Practical Machine Learning Final Report

WM Ke

20160322

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).

Data

The training data for this project are available here

The test data are available here

```
setwd("F:/R/8_Machine Learning HW/PML_Final/")

download.file(url="https://d396qusza40orc.cloudfront.net/predmachlearn/
pml-training.csv", method="curl",destfile = "train.csv")

## Warning: 執行中命令 'curl "https://d396qusza40orc.cloudfront.net/
## predmachlearn/pml-training.csv" -o "train.csv"' 已有狀態 127

## Warning in download.file(url = "https://d396qusza40orc.cloudfront.ne
t/
## predmachlearn/pml-training.csv", : download had nonzero exit status

download.file(url="https://d396qusza40orc.cloudfront.net/predmachlearn/
pml-testing.csv", method="curl",destfile = "test.csv")

## Warning: 執行中命令 'curl "https://d396qusza40orc.cloudfront.net/
## predmachlearn/pml-testing.csv" -o "test.csv"' 已有狀態 127
```

```
## Warning in download.file(url = "https://d396qusza40orc.cloudfront.ne
t/
## predmachlearn/pml-testing.csv", : download had nonzero exit status

train<-read.csv(file="train.csv",na.strings = "NA"); test<-read.csv(fil
e ="test.csv",na.strings = "NA")
```

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.

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.

Reproducibility

Due to security concerns with the exchange of R code, your code will not be run during the evaluation by your classmates. Please be sure that if they download the repo, they will be able to view the compiled HTML version of your analysis.

Data Processsing

I first check the data structure of training file and found there are lots of blank or NA column in the dataset which would not do any good on prediction so I remove them (while there are 90% of row with blank or NA). The first 7 column is the personal identifying information and they were also removed.
The processed taining file was divided into two file with ratio of 3:1 : trainsub_Sub and trainsub_Vali, one for model trianing and one for validation use.

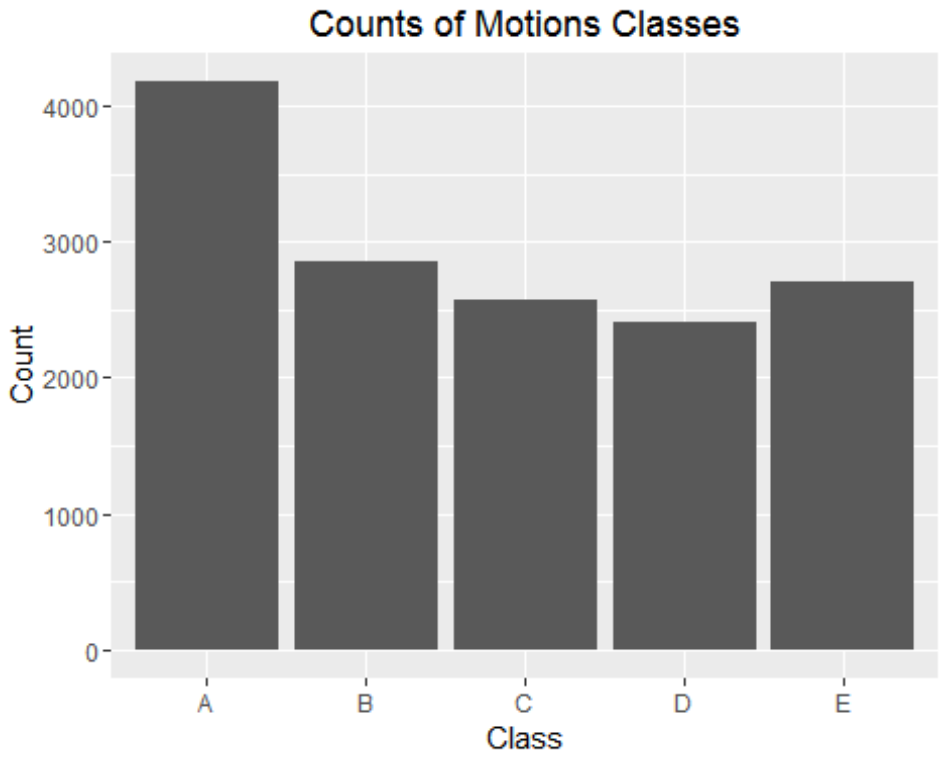
```
library(caret);

## Loading required package: lattice

## Loading required package: ggplot2

# remove column with too much NA and blank
trainsub<-train[,!colSums(is.na(train) | train=="" )/nrow(train)>=0.9 ]
testsub<-test[,!colSums(is.na(test) | test=="" )/nrow(test)>=0.9 ]
```

```
# remove the first 7 colonm with contain non-featuring information
trainsub<-trainsub[,-c(1:7)];testsub<-testsub[,-c(1:7)]

# Subset training data into traning and validation use sub
trainsub_Index<-createDataPartition(y=trainsub$classe, p=0.75, list=FAL
SE)
trainsub_Sub=trainsub[trainsub_Index,];trainsub_Vali=trainsub[-trainsub
_Index,]
```

Explortory ploting

There are 52 variable could be use to predict the type of action. I can only check the distibution of each type of motion (classe) - 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)

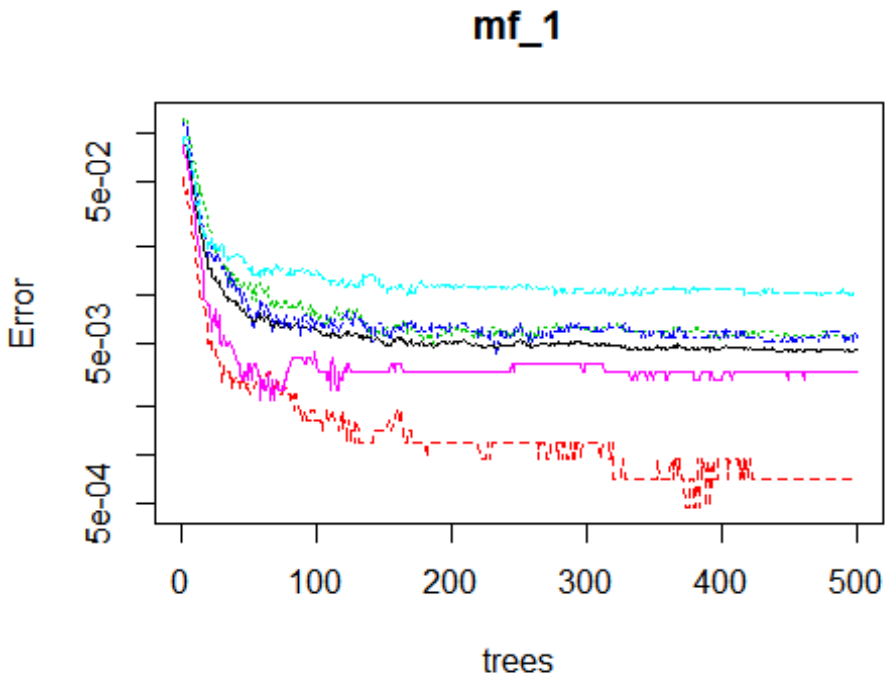
```
qplot(trainsub_Sub$classe,main="Counts of Motions Classes",xlab="Class
",ylab="Count")
```



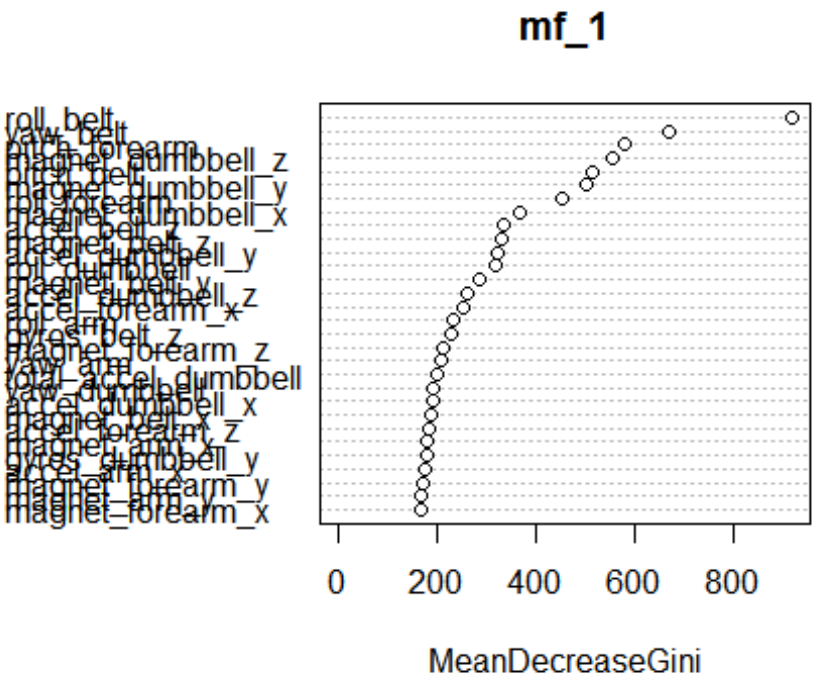
Model fit 1: Random forest

```
# Model fit
library(randomForest)
```

```
## randomForest 4.6-12

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':
##
##     margin

mf_1<-randomForest(classe~. ,data=trainsub_Sub, method="class")

#Plot
plot(mf_1, log="y")
```




```
varImpPlot(mf_1)
```




```
#Predict
p1_mf_1<-predict(mf_1,trainsub_Vali)
confusionMatrix(trainsub_Vali$classe,p1_mf_1)

## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1394    1    0    0    0
##          B    2  946    1    0    0
##          C    0    0  855    0    0
##          D    0    0    9  794    1
##          E    0    0    0    1  900
##
## Overall Statistics
##
##                Accuracy : 0.9969
##                  95% CI : (0.995, 0.9983)
##     No Information Rate : 0.2847
##     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            0.9986   0.9989   0.9884   0.9987   0.9989
## Specificity            0.9997   0.9992   1.0000   0.9976   0.9998
## Pos Pred Value         0.9993   0.9968   1.0000   0.9876   0.9989
## Neg Pred Value         0.9994   0.9997   0.9975   0.9998   0.9998
## Prevalence             0.2847   0.1931   0.1764   0.1621   0.1837
## Detection Rate         0.2843   0.1929   0.1743   0.1619   0.1835
## Detection Prevalence   0.2845   0.1935   0.1743   0.1639   0.1837
## Balanced Accuracy      0.9991   0.9991   0.9942   0.9982   0.9993
```

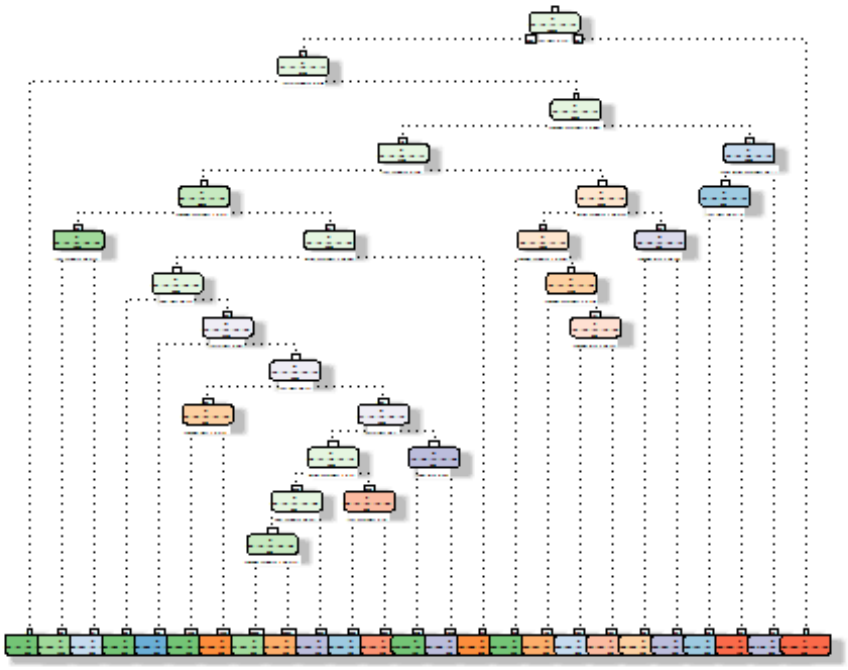
Model fit 2: Boosting with trees

```
mf_2<-train(classe~. ,data=trainsub_Sub,method="gbm")

## Loading required package: gbm

## Loading required package: survival

##
## Attaching package: 'survival'

## The following object is masked from 'package:caret':
##
##     cluster

## Loading required package: splines

## Loading required package: parallel

## Loaded gbm 2.1.1

## Loading required package: plyr

#Predict
p2_mf_1<-predict(mf_2,trainsub_Vali)
confusionMatrix(p2_mf_1,trainsub_Vali$classe)

## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1382   27    0    2    4
##          B   11  895   22    2    5
##          C    1   26  823   26    5
##          D    1    0    9  772   13
##          E    0    1    1    2  874
##
## Overall Statistics
##
##                Accuracy : 0.9678
##                  95% CI : (0.9625, 0.9725)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
```

```
##
##                   Kappa : 0.9592
##  Mcnemar's Test P-Value : 8.218e-05
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9907   0.9431   0.9626   0.9602   0.9700
## Specificity            0.9906   0.9899   0.9857   0.9944   0.9990
## Pos Pred Value         0.9767   0.9572   0.9342   0.9711   0.9954
## Neg Pred Value         0.9963   0.9864   0.9920   0.9922   0.9933
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2818   0.1825   0.1678   0.1574   0.1782
## Detection Prevalence   0.2885   0.1907   0.1796   0.1621   0.1790
## Balanced Accuracy      0.9906   0.9665   0.9741   0.9773   0.9845
```

Model fit 3: rpart

```
#loading library
library(ggplot2)
library(rpart)
library(rattle)

## Rattle: A free graphical interface for data mining with R.
## Version 4.1.0 Copyright (c) 2006-2015 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.

library(rpart.plot)
#Model fit
mf_3<-rpart(classe~. ,data=trainsub_Sub, method="class")

#Ploting

fancyRpartPlot(mf_3,main="")

## Warning: labs do not fit even at cex 0.15, there may be some overplo
tting
```




```
#Predict
p3_mf_1<-predict(mf_3,trainsub_Vali,type="class")
confusionMatrix(p3_mf_1,trainsub_Vali$classe)

## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 1286  151   12   49    9
##          B   48  576   91   56   99
##          C   30  129  654   70   77
##          D   21   47   59  548   71
##          E   10   46   39   81  645
##
## Overall Statistics
##
##                Accuracy : 0.7563
##                  95% CI : (0.7441, 0.7683)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6909
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
```

```
## Sensitivity            0.9219   0.6070   0.7649   0.6816   0.7159
## Specificity            0.9370   0.9257   0.9244   0.9517   0.9560
## Pos Pred Value         0.8534   0.6621   0.6813   0.7346   0.7856
## Neg Pred Value         0.9679   0.9075   0.9490   0.9384   0.9373
## Prevalence             0.2845   0.1935   0.1743   0.1639   0.1837
## Detection Rate         0.2622   0.1175   0.1334   0.1117   0.1315
## Detection Prevalence   0.3073   0.1774   0.1958   0.1521   0.1674
## Balanced Accuracy      0.9294   0.7663   0.8447   0.8166   0.8360
```

Model Comparison

The accuracy of three proposed model are shown. I noted the "Random forest"" model provide the best accuracy. So, this model was choose

```
p1_mf_2<-predict(mf_1,testsub)
p1_mf_2
```

```
##  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
```

Here the answer to the question!!