Acitivty Modeling

Lon Chang

June 24, 2017

Executive summary

This document is the data analysis report for the Coursera Course 8 Week 4 assignment. This assignment involves using activity tracker data to develop a model to predict what activity a participant is doing based on motion trackers. I will create random forest and linear discriminant analysis models to predict 20 holdout activities. The two models will be compared based on accuracy to the training data before deciding on the model for use in predicting the holdout activities.

The data for this project come from this source: http://groupware.les.inf.puc-rio.br/har (http://groupware.les.inf.puc-rio.br/har). The author of the linked website has permitted others to use the data with citation to the above link.

Load required libraries and inspect the data

There are 160 columns in the data. There are 5 possible activities ranging from A-E in the response variable Classe. There are no columns with no data.

```
library(caret)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
setwd("C:/Users/Lon/Documents/Coursera/Course 8 Machine Learning/Week 4")
download.file( "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-trainin
g.csv", "pml-training.csv")
download.file( "https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testin
g.csv", "pml-testing.csv")
downloadtime <- Sys.time()

fulltrain <- read.csv( "pml-training.csv", header= TRUE, sep= ",", na.strings=
"NA")
fulltest <- read.csv( "pml-testing.csv", header= TRUE, sep= ",", na.strings= "N
A")
dim(fulltest)
```

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

```
dim(fulltrain)
```

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

```
summary(fulltrain$classe)
```

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

Clean the data

The full table is quite large and there are numerous columns that contain a small percentage of non-0 values. I will remove these columns, as well as the first 7 columns, as these columns do not add value to fitting a model.

```
training <- fulltrain[, colSums(is.na(fulltrain)) == 0] #remove columns that ha
ve no data
training <- training[ , -(1:7)] #remove columns that do not appear useful to mo
del on
lowsignal <- nearZeroVar(training, saveMetrics = FALSE)
training <- training[ , -lowsignal]
x <- training[ ,1:(dim(training)[2]-1)] #split features into a separate table t
o improve modeling time
y <- training[ , dim(training)[2]] #split predictor into its own table to impro
ve modeling time
dim(x)
```

```
## [1] 19622    52
```

Set up multi-core processing

There is a large amount of data even with the above data cleansing. The code below sets up multi-core processing to improve modeling time. K-fold validation is also being set up to cut down on the amount of data used for each model in order to improve modeling time. 10 folds are used.

```
library(mlbench)
library(parallel)
library(doParallel)
```

```
## Loading required package: foreach
```

```
## Loading required package: iterators
```

```
cluster <- makeCluster(detectCores() - 1) # convention to leave 1 core for OS
registerDoParallel(cluster)

#set up 10-fold validation
fitControl <- trainControl(method = "cv",
              number = 10,
               allowParallel = TRUE)
```

Create models

Two models will be created, one based on random forests and one based on linear discriminant analysis. A generalized linear model and general boosting model was also attempted to run but no optimal model was found by caret.

```
#create random forest model
set.seed(15)
mod1 <- train( x, y, data= training, method= "rf", na.action= na.pass, trContro
l = fitControl)
```

```
## Loading required package: randomForest
```

```
## randomForest 4.6-12
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
confusionMatrix.train(mod1)
```

```
## Cross-Validated (10 fold) Confusion Matrix
##
## (entries are percentual average cell counts across resamples)
##
##           Reference
## Prediction    A    B    C    D    E
##          A 28.4  0.1  0.0  0.0  0.0
##          B  0.0 19.3  0.1  0.0  0.0
##          C  0.0  0.0 17.3  0.2  0.0
##          D  0.0  0.0  0.0 16.2  0.0
##          E  0.0  0.0  0.0  0.0 18.4
##
##  Accuracy (average) : 0.9955
```

```
pred1 <- predict(mod1, fulltest)

#create lda model
mod2 <- train( x, y, data= training, method= "lda", na.action= na.pass, trContr
ol = fitControl)
```

```
## Loading required package: MASS
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
confusionMatrix.train(mod2)
```

```
## Cross-Validated (10 fold) Confusion Matrix
##
## (entries are percentual average cell counts across resamples)
##
##           Reference
## Prediction    A    B    C    D    E
##          A 23.2  3.0  1.7  1.0  0.7
##          B  0.6 12.4  1.7  0.7  3.1
##          C  2.3  2.3 11.5  2.0  1.7
##          D  2.2  0.8  2.1 12.1  1.8
##          E  0.1  0.9  0.4  0.7 11.1
##
##  Accuracy (average) : 0.703
```

```
pred2 <- predict(mod2, fulltest)

sum(pred1==pred2)/length(pred1)
```

```
## [1] 0.7
```

```
#exit multi-core processing
stopCluster(cluster)
registerDoSEQ()
```

The random forest model has a greater overall accuracy rate than the lda model as shown in the training set. The two models also predict the same activity type 70% of the time. The random forest model has superior accuracy in the training data set so I will use the random forest predictions as my prediction for the test set. Despite the random forest's 99+% accuracy in the training set, I believe the out of sample accuracy will be lower, as the error rate on out of sample data is universally higher than in the training data. Training data is subject to collection biases and do not capture the true universe of data variation.

Predictions

```
pred1
```

```
##  [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
```