Practical Machine Learning Assignment

Executive summary

Analyze the device data from Jawbone Up, Nike FuelBand, and Fitbit for 6 participants, from their ccelerometers on the belt, forearm, arm, and dumbelldata for 5 dataset using linear regression models. This learning will quantify how much of a particular activity they do, but they rarely quantify how well they do it.

Reading the data

The Raw Data - Download the file if does not exist in local system

```
if (!file.exists("./Data/pml-training.csv")){
  download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv",
                "./Data/pml-training.csv")
}
if (!file.exists("./Data/pml-testing.csv")){
  download.file("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv",
                "./Data/pml-testing.csv")
}
```

Load the training and testing data

```
trainingdata = read.csv("./Data/pml-training.csv", na.strings = c("NA", ""))
dim(trainingdata); summary(trainingdata$classe)
```

```
## [1] 19622   160

##    A    B    C    D    E
## 5580 3797 3422 3216 3607
```

```
testingdata = read.csv("./Data/pml-testing.csv", na.strings = c("NA", ""))
```

Load the library

```
library(ggplot2); library(caret);library(randomForest)
```

Removing nerar Zero covariates

```
nzv <- nearZeroVar(trainingdata,saveMetrics=TRUE)
trainingdata <- trainingdata[,nzv$nzv==FALSE]

nzv <- nearZeroVar(testingdata,saveMetrics=TRUE)
testingdata <- testingdata[,nzv$nzv==FALSE]
```

Partioning the training datset

```
inTrain <- createDataPartition(y=trainingdata$classe, p=0.6, list=FALSE)
projTraining <- trainingdata[inTrain, ]; projTesting <- trainingdata[-inTrain, ]
dim(projTraining); dim(projTesting)
```

```
## [1] 11776   117
```

```
## [1] 7846  117
```

Killing first column of Dataset(ID Removing first ID variable) so that it does not interfer with ML Algorithms.

```
projTraining <- projTraining[c(-1)]
```

Remove the columns / Variables has too many NAs (keep only the variable > 60% threshold of NA's)

```
subprojTraining <- projTraining
for(i in 1:length(projTraining)) {
  if( sum( is.na( projTraining[, i] ) ) /nrow(projTraining) >= .6 ) {
    for(j in 1:length(subprojTraining)) {
      if( length( grep(names(projTraining[i]), names(subprojTraining)[j]) ) ==1)  {
        subprojTraining <- subprojTraining[ , -j]
      }
    }
  }
}

#To check the new NA's of observations
dim(subprojTraining); str(subprojTraining)
```

```
## [1] 11776    58
```

```
## 'data.frame':    11776 obs. of  58 variables:
##  $ user_name           : Factor w/ 6 levels "adelmo","carlitos",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ raw_timestamp_part_1: int  1323084231 1323084231 1323084232 1323084232 1323084232 1323084232 13230
##  $ raw_timestamp_part_2: int  788290 820366 304277 440390 484323 500302 528316 560359 576390 604281
##  $ cvtd_timestamp      : Factor w/ 20 levels "02/12/2011 13:32",..: 9 9 9 9 9 9 9 9 9 9 ...
##  $ num_window          : int  11 11 12 12 12 12 12 12 12 12 ...
##  $ roll_belt           : num  1.41 1.42 1.45 1.42 1.43 1.45 1.43 1.42 1.42 1.45 ...
##  $ pitch_belt          : num  8.07 8.07 8.06 8.13 8.16 8.18 8.18 8.2 8.21 8.2 ...
##  $ yaw_belt            : num  -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 -94.4 ...
##  $ total_accel_belt    : int  3 3 3 3 3 3 3 3 3 3 ...
##  $ gyros_belt_x        : num  0 0 0.02 0.02 0.02 0.03 0.02 0.02 0.02 0 ...
##  $ gyros_belt_y        : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ gyros_belt_z        : num  -0.02 -0.02 -0.02 -0.02 -0.02 -0.02 -0.02 0 -0.02 0 ...
```

```
##  $ accel_belt_x        : int  -21 -20 -21 -22 -20 -21 -22 -22 -22 -21 ...
##  $ accel_belt_y        : int  4 5 4 4 2 2 2 4 4 2 ...
##  $ accel_belt_z        : int  22 23 21 21 24 23 23 21 21 22 ...
##  $ magnet_belt_x       : int  -3 -2 0 -2 1 -5 -2 -3 -8 -1 ...
##  $ magnet_belt_y       : int  599 600 603 603 602 596 602 606 598 597 ...
##  $ magnet_belt_z       : int  -313 -305 -312 -313 -312 -317 -319 -309 -310 -310 ...
##  $ roll_arm            : num  -128 -128 -128 -128 -128 -128 -128 -128 -128 -129 ...
##  $ pitch_arm           : num  22.5 22.5 22 21.8 21.7 21.5 21.5 21.4 21.4 21.4 ...
##  $ yaw_arm             : num  -161 -161 -161 -161 -161 -161 -161 -161 -161 -161 ...
##  $ total_accel_arm     : int  34 34 34 34 34 34 34 34 34 34 ...
##  $ gyros_arm_x         : num  0 0.02 0.02 0.02 0.02 0.02 0.02 0.02 0.02 0.02 ...
##  $ gyros_arm_y         : num  0 -0.02 -0.03 -0.02 -0.03 -0.03 -0.03 -0.02 0 0 ...
##  $ gyros_arm_z         : num  -0.02 -0.02 0 0 -0.02 0 0 -0.02 -0.03 -0.03 ...
##  $ accel_arm_x         : int  -288 -289 -289 -289 -288 -290 -288 -287 -288 -289 ...
##  $ accel_arm_y         : int  109 110 111 111 109 110 111 111 111 111 ...
##  $ accel_arm_z         : int  -123 -126 -122 -124 -122 -123 -123 -124 -124 -124 ...
##  $ magnet_arm_x        : int  -368 -368 -369 -372 -369 -366 -363 -372 -371 -374 ...
##  $ magnet_arm_y        : int  337 344 342 338 341 339 343 338 331 342 ...
##  $ magnet_arm_z        : int  516 513 513 510 518 509 520 509 523 510 ...
##  $ roll_dumbbell       : num  13.1 12.9 13.4 12.8 13.2 ...
##  $ pitch_dumbbell      : num  -70.5 -70.3 -70.8 -70.3 -70.4 ...
##  $ yaw_dumbbell        : num  -84.9 -85.1 -84.5 -85.1 -84.9 ...
##  $ total_accel_dumbbell: int  37 37 37 37 37 37 37 37 37 37 ...
##  $ gyros_dumbbell_x    : num  0 0 0 0 0 0 0 0 0.02 0 ...
##  $ gyros_dumbbell_y    : num  -0.02 -0.02 -0.02 -0.02 -0.02 -0.02 -0.02 -0.02 -0.02 -0.02 ...
##  $ gyros_dumbbell_z    : num  0 0 0 0 0 0 0 -0.02 -0.02 0 ...
##  $ accel_dumbbell_x    : int  -234 -232 -234 -234 -232 -233 -233 -234 -234 -234 ...
##  $ accel_dumbbell_y    : int  47 46 48 46 47 47 47 48 48 47 ...
##  $ accel_dumbbell_z    : int  -271 -270 -269 -272 -269 -269 -270 -269 -268 -270 ...
##  $ magnet_dumbbell_x   : int  -559 -561 -558 -555 -549 -564 -554 -552 -554 -554 ...
##  $ magnet_dumbbell_y   : int  293 298 294 300 292 299 291 302 295 294 ...
##  $ magnet_dumbbell_z   : num  -65 -63 -66 -74 -65 -64 -65 -69 -68 -63 ...
##  $ roll_forearm        : num  28.4 28.3 27.9 27.8 27.7 27.6 27.5 27.2 27.2 27.2 ...
##  $ pitch_forearm       : num  -63.9 -63.9 -63.9 -63.8 -63.8 -63.8 -63.8 -63.9 -63.9 -63.9 ...
##  $ yaw_forearm         : num  -153 -152 -152 -152 -152 -152 -152 -151 -151 -151 ...
##  $ total_accel_forearm : int  36 36 36 36 36 36 36 36 36 36 ...
##  $ gyros_forearm_x     : num  0.03 0.03 0.02 0.02 0.03 0.02 0.02 0 0 0 ...
##  $ gyros_forearm_y     : num  0 -0.02 -0.02 -0.02 0 -0.02 0.02 0 -0.02 -0.02 ...
##  $ gyros_forearm_z     : num  -0.02 0 -0.03 0 -0.02 -0.02 -0.03 -0.03 -0.03 -0.02 ...
##  $ accel_forearm_x     : int  192 196 193 193 193 193 191 193 193 192 ...
##  $ accel_forearm_y     : int  203 204 203 205 204 205 203 205 202 201 ...
##  $ accel_forearm_z     : int  -215 -213 -215 -213 -214 -214 -215 -215 -214 -214 ...
##  $ magnet_forearm_x    : int  -17 -18 -9 -9 -16 -17 -11 -15 -14 -16 ...
##  $ magnet_forearm_y    : num  654 658 660 660 653 657 657 655 659 656 ...
##  $ magnet_forearm_z    : num  476 469 478 474 476 465 478 472 478 472 ...
##  $ classe              : Factor w/ 5 levels "A","B","C","D",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```r
clean1 <- colnames(subprojTraining)
clean2 <- colnames(subprojTraining[, -58]) # Remove the classe column
projTesting <- projTesting[clean1]; projTraining <- subprojTraining
testing <- testingdata[clean2]

dim(projTesting); dim(testing)
```

```
## [1] 7846   58
```

```
## [1] 20 57
```

Model Builinding ~ Train model with random forest due to its highly accuracy rate.

```
modFitB1 <- randomForest(classe ~. , data=subprojTraining)
predictionsB1 <- predict(modFitB1, projTesting, type = "class")
confMatrix <- confusionMatrix(predictionsB1, projTesting$classe)
confMatrix
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    A    B    C    D    E
##          A 2232    1    0    0    0
##          B    0 1516    6    0    0
##          C    0    1 1362    5    0
##          D    0    0    0 1281    2
##          E    0    0    0    0 1440
##
## Overall Statistics
##
##                Accuracy : 0.9981
##                  95% CI : (0.9968, 0.9989)
##     No Information Rate : 0.2845
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9976
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E
## Sensitivity            1.0000   0.9987   0.9956   0.9961   0.9986
## Specificity            0.9998   0.9991   0.9991   0.9997   1.0000
## Pos Pred Value         0.9996   0.9961   0.9956   0.9984   1.0000
## Neg Pred Value         1.0000   0.9997   0.9991   0.9992   0.9997
## Prevalence             0.2845   0.1935   0.1744   0.1639   0.1838
## Detection Rate         0.2845   0.1932   0.1736   0.1633   0.1835
## Detection Prevalence   0.2846   0.1940   0.1744   0.1635   0.1835
## Balanced Accuracy      0.9999   0.9989   0.9973   0.9979   0.9993
```

Let's have a look at the accuracy

```
confMatrix$overall[1] # Accuracy - 0.9980882
```

```
##  Accuracy
## 0.9980882
```

It looks very good, it is more then 99.85%. Random Forests yielded better Results, as expected!

Conclusion

The estimate the out of sample error is less than 1% (1 - accuracy). This is a promising result to detect exercise form to quantify how much of a particular activity they do and effective.