Machine Learning - Prediction Assignment

Garrett Richardson

The Goal of this project is to quantify how well people do a particular activity. In this case, we will take observations on the manner in which poeple exericse and grade them as follows: A - Exact to Specification, B - Throwing Elbows Out Front, C - Lifting Halfway, D = Lowering Halfway, E - Throwing Hips to the Front

First, we must read in the Datafile

```
exercise <- read.csv("pml-training.csv", header=TRUE)
```

Next we separate the datasets using cross-validation. 75% training, 25% testing.

```
inTrain <-createDataPartition(y=exercise$classe, p=0.75, list=FALSE)
training <- exercise[inTrain,]
testing <- exercise[-inTrain,]
dim(training)
```

```
## [1] 14718   160
```

```
dim(testing)
```

```
## [1] 4904  160
```

Cleaning the dataset. We create a new table with the variables we have chosen to use. We choose six variables based on their complete set of data, no NA values.

```
exercise <- subset(exercise, select=c(classe,total_accel_belt, total_accel_forearm, total_accel_arm, to
```

We partitition again on new this dataset as well via cross-validation.

```
inTrain <-createDataPartition(y=exercise$classe, p=0.75, list=FALSE)
training <- exercise[inTrain,]
testing <- exercise[-inTrain,]
```

We have now created a new data set which we will be using going forward to build our prediciton model.We will use K Folds sampling to created 20 folds for our 20 predictions.

```
set.seed(32323)
folds <- createFolds(y=training$classe, k=20, list=TRUE, returnTrain=TRUE)
sapply(folds,length)
```

```
## Fold01 Fold02 Fold03 Fold04 Fold05 Fold06 Fold07 Fold08 Fold09 Fold10
##  13982  13982  13982  13980  13983  13980  13983  13980  13983  13984
## Fold11 Fold12 Fold13 Fold14 Fold15 Fold16 Fold17 Fold18 Fold19 Fold20
##  13982  13981  13983  13983  13983  13981  13982  13984  13983  13981
```

The aforementioned represent the 20 folds we have created and their length. Now we will create 20 datasets through resampling.

```
set.seed(32323)
folds <- createResample(y=training$classe, times=20, list=TRUE)
sapply(folds, length)
```

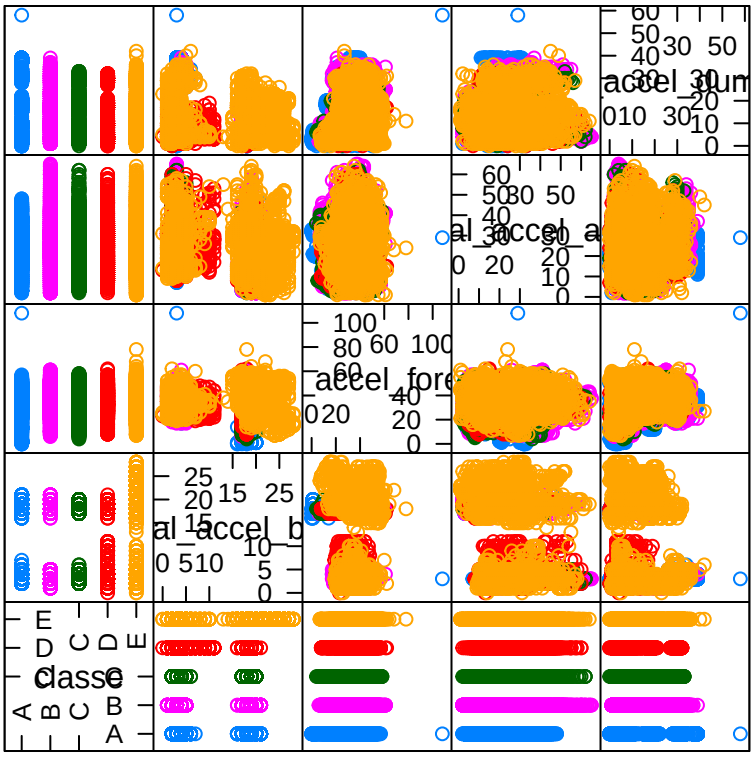
```
## Resample01 Resample02 Resample03 Resample04 Resample05 Resample06
##      14718      14718      14718      14718      14718      14718
## Resample07 Resample08 Resample09 Resample10 Resample11 Resample12
##      14718      14718      14718      14718      14718      14718
## Resample13 Resample14 Resample15 Resample16 Resample17 Resample18
##      14718      14718      14718      14718      14718      14718
## Resample19 Resample20
##      14718      14718
```

We have just displayed a couple of cross-validation techniques.

Now we will displays some figures to show data relationships.

Creating a Feature Plot

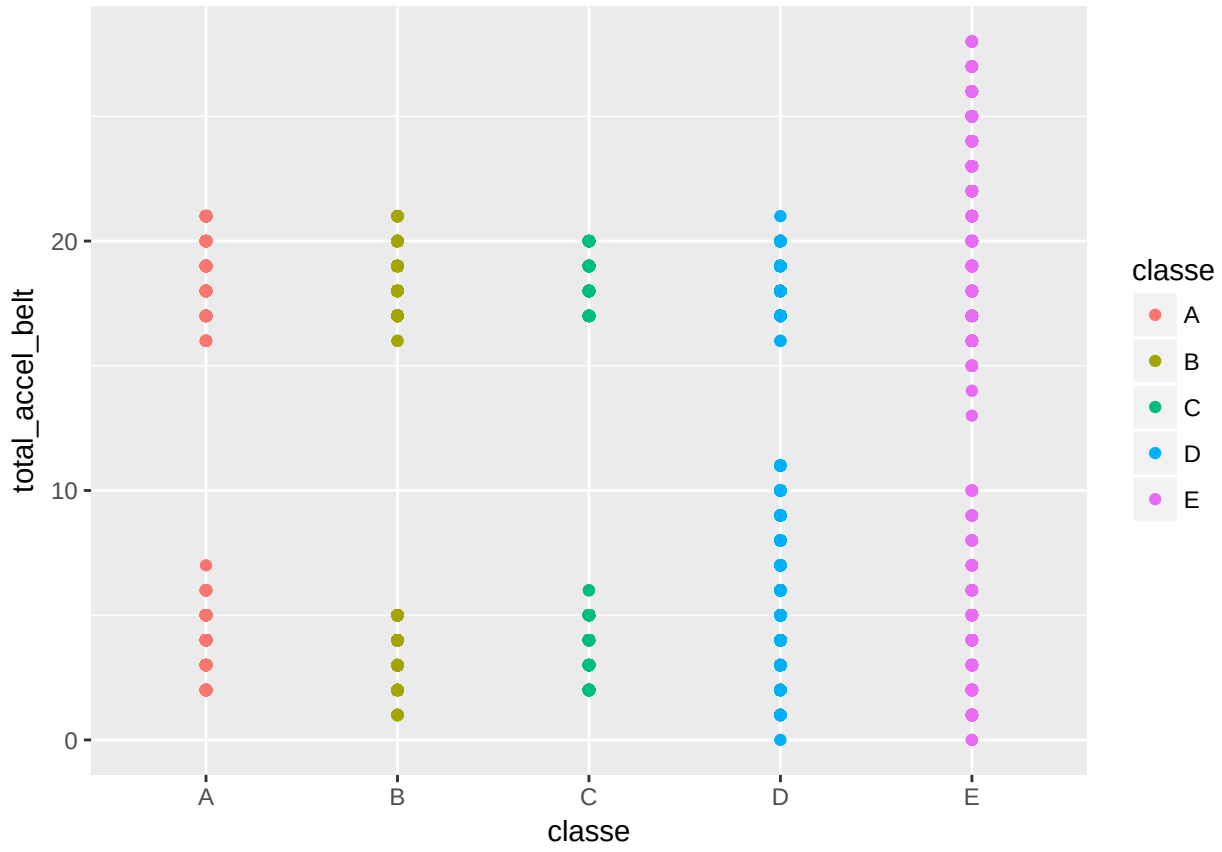
```
featurePlot(x=training[,c("classe","total_accel_belt", "total_accel_forearm", "total_accel_arm", "total
            y = training$classe, plot="pairs")
```



Scatter Plot Matrix

###Create a Qplot

```
qplot(classe,total_accel_belt,colour=classe, data=training)
```




```
plot
```

```
## standardGeneric for "plot" defined from package "graphics"
##
## function (x, y, ...)
## standardGeneric("plot")
## <environment: 0x000000001b5bf528>
## Methods may be defined for arguments: x, y
## Use  showMethods("plot")  for currently available ones.
```

Now let's make a prediction and find the error rate on 20 different test cases.

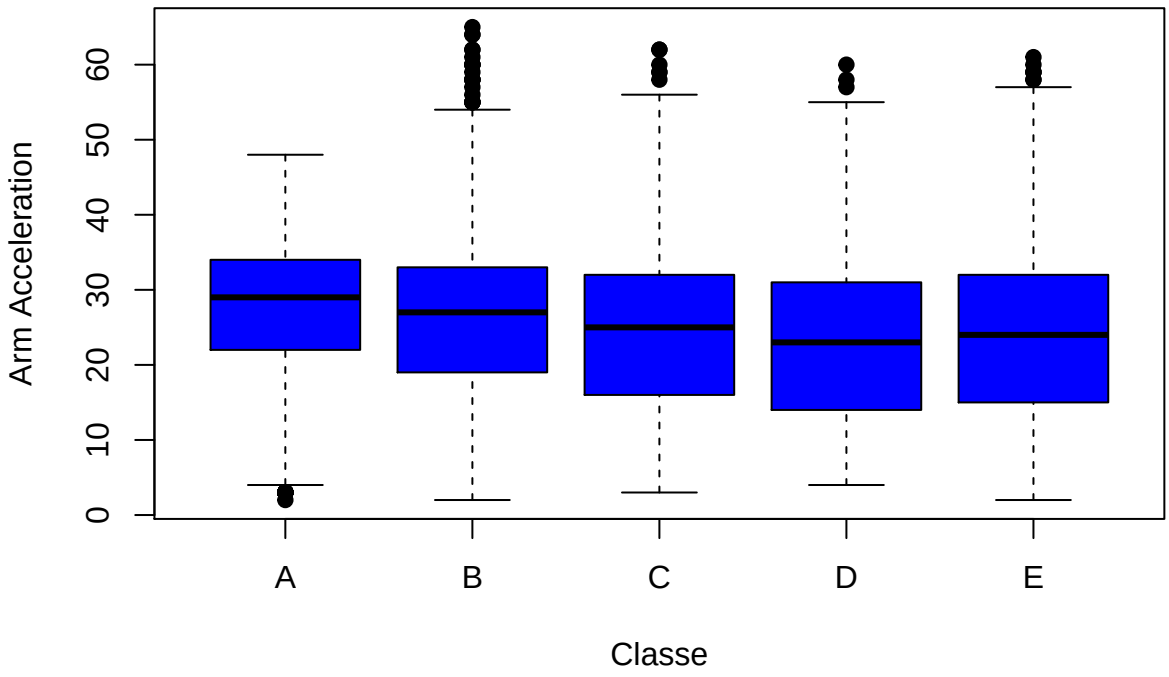
First we train, then we test and produce an Out of Sample Error.

```
inTrain <- createDataPartition(y=exercise$classe, p=0.5, list=FALSE)
trainexercise <- exercise[inTrain,]
testexercise <- exercise[-inTrain,]
lml <- lm(classe ~., data=testexercise)
```

```
## Warning in model.response(mf, "numeric"): using type = "numeric" with a
## factor response will be ignored
```

```
## Warning in Ops.factor(y, z$residuals): '-' not meaningful for factors
```

```
plot(trainexercise$classe, trainexercise$total_accel_arm, pch=19, col="blue", xlab="Classe", ylab="Arm A
```




```
predicted <- predict(lml, testexercise[21:1000, ], type = "response")
testexercise$classe = as.numeric(testexercise$classe)
actual <- testexercise[21:1000, "classe" ]
sqrt(mean(predicted - actual)^2)
```

```
## [1] 1.783479
```