Practical Machine Learning - Human Activity Recognition

=====================================================

# Background

In this project, we will use data from accelerometers on the belt, forearm, arm, and dumbell of 6 participants using self monitoring devices to improve their health outcomes. 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). The goal of our project is to predict the manner in which they did the exercise. This is the "classe" variable in the training set. We will create a report describing how we built our model, how we used cross validation, what we think the expected outcome of sample error is, and why we made the choices we did. We will also use our 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>].

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).

Only Class A is the correct way of performing the exercize, all the other 4 ways are incorrect. Our prediction is on the Class E variable above. An overall pseudo-random number generator seed was set at 1234 for all code. In order to reproduce the results below, the same seed should be used.

### Cross-validation

Cross-validation will be performed by subsampling our training data set randomly without replacement into 2 subsamples: subTraining data (70% of the original Training data set) and subTesting data (30%). Our models will be fitted on the subTraining data set, and tested on the subTesting data. Once the most accurate model is choosen, it will be tested on the given Testing data set. Cross-validation provides good error estimates with minimal assumptions.It provides more confidence and security in the resulting conclusions.

### Expected out-of-sample error

The expected out-of-sample error will correspond to the quantity: 1-accuracy in the cross-validation data. Since the model is trained on the subTraining data set, accuracy is the proportion of correctly classified observations over the total sample in the subTesting data set. "Expected accuracy"" is the expected accuracy in the out-of-sample data set (i.e. given testing data set). Thus, the expected value of the out-of-sample error will correspond to the expected number of missclassified observations/total observations in the Test data set, which is the quantity: 1-accuracy found from the cross-validation data set.

### Reasons for my choices

Our outcome variable "classe" is an unordered factor variable. Thus, we can choose our error type as 1-accuracy. We have a large sample size with N= 19622 in the Training data set. This allow us to divide our Training sample into subTraining and subTesting to allow cross-validation. Features with all missing values will be discarded as well as features that are irrelevant. All other features will be kept as relevant variables. I chose Decision tree and random forest algorithms, as they are known for their ability of detecting the features that are important for classification, and work well when relationships aren't linear and complex relationships exist in data. I will use the Caret package which inherently selects the best features and the best model fit.

### Acknowledgement:

Thanks to the following folks for making the data available publicly for learning and analysis: Velloso, E.; Bulling, A.; Gellersen, H.; Ugulino, W.; Fuks, H. Qualitative Activity Recognition of Weight Lifting Exercises. Proceedings of 4th International Conference in Cooperation with SIGCHI (Augmented Human '13) . Stuttgart, Germany: ACM SIGCHI, 2013.

## Exploratory Data Analysis:

Lets load the data and provide a basic summary of the data. We will generate some basic plots to see relationships. Since we like to see the effect of the various variables against the Outcome (classe).

#install.packages("caret")  
#install.packages("randomForest")  
#install.packages("rpart")  
library(caret)

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

library(randomForest)

## randomForest 4.6-10  
## Type rfNews() to see new features/changes/bug fixes.

library(rpart)  
library(rpart.plot)  
set.seed(1234)  
  
# Loading the training data set replacing all missing with "NA"  
trainUrl <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv"  
testUrl <- "http://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv"  
  
trainingSet <- read.csv(url(trainUrl), na.strings=c("NA","#DIV/0!",""))  
testingSet <- read.csv(url(testUrl), na.strings=c("NA","#DIV/0!",""))  
  
# Check number of variables and number of observations  
# dim(trainingSet)  
# dim(testingSet)  
  
# Delete columns with all missing values  
trainingSet<-trainingSet[,colSums(is.na(trainingSet)) == 0]  
testingSet <-testingSet[,colSums(is.na(testingSet)) == 0]

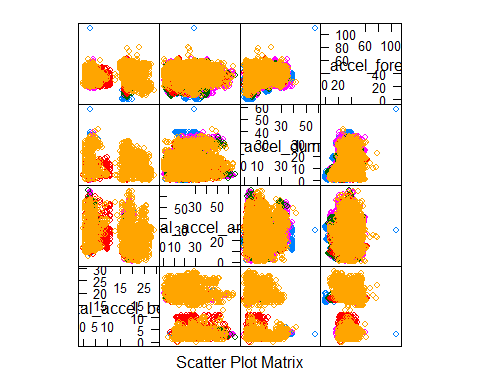
Now we have somewhat reduced the sets to relevant variables. View the data to see if all the remaining variables make sense for relationships to class variables.The first seven columns are irrelevant - the row number, the timestamps and the window information can be removed from the data. The remaining 53 variables all seem relevant to our prediction data.

# dim(trainingSet)  
# dim(testingSet)  
  
# head(trainingSet)  
  
trainingSet <-trainingSet[,-c(1:7)]  
testingSet <-testingSet[,-c(1:7)]  
  
# dim(trainingSet)  
# dim(testingSet)

### Cross validation data setup

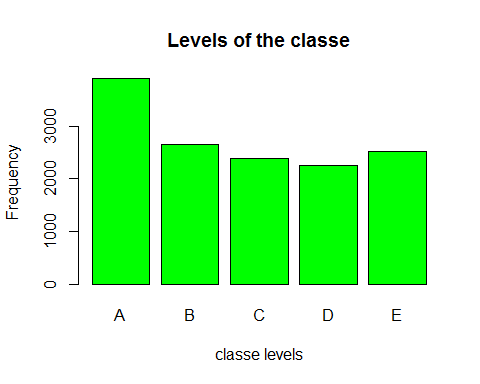
We will partition the training data ( 19622 observations) into subTraining and subTesting data sets in the ratio of 70:30. The given testingSet is only 20 observations, so the accuracy determined on our subTesting data set should be relatively close to that determined on the given testingSet.

subsamples <- createDataPartition(y=trainingSet$classe, p=0.70, list=FALSE)  
subTraining <- trainingSet[subsamples, ]   
subTesting <- trainingSet[-subsamples, ]  
# dim(subTraining)  
# dim(subTesting)  
# head(subTraining)  
# head(subTesting)  
  
featurePlot(x=trainingSet[,c("total\_accel\_belt","total\_accel\_arm","total\_accel\_dumbbell","total\_accel\_forearm" )],  
 y = trainingSet$classe,  
 plot="pairs")



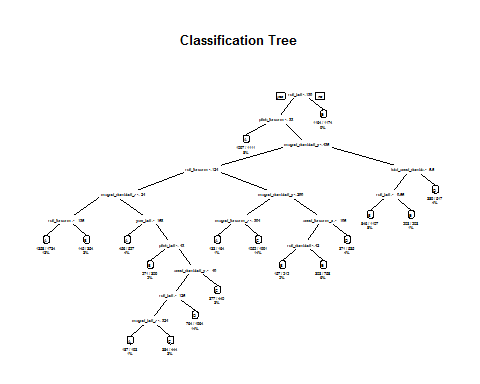
The feature plot is not very useful, let us determine the levels in the classe variable, supposed to show the positions of the subjects when the exercize was performed.

plot(subTraining$classe, col="green", main="Levels of the classe", xlab="classe levels", ylab="Frequency")



This plot shows that classe has 5 different values. Let us now train the model on the training data in classification tree method, and use the model in predicting the subTesting data

model1 <- rpart(classe ~ ., data=subTraining, method="class")  
  
# Predicting:  
prediction1 <- predict(model1, subTesting, type = "class")  
  
# Plot of the Decision Tree  
rpart.plot(model1, main="Classification Tree", extra=102, under=TRUE, faclen=0)



# Test results on our subTesting data set:  
confusionMatrix(prediction1, subTesting$classe)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 1364 169 24 48 16  
## B 60 581 46 79 74  
## C 52 137 765 129 145  
## D 183 194 125 650 159  
## E 15 58 66 58 688  
##   
## Overall Statistics  
##   
## Accuracy : 0.6879   
## 95% CI : (0.6758, 0.6997)  
## No Information Rate : 0.2845   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.6066   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Statistics by Class:  
##   
## Class: A Class: B Class: C Class: D Class: E  
## Sensitivity 0.8148 0.51010 0.7456 0.6743 0.6359  
## Specificity 0.9390 0.94543 0.9047 0.8657 0.9590  
## Pos Pred Value 0.8415 0.69167 0.6230 0.4958 0.7774  
## Neg Pred Value 0.9273 0.88940 0.9440 0.9314 0.9212  
## Prevalence 0.2845 0.19354 0.1743 0.1638 0.1839  
## Detection Rate 0.2318 0.09873 0.1300 0.1105 0.1169  
## Detection Prevalence 0.2754 0.14274 0.2087 0.2228 0.1504  
## Balanced Accuracy 0.8769 0.72776 0.8252 0.7700 0.7974

As we see, it gives an overall accuracy of only 69%. Let us now train our model using the random forest method

model2 <- randomForest(classe ~. , data=subTraining, method="class")  
  
# Predicting:  
prediction2 <- predict(model2, subTesting, type = "class")  
  
# Test results on subTesting data set:  
confusionMatrix(prediction2, subTesting$classe)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction A B C D E  
## A 1674 5 0 0 0  
## B 0 1133 6 0 0  
## C 0 1 1020 4 0  
## D 0 0 0 959 1  
## E 0 0 0 1 1081  
##   
## Overall Statistics  
##   
## Accuracy : 0.9969   
## 95% CI : (0.9952, 0.9982)  
## No Information Rate : 0.2845   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.9961   
## Mcnemar's Test P-Value : NA   
##   
## Statistics by Class:  
##   
## Class: A Class: B Class: C Class: D Class: E  
## Sensitivity 1.0000 0.9947 0.9942 0.9948 0.9991  
## Specificity 0.9988 0.9987 0.9990 0.9998 0.9998  
## Pos Pred Value 0.9970 0.9947 0.9951 0.9990 0.9991  
## Neg Pred Value 1.0000 0.9987 0.9988 0.9990 0.9998  
## Prevalence 0.2845 0.1935 0.1743 0.1638 0.1839  
## Detection Rate 0.2845 0.1925 0.1733 0.1630 0.1837  
## Detection Prevalence 0.2853 0.1935 0.1742 0.1631 0.1839  
## Balanced Accuracy 0.9994 0.9967 0.9966 0.9973 0.9994

As we can see, it gives a much better accuracy of 99.6% on our subTesting data.

### Conclusion

Random Forest algorithm performed better than Decision Trees. Accuracy for Random Forest model was 0.996 (95% CI: (0.9952, 0.9982) compared to 0.6879 and 95% CI : (0.6758, 0.6997) for Decision Tree model. The random Forest model is choosen. The accuracy of the model is 0.996. The expected out-of-sample error is estimated at 0.004, or 0.4%. The expected out-of-sample error is calculated as 1 - accuracy for predictions made against the cross-validation set. Our Test data set comprises 20 cases. With an accuracy above 99% on our cross-validation data, we can expect that very few, or none, of the test samples will be missclassified.

### Testing Set and Submission

We will now run our prediction on the given testing data.

# predict outcome levels on the original Testing data set using Random Forest algorithm  
predictfinal <- predict(model2, testingSet, type="class")  
predictfinal

## 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

#Write the files for submission  
  
# Write files for submission  
pml\_write\_files = function(x){  
 n = length(x)  
 for(i in 1:n){  
 filename = paste0("problem\_id\_",i,".txt")  
 write.table(x[i],file=filename,quote=FALSE,row.names=FALSE,col.names=FALSE)  
 }  
}  
pml\_write\_files(predictfinal)