BARBELL LIFTS MODEL

Synopsys

This study is conducted to determine how the barbell lifts are done, based on data get from accelerometers on the belt, forearm, arm, and dumbell.

This "classification model" was tested using decission trees and random forest (too much time processing data on a "MAC 8gb ram and core i7").

The training data was split in Training and Testing data. Training 70%, Testing 30%.

The cross validation was conducted using three random samples, and the out of sample error is estimated by average of three models.

The model reports 66.17% accuracy average on testing data (extracted from training dataset): Model 1: 66.15% Model 2: 66.17% Model 3: 66.19%

Out of sample error expected = 33%

Loading data and Exploratory Analysis

```
library (rattle)
```

```
## Warning: package 'rattle' was built under R version 3.1.1
```

```
## Rattle: A free graphical interface for data mining with R.
## Versi'on 3.1.0 Copyright (c) 2006-2014 Togaware Pty Ltd.
## Escriba 'rattle()' para agitar, sacudir y  rotar sus datos.
```

```
library(AppliedPredictiveModeling)
library(caret)
```

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

```
library(Hmisc)
```

```
## Loading required package: grid
## Loading required package: survival
## Loading required package: splines
##
## Attaching package: 'survival'
##
## The following object is masked from 'package:caret':
##
##     cluster
##
## Loading required package: Formula
```

```
## Warning: package 'Formula' was built under R version 3.1.1
```

```
##
## Attaching package: 'Hmisc'
##
## The following objects are masked from 'package:base':
##
##     format.pval, round.POSIXt, trunc.POSIXt, units
```

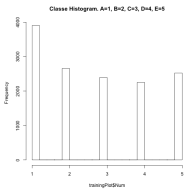
```
library(e1071)
```

```
##
## Attaching package: 'e1071'
##
## The following object is masked from 'package:Hmisc':
##
##     impute
```

```
library(psych)
```

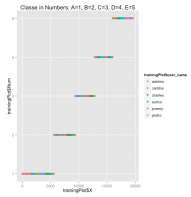
```
##
## Attaching package: 'psych'
##
## The following object is masked from 'package:Hmisc':
##
##     describe
##
## The following object is masked from 'package:ggplot2':
##
##     %+%
```

```
setwd("~/Coursera/Data Science/Practical Machine Learning/project1")
trainingSetOR <- read.csv("pml-training.csv")
testingSetOR <- read.csv("pml-testing.csv")

trainingSet <-
trainingSetOR[,-c(12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,50,51,52,53,54,55,56,57,58,59,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,87,88,89,90,91,92,93,94,95,96,97,98,99,100,101,103,104,105,106,107,108,109,110,111,112,125,126,127,128,129,130,131,132,133,134,135,136,137,138,139,141,142,143,144,145,146,147,1

testingSet <-
testingSetOR[,-c(12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,50,51,52,53,54,55,56,57,58,59,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,87,88,89,90,91,92,93,94,95,96,97,98,99,100,101,103,104,105,106,107,108,109,110,111,112,125,126,127,128,129,130,131,132,133,134,135,136,137,138,139,141,142,143,144,145,146,147,14



#str(trainingSet)
#describe(trainingSet)
trainIndex = createDataPartition(y=trainingSet$classe, p = 0.70,list=FALSE) #70% for training
training = trainingSet[trainIndex,]
testing = trainingSet[-trainIndex,]

#RF<-train(classe ~ ., data=training, method="rf", prox=TRUE)

# Assign values to classe ... for ploting purposses
# A=1, B=2, C=3, D=4, E=5
trainingPlot <- training
trainingPlot$Num <- 0
colNum=1
trainingPlot[trainingPlot$classe=="A",colNum] <- 1
trainingPlot[trainingPlot$classe=="B",colNum] <- 2
trainingPlot[trainingPlot$classe=="C",colNum] <- 3
trainingPlot[trainingPlot$classe=="D",colNum] <- 4
trainingPlot[trainingPlot$classe=="E",colNum] <- 5
hist(trainingPlot$Num, main="Classe Histogram. A=1, B=2, C=3, D=4, E=5")
```



Little skewness at right. But assume normal distribution.

```
qplot(trainingPlot$X,trainingPlot$Num, colour=trainingPlot$user_name, data=trainingPlot, main="Classe in Numbers: A=1, B=2, C=3, D=4, E=5")
```



X is an important variable as looks like each User were asked to do de exercise in a different way at by sequence in time.

Modelling: 3 random models for cross validation

```
# Model 1
modFit1 <- train(classe ~ ., method="rpart", data=training) #Commented for publishing results
```

```
## Loading required package: rpart
```

```
confusionMatrix(testing$classe, predict(modFit1, newdata=testing))
```

```
## Confusion Matrix and Statistics
##
##          Reference
## Prediction    A    B    C    D    E
##          A 1674    0    0    0    0
##          B    0 1139    0    0    0
##          C    0    0    0    0 1026
##          D    0    0    0    0  964
##          E    0    0    0    0 1082
##
## Overall Statistics
##
##                Accuracy : 0.662
##                  95% CI : (0.65, 0.674)
##     No Information Rate : 0.522
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.57
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity             1.000    1.000       NA       NA    0.352
## Specificity             1.000    1.000    0.826    0.836    1.000
## Pos Pred Value          1.000    1.000       NA       NA    1.000
## Neg Pred Value          1.000    1.000       NA       NA    0.586
## Prevalence              0.284    0.194    0.000    0.000    0.522
## Detection Rate          0.284    0.194    0.000    0.000    0.184
## Detection Prevalence    0.284    0.194    0.174    0.164    0.184
## Balanced Accuracy       1.000    1.000       NA       NA    0.676
```

** Model 1 Accuracy: 66.17%. Out of sample Error = 34% **

```
# Model 2
trainIndex2 = createDataPartition(y=trainingSet$classe, p = 0.70,list=FALSE) #70% for training
training2 = trainingSet[trainIndex2,]
testing2 = trainingSet[-trainIndex2,]

modFit2 <- train(classe ~ ., method="rpart", data=training2)  #Commented for publishing results
confusionMatrix(testing2$classe, predict(modFit2, newdata=testing2))
```

```
## Confusion Matrix and Statistics
##
##          Reference
## Prediction    A    B    C    D    E
##          A 1674    0    0    0    0
##          B    0 1139    0    0    0
##          C    0    0    0    0 1026
##          D    0    0    0    0  964
##          E    0    0    0    0 1082
##
## Overall Statistics
##
##                Accuracy : 0.662
##                  95% CI : (0.65, 0.674)
##     No Information Rate : 0.522
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.57
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity             1.000    1.000       NA       NA    0.352
## Specificity             1.000    1.000    0.826    0.836    1.000
## Pos Pred Value          1.000    1.000       NA       NA    1.000
## Neg Pred Value          1.000    1.000       NA       NA    0.586
## Prevalence              0.284    0.194    0.000    0.000    0.522
## Detection Rate          0.284    0.194    0.000    0.000    0.184
## Detection Prevalence    0.284    0.194    0.174    0.164    0.184
## Balanced Accuracy       1.000    1.000       NA       NA    0.676
```

"Model 2 Accuracy: 66.15% . Out of sample Error = 34% "

```
# Model 3
trainIndex3 = createDataPartition(y=trainingSet$classe, p = 0.70,list=FALSE) #70% for training
training3 = trainingSet[trainIndex3,]
testing3 = trainingSet[-trainIndex3,]

modFit3 <- train(classe ~ ., method="rpart", data=training3)  #Commented for publishing results
confusionMatrix(testing3$classe, predict(modFit3, newdata=testing3))
```

```
## Confusion Matrix and Statistics
##
##          Reference
## Prediction    A    B    C    D    E
##          A 1673    0    0    0    1
##          B    0 1138    0    0    1
##          C    0    0    0    0 1026
##          D    0    0    0    0  964
##          E    0    0    0    0 1082
##
## Overall Statistics
##
##                Accuracy : 0.662
##                  95% CI : (0.649, 0.674)
##     No Information Rate : 0.522
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.569
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.000    0.999       NA       NA    0.352
## Specificity            1.000    1.000    0.826    0.836    1.000
## Pos Pred Value         0.999    0.999       NA       NA    1.000
## Neg Pred Value         1.000    1.000       NA       NA    0.585
## Prevalence             0.284    0.194    0.000    0.000    0.522
## Detection Rate         0.284    0.193    0.000    0.000    0.184
## Detection Prevalence   0.284    0.194    0.174    0.164    0.184
## Balanced Accuracy      1.000    0.999       NA       NA    0.676
```

Model 3 Accuracy: 66.19%. Out of sample Error = 34%

Applying model to test dataset

```
results <- predict(modFit1, newdata=testingSet)

class(results)
```

```
## [1] "factor"
```

```
answers <- as.vector(results, mode = "character")
is.vector(answers)
```

```
## [1] TRUE
```

```
class(answers)
```

```
## [1] "character"
```

```
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(answers)
```