Executive Summary

This analysis will examine the Weight Lifting Execise dataset available from the Human Activity Recognition website: http://groupware.les.inf.puc-rio.br/har. The WLE dataset was gathered to assess correctness in performance of weight lifting exercises (barbell curls in this case) based on various biometric measurements.

In this project we will use a training dataset to build a model that we will then assess against the testing dataaet provided. The model will be used to predict the 'classe' feature. The test dataset will be used to assess the out of band error rate.

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.0.3
```

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

```
## Warning: package 'ggplot2' was built under R version 3.0.3
```

```
#load the WLE training and test datasets
wle_training <- read.csv("pml-training.csv", header=TRUE)
wle_testing <- read.csv("pml-testing.csv", header=TRUE)
```

Exploratory Analysis

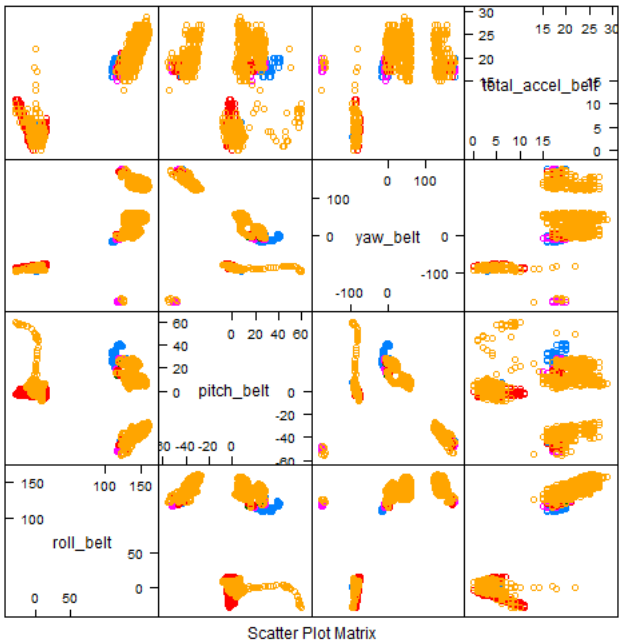
The WLE training dataset consists of a dataframe with 19622 observations on 160 variables. We'll use the variables that provide the average for each measurement type. These are the columns that contain the string 'avg', so we subset the training and test data on those columns.

We review the summary and a scattter plot matrix of a subset of the features, to determine if we obsevre trends.

```
#use the main variables for belt, arm, dumbbell, and forearm to build the models
trainData <- subset(wle_training, select=c("roll_belt", "pitch_belt","yaw_belt", "total_accel_belt",
                                           "roll_arm", "pitch_arm","yaw_arm", "total_accel_arm",
                                           "roll_dumbbell", "pitch_dumbbell","yaw_dumbbell",
"total_accel_dumbbell",
                                           "roll_forearm", "pitch_forearm","yaw_forearm",
"total_accel_forearm"))
trainData$classe <- wle_training$classe
testData <- subset(wle_testing, select=c("roll_belt", "pitch_belt","yaw_belt", "total_accel_belt",
                                         "roll_arm", "pitch_arm","yaw_arm", "total_accel_arm",
                                         "roll_dumbbell", "pitch_dumbbell","yaw_dumbbell",
"total_accel_dumbbell",
                                         "roll_forearm", "pitch_forearm","yaw_forearm",
"total_accel_forearm"))
summary(trainData)
```

```
##    roll_belt       pitch_belt        yaw_belt      total_accel_belt
##  Min.   :-28.9   Min.   :-55.80   Min.   :-180.0   Min.   : 0.0
##  1st Qu.:  1.1   1st Qu.:  1.76   1st Qu.: -88.3   1st Qu.: 3.0
##  Median :113.0   Median :  5.28   Median : -13.0   Median :17.0
##  Mean   : 64.4   Mean   :  0.31   Mean   : -11.2   Mean   :11.3
##  3rd Qu.:123.0   3rd Qu.: 14.90   3rd Qu.:  12.9   3rd Qu.:18.0
##  Max.   :162.0   Max.   : 60.30   Max.   : 179.0   Max.   :29.0
##     roll_arm        pitch_arm         yaw_arm        total_accel_arm
##  Min.   :-180.0   Min.   :-88.80   Min.   :-180.00   Min.   : 1.0
##  1st Qu.: -31.8   1st Qu.:-25.90   1st Qu.: -43.10   1st Qu.:17.0
##  Median :   0.0   Median :  0.00   Median :   0.00   Median :27.0
##  Mean   :  17.8   Mean   : -4.61   Mean   :  -0.62   Mean   :25.5
##  3rd Qu.:  77.3   3rd Qu.: 11.20   3rd Qu.:  45.88   3rd Qu.:33.0
##  Max.   : 180.0   Max.   : 88.50   Max.   : 180.00   Max.   :66.0
##  roll_dumbbell     pitch_dumbbell     yaw_dumbbell      total_accel_dumbbell
##  Min.   :-153.7   Min.   :-149.6   Min.   :-150.87   Min.   : 0.0
##  1st Qu.: -18.5   1st Qu.: -40.9   1st Qu.: -77.64   1st Qu.: 4.0
##  Median :  48.2   Median : -21.0   Median :  -3.32   Median :10.0
##  Mean   :  23.8   Mean   : -10.8   Mean   :   1.67   Mean   :13.7
##  3rd Qu.:  67.6   3rd Qu.:  17.5   3rd Qu.:  79.64   3rd Qu.:19.0
##  Max.   : 153.6   Max.   : 149.4   Max.   : 154.95   Max.   :58.0
##   roll_forearm     pitch_forearm      yaw_forearm      total_accel_forearm
##  Min.   :-180.00   Min.   :-72.50   Min.   :-180.0   Min.   :  0.0
##  1st Qu.:  -0.74   1st Qu.:  0.00   1st Qu.: -68.6   1st Qu.: 29.0
##  Median :  21.70   Median :  9.24   Median :   0.0   Median : 36.0
##  Mean   :  33.83   Mean   : 10.71   Mean   :  19.2   Mean   : 34.7
##  3rd Qu.: 140.00   3rd Qu.: 28.40   3rd Qu.: 110.0   3rd Qu.: 41.0
##  Max.   : 180.00   Max.   : 89.80   Max.   : 180.0   Max.   :108.0
##  classe
##  A:5580
##  B:3797
##  C:3422
##  D:3216
##  E:3607
##
```

```
featurePlot(x=trainData[,c("roll_belt", "pitch_belt","yaw_belt",
"total_accel_belt")],y=trainData$classe, plot="pairs")
```



Scatter Plot Matrix

Build a Model

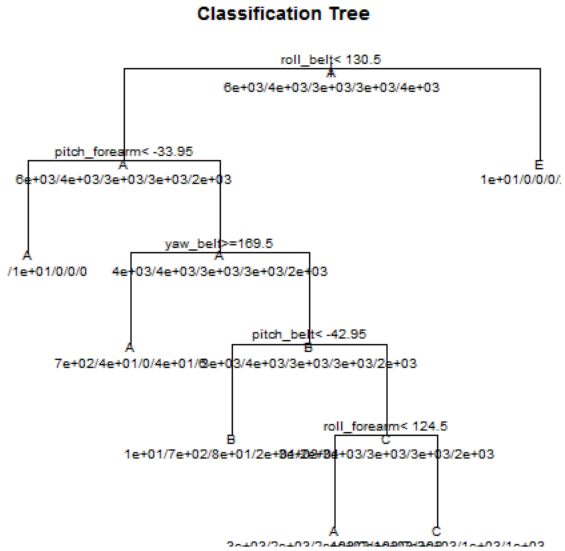
Partitioning Model

First, We build a classifiction by tree model:

```
modFit_rpart <- train(classe~.,method="rpart",data=trainData)
```

```
## Loading required package: rpart
## Loading required namespace: e1071
```

```
plot(modFit_rpart$finalModel, uniform=TRUE, main="Classification Tree")
text(modFit_rpart$finalModel, use.n=TRUE, all=TRUE, cex=.8)
```



Bootstrapped Aggregation Model with TreeBag

We build a model using Boostrapped Aggregation and then compare accuracy.

```
modFit_bag <- train(classe~.,method="treebag",data=trainData)
```

```
## Loading required package: ipred
```

```
## Warning: package 'ipred' was built under R version 3.0.3
```

```
## Loading required package: plyr
```

```
## Warning: package 'plyr' was built under R version 3.0.3
## Warning: some row.names duplicated:
2,4,6,8,9,10,11,14,18,20,22,24,26,30,32,33,36,39,41,42,46,54,56,57,59,61,65,66,68,71,72,82,86,90,94,98,99,10

## Warning: some row.names duplicated:
2,6,10,11,14,15,17,19,22,25,27,33,35,40,42,43,44,50,52,56,61,64,67,69,72,76,77,78,80,82,87,89,93,96,98,103,1

## Warning: some row.names duplicated:
3,5,6,9,10,12,15,18,19,20,23,24,26,27,29,32,33,35,42,44,46,49,51,53,54,55,62,64,67,72,73,76,77,79,82,83,86,8

## Warning: some row.names duplicated:
2,3,7,14,15,16,21,22,24,26,33,35,38,40,43,44,49,51,53,58,59,60,63,65,69,72,73,79,81,83,84,87,89,91,93,94,101

## Warning: some row.names duplicated:
2,5,10,11,13,15,17,19,20,21,25,26,31,33,34,36,38,40,41,46,47,52,58,64,70,74,75,77,78,89,91,95,99,101,102,103

## Warning: some row.names duplicated:
3,7,9,10,12,13,15,17,20,24,25,28,29,31,33,36,39,41,43,45,46,48,51,53,58,60,61,63,70,71,73,77,80,85,90,91,94,

## Warning: some row.names duplicated:
4,6,9,11,12,14,16,19,21,23,26,31,33,41,43,50,51,53,56,58,59,62,64,69,72,74,76,82,84,86,88,89,91,95,96,98,99,

## Warning: some row.names duplicated:
3,6,8,11,13,14,16,24,26,27,28,29,33,37,39,40,41,44,46,47,48,51,52,53,54,57,58,59,61,62,63,68,70,76,78,82,84,

## Warning: some row.names duplicated:
6,9,17,18,19,21,22,24,25,27,28,29,35,37,40,41,44,45,49,52,55,58,65,69,70,71,74,82,83,86,90,92,93,95,98,100,1

## Warning: some row.names duplicated:
3,4,8,11,13,15,17,19,25,27,29,42,43,47,50,53,54,55,58,62,63,69,70,71,74,76,79,81,82,84,85,93,96,98,101,103,1

## Warning: some row.names duplicated:
2,8,9,11,12,14,18,19,20,21,24,25,27,30,31,35,37,40,43,44,51,53,54,56,61,63,64,66,69,72,74,76,78,82,84,89,92,

## Warning: some row.names duplicated:
2,7,8,9,12,14,15,16,18,19,20,22,24,27,28,29,30,32,33,36,37,42,44,46,47,49,53,54,56,57,58,60,64,66,68,69,72,7

## Warning: some row.names duplicated:
3,6,9,11,12,14,19,26,34,35,37,41,42,45,49,52,59,61,64,65,69,73,75,76,80,82,83,86,91,93,95,100,102,109,114,11

## Warning: some row.names duplicated:
2,4,5,7,8,10,12,13,14,15,17,18,24,25,28,32,34,36,39,41,46,49,50,56,59,61,62,67,69,71,73,75,76,80,86,89,95,96

## Warning: some row.names duplicated:
3,7,8,10,13,14,15,18,19,25,26,28,29,30,32,35,37,45,47,50,52,56,63,64,67,70,71,74,77,79,82,83,85,86,88,91,96,

## Warning: some row.names duplicated:
2,3,9,12,17,20,24,26,27,30,32,34,35,40,44,47,50,52,53,59,60,66,67,68,74,76,79,82,83,86,88,89,91,92,94,95,96,

## Warning: some row.names duplicated:
9,10,14,16,21,23,25,26,28,30,33,37,40,41,43,45,52,54,56,59,60,62,66,68,73,80,81,83,84,86,87,93,95,97,101,102

## Warning: some row.names duplicated:
2,5,8,9,10,13,16,20,23,24,25,27,28,30,33,34,37,38,41,43,45,46,48,56,57,64,68,70,73,75,78,83,85,86,87,91,96,9

## Warning: some row.names duplicated:
3,6,9,10,13,14,19,24,29,35,37,48,51,52,56,57,60,62,65,67,68,71,73,74,76,80,84,85,92,98,102,103,105,106,107,1

## Warning: some row.names duplicated:
2,18,20,22,24,26,28,31,32,35,36,37,39,41,44,47,51,56,57,61,64,68,72,74,76,78,79,83,85,86,90,92,94,95,98,100,

## Warning: some row.names duplicated:
2,5,7,8,9,13,14,15,21,24,26,28,29,31,32,36,38,39,42,43,45,47,50,54,57,59,62,64,66,72,77,79,80,82,84,85,86,92

## Warning: some row.names duplicated:
2,3,9,10,11,14,16,17,19,21,22,24,25,28,29,34,39,41,45,47,48,53,55,57,58,59,60,63,64,71,72,74,78,80,83,86,89,

## Warning: some row.names duplicated:
3,6,9,12,14,17,20,25,28,31,35,37,39,40,42,45,47,49,50,52,53,55,56,58,59,60,61,66,67,69,71,73,74,75,80,83,86,

## Warning: some row.names duplicated:
2,8,10,14,18,22,23,27,28,34,35,37,38,40,41,44,51,55,56,58,62,63,64,65,67,70,71,74,80,81,84,85,87,88,91,93,96

## Warning: some row.names duplicated:
3,4,5,7,10,11,13,15,17,19,20,24,25,27,33,34,37,39,41,43,46,47,49,50,51,54,56,58,60,61,65,67,68,70,71,72,76,7
```

```
print(modFit_bag)
```

```
## Bagged CART
##
## 19622 samples
##    16 predictor
##     5 classes: 'A', 'B', 'C', 'D', 'E'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
##
## Summary of sample sizes: 19622, 19622, 19622, 19622, 19622, 19622, ...
##
## Resampling results
##
##   Accuracy  Kappa   Accuracy SD  Kappa SD
##   0.9798    0.9745  0.004187     0.005293
##
##
```

Predict Test Set Values Using the Models

Apply the model to the test data set.

```
predict_rpart <- predict(modFit_rpart, newdata=testData)
print(predict_rpart)
```

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

```
predict_bag <- predict(modFit_bag, newdata=testData)
print(predict_bag)
```

```
##  [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
```

Estimate the Out of Sample Error

Accuracy based on Cross Validation with the Training Set

We will estimate the Out of Sample Accuracy by calculating the error rate based on the training data. First, we create predictions for the training set. Then plot a confsion matrix. We see that the bagging model has higher accuracy than the classifaction tree based model.

```
trainPredict_rpart <- predict(modFit_rpart, newdata=trainData)
confusionMatrix(trainPredict_rpart, trainData$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 5152 1866 1571 1977  713
##          B   12  680   75   21   20
##          C  402 1251 1776 1218 1243
##          D    0    0    0    0    0
##          E   14    0    0    0 1631
##
## Overall Statistics
##
##                Accuracy : 0.471
##                  95% CI : (0.464, 0.478)
##     No Information Rate : 0.284
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.304
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity             0.923   0.1791   0.5190    0.000   0.4522
## Specificity             0.564   0.9919   0.7460    1.000   0.9991
## Pos Pred Value          0.457   0.8416   0.3015      NaN   0.9915
## Neg Pred Value          0.949   0.8343   0.8801    0.836   0.8901
## Prevalence              0.284   0.1935   0.1744    0.164   0.1838
## Detection Rate          0.263   0.0347   0.0905    0.000   0.0831
## Detection Prevalence    0.575   0.0412   0.3002    0.000   0.0838
## Balanced Accuracy       0.743   0.5855   0.6325    0.500   0.7257
```

```
trainPredict_bag <- predict(modFit_bag, newdata=trainData)
confusionMatrix(trainPredict_bag, trainData$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 5580    0    0    0    0
##          B    0 3797    1    0    0
##          C    0    0 3421    1    2
##          D    0    0    0 3215    0
##          E    0    0    0    0 3605
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (0.999, 1)
##     No Information Rate : 0.284
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 1
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity             1.000    1.000    1.000    1.000    0.999
## Specificity             1.000    1.000    1.000    1.000    1.000
## Pos Pred Value          1.000    1.000    0.999    1.000    1.000
## Neg Pred Value          1.000    1.000    1.000    1.000    1.000
## Prevalence              0.284    0.194    0.174    0.164    0.184
## Detection Rate          0.284    0.194    0.174    0.164    0.184
## Detection Prevalence    0.284    0.194    0.174    0.164    0.184
## Balanced Accuracy       1.000    1.000    1.000    1.000    1.000
```

Conclusion

We conclude that the complex relationship among the variables is best modelled using methods that support non-linear relationships. The more simplistic classification tree approach enables better interpretation of the model, but is unable to repreasent the interplay of the varuables sufficiently.