# Weight Training Analysis

## Summary

In this document a random forest model is used to classify weight training data from the WLE dataset available at http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har into one of 5 categories indicating whether the subject is correctly performing the exercise (class A) or incorrectly performing the exercise in one of four distinct ways (classes B,C,D,E). Random forests are chosen here as they provide a well understood easy to implement baseline classifier that typically performs well, but other options would be neural networks, k-means clustering (which an exploratory analysis not shown here indicates would be a bad approach), and naive Bayes classification. These other approaches may be explored outside of this course assignment.

The WLE dataset consists of 19622 entries of 160 variables. The data must be preprocessed as there are many variables which are predominantly NaNs and so should be removed before building the classifier. After preprocessing, the random forest model is built obtaing an out of sample error rate of about 0.5%. Note that all code for this document is presented at an appendix at the end for clarity of presentation.

## Preprocessing

The dataset is preprocessed by first removing variables at the beginning of the frame that shouldn't be used in the final model. These variables include the variables X through num_window which have nothing to do with the actual physical activity and shouldn't have any bearing on classification. Next I remove those columns that consist of more than 5% NaNs as these data should not be trusted in building the classifier, and there is no good way to impute the missing values, leaving a total of 52 predictor variables that can be used in the chosen model. Then I proceed by splitting the dataset by random sampling into training (80% of total dataset) and testing (20% of total dataset) sets. Finally I center and scale the data before applying the random forest model.

## Model Selection and Characterization

In order to determine the appropriate features for our model, I now perform a $5-$fold cross validation analysis using the $N$ most important features for $N = 52, 26, 13, 6, 3, 1$. This shows that we get significant gains in accuracy at least up to $N = 26$ of the most important features, with still apparent if modest gains beyond that up to $N = 52$ features.

| 52 | 26 | 13 | 6 | 3 | 1 |
|---|---|---|---|---|---|
| 0.005351 | 0.007899 | 0.06727 | 0.109 | 0.1456 | 0.4626 |

The confusion matrix for the testing set given earlier with a validation accuracy of 0.9961764 is shown below.

| | A | B | C | D | E |
|---|---|---|---|---|---|
| **A** | 1116 | 3 | 0 | 0 | 0 |
| **B** | 0 | 756 | 1 | 0 | 0 |
| **C** | 0 | 0 | 683 | 6 | 0 |
| **D** | 0 | 0 | 0 | 637 | 2 |
| **E** | 0 | 0 | 0 | 0 | 719 |

## Appendix

```
knitr::opts_chunk$set(echo = FALSE, results='asis',dev='pdf',cache=TRUE)
#knitr::opts_chunk$set(dev = 'pdf')
options(xtable.type = 'html')

##Preprocessing the data


library(caret)
library(randomForest)
library(dplyr)
library(pander)
set.seed(123453)
current_dir<-getwd()

data_dir<-"/Users/William/Desktop/Desktop/Projects/coursera-dsc/course-8/ProjectData"

setwd(data_dir)

#Load the relevant data.
#Note that the training data in this case should be split into testing and training data.
training<-read.csv("training.csv",na.strings=c("","#DIV/0!","NA"),
                   stringsAsFactors = FALSE)

#The first seven columms are data that we shouldn't build a model on and are removed
training<-select(training,roll_belt:classe)

#Also there are a number of columns consisting mostly of NAs.
#If the number of NAs is too large the column is removed
training<-training[,colSums(!is.na(training))/length(training[,"classe"])>=0.95]

#Create the partition
inTraining<-createDataPartition(training$classe,p=0.8,list=FALSE)

train<-training[inTraining,]

test<-training[-inTraining,]

#Note, the rest of the columns have been checked and there are actually no more NAs

#Now center and scale the data excluding the classification.
preProc<-preProcess(select(train,roll_belt:magnet_forearm_z),method=c("center","scale"))

trainPP<-predict(preProc,select(train,roll_belt:magnet_forearm_z))

testPP<-predict(preProc,test)

trainPP$classe<-as.factor(train$classe)

testPP$classe<-as.factor(test$classe)

rfmod<-randomForest(classe~.,data=trainPP,ntree=200,importance=TRUE)
```

```
pred<-predict(rfmod,testPP)

CM<-confusionMatrix(pred,testPP$classe)

setwd(current_dir)
CV<-rfcv(select(train,roll_belt:magnet_forearm_z),trainPP$classe,recursive=TRUE)
pander(CV$error.cv, type = 'grid')
pander(CM$table, type = 'grid')
```