Machine Learning - Fitness Tracking

With the proliferation of wearable fitness trackers comes an enormous amount of easily accessible data on movement and activity. With this exercise, we are trying to determine the quality of an individual's exercise (specifically, a Unilateral Dumbbell Biceps Curl) based on their movement data. For more details on the data go to: http://groupware.les.inf.puc-rio.br/har.

First, we load the training and test data. The data contains a mix of motion data from sensors located on the arm, forearm, belt, and dumbbell. The training data has 19,622 observations, while the final testing data contains 20 observations for which we will try to predict 'classe'.

Classe Categories

- A: Bicep Curl exactly according to specification
- B: Throwing the elbows to the front
- C: Lifting the dumbbell only halfway
- D: Lowering the dumbbell only halfway
- E: Throwing the hips to the front

```
library(caret)
##Load Training Data
trainingdata<-read.csv("~/Documents/Learning/Coursera/machinelearning/predictionproject/pml-training.csv",
                       header=TRUE,na.strings="NA")
##Load Testing Data
testingdata<-read.csv("~/Documents/Learning/Coursera/machinelearning/predictionproject/pml-testing.csv",
                       header=TRUE,na.strings="NA")
```

In the exploratory phase, we do some plotting, while coloring by classe in order to try to uncover some details about the underlying data.

There appears to be a number of columns that contain a lot of NA's and very little usable data. `nearZeroVar` allows us to identify columns of data with low variance, and we can transform the data to remove those columns. This still leaves us with a lot of columns with mostly missing values. So we will also transform the training data to keep only columns with more than half the rows populated with data.

```
nearzerovar <- nearZeroVar(trainingdata,freqCut=200/5,saveMetrics=FALSE)
cleantrainingdata <- trainingdata[,-nearzerovar]
cleantrainingdata<-cleantrainingdata[,colSums(is.na(cleantrainingdata)) < .5*nrow(cleantrainingdata)]
```

In order to cross-validate, we will need to split our training set into a subset of training and testing sets.

```
inTrain <- createDataPartition(y=cleantrainingdata$classe,p=0.8,list=FALSE)
SubTrain <- cleantrainingdata[inTrain,]
SubTest <- cleantrainingdata[-inTrain,]
```

We are trying to predict categorical observations of the individual's exercise classe, so the machine learning method we will use is classification trees.

```
set.seed(123)
modFit<-train(classe ~ .,method="rpart",data=SubTrain)
modFit
```

```
## CART
##
## 15699 samples
##    58 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
##
## Summary of sample sizes: 15699, 15699, 15699, 15699, 15699, 15699, ...
##
## Resampling results across tuning parameters:
##
##   cp   Accuracy  Kappa  Accuracy SD  Kappa SD
##   0.2  0.7       0.6    0.09         0.1
##   0.3  0.6       0.5    0.09         0.1
##   0.3  0.4       0.2    0.09         0.2
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was cp = 0.2437.
```

We then use the `predict` function to apply our model to our test set, to see how well our model works.

```
pred <- predict(modFit,SubTest); SubTest$predRight <- pred==SubTest$classe
table(pred,SubTest$classe)
```

```
##
## pred    A    B    C    D    E
##    A 1115    0    0    0    0
##    B    1  759    0    0    0
##    C    0    0    0    0    0
##    D    0    0    0    0    0
##    E    0    0  684  643  721
```

Our model tests well for some of the classe; however, it seems to struggle with C and D. The accuracy of our test was 0.66. Our model accuracy was 0.736 with an SD of 0.089, so our test falls in line. Our out-of-sample accuracy would likely follow suit.

Finally, we use our model to predict the classe on our original test set of 20 observations.

```
finalPrediction <- predict(modFit,testingdata)
finalPrediction
```

```
##  [1] A A A A A A A A A A A A A A A A A A A A
## Levels: A B C D E
```

Oh boy…