Weighting Lifting Exercise prediction analysis

```
library(caret)
library(gbm)
library(dplyr)
library(tidyverse)
```

Data Loading

The data in this report comes from http://web.archive.org/web/20161224072740/http:/groupware.les.inf.puc-rio.br/har. We start off by downloading the training and testing datasets into R.

```
download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv", destfile="pml-tra
download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv", destfile="pml-test
training <- read.csv("pml-training.csv")
testing <- read.csv("pml-testing.csv")
```

Data Cleanup

The original data is organized in a way that there are some measurement lines, followed by a summary line. The summary line contains a lot of NA / DIV!0 values with the measurements, and is denoted in the data by the new_window = 'yes' variable. Here we filter them out as summary is simply transformation of measurement statistics, and would be expected to be highly-correlated with them. Besides, we take out the independent variables used in our model.

```
set.seed(12345)
training_clean <- training %>% filter(new_window == 'no') %>% select(starts_with("gyros"), starts_with(
```

Model Fitting

We will begin by fitting the data to 3 models, random forest, boosting and linear discriminant analysis, applying them to the training dataset. As can be seen below, random forest attained the highest level of accuracy (100%) with a tight 95% confidence interval (0.9998, 1). This is much better than the estimation from boosting and linear discriminant analysis models.

```
mod1 <- train(classe ~ ., method="rf", data=training_clean)
mod2 <- train(classe ~ ., method="gbm", data=training_clean, verbose=FALSE)
mod3 <- train(classe ~ ., method="lda", data=training_clean)

pred1 <- predict(mod1, training_clean)
pred2 <- predict(mod2, training_clean)
pred3 <- predict(mod3, training_clean)

confusionMatrix(as.factor(training_clean$classe), pred1)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 5471    0    0    0    0
##          B    0 3718    0    0    0
##          C    0    0 3352    0    0
##          D    0    0    0 3147    0
##          E    0    0    0    0 3528
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (0.9998, 1)
##     No Information Rate : 0.2847
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   1.0000   1.0000   1.0000   1.0000
## Specificity            1.0000   1.0000   1.0000   1.0000   1.0000
## Pos Pred Value         1.0000   1.0000   1.0000   1.0000   1.0000
## Neg Pred Value         1.0000   1.0000   1.0000   1.0000   1.0000
## Prevalence             0.2847   0.1935   0.1744   0.1638   0.1836
## Detection Rate         0.2847   0.1935   0.1744   0.1638   0.1836
## Detection Prevalence   0.2847   0.1935   0.1744   0.1638   0.1836
## Balanced Accuracy      1.0000   1.0000   1.0000   1.0000   1.0000
```

```
confusionMatrix(as.factor(training_clean$classe), pred2)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 5311   42   49   66    3
##          B  229 3260  173   38   18
##          C   76  122 3090   53   11
##          D   57   27  199 2824   40
##          E   26   70   75   86 3271
##
## Overall Statistics
##
##                Accuracy : 0.924
##                  95% CI : (0.9202, 0.9277)
##     No Information Rate : 0.2966
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9038
##
```

```
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.9319   0.9259   0.8617   0.9208   0.9785
## Specificity            0.9882   0.9708   0.9832   0.9800   0.9838
## Pos Pred Value         0.9708   0.8768   0.9218   0.8974   0.9272
## Neg Pred Value         0.9718   0.9832   0.9687   0.9849   0.9954
## Prevalence             0.2966   0.1832   0.1866   0.1596   0.1740
## Detection Rate         0.2764   0.1697   0.1608   0.1470   0.1702
## Detection Prevalence   0.2847   0.1935   0.1744   0.1638   0.1836
## Balanced Accuracy      0.9600   0.9483   0.9225   0.9504   0.9811
```

```
confusionMatrix(as.factor(training_clean$classe), pred3)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 4181  252  533  441   64
##          B  687 2253  418  153  207
##          C  599  282 2014  382   75
##          D  243  244  431 1912  317
##          E  239  657  288  498 1846
##
## Overall Statistics
##
##                Accuracy : 0.6352
##                  95% CI : (0.6283, 0.642)
##     No Information Rate : 0.3096
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5373
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            0.7028   0.6109   0.5467   0.5647  0.73575
## Specificity            0.9028   0.9057   0.9139   0.9220  0.89932
## Pos Pred Value         0.7642   0.6060   0.6008   0.6076  0.52324
## Neg Pred Value         0.8714   0.9074   0.8947   0.9083  0.95774
## Prevalence             0.3096   0.1919   0.1917   0.1762  0.13057
## Detection Rate         0.2176   0.1172   0.1048   0.0995  0.09607
## Detection Prevalence   0.2847   0.1935   0.1744   0.1638  0.18360
## Balanced Accuracy      0.8028   0.7583   0.7303   0.7433  0.81754
```

Model Testing

Finally, we apply the selected model to the testing dataset.

```
predict(mod1, testing)
```

```
##  [1] 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
```