : (0.8461, 0.8618)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8149
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9637   0.7352   0.9189   0.6843   0.8994
## Specificity            0.9596   0.9692   0.9370   0.9832   0.9680
## Pos Pred Value         0.9045   0.8513   0.7550   0.8889   0.8635
## Neg Pred Value         0.9852   0.9385   0.9820   0.9408   0.9771
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2742   0.1422   0.1602   0.1122   0.1653
## Detection Prevalence   0.3031   0.1671   0.2122   0.1262   0.1914
## Balanced Accuracy      0.9616   0.8522   0.9279   0.8338   0.9337
```

To validate, RandomForest yielded a 0.2% error rate. This outperformed the rpart model but could be slightly overfitting. RandomForest does a better job selecting the most important features via gini index and is more resilient to outliers.

```
model2 <- randomForest(classe ~. , data=myTraining, trControl = trainControl(method = "cv", 5))
predResults2 <- predict(model2, myTesting, type = "class")
confusionMatrix(predResults2, myTesting$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2232    1    0    0    0
##          B    0 1517    2    0    0
##          C    0    0 1366    2    0
##          D    0    0    0 1283    4
##          E    0    0    0    1 1438
##
## Overall Statistics
##
##                Accuracy : 0.9987
##                  95% CI : (0.9977, 0.9994)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9984
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9993   0.9985   0.9977   0.9972
## Specificity            0.9998   0.9997   0.9997   0.9994   0.9998
## Pos Pred Value         0.9996   0.9987   0.9985   0.9969   0.9993
## Neg Pred Value         1.0000   0.9998   0.9997   0.9995   0.9994
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2845   0.1933   0.1741   0.1635   0.1833
## Detection Prevalence   0.2846   0.1936   0.1744   0.1640   0.1834
## Balanced Accuracy      0.9999   0.9995   0.9991   0.9985   0.9985
```

Now for the final testing rounds:

```
predFinal1 <- predict(model1, testing2, type = "class")
predFinal1
```

```
##  2  1 21  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  A  B  A  C  A  A  E  D  C  A  A  B  C  B  A  E  E  A  B  B  B
## Levels: A B C D E
```

```
#Ignore the 31 (stitched from myTraining) in the model
predFinal2 <- predict(model2, testing2, type = "class")
predFinal2 <- predFinal2[-4]
predFinal2
```

```
##  2  1 21  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  A  B  A  A  A  E  D  B  A  A  B  C  B  A  E  E  A  B  B  B
## Levels: A B C D E
```

Write out results:

```
pml_write_files = function(x) {
    n = length(x)
    for (i in 1:n) {
        filename = paste0("problem_id_", i, ".txt")
        write.table(x[i], file = filename, quote = FALSE, row.names = FALSE,
            col.names = FALSE)
    }
}
pml_write_files(predFinal2)
```