Exercise Predictions from Personal Activity Monitors

We want to predict the manner in which 6 participants performed barbell lifts, using the 'classe' variable in the datasets. We are using data from accelerometers on the belt, forearm, arm, and dumbbell. Participants lifted correctly and incorrectly in 5 different ways.

First, let's download the training and testing data:

Let's look at the dimensions:

```
training <- read.csv("http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv")
testing <- read.csv("http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv")
```

Let's look at the dimensions:

```
dim(training)

## [1] 19622   160

dim(testing)

## [1] 20  160
```

The training and testing datasets have 160 variables. Training data has 19,622 observations and the testing data has 20. The variables are made up of averages, minimums, maximums, variances, standard deviations, and kurtosis.

To do this analysis, we will build a model. We will first load the caret package.

```
library(caret)
```

I decided to use a random forest model since it is known for its accuracy. I ruled out glm models due to the fact that our outcome variable has 5 different types. Random forest does not work with NA values, so I eliminated the columns in the test data that had NA in each row and also eliminated these same rows in the testing data.

```
training11 <- training[,!apply(is.na(testing),2,all)]
testing1 <- testing[,!apply(testing,2,function(x) all(is.na(x)))]
```

Furthermore, I removed columns 1-7 that had information about the users, dates, and time stamps, as I wanted to only include numeric or integer data. I again did this both in the training and test since they must be processed in the same way. We are measuring how well an individual performed an activity, and while the time of day may affect a person's performance, the time itself is not a measure of performance. I also thought it made sense to eliminate the 'total' columns since we were including averages and max and min's,

which are included in the total.

```
training12 <- training11[,-c(1:7,11,24,37,50)]
testing12 <- testing1[,-c(1:7,11,24,37,50)]
```

I started cross validation by using my training set and splitting it into a training and validation set. I picked 60% training, 40% validation as my proportions.

```
inTrain <- createDataPartition(y=training12$classe,p=0.6,list=FALSE)
training14 <- training12[inTrain,]
testing6 <- training12[-inTrain,]
```

I then built a random forest with no preProc model on the training set.

```
modelFit <- train(classe ~ .,method="rf",data=training14)
modelFit

## modelFit
## Random Forest
## 7846 samples
## 48 predictors
## 5 classes: 'A', 'B', 'C', 'D', 'E'
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 7846, 7846, 7846, 7846, 7846, 7846, ...
## Resampling results across tuning parameters:
##       mtry    Accuracy    Kappa    Accuracy SD
## 2         0.979         0.974    0.00356
## 25        0.979         0.974    0.00316
## 48        0.972         0.964    0.00323
## Kappa SD
## 0.0045
## 0.004
## 0.0041
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was mtry = 2.
```

After running the model, I evaluated the data on the validation set.

```
p1 <- predict(modelFit,testing6)
```

Here is a table comparing the predictions to the original observations in the validation set.

```
t1 <- table(p1,testing6$classe)
```

From testing on a new dataset, we can estimate the out of sample error. We will use the accuracy standard deviation as our measure. For this model, it was 0.003. This number

is low which makes it seem that our predictions are not too far off the actual.

To continue cross validation, I built another model with the training set and predicted on the validation set. I decided to add preprocessing by principal component analysis this time to the random forest model.

```
modelFitnew2 <- train(classe ~ .,method="rf",preProcess="pca",data=training14)
modelFitnew2

## modelFitnew2
## Random Forest
## 11776 samples
## 48 predictors
## 5 classes: 'A', 'B', 'C', 'D', 'E'
## Pre-processing: principal component
## signal extraction, scaled, centered
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 11776, 11776, 11776, 11776, 11776, 11776, ...
## Resampling results across tuning parameters:
##   mtry  Accuracy  Kappa  Accuracy SD  Kappa SD
## 2       0.954     0.942  0.00406      0.00512
## 25      0.935     0.918  0.00503      0.00633
## 48      0.935     0.918  0.00498      0.00627
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 2.
```

I again tested on the validation set and produced a table comparing the findings.

```
p2 <- predict(modelFitnew2,testing6)
t2 <- table(p2,testing6$classe)
```

The accuracy in the second model was smaller (95.4% vs. 98.7%) when just looking at the variable modelFitnew and modelFitnew2 and viewing the number of observations off the diagonals in the two tables. The accuracy standard deviation was also just a little bit higher (0.004). So, I decided the first model was the one to use for predicting on my final test set.

```
predict(modelFit,testing12)

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

This shows how the 20 test cases performed in each exercise.

Source: Ugulino, W.; Cardador, D.; Vega, K.; Velloso, E.; Milidiu, R.; Fuks, H. Wearable Computing: Accelerometers' Data Classification of Body Postures and Movements.

Proceedings of 21st Brazilian Symposium on Artificial Intelligence. Advances in Artificial Intelligence - SBIA 2012. In: Lecture Notes in Computer Science. , pp. 52-61. Curitiba, PR: Springer Berlin / Heidelberg, 2012. ISBN 978-3-642-34458-9. DOI: 10.1007/978-3-642-34459-6_6.