Synopsis

Using devices such as Jawbone Up, Nike FuelBand, and Fitbit it is now possible to collect a large amount of data about personal activity relatively inexpensively. These type of devices are part of the quantified self movement – a group of enthusiasts who take measurements about themselves regularly to improve their health, to find patterns in their behavior, or because they are tech geeks. One thing that people regularly do is quantify how much of a particular activity they do, but they rarely quantify how well they do it. In this project, your goal will be to use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants. They were asked to perform barbell lifts correctly and incorrectly in 5 different ways. More information is available from the website here: http://groupware.les.inf.puc-rio.br/har (see the section on the Weight Lifting Exercise Dataset).

Six young health participants were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in five different fashions: exactly according to the specification (Class A), 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).

Read more: http://groupware.les.inf.puc-rio.br/har#weight_lifting_exercises#ixzz4eTUzCYNt

The Goal of this project

The goal of your project is to predict the manner in which they did the exercise. This is the "classe" variable in the training set. You may use any of the other variables to predict with. You should create a report describing how you built your model, how you used cross validation, what you think the expected out of sample error is, and why you made the choices you did. You will also use your prediction model to predict 20 different test cases.

Data

The training data for this project are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv

The test data are available here:

https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv

The data for this project come from this source: http://groupware.les.inf.puc-rio.br/har. If you use the document you create for this class for any purpose please cite them as they have been very generous in allowing their data to be used for this kind of assignment.

Finding all the ratios of invalid column data

```
navaluestrain <- colSums(is.na(training))/nrow(training)
# Ratio of Invalid column data in training data set
str(navaluestrain)
```

```
##  Named num [1:160] 0 0 0 0 0 0 0 0 0 0 ...
##  - attr(*, "names")= chr [1:160] "X" "user_name" "raw_timestamp_part_1" "
```

Remove the columns that has more than 75% of invalid data which is either null or not applicable data. We do this so while training the model without error and poor accuracy.

```
trainingset <- training[,navaluestrain < .25]

# convert the classe variable into a factor variable
trainingset$classe <- as.factor(trainingset$classe)
```

Now partition the training data such that 75% is training set and 25% testins set to test the model I am trained

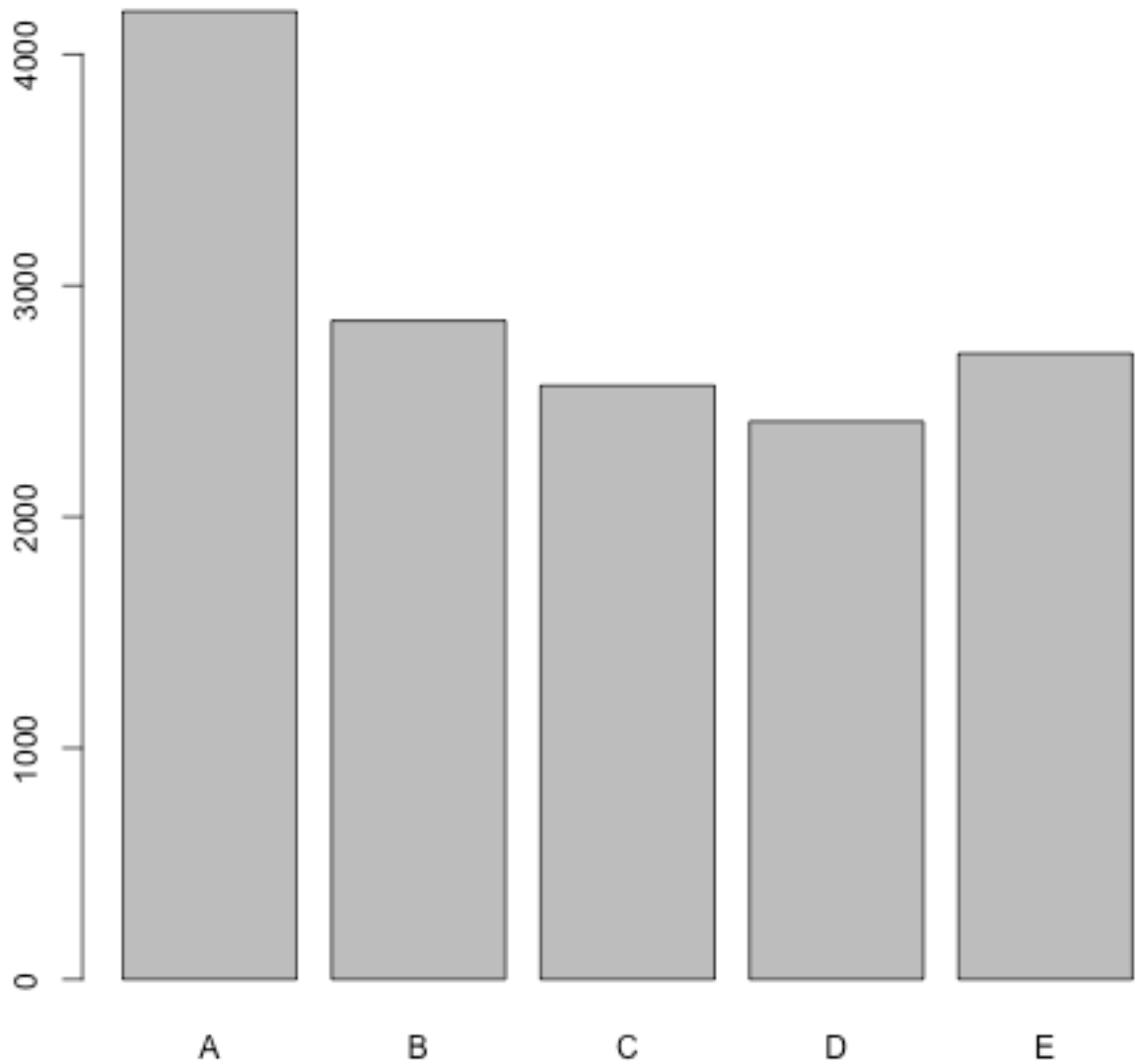
```
intrainset <- createDataPartition(y=trainingset$classe,p=0.75,list = FALSE)
intrain <- trainingset[intrainset,]
intest <- trainingset[-intrainset,]
```

Remove the columns that are not needed for training the model

```
intrain <- intrain[,-c(1:7)]
intest <- intest[,-c(1:7)]
```

Plot the Classe data to see how many people are getting trained with dumabells in each of the 5 different fashions mentioned in the synopsis.

```
plot(intrain$classe)
```



We could see from the above graph, that larger amount of people are trained with dumbbells with exact specification who fall under the category A, following is the people who have a fashion of throwing the hips front.

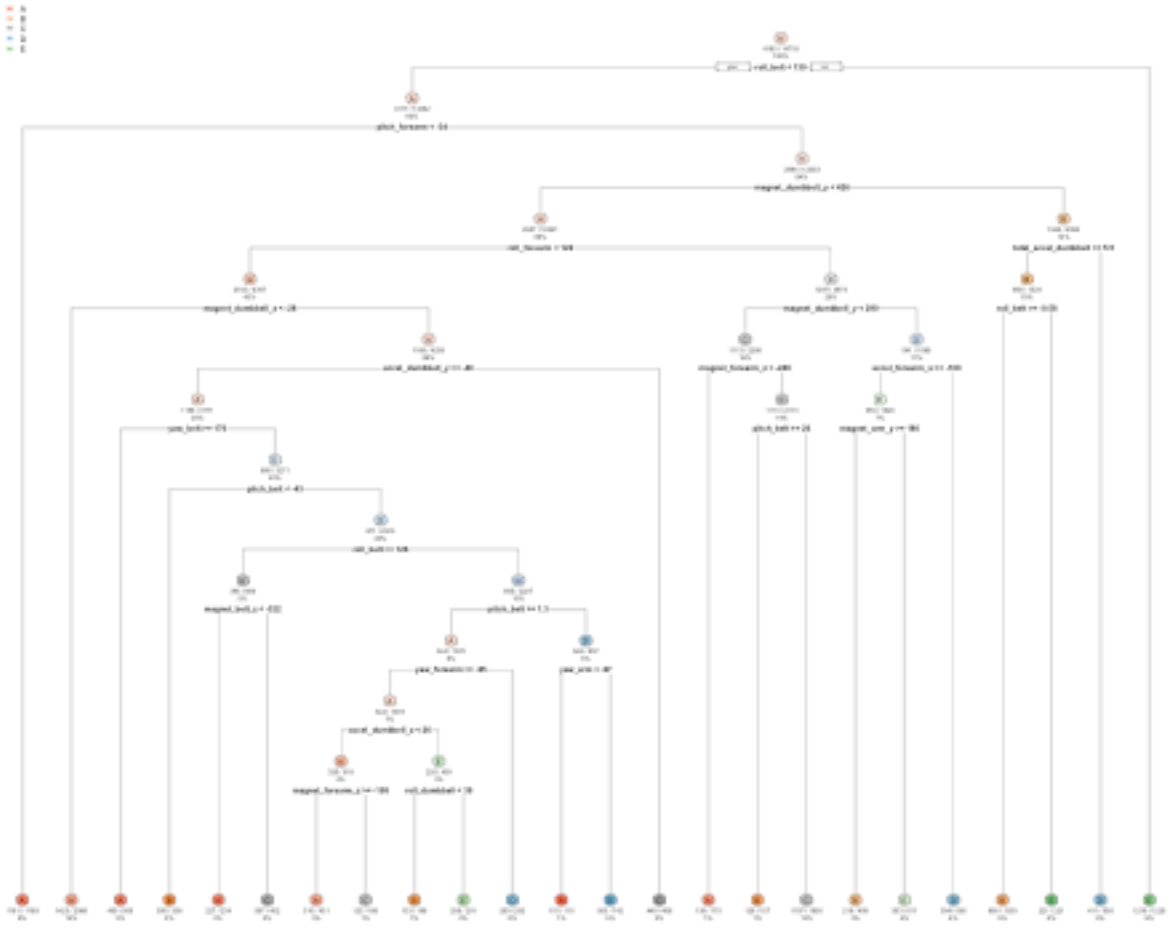
Training the Model

Prediction Model 1: Decision Tree

```
fitrptrain <- rpart(classe~.,data=intrain,method = "class")
predictrp <- predict(fitrptrain,intest,type = "class")
confusionMatrix(predictrp,intest$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1271  191   13   86   38
##          B   49  524   70   43   69
##          C   35  112  695  129  112
##          D   17   75   56  492   44
##          E   23   47   21   54  638
##
## Overall Statistics
##
##                Accuracy : 0.7382
##                  95% CI : (0.7256, 0.7504)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6672
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9111   0.5522   0.8129   0.6119   0.7081
## Specificity            0.9065   0.9416   0.9042   0.9532   0.9638
## Pos Pred Value         0.7949   0.6940   0.6417   0.7193   0.8148
## Neg Pred Value         0.9625   0.8976   0.9581   0.9261   0.9362
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2592   0.1069   0.1417   0.1003   0.1301
## Detection Prevalence   0.3261   0.1540   0.2208   0.1395   0.1597
## Balanced Accuracy      0.9088   0.7469   0.8585   0.7826   0.8359
```

```
rpart.plot(fitrptrain, main="Classification Tree", extra=102, under=TRUE, fa
```



Prediction Model 2: Random Forest Model

```
fitrftrain <- randomForest(classe~.,data=intrain,method = "class")
predictrf <- predict(fitrftrain,intest)
confusionMatrix(predictrf,intest$classe)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1395    9    0    0    0
##          B    0  938    3    0    0
##          C    0    2  852    8    0
##          D    0    0    0  796    1
##          E    0    0    0    0  900
##
## Overall Statistics
##
##                Accuracy : 0.9953
##                  95% CI : (0.993, 0.997)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9941
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9884   0.9965   0.9900   0.9989
## Specificity            0.9974   0.9992   0.9975   0.9998   1.0000
## Pos Pred Value         0.9936   0.9968   0.9884   0.9987   1.0000
## Neg Pred Value         1.0000   0.9972   0.9993   0.9981   0.9998
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2845   0.1913   0.1737   0.1623   0.1835
## Detection Prevalence   0.2863   0.1919   0.1758   0.1625   0.1835
## Balanced Accuracy      0.9987   0.9938   0.9970   0.9949   0.9994
```

From the above model accuracy, we see that the random forest has performed better with an accuracy of 99%, when compared to the decision tree algorithm with very high expected out of sample error and accuracy of ony 66.5% and the random forest algorithm - shows the confusion matrix table with prediction and reference shows that this is a best model by only a small amount of samples are falling away from the respective classe and we ould fit most of the testing data right. So we will choose the Random forest Prediction Model.

Submission

Let us apply the RF model against the testing data

```
library(caret)
predicttest <- predict(fitrftrain,testing,type= "class")
predicttest
```

```
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20
##  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
```