Practical Machine Learning Project

Louis Jordan

Sunday, April 10, 2016

Executive Summary

Participants in the Human Activity Recognition (HAR) Project were asked to perform various exercises correctly and incorrectly in 5 different ways.Using performance data collected from accelerometers fed by multiple quantified self movement devices, the goal of this project is to train a model that could be used to predict the manner in which the participants performed the exercises.

The data for this project come from this source: http://groupware.les.inf.puc-rio.br/har.
The training data for this project are available here:
https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv
The test data for this project are available here:
https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

Preliminaries

Set Seed for Reproducibility
Load Required Libraries

```
print(currentDate <- date())
```

```
## [1] "Sat Apr 16 20:48:01 2016"
```

```
set.seed(212061996)
library(caret)
library(gmodels)
library(randomForest)
```

Load the Data Sets

```
trainChunk <- read.csv("pmlTrain.csv", header = TRUE, stringsAsFactors = FALSE,
                 sep = ",", na.strings = c("NA", "", "#DIV/0!"))

testChunk <- read.csv("pmlTest.csv", header = TRUE, stringsAsFactors = FALSE,
                 sep = ",", na.strings = c("NA", "", "#DIV/0!"))

trainChunk$classe <- as.factor(trainChunk$classe)
```

Preprocess (Examine & Clean) the Data

Remove missing values, irrelevant columns of data, and other items from the data set that do not contribute to the scope of the project

Examine the Data

```
# summary(trainChunk)
# str(trainChunk)
dim(trainChunk)

## [1] 19622   160
```

Remove RowID Column

```
removeIDCol <- trainChunk[, -1]
processedTrainChunk <- removeIDCol
dim(processedTrainChunk)

## [1] 19622   159
```

Find & Remove Missing Values

```
NAs <- apply(processedTrainChunk, 2, function(x) {sum(is.na(x))})
removeNAs <- processedTrainChunk[, which (NAs == 0)]
processedTrainChunk <- removeNAs
dim(processedTrainChunk)

## [1] 19622    59
```

Remove NZV Values

```
removeNZV <- nearZeroVar(processedTrainChunk)
processedTrainChunk <- processedTrainChunk[, -removeNZV]
dim(processedTrainChunk)

## [1] 19622    58
```

Find & Remove Useless Predictors (features)

```
uselessPredictors <- grep("cvtd_timestamp|X|user_name|num_window",
                          names (trainChunk))
processedTrainChunk <- processedTrainChunk[, -uselessPredictors]
dim(processedTrainChunk)

## [1] 19622    54
```

Define Data Partitions

Partition Training Data into Training and Validating Data Subsets

```r
inTrain <- createDataPartition(y = trainChunk$classe, p = 0.25, list = FALSE)
training <- processedTrainChunk[inTrain, ]
dim(training)
```

```
## [1] 4907   54
```

```r
# create test data for future use in cross validation
validating <- processedTrainChunk[-inTrain, ]
dim(validating)
```

```
## [1] 14715    54
```

Modeling

Fit the Model Using the Random Forest Algorithm (5-fold cross validation)

```r
ctrl <- trainControl(method = "cv", number = 5, allowParallel = TRUE)
myModel <- train(training$classe ~ ., data = training, method = "rf",
          prof = TRUE, trControl = ctrl)
myModel
```

```
## Random Forest
##
## 4907 samples
##   53 predictor
##    5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 3925, 3925, 3927, 3926, 3925
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9606698  0.9501931
##   36    0.9743223  0.9674952
##   71    0.9698371  0.9618124
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 36.
```

Evaluate the Model

```
cvPrediction <- predict(myModel, newdata = validating)
confusionMatrix(cvPrediction, validating$classe)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction    A    B    C    D    E
##          A 4180   35    0    0    0
##          B    3 2739   70    0    0
##          C    2   73 2469   28    6
##          D    0    0   27 2370   11
##          E    0    0    0   14 2688
## 
## Overall Statistics
## 
##                Accuracy : 0.9817
##                  95% CI : (0.9794, 0.9838)
##     No Information Rate : 0.2844
##     P-Value [Acc > NIR] : < 2.2e-16
## 
##                   Kappa : 0.9769
##  Mcnemar's Test P-Value : NA
## 
## Statistics by Class:
## 
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9988   0.9621   0.9622   0.9826   0.9937
## Specificity            0.9967   0.9938   0.9910   0.9969   0.9988
## Pos Pred Value         0.9917   0.9740   0.9577   0.9842   0.9948
## Neg Pred Value         0.9995   0.9909   0.9920   0.9966   0.9986
## Prevalence             0.2844   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2841   0.1861   0.1678   0.1611   0.1827
## Detection Prevalence   0.2864   0.1911   0.1752   0.1636   0.1836
## Balanced Accuracy      0.9977   0.9780   0.9766   0.9897   0.9963
```

```
cvPrediction <- predict(myModel, newdata = validating)
accuraccy <- c(as.numeric(cvPrediction == validating$classe))
accuraccy <- sum(accuraccy) * 100/nrow(validating)
oosError <- 100 - accuraccy
```

The CrossTable function of the gmodels package yields a more detailed confusion matrix.

```
CrossTable(cvPrediction, validating$classe)
```

```
## 
##    Cell Contents
## |-------------------------|
## |                       N |
## | Chi-square contribution |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
## 
## 
## Total Observations in Table:  14715 
## 
##              | validating$classe 
## cvPrediction |         A |         B |         C |         D |         E | Row Total | 
## -------------|-----------|-----------|-----------|-----------|-----------|-----------|
##            A |      4180 |        35 |         0 |         0 |         0 |      4215 | 
##              |  7414.138 |   747.004 |   735.011 |   690.899 |   774.827 |           | 
##              |     0.992 |     0.008 |     0.000 |     0.000 |     0.000 |     0.286 | 
##              |     0.999 |     0.012 |     0.000 |     0.000 |     0.000 |           | 
##              |     0.284 |     0.002 |     0.000 |     0.000 |     0.000 |           | 
## -------------|-----------|-----------|-----------|-----------|-----------|-----------|
##            B |         3 |      2738 |        70 |         0 |         0 |      2811 | 
##              |   793.470 |  8851.975 |   360.178 |   460.763 |   516.735 |           | 
##              |     0.001 |     0.974 |     0.025 |     0.000 |     0.000 |     0.191 | 
##              |     0.001 |     0.962 |     0.027 |     0.000 |     0.000 |           | 
##              |     0.000 |     0.186 |     0.005 |     0.000 |     0.000 |           | 
## -------------|-----------|-----------|-----------|-----------|-----------|-----------|
##            C |         2 |        74 |      2469 |        28 |         6 |      2579 | 
##              |   729.483 |   361.949 |  9066.567 |   368.590 |   462.163 |           | 
##              |     0.001 |     0.029 |     0.957 |     0.011 |     0.002 |     0.175 | 
##              |     0.000 |     0.026 |     0.962 |     0.012 |     0.002 |           | 
##              |     0.000 |     0.005 |     0.168 |     0.002 |     0.000 |           | 
## -------------|-----------|-----------|-----------|-----------|-----------|-----------|
##            D |         0 |         0 |        27 |      2370 |        11 |      2408 | 
##              |   684.844 |   465.890 |   367.643 |  9885.305 |   420.926 |           | 
##              |     0.000 |     0.000 |     0.011 |     0.984 |     0.005 |     0.164 | 
##              |     0.000 |     0.000 |     0.011 |     0.983 |     0.004 |           | 
##              |     0.000 |     0.000 |     0.002 |     0.161 |     0.001 |           | 
## -------------|-----------|-----------|-----------|-----------|-----------|-----------|
##            E |         0 |         0 |         0 |        14 |      2688 |      2702 | 
##              |   768.459 |   522.772 |   471.174 |   415.339 |  9667.455 |           | 
##              |     0.000 |     0.000 |     0.000 |     0.005 |     0.995 |     0.184 | 
##              |     0.000 |     0.000 |     0.000 |     0.006 |     0.994 |           | 
##              |     0.000 |     0.000 |     0.000 |     0.001 |     0.183 |           | 
## -------------|-----------|-----------|-----------|-----------|-----------|-----------|
## Column Total |      4185 |      2847 |      2566 |      2412 |      2705 |     14715 | 
##              |     0.284 |     0.193 |     0.174 |     0.164 |     0.184 |           | 
## -------------|-----------|-----------|-----------|-----------|-----------|-----------|
## 
## 
```

Predictions

Predict on Test Data & Write to File

```
      tcPrediction <- predict(myModel, testChunk, type = "raw")
      tcPrediction

##  [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

      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(tcPrediction)
```

Conclusion

The kappa statistic ranges from 0 to 1, inclusive, with 1 indicating perfect agreement between the model's prediction and the true values. Though the interpretation can be subjective, generally speaking, a good agreement typically ranges between 0.60 - 0.80.

The model accuracccy is 98.17 %.
The out-of-sample error is 1.83 %.
The kappa value is 0.97.

```
print(currentDate <- date())

## [1] "Sat Apr 16 20:52:46 2016"
```