Project - Practical Machine Learning

David Berkowitz

March 3, 2015

```
library (caret, quietly=TRUE)
library (randomForest, quietly=TRUE)
set.seed (123)
```

Read in the data

```
## [1] "RDS exists, reading it"
## [1] "."
```

Feature selection (reduce the number of columns)

Many of the columns are statistical calculations (min_, max_, avg_, stddev_, var_, skewness_, kurtosis_) on the raw data measurements. I chose to build my classifier only on raw data columns: gyro_, accel_ and magnet_.

```
names = names (data.raw) # get the list of column names

gyros = grep ("^gyros", names)
accel = grep ("^accel", names)
magnet = grep ("^magnet", names)
class = grep ("^classe", names) # add the activity column
user = grep ("^user", names) # add the user name column

data = data.raw [, c(gyros, accel, magnet, class, user)] # only include wanted columns
data = na.omit (data) # omit NA values which are not appreciated by later functions
```

Partition the data

```
dp = createDataPartition (y = data$classe, p = 0.6, list=FALSE)
myTraining = data [dp,] # training set has random 60%
myTesting = data [-dp, ] # testing set has the remaining 40%

a = nrow (data) ; b = nrow (myTraining) ; c = nrow (myTesting)
check = a - b - c
cbind (a, b, c, check) # check should equal 0
```

```
##          a     b    c check
## [1,] 19622 11776 7846     0
```

Create the classifier based on my training set

```
rf = randomForest (classe ~ ., data=myTraining); rf
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = myTraining)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 6
##
##         OOB estimate of  error rate: 1.55%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 3338    2    2    4    2 0.002986858
## B   30 2234   13    2    0 0.019745502
## C    3   39 2010    2    0 0.021421616
## D   13    0   53 1858    6 0.037305699
## E    1    2    3    6 2153 0.005542725
```

Evaluate the classifier based on my testing set

```
predictions = predict (rf, myTesting)
confusionMatrix (predictions, myTesting$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2226   22    2    5    2
##          B    2 1484   18    0    0
##          C    0   12 1345   40    7
##          D    4    0    2 1237    6
##          E    0    0    1    4 1427
##
## Overall Statistics
##
##                Accuracy : 0.9838
##                  95% CI : (0.9808, 0.9865)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9795
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9973   0.9776   0.9832   0.9619   0.9896
## Specificity            0.9945   0.9968   0.9909   0.9982   0.9992
## Pos Pred Value         0.9863   0.9867   0.9580   0.9904   0.9965
## Neg Pred Value         0.9989   0.9946   0.9964   0.9926   0.9977
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2837   0.1891   0.1714   0.1577   0.1819
## Detection Prevalence   0.2877   0.1917   0.1789   0.1592   0.1825
## Balanced Accuracy      0.9959   0.9872   0.9870   0.9800   0.9944
```

This simple predictor does quite well with 98.38% accuracy, which is in line with the OOB estimated error rate of 1.55%.

Predict based on the real testing set

```
testingData = read.csv ("pml-testing.csv")
testingData = testingData [, c(gyros, accel, magnet, class, user)] # same dataset filtering a
s before
testingData = na.omit (testingData)

predictions = predict (rf, testingData); predictions
```

```
##  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
```

This “simple” classifier was sufficient to correctly predict all 20 values for the course project submission.

Cross Validation

```
table (data$user_name, data$classe)
```

```
##
##               A    B    C    D    E
##   adelmo   1165  776  750  515  686
##   carlitos  834  690  493  486  609
##   charles   899  745  539  642  711
##   eurico    865  592  489  582  542
##   jeremy   1177  489  652  522  562
##   pedro     640  505  499  469  497
```

```
#ggplot (data, aes (x=classe)) + geom_bar() + facet_wrap (~ user_name)
```

There appear to be sufficient data points for each activity for each user. I will cross validate by user_name.

```
crossValidate = function (name)
  {
  testingRowNumbers = training = testing = NULL
  a = b = c = check = NULL
  rf = cm = NULL

  testingRowNumbers = grep (name, data$user_name)
  training = data [-testingRowNumbers,]
  testing = data [testingRowNumbers,]

  a = nrow (data) ; b = nrow (training) ; c = nrow (testing); check = a - b - c
  cbind (a, b, c, check) # check should equal 0

  rf = randomForest (classe ~ ., data=training)
  predictions = predict (rf, testing)
  cm = confusionMatrix (predictions, testing$classe)
  cat (name, "accuracy= ", cm$overall ["Accuracy"], "\n")
  return (rf);
  }
```

```
rf = crossValidate ("adelmo"); rf
```

```
## adelmo accuracy=  0.2834018
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = training)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 6
##
##         OOB estimate of  error rate: 0.83%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 4409    1    0    4    1 0.001359003
## B   24 2977   20    0    0 0.014564714
## C    3   14 2655    0    0 0.006362275
## D    4    0   46 2649    2 0.019252129
## E    0    2    2    7 2910 0.003765834
```

```
rf = crossValidate ("carlitos"); rf
```

```
## carlitos accuracy=  0.5780848
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = training)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 6
##
##         OOB estimate of  error rate: 0.79%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 4741    2    0    2    1 0.001053519
## B   14 3079   13    1    0 0.009011909
## C    2   25 2901    1    0 0.009559577
## D    5    0   54 2667    4 0.023076923
## E    0    1    0    5 2992 0.002001334
```

```
rf = crossValidate ("charles"); rf
```

```
## charles accuracy=  0.5616516
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = training)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 6
##
##         OOB estimate of  error rate: 0.75%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 4673    2    0    5    1 0.001709037
## B   18 3023   11    0    0 0.009501966
## C    2   20 2861    0    0 0.007630940
## D    5    0   46 2521    2 0.020590521
## E    0    1    2    6 2887 0.003107735
```

```
rf = crossValidate ("eurico"); rf
```

```
## eurico accuracy=  0.1765472
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = training)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 6
##
##         OOB estimate of  error rate: 0.9%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 4710    1    0    4    0 0.001060445
## B   21 3166   18    0    0 0.012168487
## C    3   25 2905    0    0 0.009546539
## D    3    0   61 2567    3 0.025436598
## E    0    0    3    7 3055 0.003262643
```

```
rf = crossValidate ("jeremy"); rf
```

```
## jeremy accuracy=  0.5684891
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = training)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 6
##
##         OOB estimate of  error rate: 0.86%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 4392    2    3    5    1 0.002498297
## B   24 3264   20    0    0 0.013301088
## C    2   20 2747    1    0 0.008303249
## D    4    0   50 2636    4 0.021529324
## E    0    1    0    3 3041 0.001313629
```

```
rf = crossValidate ("pedro"); rf
```

```
## pedro accuracy=  0.1938697
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = training)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 6
##
##         OOB estimate of  error rate: 0.81%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 4935    2    0    2    1 0.001012146
## B   18 3252   21    1    0 0.012150668
## C    0   19 2904    0    0 0.006500171
## D    1    1   59 2682    4 0.023662177
## E    0    1    2    6 3101 0.002893891
```

Results

```
cbind (names, oobErrorRates, accuracy)
```

```
##      names      oobErrorRates accuracy
## [1,] "adelmo"   "0.82"        "0.27"
## [2,] "carlitos" "0.84"        "0.58"
## [3,] "charles"  "0.82"        "0.58"
## [4,] "eurico"   "0.93"        "0.18"
## [5,] "jeremy"   "0.83"        "0.55"
## [6,] "pedro"    "0.81"        "0.22"
```

```
mean (oobErrorRates) # average expected error rate (percentage)
```

```
## [1] 0.8416667
```

```
mean (accuracy) # average measured accuracy
```

```
## [1] 0.3966667
```