Course Project: Weight Lifting Activity Prediction

1. Executive Summary

The goal of this analysis was to predict how well a weight lifting activity was performed using data from activity tracking devices. The random forest CART model generated in this analysis can predict if the exercise was performed correctly (Class A) with 98.9% accuracy.

2. Introduction/Dataset

The Weight Lifting Exercises dataset was obtained from Groupware@LES (mailto:Groupware@LES) (http://groupware.les.inf.puc-rio.br/har (http://groupware.les.inf.puc-rio.br/har)). 6 male subjects aged 20-28 were asked to perform biceps curls either correctly (Class A), or displaying one of 4 common mistakes: throwing the elbows to the front (Class B), lifting the dumbbell only halfway (Class C), lowering the dumbbell only halfway (Class D) and throwing the hips to the front (Class E). The dataset contains variables recorded by the activity trackers during the 5 variations of this exercise.

3. Methods

Loading libraries and datasets:

```
library(caret)
library(rpart)
library(randomForest)
library(rattle)
library(rpart.plot)

training <- read.delim("pml-training.csv", sep = ",", na.strings = c("NA", "#DIV/0!"))
testing <- read.delim("pml-testing.csv", sep = ",", na.strings = c("NA", "#DIV/0!"))
```

Cleaning the training dataset:

```
table(is.na(training))
```

```
##
##   FALSE    TRUE
## 1214418 1925102
```

```
good <- which(colSums(is.na(training)) == 0) # selecting columns without NA's
clean.train <- training[good]
Train <- clean.train[, -c(1:7)]
```

The original raw data sets contained a large number of NA’s that were confined to certain columns which almost

exclusively contained NA's. Variables for which the vast majority of observations were NA were removed from the dataset, because imputing these values would not be a good idea given that there were so few actual values in these columns. The first 7 columns of the dataset were removed as well since they did not contain any predictor variables. The resulting clean data set contains 19622 observations of 53 variables.

Split data set into training and validation sets:

```
set.seed(12345)
inTrain <- createDataPartition(y=Train$classe, p=0.1, list=FALSE)
train <- Train[inTrain,]
val <- Train[-inTrain,]
dim(train); dim(val)
```

```
## [1] 1964   53
```

```
## [1] 17658    53
```

In order to keep the processing time for the model fit reasonable, only 10% of the data set were allocated to the training set and 90% were set aside for validation. If I had had more time I would have preferred a 70%:30% split. I tried running a random forest model with 70% of the training data, but the model fit could not be completed in a reasonable amount of time.

Fit prediction model using the random forest method:

```
set.seed(12345)
Fit <- train(classe ~ ., data = train, method = "rf")
```

4. Results

Final Model:

```
Fit$finalModel
```

```
##
## Call:
##  randomForest(x = x, y = y, mtry = param$mtry)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 27
##
##         OOB estimate of  error rate: 5.7%
## Confusion matrix:
##     A   B   C   D   E class.error
## A 548   6   2   2   0     0.01792
## B  19 342  17   2   0     0.10000
## C   0  14 326   3   0     0.04956
## D   2   2  23 294   1     0.08696
## E   1   4   7   7 342     0.05263
```

The predicted out of sample error of the final model is 5.7%.

Prediction on validation data:

```
predictions <- predict(Fit, newdata = val)
```

Cross Validation:

```
confusionMatrix(predictions, val$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 4957  111    2    3    8
##          B   14 3097  119    9   81
##          C   25  184 2945  143   76
##          D   11   19   13 2723   25
##          E   15    6    0   16 3056
##
## Overall Statistics
##
##                Accuracy : 0.95
##                  95% CI : (0.947, 0.953)
##     No Information Rate : 0.284
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.937
##  Mcnemar's Test P-Value : <2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity             0.987    0.906    0.956    0.941    0.941
## Specificity             0.990    0.984    0.971    0.995    0.997
## Pos Pred Value          0.976    0.933    0.873    0.976    0.988
## Neg Pred Value          0.995    0.978    0.991    0.988    0.987
## Prevalence              0.284    0.194    0.174    0.164    0.184
## Detection Rate          0.281    0.175    0.167    0.154    0.173
## Detection Prevalence    0.288    0.188    0.191    0.158    0.175
## Balanced Accuracy       0.989    0.945    0.964    0.968    0.969
```

When validating the model on the validation data set, the out of sample error is 5.0%.

Predicting Test Data:

```
test.pred <- predict(Fit, newdata = testing)
test.pred
```

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

5. Discussion

The overall out of sample error of the prediction model is 5%, but there are some differences in the prediction accuracy between the 5 different exercise classes. Class B has the lowest balanced accuracy (94.5%), while Class A has the highest (98.9%). Limitations of this model: If I had had more time and hardware resources, I would have used

a larger fraction of the training set (60-70%) to build the predition model which probably would have yielded even more accurate predictions.