Predictive Modeling

Larry Colucci

December 17, 2017

Predictive Modeling Final Assignment

Project is to review data from physical measurement devices used when test subjects were performing exercises correctly and incorrectly. Based on test data, predict 20 new samples.

How I built the model

I decided to try a random forest model, and see how that worked. I started by importing the data, and doing some initial exploration to assess pre-processing needs. I did a histogram of the response variable 'classe'; it was slightly skewed to the value 'A' but not enough to justify normalization or other transforms.

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.3.3
```

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

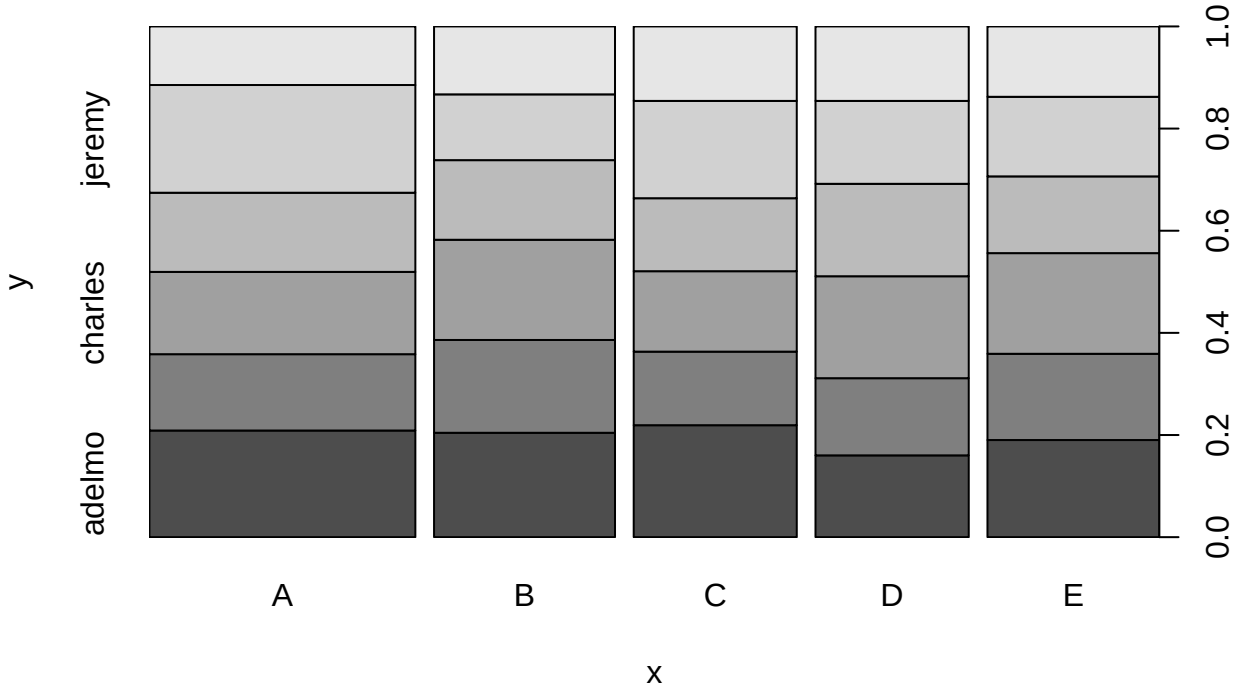
```
## Warning: package 'lattice' was built under R version 3.3.3
```

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

```
## Warning: package 'ggplot2' was built under R version 3.3.3
```

```
set.seed(1122)

inTrain <- read.csv("c:/devl/coursera/stat inf/pml-training.csv", na.strings=c("NA","#DIV/0!",""),string
inTest <- read.csv('c:/devl/coursera/stat inf/pml-testing.csv', na.strings=c("NA","#DIV/0!",""),strings
plot(as.factor(inTrain$classe), as.factor(inTrain$user_name))
```



I plotted a few other variables, and decided to remove the ones that were character/time based, and also those which were completely or largely NAs. This left 53 predictor variables.

```
char.fields <- grepl("X|timestamp|user_name|new_window|problem_id", names(inTrain))
inTrain <- inTrain[, which(char.fields ==FALSE)]
char.fields <- grepl("X|timestamp|user_name|new_window|problem_id", names(inTest))
inTest <- inTest[, which(char.fields == FALSE)]

#remove fields with NAs
inTrain <-inTrain[,colSums(is.na(inTrain)) == 0]
inTest <-inTest[,colSums(is.na(inTest)) == 0]
```

I ran a correlation analysis to see if variables had strong correlation; there were many that were greater than abs(.8). I plotted and visually inspected a few relationships; none that I viewed were simple ratios. I did notice one set that had a single outlier, and so removed that row.

```
#check correlation
M <- abs(cor(inTrain[,-54]))
diag(M) <- 0
which(M > 0.8, arr.ind=T)
```

```
##                  row col
## yaw_belt           4   2
## total_accel_belt   5   2
## accel_belt_y      10   2
## accel_belt_z      11   2
## accel_belt_x       9   3
## magnet_belt_x     12   3
```

```
## roll_belt          2   4
## roll_belt          2   5
## accel_belt_y      10   5
## accel_belt_z      11   5
## pitch_belt         3   9
## magnet_belt_x     12   9
## roll_belt          2  10
## total_accel_belt   5  10
## accel_belt_z      11  10
## roll_belt          2  11
## total_accel_belt   5  11
## accel_belt_y      10  11
## pitch_belt         3  12
## accel_belt_x       9  12
## gyros_arm_y       20  19
## gyros_arm_x       19  20
## magnet_arm_x      25  22
## accel_arm_x       22  25
## magnet_arm_z      27  26
## magnet_arm_y      26  27
## accel_dumbbell_x  35  29
## accel_dumbbell_z  37  30
## gyros_dumbbell_z  34  32
## gyros_forearm_z   47  32
## gyros_dumbbell_x  32  34
## gyros_forearm_z   47  34
## pitch_dumbbell    29  35
## yaw_dumbbell      30  37
## gyros_forearm_z   47  46
## gyros_dumbbell_x  32  47
## gyros_dumbbell_z  34  47
## gyros_forearm_y   46  47
```

```
#many with some corr; none have obvious simple relationship
#46/47 correlation has an obvious outlier, remove row
inTrain <- subset(inTrain, inTrain[47] < 10)
```

To be able to compute out of sample error, I segmented the training data into a train and test set. I chose 80% in the train set, to include as much data in the model build as is safe.

From there I ran the random forest model (from the caret package). Though processing time was large, it seemed to come back with a solid model.

```
knitr::opts_chunk$set(cache=TRUE)
dp <- createDataPartition(inTrain$classe, p=0.8, list = FALSE)

inTrain.train <- inTrain[dp,]
inTrain.test <- inTrain[-dp,]
whole.rf <- train(classe ~ ., data = inTrain.train, method='rf')
```

```
## Warning: package 'randomForest' was built under R version 3.3.3
```

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

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

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

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

```
pred.rf <- predict(whole.rf, newdata = inTest)
```

Cross validation

Due to the nature of the random forest model, stand alone cross validation was not used. The random forest model used a bootstrapped resample 25 times.

OUt of Sample error

```
confusionMatrix(inTrain.test$classe, predict(whole.rf, inTrain.test))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1115    0    0    0    0
##          B    1  757    1    0    0
##          C    0    3  681    0    0
##          D    0    0    0  643    0
##          E    0    0    0    2  719
##
## Overall Statistics
##
##                Accuracy : 0.9982
##                  95% CI : (0.9963, 0.9993)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9977
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9991   0.9961   0.9985   0.9969   1.0000
## Specificity            1.0000   0.9994   0.9991   1.0000   0.9994
## Pos Pred Value         1.0000   0.9974   0.9956   1.0000   0.9972
## Neg Pred Value         0.9996   0.9991   0.9997   0.9994   1.0000
## Prevalence             0.2845   0.1938   0.1739   0.1645   0.1833
## Detection Rate         0.2843   0.1930   0.1736   0.1639   0.1833
## Detection Prevalence   0.2843   0.1935   0.1744   0.1639   0.1838
## Balanced Accuracy      0.9996   0.9977   0.9988   0.9984   0.9997
```

```
out.of.sample.error <- sum(predict(whole.rf, inTrain.test) == inTrain.test$classe) / length(predict(whol
```

The out of sample error is 0.0017.