Coursera - Practical Machine Learning Assignment Writeup

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 geek. 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).

The goal of your project is to predict the manner in which they did the exercise. The five classes we are having to predict are:
1. **A** Exact bicep curl 2. **B** Throwing elbows to the front 3. **C** Lifting the dumbbell only halfway 4. **D** Lowering the dumbbell only halfway 5. **E** Throwing the hip forward

Loading the necessary packages and data

```
library(caret)
library(randomForest)
train <- read.csv("pml-training.csv", na.strings=c("NA",""), strip.white=T)
test <- read.csv("pml-testing.csv", na.strings=c("NA",""), strip.white=T)
```

Pre-processing

Lot of columns have empty data like 'NA' or ' "" '. Let's remove them. Also, remove columns without predictive power like user_name, new_window, num_window, raw_timestamp_part_1, raw_timestamp_part_2, cvtd_timestampr.

```
isNA <- apply(train, 2, function(x) { sum(is.na(x)) })
training <- subset(train[, which(isNA == 0)],
                    select=-c(X, user_name, new_window, num_window, raw_timestamp_part_1, raw_timestamp_
isNA <- apply(test, 2, function(x) { sum(is.na(x)) })
testing <- subset(test[, which(isNA == 0)],
                    select=-c(X, user_name, new_window, num_window, raw_timestamp_part_1, raw_timestamp_
dim(training)
```

```
[1] 19622    53
```

```
dim(testing)
```

```
[1] 20 53
```

Training a Random Forest Model

Given that the problem is a high-dimensional classification problem with number of observations much exceeding the number of predictors, random forest seems like a good choice.

```
set.seed(12345)
model <- randomForest(classe ~ ., data = training)
```

```
model
```

```
##
## Call:
##  randomForest(formula = classe ~ ., data = training)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 7
##
##         OOB estimate of  error rate: 0.29%
## Confusion matrix:
##      A    B    C    D    E class.error
## A 5577    2    0    0    1   0.0005376
## B    9 3785    3    0    0   0.0031604
## C    0   11 3409    2    0   0.0037989
## D    0    0   19 3194    3   0.0068408
## E    0    0    2    5 3600   0.0019407
```

OOB estimate of error rate: 0.29% looks excellent. The confusion matrix also looks excellent. Training set has lots of observations so model fits very well. If we had more data to train,
model training would become time intensive so we could switch to parallel using doMC library and/or using less variable according to variable importance. Let's look at the variable importance.

```
imp <- varImp(model)
imp$Variable <- row.names(imp)
imp[order(imp$Overall, decreasing = T),]
```

```
##                     Overall           Variable
## roll_belt           1255.40          roll_belt
## yaw_belt             901.01           yaw_belt
## magnet_dumbbell_z    748.82  magnet_dumbbell_z
## pitch_forearm        736.67      pitch_forearm
## pitch_belt           716.83         pitch_belt
## magnet_dumbbell_y    668.27  magnet_dumbbell_y
## roll_forearm         628.33       roll_forearm
## magnet_dumbbell_x    488.62  magnet_dumbbell_x
## roll_dumbbell        419.53      roll_dumbbell
## accel_dumbbell_y     407.78   accel_dumbbell_y
## magnet_belt_z        404.58      magnet_belt_z
## accel_belt_z         395.56       accel_belt_z
## magnet_belt_y        387.27      magnet_belt_y
## accel_forearm_x      325.48    accel_forearm_x
## roll_arm             322.68           roll_arm
## accel_dumbbell_z     316.62   accel_dumbbell_z
```

```
## gyros_belt_z          315.20         gyros_belt_z
## magnet_forearm_z      289.77     magnet_forearm_z
## gyros_dumbbell_y      265.26     gyros_dumbbell_y
## total_accel_dumbbell  264.21 total_accel_dumbbell
## magnet_arm_x          261.34         magnet_arm_x
## accel_dumbbell_x      254.31     accel_dumbbell_x
## magnet_belt_x         250.63        magnet_belt_x
## yaw_dumbbell          249.74         yaw_dumbbell
## yaw_arm               249.02              yaw_arm
## accel_forearm_z       248.61      accel_forearm_z
## accel_arm_x           244.88          accel_arm_x
## magnet_forearm_y      232.17     magnet_forearm_y
## magnet_forearm_x      229.37     magnet_forearm_x
## magnet_arm_y          225.68         magnet_arm_y
## total_accel_belt      217.18     total_accel_belt
## magnet_arm_z          179.39         magnet_arm_z
## pitch_arm             177.17            pitch_arm
## yaw_forearm           175.91          yaw_forearm
## pitch_dumbbell        172.61       pitch_dumbbell
## accel_arm_y           142.44          accel_arm_y
## accel_forearm_y       141.25      accel_forearm_y
## gyros_arm_y           138.62          gyros_arm_y
## gyros_arm_x           131.69          gyros_arm_x
## gyros_dumbbell_x      129.73     gyros_dumbbell_x
## accel_arm_z           128.68          accel_arm_z
## gyros_forearm_y       125.82      gyros_forearm_y
## accel_belt_y          118.73         accel_belt_y
## accel_belt_x          115.17         accel_belt_x
## gyros_belt_y          111.58         gyros_belt_y
## total_accel_forearm   107.80  total_accel_forearm
## total_accel_arm       106.83      total_accel_arm
## gyros_belt_x           99.16         gyros_belt_x
## gyros_forearm_z        84.41      gyros_forearm_z
## gyros_dumbbell_z       77.31     gyros_dumbbell_z
## gyros_forearm_x        73.12      gyros_forearm_x
## gyros_arm_z            55.73          gyros_arm_z
```

```r
predict(model, testing)
```

This simple, almost basic model achieves the perfect 100% accuracy on the testing set.

Conclusion

Model is perfect for the given testing data so any further analysis is not necessary.