**Background**

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).

## ****What you should submit****

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.

### **Approach**

Our outcome variable is ‘classe’, a factor variable. For this data set, “participants were asked to perform one set of 10 repetitions of the Unilateral Dumbbell Biceps Curl in 5 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) - throwing the hips to the front (Class E)

We will test 2 models using ‘Decision Tree’ and ‘Random Forest’. The model with the highest Accuracy will be chosen as our final model.

#### **Cross-validation**

We will perform Cross-validation by subsampling our training data set randomly without replacement into 2 subsamples: TrainTrainingSet data (75% of the original Training data set) and TestTrainingSet data (25%).

Our models will be fitted on the TrainTrainingSet data set, and tested on the TestTrainingSet data. Once the most accurate model is chosen, it will be tested on the original Testing data set.

#### **Expected out-of-sample error**

The expected out-of-sample error will correspond to the quantity: 1-accuracy in the cross-validation data.

Accuracy is the proportion of correct classified observation over the total sample in the TestTrainingSet data set. Expected accuracy is the expected accuracy in the out-of-sample data set (i.e. original testing data set). Thus, the expected value of the out-of-sample error will correspond to the expected number of misclassified observations/total observations in the Test data set, which is the quantity: 1-accuracy found from the cross-validation data set.

**Algorithm**

1. **Load necessary libraries**

library(RCurl)

library(caret)

library(rpart)

library(rpart.plot)

library(randomForest)

1. **Loading Testing and Training sets**

testingUrl <- getURL("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv")

testingset <- read.csv(text = testingUrl, na.strings=c("NA","#DIV/0!", ""))

trainingUrl <- getURL("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv")

trainingset <- read.csv(text = trainingUrl, na.strings=c("NA","#DIV/0!", ""))

1. **Perform exploratory analysis**

dim(trainingset);

dim(testingset);

summary(trainingset);

summary(testingset);

str(trainingset);

str(testingset);

head(trainingset);

head(testingset);

1. **Prepare Data**

4.1) **Delete columns with all missing values**

trainingset<-trainingset[,colSums(is.na(trainingset)) == 0]

testingset <-testingset[,colSums(is.na(testingset)) == 0]

4.2) **Delete all irrelevant variables: user\_name, raw\_timestamp\_part\_1, raw\_timestamp\_part\_2, cvtd\_timestamp, new\_window, and num\_window (columns 1 to 7).**

trainingset <-trainingset[,-c(1:7)]

testingset <-testingset[,-c(1:7)]

1. **Partition the data**

Partition the data so that 75% of the training dataset into training and the remaining 25% to testing

traintrainset <- createDataPartition(y=trainingset$classe, p=0.75, list=FALSE)

TrainTrainingSet <- trainingset[traintrainset, ]

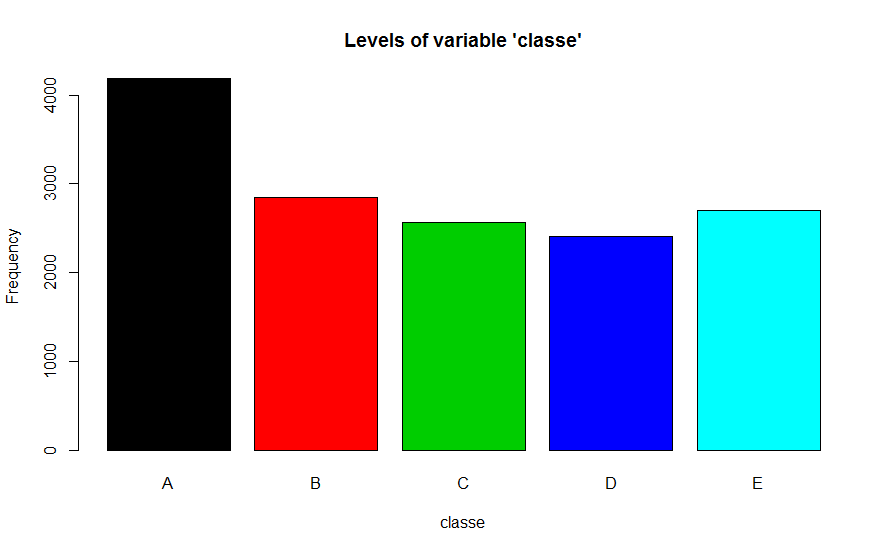
TestTrainingSet <- trainingset[-traintrainset, ]

1. **Analyze the data via plotting**

We will crate a plot of the outcome variable. It will help us to see the frequency of each levels in the TrainTrainingSet data set and compare them.

plot(TrainTrainingSet$classe, col=1:length(TrainTrainingSet$classe),

main="Levels of variable 'classe'", xlab="classe", ylab="Frequency")



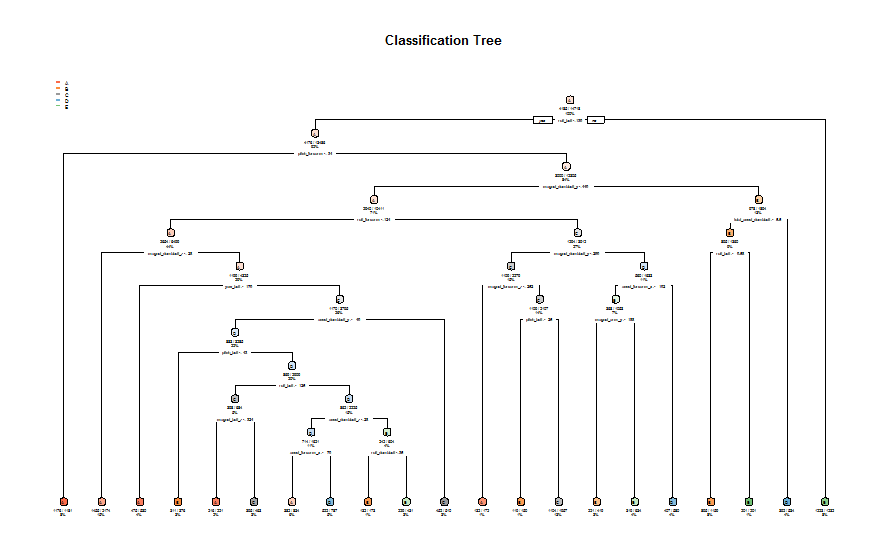
As we can see on the plot, Level A is the most frequent while level D is the least frequent.

1. **Create Decision Tree Prediction Model**

7.1) **Building Decision Tree Prediction model**

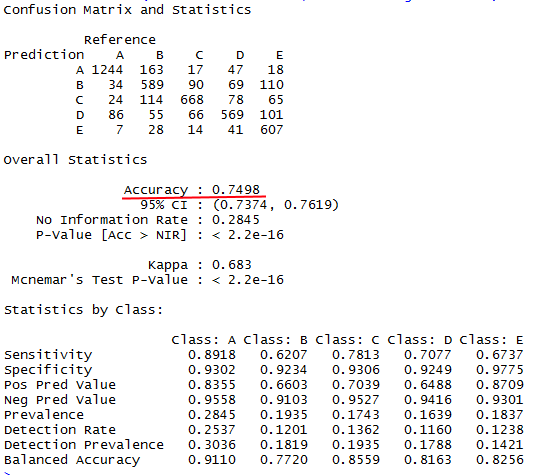
predictionTreeModel <- rpart(classe ~ ., data=TrainTrainingSet, method="class")

treeModelPrediction <- predict(predictionTreeModel, TestTrainingSet, type = "class")

7.2) **Plot the Decision Tree**rpart.plot(predictionTreeModel, main="Classification Tree", extra=102, under=TRUE, faclen=0, tweak=2)

7.3) **Test results on our TestTrainingSet data set**

confusionMatrix(treeModelPrediction, TestTrainingSet$classe)



1. **Create Random Forest Prediction Model**

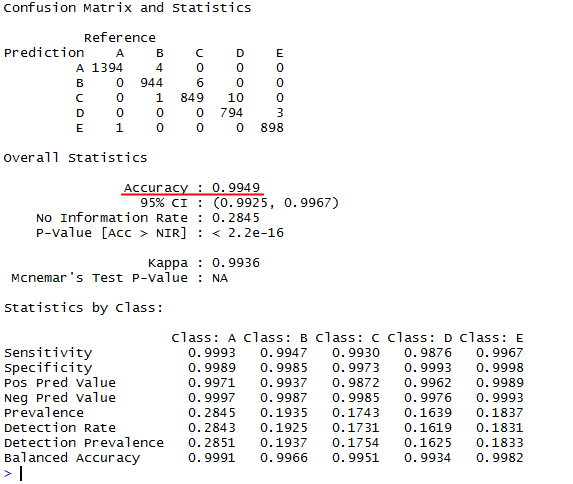
8.1) **Building Random Forest Prediction model**

randomForestModel <- randomForest(classe ~. , data=TrainTrainingSet, method="class")

randomForestPrediction <- predict(randomForestModel, TestTrainingSet, type = "class")

8.2) **Test results on TestTrainingSet data set**

confusionMatrix(randomForestPrediction, TestTrainingSet$classe)



**Choosing the prediction model**

Random Forest algorithm performed better than Decision Trees. Accuracy for Random Forest model was 0.995 (95% CI: (0.993, 0.997)) compared to Decision Tree model with 0.739 (95% CI: (0.727, 0.752)). The Random Forests model is chosen. The expected out-of-sample error is estimated at 0.005, or 0.5%.

**Submission**

Here is the final outcome based on the Random Forest Prediction Model applied against the Testing dataset

Predict outcome levels on the original Testing data set using Random Forest algorithm

predictfinal <- predict(randomForestModel, testingset, type="class")

predictfinal

