Predicting Types of Activities from Body Accelerometers

J.E. Panzik **June 12, 2020**

System Details

The following was run on:
x86_64-apple-darwin15.6.0
R version 3.6.3 (2020-02-29)

Data Information

The data used for this project comes from http://groupware.les.inf.puc-rio.br/har, and contains information from accelerometers that are used to classify motion/activity types of the participants.

Reading & Cleaning the Data

Import the data straight from the website. The data was downloaded: **June 12, 2020 17:55:18**

Data is imported setting both NA and blanks as NA values since both exist in the imported data.

```
trainRaw <- read.csv("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-training.csv", na.st
validRaw <- read.csv("https://d396qusza40orc.cloudfront.net/predmachlearn/pml-testing.csv", na.str
```

```
dim(trainRaw)
```

```
## [1] 19622   160
```

```
str(trainRaw)
```

```
## 'data.frame':    19622 obs. of  160 variables:
##  $ X                       : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ 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 1323084231 1323084232 1323084232 132308
##  $ raw_timestamp_part_2    : int  788290 808298 820366 120339 196328 304277 368296 440390 48432
##  $ cvtd_timestamp          : Factor w/ 20 levels "02/12/2011 13:32",..: 9 9 9 9 9 9 9 9 9 9 ...
##  $ new_window              : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ num_window              : int  11 11 11 12 12 12 12 12 12 12 ...
##  $ roll_belt               : num  1.41 1.41 1.42 1.48 1.48 1.45 1.42 1.42 1.43 1.45 ...
##  $ pitch_belt              : num  8.07 8.07 8.07 8.05 8.07 8.06 8.09 8.13 8.16 8.17 ...
##  $ 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 ...
##  $ kurtosis_roll_belt      : Factor w/ 396 levels "-0.016850","-0.021024",..: NA NA NA NA NA NA
##  $ kurtosis_picth_belt     : Factor w/ 316 levels "-0.021887","-0.060755",..: NA NA NA NA NA NA
##  $ kurtosis_yaw_belt       : Factor w/ 1 level "#DIV/0!": NA NA NA NA NA NA NA NA NA NA ...
##  $ skewness_roll_belt      : Factor w/ 394 levels "-0.003095","-0.010002",..: NA NA NA NA NA NA
##  $ skewness_roll_belt.1    : Factor w/ 337 levels "-0.005928","-0.005960",..: NA NA NA NA NA NA
##  $ skewness_yaw_belt       : Factor w/ 1 level "#DIV/0!": NA NA NA NA NA NA NA NA NA NA ...
##  $ max_roll_belt           : num  NA NA NA NA NA NA NA NA NA NA ...
```

```
##  $ max_picth_belt          : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_yaw_belt            : Factor w/ 67 levels "-0.1","-0.2",..: NA NA NA NA NA NA NA NA NA N
##  $ min_roll_belt           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_pitch_belt          : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_yaw_belt            : Factor w/ 67 levels "-0.1","-0.2",..: NA NA NA NA NA NA NA NA NA N
##  $ amplitude_roll_belt     : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_pitch_belt    : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_yaw_belt      : Factor w/ 3 levels "#DIV/0!","0.00",..: NA NA NA NA NA NA NA NA NA
##  $ var_total_accel_belt    : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_roll_belt           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_roll_belt        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_roll_belt           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_pitch_belt          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_pitch_belt       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_pitch_belt          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_yaw_belt            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_yaw_belt         : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_yaw_belt            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ gyros_belt_x            : num  0 0.02 0 0.02 0.02 0.02 0.02 0.02 0.02 0.03 ...
##  $ gyros_belt_y            : num  0 0 0 0 0.02 0 0 0 0 0 ...
##  $ gyros_belt_z            : num  -0.02 -0.02 -0.02 -0.03 -0.02 -0.02 -0.02 -0.02 -0.02 0 ...
##  $ accel_belt_x            : int  -21 -22 -20 -22 -21 -21 -22 -22 -20 -21 ...
##  $ accel_belt_y            : int  4 4 5 3 2 4 3 4 2 4 ...
##  $ accel_belt_z            : int  22 22 23 21 24 21 21 21 24 22 ...
##  $ magnet_belt_x           : int  -3 -7 -2 -6 -6 0 -4 -2 1 -3 ...
##  $ magnet_belt_y           : int  599 608 600 604 600 603 599 603 602 609 ...
##  $ magnet_belt_z           : int  -313 -311 -305 -310 -302 -312 -311 -313 -312 -308 ...
##  $ roll_arm                : num  -128 -128 -128 -128 -128 -128 -128 -128 -128 -128 ...
##  $ pitch_arm               : num  22.5 22.5 22.5 22.1 22.1 22 21.9 21.8 21.7 21.6 ...
##  $ 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 ...
##  $ var_accel_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_roll_arm            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_roll_arm         : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_roll_arm            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_pitch_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_pitch_arm        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_pitch_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_yaw_arm             : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_yaw_arm          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_yaw_arm             : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ gyros_arm_x             : num  0 0.02 0.02 0.02 0 0.02 0 0.02 0.02 0.02 ...
##  $ gyros_arm_y             : num  0 -0.02 -0.02 -0.03 -0.03 -0.03 -0.03 -0.02 -0.03 -0.03 ...
##  $ gyros_arm_z             : num  -0.02 -0.02 -0.02 0.02 0 0 0 0 -0.02 -0.02 ...
##  $ accel_arm_x             : int  -288 -290 -289 -289 -289 -289 -289 -289 -288 -288 ...
##  $ accel_arm_y             : int  109 110 110 111 111 111 111 111 109 110 ...
##  $ accel_arm_z             : int  -123 -125 -126 -123 -123 -122 -125 -124 -122 -124 ...
##  $ magnet_arm_x            : int  -368 -369 -368 -372 -374 -369 -373 -372 -369 -376 ...
##  $ magnet_arm_y            : int  337 337 344 344 337 342 336 338 341 334 ...
##  $ magnet_arm_z            : int  516 513 513 512 506 513 509 510 518 516 ...
```

```
##  $ kurtosis_roll_arm       : Factor w/ 329 levels "-0.02438","-0.04190",..: NA NA NA NA NA NA N
##  $ kurtosis_picth_arm      : Factor w/ 327 levels "-0.00484","-0.01311",..: NA NA NA NA NA NA N
##  $ kurtosis_yaw_arm        : Factor w/ 394 levels "-0.01548","-0.01749",..: NA NA NA NA NA NA N
##  $ skewness_roll_arm       : Factor w/ 330 levels "-0.00051","-0.00696",..: NA NA NA NA NA NA N
##  $ skewness_pitch_arm      : Factor w/ 327 levels "-0.00184","-0.01185",..: NA NA NA NA NA NA N
##  $ skewness_yaw_arm        : Factor w/ 394 levels "-0.00311","-0.00562",..: NA NA NA NA NA NA N
##  $ max_roll_arm            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_picth_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_yaw_arm             : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_roll_arm            : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_pitch_arm           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_yaw_arm             : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_roll_arm      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_pitch_arm     : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_yaw_arm       : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ roll_dumbbell           : num  13.1 13.1 12.9 13.4 13.4 ...
##  $ pitch_dumbbell          : num  -70.5 -70.6 -70.3 -70.4 -70.4 ...
##  $ yaw_dumbbell            : num  -84.9 -84.7 -85.1 -84.9 -84.9 ...
##  $ kurtosis_roll_dumbbell  : Factor w/ 397 levels "-0.0035","-0.0073",..: NA NA NA NA NA NA NA
##  $ kurtosis_picth_dumbbell : Factor w/ 400 levels "-0.0163","-0.0233",..: NA NA NA NA NA NA NA
##  $ kurtosis_yaw_dumbbell   : Factor w/ 1 level "#DIV/0!": NA NA NA NA NA NA NA NA NA NA ...
##  $ skewness_roll_dumbbell  : Factor w/ 400 levels "-0.0082","-0.0096",..: NA NA NA NA NA NA NA
##  $ skewness_pitch_dumbbell : Factor w/ 401 levels "-0.0053","-0.0084",..: NA NA NA NA NA NA NA
##  $ skewness_yaw_dumbbell   : Factor w/ 1 level "#DIV/0!": NA NA NA NA NA NA NA NA NA NA ...
##  $ max_roll_dumbbell       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_picth_dumbbell      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_yaw_dumbbell        : Factor w/ 72 levels "-0.1","-0.2",..: NA NA NA NA NA NA NA NA NA N
##  $ min_roll_dumbbell       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_pitch_dumbbell      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_yaw_dumbbell        : Factor w/ 72 levels "-0.1","-0.2",..: NA NA NA NA NA NA NA NA NA N
##  $ amplitude_roll_dumbbell : num  NA NA NA NA NA NA NA NA NA NA ...
##   [list output truncated]
```

```
NA_perc <- 100*sum(is.na(trainRaw))/(dim(trainRaw)[1]*dim(trainRaw)[2])
```

The training data has **19622** rows and **160** columns, for a total of **3139520** data entries. Of these entries, **1921600 (61.2068087%)** are NA values.

The investigate if missing data is related to specific activity types ($classe) shows that all activity types appear to have the same proportion of NA values.

```
colnames(trainRaw[, colSums(is.na(trainRaw))/dim(trainRaw)[1] >=0.97])
```

```
##   [1] "kurtosis_roll_belt"       "kurtosis_picth_belt"
##   [3] "kurtosis_yaw_belt"        "skewness_roll_belt"
##   [5] "skewness_roll_belt.1"     "skewness_yaw_belt"
##   [7] "max_roll_belt"            "max_picth_belt"
##   [9] "max_yaw_belt"             "min_roll_belt"
##  [11] "min_pitch_belt"           "min_yaw_belt"
##  [13] "amplitude_roll_belt"      "amplitude_pitch_belt"
##  [15] "amplitude_yaw_belt"       "var_total_accel_belt"
##  [17] "avg_roll_belt"            "stddev_roll_belt"
```

```
##  [19] "var_roll_belt"            "avg_pitch_belt"
##  [21] "stddev_pitch_belt"        "var_pitch_belt"
##  [23] "avg_yaw_belt"             "stddev_yaw_belt"
##  [25] "var_yaw_belt"             "var_accel_arm"
##  [27] "avg_roll_arm"             "stddev_roll_arm"
##  [29] "var_roll_arm"             "avg_pitch_arm"
##  [31] "stddev_pitch_arm"         "var_pitch_arm"
##  [33] "avg_yaw_arm"              "stddev_yaw_arm"
##  [35] "var_yaw_arm"              "kurtosis_roll_arm"
##  [37] "kurtosis_picth_arm"       "kurtosis_yaw_arm"
##  [39] "skewness_roll_arm"        "skewness_pitch_arm"
##  [41] "skewness_yaw_arm"         "max_roll_arm"
##  [43] "max_picth_arm"            "max_yaw_arm"
##  [45] "min_roll_arm"             "min_pitch_arm"
##  [47] "min_yaw_arm"              "amplitude_roll_arm"
##  [49] "amplitude_pitch_arm"      "amplitude_yaw_arm"
##  [51] "kurtosis_roll_dumbbell"   "kurtosis_picth_dumbbell"
##  [53] "kurtosis_yaw_dumbbell"    "skewness_roll_dumbbell"
##  [55] "skewness_pitch_dumbbell"  "skewness_yaw_dumbbell"
##  [57] "max_roll_dumbbell"        "max_picth_dumbbell"
##  [59] "max_yaw_dumbbell"         "min_roll_dumbbell"
##  [61] "min_pitch_dumbbell"       "min_yaw_dumbbell"
##  [63] "amplitude_roll_dumbbell"  "amplitude_pitch_dumbbell"
##  [65] "amplitude_yaw_dumbbell"   "var_accel_dumbbell"
##  [67] "avg_roll_dumbbell"        "stddev_roll_dumbbell"
##  [69] "var_roll_dumbbell"        "avg_pitch_dumbbell"
##  [71] "stddev_pitch_dumbbell"    "var_pitch_dumbbell"
##  [73] "avg_yaw_dumbbell"         "stddev_yaw_dumbbell"
##  [75] "var_yaw_dumbbell"         "kurtosis_roll_forearm"
##  [77] "kurtosis_picth_forearm"   "kurtosis_yaw_forearm"
##  [79] "skewness_roll_forearm"    "skewness_pitch_forearm"
##  [81] "skewness_yaw_forearm"     "max_roll_forearm"
##  [83] "max_picth_forearm"        "max_yaw_forearm"
##  [85] "min_roll_forearm"         "min_pitch_forearm"
##  [87] "min_yaw_forearm"          "amplitude_roll_forearm"
##  [89] "amplitude_pitch_forearm"  "amplitude_yaw_forearm"
##  [91] "var_accel_forearm"        "avg_roll_forearm"
##  [93] "stddev_roll_forearm"      "var_roll_forearm"
##  [95] "avg_pitch_forearm"        "stddev_pitch_forearm"
##  [97] "var_pitch_forearm"        "avg_yaw_forearm"
##  [99] "stddev_yaw_forearm"       "var_yaw_forearm"
```

```
colnames(trainRaw[, colSums(is.na(trainRaw))/dim(trainRaw)[1] >=0.98])
```

```
## character(0)
```

The missing data is concentrated in 100 columns of the data and show that they are missing 97-98% of the data. These columns are removed from the training set and test set. The first 7 columns are also removed because they do not contribute anything to determining what type of activity is being performed. The $classe column is already a factor variable and does not need to be converted. The imported validation/test data has an additional column of problem_id at the end which will be removed.

```
include <- which(colSums(is.na(trainRaw))<0.95*dim(trainRaw)[1])
trainClean <- trainRaw[,include]
trainClean <- trainClean[,-c(1:7)]

validClean <- validRaw[,include]
validClean <- validClean[, -c(1:7)]
validClean <- validClean[, -dim(validClean)[2]]
```

Create Training Models

The training data will be fit using 2 model types and compare the relative accuracy of each:
-Random Forest (rf)
-Gradiant Boosting Method (gbm)

The training set is broken up into a training and initial test set.

```
set.seed(2425)
library(caret)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
inTrain <- createDataPartition(trainClean$classe, p=0.70, list=FALSE)
train <- trainClean[inTrain, ]
test <- trainClean[-inTrain, ]

cv <- trainControl(method="cv", 5)
rf <- train(classe~., data=train, method="rf", trControl=cv, verbose=FALSE, ntree=250)

gbm <- train(classe~., data=train, method="gbm", trControl=cv, verbose=FALSE)
```

Testing Models

```
library(caret); library(knitr)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

```
rf_predict <- predict(rf, newdata=test)
rf_acc <- confusionMatrix(test$classe, rf_predict)$overall['Accuracy']*100

gbm_predict <- predict(gbm, newdata=test)
gbm_acc <- confusionMatrix(test$classe, gbm_predict)$overall['Accuracy']*100
```

Random Forest Model

The random forest model fit the subset of training data used as a test with **99.3033135%** accuracy, and an estimated out of sample error of **0.6966865%**.

The predicted vs actual activities are shown in the table below.

	A	B	C	D	E
A	1672	1	0	0	1
B	6	1133	0	0	0
C	0	7	1019	0	0
D	0	0	24	939	1
E	0	0	0	1	1081

Gradiant Boosting Method Model

The gradiant boosting method fit the subset of training data used as a test with **96.4316058%** accuracy, and an estimated out of sample error of **3.5683942%**.

The predicted vs actual activities are shown in the table below.

	A	B	C	D	E
A	1655	8	7	3	1
B	45	1065	29	0	0
C	0	29	991	5	1
D	0	4	34	917	9
E	2	10	11	12	1047

Model Conclusions

The comparison between the two model methods shows that the random forest has the the highest accuracy, with very few misclassifications. The gradiant boosting method still has >95% accuracy, but the misclassifications are widely spread out. The random forest will be used as a more robust method of classifying activitiy types.

Applying the Random Forest Model to the Provided Test Data

The results from the random forest model will be applied to the provided test data that was labelled as validation data. This will sort the data into activity types.

```
valid_predict <- predict(rf, newdata=validClean)
valid_predict
```

```
##  [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
```