WeightLiftMachineLearning.Rmd

Jim Callahan

October 18-23, 2015

Executive Summary

People in the "quantified self movement" often quantify how much of a particular activity they do, but they rarely quantify how well they do it.

Telemetry data from weight lifters was gathered in a project described in "**Qualitative Activity Recognition of Weight Lifting Exercises**" a paper presented by Velloso, Bulling, Gellersen, Ugulino and Fuks at the **Augmented Human '13** ACM conference held in Stuttgart, Germany. Their paper is available on the web at: http://groupware.les.inf.puc-rio.br/public/papers/2013.Velloso.QAR-WLE.pdf

The goal of their project and this project is to build a model to predict the manner in which 6 participants did a weight lifting exercise from data gathered from from accelerometers (and other sensors) on the belt, arm and forearm of the participants as well as on the dumbell they lifted.

The participants were asked to perform barbell lifts correctly and then incorrectly in 5 different ways. The correct lift (labeled "A") and the four incorrect lifts (labeled "B" through "E") form the "**classe**" variable in the training set. The model predicting the "**classe**" variable was built using a subset of the variables in the training set.

The authors describe their instrumentation as: "For data recording we used four 9 degrees of freedom **Razor inertial measurement units (IMU)**, which provide three-axes acceleration, gyroscope and magnetometer data at a joint sampling rate of 45 Hz."

Exploratory Data Analysis (EDA)

The first step in working with new data is to verify the data matches any description supplied with the data. In this project there was a substantial mismatch between the published article and the data supplied so an extensive reconcilation process had to be undertaken.

The training data file provided on the website consisted of 160 variables. There was no data dictionary provided for the training or test files.

Therefore the data file had to be read in and selectively dumped, so reasonable assumptions could be made about what data to include in the analysis.

```
# read in data
rawtraining <- read.csv("~\\GitHub\\MachineLearning\\Data\\pml-training.csv",
                        stringsAsFactors = FALSE, na.strings = c("NA", ""))
# dump the names of the variables
names(rawtraining)
```

```
##   [1] "X"                        "user_name"
##   [3] "raw_timestamp_part_1"     "raw_timestamp_part_2"
##   [5] "cvtd_timestamp"           "new_window"
##   [7] "num_window"               "roll_belt"
##   [9] "pitch_belt"               "yaw_belt"
##  [11] "total_accel_belt"         "kurtosis_roll_belt"
##  [13] "kurtosis_picth_belt"      "kurtosis_yaw_belt"
##  [15] "skewness_roll_belt"       "skewness_roll_belt.1"
```

```
##  [17] "skewness_yaw_belt"        "max_roll_belt"
##  [19] "max_picth_belt"           "max_yaw_belt"
##  [21] "min_roll_belt"            "min_pitch_belt"
##  [23] "min_yaw_belt"             "amplitude_roll_belt"
##  [25] "amplitude_pitch_belt"     "amplitude_yaw_belt"
##  [27] "var_total_accel_belt"     "avg_roll_belt"
##  [29] "stddev_roll_belt"         "var_roll_belt"
##  [31] "avg_pitch_belt"           "stddev_pitch_belt"
##  [33] "var_pitch_belt"           "avg_yaw_belt"
##  [35] "stddev_yaw_belt"          "var_yaw_belt"
##  [37] "gyros_belt_x"             "gyros_belt_y"
##  [39] "gyros_belt_z"             "accel_belt_x"
##  [41] "accel_belt_y"             "accel_belt_z"
##  [43] "magnet_belt_x"            "magnet_belt_y"
##  [45] "magnet_belt_z"            "roll_arm"
##  [47] "pitch_arm"                "yaw_arm"
##  [49] "total_accel_arm"          "var_accel_arm"
##  [51] "avg_roll_arm"             "stddev_roll_arm"
##  [53] "var_roll_arm"             "avg_pitch_arm"
##  [55] "stddev_pitch_arm"         "var_pitch_arm"
##  [57] "avg_yaw_arm"              "stddev_yaw_arm"
##  [59] "var_yaw_arm"              "gyros_arm_x"
##  [61] "gyros_arm_y"              "gyros_arm_z"
##  [63] "accel_arm_x"              "accel_arm_y"
##  [65] "accel_arm_z"              "magnet_arm_x"
##  [67] "magnet_arm_y"             "magnet_arm_z"
##  [69] "kurtosis_roll_arm"        "kurtosis_picth_arm"
##  [71] "kurtosis_yaw_arm"         "skewness_roll_arm"
##  [73] "skewness_pitch_arm"       "skewness_yaw_arm"
##  [75] "max_roll_arm"             "max_picth_arm"
##  [77] "max_yaw_arm"              "min_roll_arm"
##  [79] "min_pitch_arm"            "min_yaw_arm"
##  [81] "amplitude_roll_arm"       "amplitude_pitch_arm"
##  [83] "amplitude_yaw_arm"        "roll_dumbbell"
##  [85] "pitch_dumbbell"           "yaw_dumbbell"
##  [87] "kurtosis_roll_dumbbell"   "kurtosis_picth_dumbbell"
##  [89] "kurtosis_yaw_dumbbell"    "skewness_roll_dumbbell"
##  [91] "skewness_pitch_dumbbell"  "skewness_yaw_dumbbell"
##  [93] "max_roll_dumbbell"        "max_picth_dumbbell"
##  [95] "max_yaw_dumbbell"         "min_roll_dumbbell"
##  [97] "min_pitch_dumbbell"       "min_yaw_dumbbell"
##  [99] "amplitude_roll_dumbbell"  "amplitude_pitch_dumbbell"
## [101] "amplitude_yaw_dumbbell"   "total_accel_dumbbell"
## [103] "var_accel_dumbbell"       "avg_roll_dumbbell"
## [105] "stddev_roll_dumbbell"     "var_roll_dumbbell"
## [107] "avg_pitch_dumbbell"       "stddev_pitch_dumbbell"
## [109] "var_pitch_dumbbell"       "avg_yaw_dumbbell"
## [111] "stddev_yaw_dumbbell"      "var_yaw_dumbbell"
## [113] "gyros_dumbbell_x"         "gyros_dumbbell_y"
## [115] "gyros_dumbbell_z"         "accel_dumbbell_x"
## [117] "accel_dumbbell_y"         "accel_dumbbell_z"
## [119] "magnet_dumbbell_x"        "magnet_dumbbell_y"
## [121] "magnet_dumbbell_z"        "roll_forearm"
## [123] "pitch_forearm"            "yaw_forearm"
```

```
## [125] "kurtosis_roll_forearm"    "kurtosis_picth_forearm"
## [127] "kurtosis_yaw_forearm"     "skewness_roll_forearm"
## [129] "skewness_pitch_forearm"   "skewness_yaw_forearm"
## [131] "max_roll_forearm"         "max_picth_forearm"
## [133] "max_yaw_forearm"          "min_roll_forearm"
## [135] "min_pitch_forearm"        "min_yaw_forearm"
## [137] "amplitude_roll_forearm"   "amplitude_pitch_forearm"
## [139] "amplitude_yaw_forearm"    "total_accel_forearm"
## [141] "var_accel_forearm"        "avg_roll_forearm"
## [143] "stddev_roll_forearm"      "var_roll_forearm"
## [145] "avg_pitch_forearm"        "stddev_pitch_forearm"
## [147] "var_pitch_forearm"        "avg_yaw_forearm"
## [149] "stddev_yaw_forearm"       "var_yaw_forearm"
## [151] "gyros_forearm_x"          "gyros_forearm_y"
## [153] "gyros_forearm_z"          "accel_forearm_x"
## [155] "accel_forearm_y"          "accel_forearm_z"
## [157] "magnet_forearm_x"         "magnet_forearm_y"
## [159] "magnet_forearm_z"         "classe"
```

The first seven variables appear to be the "coordinates" of the rest of the data. In order to subset the data, the first seven variables will be refered to as "Block0" (block zero). Zero because there will be four additional data blocks one through four.

```
Block0 <- c(1:7)
str(rawtraining[ , Block0])
```

```
## 'data.frame':    19622 obs. of  7 variables:
##  $ X                    : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ user_name            : chr  "carlitos" "carlitos" "carlitos" "carlitos" ...
##  $ raw_timestamp_part_1: int  1323084231 1323084231 1323084231 1323084232 1323084232 1323084232 1323
##  $ raw_timestamp_part_2: int  788290 808298 820366 120339 196328 304277 368296 440390 484323 484434
##  $ cvtd_timestamp       : chr  "05/12/2011 11:23" "05/12/2011 11:23" "05/12/2011 11:23" "05/12/2011 1
##  $ new_window           : chr  "no" "no" "no" "no" ...
##  $ num_window           : int  11 11 11 12 12 12 12 12 12 12 ...
```

X: sequence number
The variable "**X**" appears to be a sequence number.

```
str(rawtraining$X)
```

```
##  int [1:19622] 1 2 3 4 5 6 7 8 9 10 ...
```

```
summary(rawtraining$X)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       1    4906    9812    9812   14720   19620
```

The data set has 19,622 observations (rows) and the **X** variable only runs from 1 to 19,620 (off by 2), but we can't sweat the small stuff. We probably need to exclude the sequence number from the analysis anyway to avoid potential "data leakage".

User name: the six participants
The six participants are identified in the variable, "**user_name**"

```
summary(as.factor(rawtraining$user_name))
```

```
##   adelmo carlitos  charles   eurico   jeremy    pedro
##     3892     3112     3536     3070     3402     2610
```

The numbers underneath each name are the number of observations (rows) associated with each name. Although the numbers are not the same, it appears to reasonably well balanced. There does not appear to be an impossible to overcome class imbalance.

The target: classe
The last variable, "**classe**", should be the labels "A" through "E" indicating the correct ("A") and incorrect ("B" through "E") weight lifts.

```
summary(as.factor(rawtraining$classe))
```

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

According to the paper, class "**A**" corresponds to the [correct] specified execution of the exercise, while the other 4 classes correspond to common mistakes."

A. Correct – "exactly according to the specification"
B. Incorrect – "throwing the elbows to the front"
C. Incorrect – "lifting the dumbbell only halfway"
D. Incorrect – "lowering the dumbbell only halfway"
E. Incorrect – "throwing the hips to the front"

The "**user_name**", "**classe**" and "**new_window**" variables should be permanently converted to factors.

```
rawtraining$user_name <- as.factor(rawtraining$user_name)
rawtraining$classe    <- as.factor(rawtraining$classe)
rawtraining$new_window <- as.factor(rawtraining$new_window)
```

We should be able to tabulate "**user_name**" by "**classe**"

```
t1 <- table(rawtraining$user_name, rawtraining$classe)
t1
```

```
##
##               A    B    C    D    E
##   adelmo   1165  776  750  515  686
##   carlitos  834  690  493  486  609
##   charles   899  745  539  642  711
##   eurico    865  592  489  582  542
##   jeremy   1177  489  652  522  562
##   pedro     640  505  499  469  497
```

```
# row proportions
round(prop.table(t1, 1),2)
```

```
##
##               A    B    C    D    E
##   adelmo   0.30 0.20 0.19 0.13 0.18
##   carlitos 0.27 0.22 0.16 0.16 0.20
##   charles  0.25 0.21 0.15 0.18 0.20
##   eurico   0.28 0.19 0.16 0.19 0.18
##   jeremy   0.35 0.14 0.19 0.15 0.17
##   pedro    0.25 0.19 0.19 0.18 0.19
```

```
# column proportions
round(prop.table(t1, 2),2)
```

```
##
##               A    B    C    D    E
##   adelmo   0.21 0.20 0.22 0.16 0.19
##   carlitos 0.15 0.18 0.14 0.15 0.17
##   charles  0.16 0.20 0.16 0.20 0.20
##   eurico   0.16 0.16 0.14 0.18 0.15
##   jeremy   0.21 0.13 0.19 0.16 0.16
##   pedro    0.11 0.13 0.15 0.15 0.14
```

Window Number?
Each unique exercise (for which there may be serveral measurement observations) is presumably tracked by the "**num_window**" variable:

```
length(unique(rawtraining$num_window))
```

```
## [1] 858
```

There seeem to be 858 "windows" numbered 1 through 864 with 6 "windows" not included. We should be able to tabulate "**user_name**" by "**num_window**" (omitted).

```
# Multiple pages of output omitted
# t2 <- table(rawtraining$user_name, rawtraining$num_window)
# t2
```

Only one participant appears to be active in each "window" (all of the other observations are zero).

We can also look at "**classe**" by "**num_window**" (omitted).

```
# Multiple pages of output omitted
# t3 <- table(rawtraining$classe, rawtraining$num_window)
# t3
```

Date and Time
There are three date or time variables in columns #3, #4 and #5.

```
str(rawtraining[ , 3:5])
```

```
## 'data.frame':    19622 obs. of  3 variables:
##  $ raw_timestamp_part_1: int  1323084231 1323084231 1323084231 1323084232 1323084232 1323084232 1323
##  $ raw_timestamp_part_2: int  788290 808298 820366 120339 196328 304277 368296 440390 484323 484434
##  $ cvtd_timestamp      : chr  "05/12/2011 11:23" "05/12/2011 11:23" "05/12/2011 11:23" "05/12/2011 1
```

```r
#
# Look at the time and "window" and variables.
timewindowvars <- c("new_window", "num_window", "raw_timestamp_part_1", "raw_timestamp_part_2",
                    "cvtd_timestamp")
head(rawtraining[ , timewindowvars], 26)
```

```
##    new_window num_window raw_timestamp_part_1 raw_timestamp_part_2
## 1          no         11           1323084231               788290
## 2          no         11           1323084231               808298
## 3          no         11           1323084231               820366
## 4          no         12           1323084232               120339
## 5          no         12           1323084232               196328
## 6          no         12           1323084232               304277
## 7          no         12           1323084232               368296
## 8          no         12           1323084232               440390
## 9          no         12           1323084232               484323
## 10         no         12           1323084232               484434
## 11         no         12           1323084232               500302
## 12         no         12           1323084232               528316
## 13         no         12           1323084232               560359
## 14         no         12           1323084232               576390
## 15         no         12           1323084232               604281
## 16         no         12           1323084232               644302
## 17         no         12           1323084232               692324
## 18         no         12           1323084232               732306
## 19         no         12           1323084232               740353
## 20         no         12           1323084232               788335
## 21         no         12           1323084232               876301
## 22         no         12           1323084232               892313
## 23         no         12           1323084232               932285
## 24        yes         12           1323084232               996313
## 25         no         13           1323084233                28311
## 26         no         13           1323084233                56286
##      cvtd_timestamp
## 1  05/12/2011 11:23
## 2  05/12/2011 11:23
## 3  05/12/2011 11:23
## 4  05/12/2011 11:23
## 5  05/12/2011 11:23
## 6  05/12/2011 11:23
## 7  05/12/2011 11:23
## 8  05/12/2011 11:23
## 9  05/12/2011 11:23
## 10 05/12/2011 11:23
## 11 05/12/2011 11:23
## 12 05/12/2011 11:23
## 13 05/12/2011 11:23
## 14 05/12/2011 11:23
## 15 05/12/2011 11:23
## 16 05/12/2011 11:23
## 17 05/12/2011 11:23
## 18 05/12/2011 11:23
## 19 05/12/2011 11:23
```

```
## 20 05/12/2011 11:23
## 21 05/12/2011 11:23
## 22 05/12/2011 11:23
## 23 05/12/2011 11:23
## 24 05/12/2011 11:23
## 25 05/12/2011 11:23
## 26 05/12/2011 11:23
```

Even the first six rows are problematic. Notice that the "**cvtd_timestamp**" and "**new_window**" are the same, but the window number, "**num_window**" changes. Why?

The change in "**num_window**" seems to be correlated with a change in part 1 of the the timestamp, "**raw_timestamp_part_1**". Unclear, what if anything, this means.

```
summary(rawtraining[ , timewindowvars])
```

```
##  new_window    num_window    raw_timestamp_part_1 raw_timestamp_part_2
##  no :19216   Min.   :  1.0   Min.   :1.322e+09    Min.   :   294
##  yes:  406   1st Qu.:222.0   1st Qu.:1.323e+09    1st Qu.:252912
##              Median :424.0   Median :1.323e+09    Median :496380
##              Mean   :430.6   Mean   :1.323e+09    Mean   :500656
##              3rd Qu.:644.0   3rd Qu.:1.323e+09    3rd Qu.:751891
##              Max.   :864.0   Max.   :1.323e+09    Max.   :998801
##  cvtd_timestamp
##  Length:19622
##  Class :character
##  Mode  :character
##
##
##
```

(Other) :11007

The converted timestamp variable, "**cvtd_timestamp**" has 11,007 "Other" values. What is "Other"? and why does the converted timestamp variable have "Other" when the part_1 and part_2 variables appear to be complete? Is "Other" blank, do I need to change how I read in the converted timestamp variable?

The other 152 variables
In the 160 variable data set, beyond "**classe**" and the 7 identifcation variables, the remaining 152 variables seem to be made up of **4 blocks** of **38 variables** each. This is different from the **96 features** described in the paper, but we have to work with the data we have and not rely on (differing) data descriptions in the paper.

```
4*38
```

```
## [1] 152
```

The **first block of 38 variables**, variables #8 through #45 appear to relate to the sensor on the **belt** of the participant.

```
# First Block of 38
str(rawtraining[ , 8:45])
```

```
## 'data.frame':    19622 obs. of  38 variables:
##  $ 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 ...
##  $ kurtosis_roll_belt   : chr  NA NA NA NA ...
##  $ kurtosis_picth_belt  : chr  NA NA NA NA ...
##  $ kurtosis_yaw_belt    : chr  NA NA NA NA ...
##  $ skewness_roll_belt   : chr  NA NA NA NA ...
##  $ skewness_roll_belt.1 : chr  NA NA NA NA ...
##  $ skewness_yaw_belt    : chr  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         : chr  NA NA NA NA ...
##  $ 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         : chr  NA NA NA NA ...
##  $ 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   : chr  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 ...
```

We want the raw accelerometer, magnetometer and gyroscopic data as well as the euler angle data (which may be synthesized from several measurements, but is not otherwise transformed).

So, from the first block we want the first four (#8, #9, #10 and #11) and the last nine (#37, #38, #39, #40, #41, #42, #43, #44, #45).

```
# First Block: untransformed data and Euler angle data
Block1 <- c(8:11, 37:45)
str(rawtraining[ , Block1])
```

```
## 'data.frame':    19622 obs. of  13 variables:
##  $ 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 ...
##  $ 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 ...
```

The **second block of 38 variables**, variables #46 through #83 appear to relate to the armband sensor on the **arm** of the participant.

```
str(rawtraining[ , 46:83])
```

```
## 'data.frame':    19622 obs. of  38 variables:
##  $ 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  : chr  NA NA NA NA ...
##  $ kurtosis_picth_arm : chr  NA NA NA NA ...
```

```
##  $ kurtosis_yaw_arm   : chr  NA NA NA NA ...
##  $ skewness_roll_arm  : chr  NA NA NA NA ...
##  $ skewness_pitch_arm : chr  NA NA NA NA ...
##  $ skewness_yaw_arm   : chr  NA NA NA NA ...
##  $ 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 ...
```

From the second block we again want the first four (#46, #47, #48 and #49) and but, the order of varaibles has changed. We want to skip 10 and the pick up the next nine (#60, #61, #62, #63, #64, #65, #66, #67 and #68), then skip the rest.

```
# Second Block: untransformed data and Euler angle data
Block2 <- c(46:49, 60:68)
str(rawtraining[ , Block2])
```

```
## 'data.frame':    19622 obs. of  13 variables:
##  $ 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 ...
##  $ 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 ...
```

The **third block of 38 variables**, variables #84 through #121 appear to relate to the sensor on the **dumbell** weight lifted by the participant.

```
str(rawtraining[ , 84:121])
```

```
## 'data.frame':    19622 obs. of  38 variables:
##  $ 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   : chr  NA NA NA NA ...
##  $ kurtosis_picth_dumbbell  : chr  NA NA NA NA ...
##  $ kurtosis_yaw_dumbbell    : chr  NA NA NA NA ...
##  $ skewness_roll_dumbbell   : chr  NA NA NA NA ...
```

```
##  $ skewness_pitch_dumbbell : chr  NA NA NA NA ...
##  $ skewness_yaw_dumbbell   : chr  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        : chr  NA NA NA NA ...
##  $ 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        : chr  NA NA NA NA ...
##  $ amplitude_roll_dumbbell : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_pitch_dumbbell: num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_yaw_dumbbell  : chr  NA NA NA NA ...
##  $ total_accel_dumbbell    : int  37 37 37 37 37 37 37 37 37 37 ...
##  $ var_accel_dumbbell      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_roll_dumbbell       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_roll_dumbbell    : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_roll_dumbbell       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_pitch_dumbbell      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_pitch_dumbbell   : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_pitch_dumbbell      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_yaw_dumbbell        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_yaw_dumbbell     : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_yaw_dumbbell        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ gyros_dumbbell_x        : num  0 0 0 0 0 0 0 0 0 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 -0.02 0 0 0 0 0 0 ...
##  $ accel_dumbbell_x        : int  -234 -233 -232 -232 -233 -234 -232 -234 -232 -235 ...
##  $ accel_dumbbell_y        : int  47 47 46 48 48 48 47 46 47 48 ...
##  $ accel_dumbbell_z        : int  -271 -269 -270 -269 -270 -269 -270 -272 -269 -270 ...
##  $ magnet_dumbbell_x       : int  -559 -555 -561 -552 -554 -558 -551 -555 -549 -558 ...
##  $ magnet_dumbbell_y       : int  293 296 298 303 292 294 295 300 292 291 ...
##  $ magnet_dumbbell_z       : num  -65 -64 -63 -60 -68 -66 -70 -74 -65 -69 ...
```

From the third block the order of variables has changed again, so we only want the first three (#84, #85 and #86) and then We want to skip 15 and the pick up only one (#102) and then skip another 10 and pick up the last 9 (#113, #114, #115, #116, #117, #118, #119, #120 and #121).

```
# Third Block: untransformed data and Euler angle data
Block3 <- c(84:86, 102, 113:121)
str(rawtraining[ , Block3])
```

```
## 'data.frame':    19622 obs. of  13 variables:
##  $ 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 ...
##  $ 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 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 -0.02 0 0 0 0 0 0 ...
##  $ accel_dumbbell_x    : int  -234 -233 -232 -232 -233 -234 -232 -234 -232 -235 ...
##  $ accel_dumbbell_y    : int  47 47 46 48 48 48 47 46 47 48 ...
##  $ accel_dumbbell_z    : int  -271 -269 -270 -269 -270 -269 -270 -272 -269 -270 ...
##  $ magnet_dumbbell_x   : int  -559 -555 -561 -552 -554 -558 -551 -555 -549 -558 ...
##  $ magnet_dumbbell_y   : int  293 296 298 303 292 294 295 300 292 291 ...
##  $ magnet_dumbbell_z   : num  -65 -64 -63 -60 -68 -66 -70 -74 -65 -69 ...
```

The **fourth block of 38 variables**, variables #122 through #159 appear to relate to the glove sensor on the **forearm** (wrist) of the participant.

```
str(rawtraining[ , 122:159])
```

```
## 'data.frame':    19622 obs. of  38 variables:
##  $ roll_forearm           : num  28.4 28.3 28.3 28.1 28 27.9 27.9 27.8 27.7 27.7 ...
##  $ pitch_forearm          : num  -63.9 -63.9 -63.9 -63.9 -63.9 -63.9 -63.9 -63.8 -63.8 -63.8 ...
##  $ yaw_forearm            : num  -153 -153 -152 -152 -152 -152 -152 -152 -152 -152 ...
##  $ kurtosis_roll_forearm  : chr  NA NA NA NA ...
##  $ kurtosis_picth_forearm : chr  NA NA NA NA ...
##  $ kurtosis_yaw_forearm   : chr  NA NA NA NA ...
##  $ skewness_roll_forearm  : chr  NA NA NA NA ...
##  $ skewness_pitch_forearm : chr  NA NA NA NA ...
##  $ skewness_yaw_forearm   : chr  NA NA NA NA ...
##  $ max_roll_forearm       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_picth_forearm      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ max_yaw_forearm        : chr  NA NA NA NA ...
##  $ min_roll_forearm       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_pitch_forearm      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ min_yaw_forearm        : chr  NA NA NA NA ...
##  $ amplitude_roll_forearm : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_pitch_forearm: num  NA NA NA NA NA NA NA NA NA NA ...
##  $ amplitude_yaw_forearm  : chr  NA NA NA NA ...
##  $ total_accel_forearm    : int  36 36 36 36 36 36 36 36 36 36 ...
##  $ var_accel_forearm      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_roll_forearm       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_roll_forearm    : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_roll_forearm       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_pitch_forearm      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_pitch_forearm   : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_pitch_forearm      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ avg_yaw_forearm        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ stddev_yaw_forearm     : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ var_yaw_forearm        : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ gyros_forearm_x        : num  0.03 0.02 0.03 0.02 0.02 0.02 0.02 0.02 0.03 0.02 ...
##  $ gyros_forearm_y        : num  0 0 -0.02 -0.02 0 -0.02 0 -0.02 0 0 ...
##  $ gyros_forearm_z        : num  -0.02 -0.02 0 0 -0.02 -0.03 -0.02 0 -0.02 -0.02 ...
##  $ accel_forearm_x        : int  192 192 196 189 189 193 195 193 193 190 ...
##  $ accel_forearm_y        : int  203 203 204 206 206 203 205 205 204 205 ...
##  $ accel_forearm_z        : int  -215 -216 -213 -214 -214 -215 -215 -213 -214 -215 ...
##  $ magnet_forearm_x       : int  -17 -18 -18 -16 -17 -9 -18 -9 -16 -22 ...
##  $ magnet_forearm_y       : num  654 661 658 658 655 660 659 660 653 656 ...
##  $ magnet_forearm_z       : num  476 473 469 469 473 478 470 474 476 473 ...
```

From the fourth block we only want the first three (#122, #123 and #124) and then we want to skip 15 and the pick up only one (#140) and then skip another 10 and pick up the last 9 (#151, #152, #153, #154, #155, #156, #157, #158 and #159) and the **classe** variable (#160).

```
# Fourth Block: untransformed data and Euler angle data
Block4 <- c(122:124, 140, 151:159, 160)
str(rawtraining[ , Block4])
```

```
## 'data.frame':    19622 obs. of  14 variables:
##  $ roll_forearm       : num  28.4 28.3 28.3 28.1 28 27.9 27.9 27.8 27.7 27.7 ...
##  $ pitch_forearm      : num  -63.9 -63.9 -63.9 -63.9 -63.9 -63.9 -63.9 -63.8 -63.8 -63.8 ...
##  $ yaw_forearm        : num  -153 -153 -152 -152 -152 -152 -152 -152 -152 -152 ...
##  $ total_accel_forearm: int  36 36 36 36 36 36 36 36 36 36 ...
##  $ gyros_forearm_x    : num  0.03 0.02 0.03 0.02 0.02 0.02 0.02 0.02 0.03 0.02 ...
##  $ gyros_forearm_y    : num  0 0 -0.02 -0.02 0 -0.02 0 -0.02 0 0 ...
##  $ gyros_forearm_z    : num  -0.02 -0.02 0 0 -0.02 -0.03 -0.02 0 -0.02 -0.02 ...
##  $ accel_forearm_x    : int  192 192 196 189 189 193 195 193 193 190 ...
##  $ accel_forearm_y    : int  203 203 204 206 206 203 205 205 204 205 ...
##  $ accel_forearm_z    : int  -215 -216 -213 -214 -214 -215 -215 -213 -214 -215 ...
##  $ magnet_forearm_x   : int  -17 -18 -18 -16 -17 -9 -18 -9 -16 -22 ...
##  $ magnet_forearm_y   : num  654 661 658 658 655 660 659 660 653 656 ...
##  $ magnet_forearm_z   : num  476 473 469 469 473 478 470 474 476 473 ...
##  $ classe             : Factor w/ 5 levels "A","B","C","D",..: 1 1 1 1 1 1 1 1 1 1 ...
```

Revised Training Set

So, our revised training set will consist of Block zero plus blocks one through four.

```
training <- rawtraining[ ,c(Block0, Block1, Block2, Block3, Block4)]
str(training)
```

```
## 'data.frame':    19622 obs. of  60 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 1323084232 1323
##  $ raw_timestamp_part_2: int  788290 808298 820366 120339 196328 304277 368296 440390 484323 484434
##  $ cvtd_timestamp       : chr  "05/12/2011 11:23" "05/12/2011 11:23" "05/12/2011 11:23" "05/12/2011 1
##  $ 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 ...
##  $ 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 ...
##  $ 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 ...
##  $ 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 ...
##  $ 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 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 -0.02 0 0 0 0 0 0 ...
##  $ accel_dumbbell_x    : int  -234 -233 -232 -232 -233 -234 -232 -234 -232 -235 ...
##  $ accel_dumbbell_y    : int  47 47 46 48 48 48 47 46 47 48 ...
##  $ accel_dumbbell_z    : int  -271 -269 -270 -269 -270 -269 -270 -272 -269 -270 ...
##  $ magnet_dumbbell_x   : int  -559 -555 -561 -552 -554 -558 -551 -555 -549 -558 ...
##  $ magnet_dumbbell_y   : int  293 296 298 303 292 294 295 300 292 291 ...
##  $ magnet_dumbbell_z   : num  -65 -64 -63 -60 -68 -66 -70 -74 -65 -69 ...
##  $ roll_forearm        : num  28.4 28.3 28.3 28.1 28 27.9 27.9 27.8 27.7 27.7 ...
##  $ pitch_forearm       : num  -63.9 -63.9 -63.9 -63.9 -63.9 -63.9 -63.9 -63.8 -63.8 -63.8 ...
##  $ yaw_forearm         : num  -153 -153 -152 -152 -152 -152 -152 -152 -152 -152 ...
##  $ total_accel_forearm : int  36 36 36 36 36 36 36 36 36 36 ...
##  $ gyros_forearm_x     : num  0.03 0.02 0.03 0.02 0.02 0.02 0.02 0.02 0.03 0.02 ...
##  $ gyros_forearm_y     : num  0 0 -0.02 -0.02 0 -0.02 0 -0.02 0 0 ...
##  $ gyros_forearm_z     : num  -0.02 -0.02 0 0 -0.02 -0.03 -0.02 0 -0.02 -0.02 ...
##  $ accel_forearm_x     : int  192 192 196 189 189 193 195 193 193 190 ...
##  $ accel_forearm_y     : int  203 203 204 206 206 203 205 205 204 205 ...
##  $ accel_forearm_z     : int  -215 -216 -213 -214 -214 -215 -215 -213 -214 -215 ...
##  $ magnet_forearm_x    : int  -17 -18 -18 -16 -17 -9 -18 -9 -16 -22 ...
##  $ magnet_forearm_y    : num  654 661 658 658 655 660 659 660 653 656 ...
##  $ magnet_forearm_z    : num  476 473 469 469 473 478 470 474 476 473 ...
##  $ classe              : Factor w/ 5 levels "A","B","C","D",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```
# Check for NAs
summary(training)
```

```
##        X            user_name    raw_timestamp_part_1 raw_timestamp_part_2
##  Min.   :    1   adelmo  :3892   Min.   :1.322e+09    Min.   :   294
##  1st Qu.: 4906   carlitos:3112   1st Qu.:1.323e+09    1st Qu.:252912
##  Median : 9812   charles :3536   Median :1.323e+09    Median :496380
##  Mean   : 9812   eurico  :3070   Mean   :1.323e+09    Mean   :500656
##  3rd Qu.:14717   jeremy  :3402   3rd Qu.:1.323e+09    3rd Qu.:751891
##  Max.   :19622   pedro   :2610   Max.   :1.323e+09    Max.   :998801
##  cvtd_timestamp     new_window    num_window      roll_belt
##  Length:19622       no :19216   Min.   :  1.0   Min.   :-28.90
##  Class :character   yes:  406   1st Qu.:222.0   1st Qu.:  1.10
##  Mode  :character               Median :424.0   Median :113.00
##                                 Mean   :430.6   Mean   : 64.41
##                                 3rd Qu.:644.0   3rd Qu.:123.00
##                                 Max.   :864.0   Max.   :162.00
##    pitch_belt          yaw_belt       total_accel_belt  gyros_belt_x
```

```
##  Min.   :-55.8000   Min.   :-180.00   Min.   : 0.00    Min.   :-1.040000
##  1st Qu.:  1.7600   1st Qu.: -88.30   1st Qu.: 3.00    1st Qu.:-0.030000
##  Median :  5.2800   Median : -13.00   Median :17.00    Median : 0.030000
##  Mean   :  0.3053   Mean   : -11.21   Mean   :11.31    Mean   :-0.005592
##  3rd Qu.: 14.9000   3rd Qu.:  12.90   3rd Qu.:18.00    3rd Qu.: 0.110000
##  Max.   : 60.3000   Max.   : 179.00   Max.   :29.00    Max.   : 2.220000
##   gyros_belt_y       gyros_belt_z      accel_belt_x       accel_belt_y
##  Min.   :-0.64000   Min.   :-1.4600   Min.   :-120.000   Min.   :-69.00
##  1st Qu.: 0.00000   1st Qu.:-0.2000   1st Qu.: -21.000   1st Qu.:  3.00
##  Median : 0.02000   Median :-0.1000   Median : -15.000   Median : 35.00
##  Mean   : 0.03959   Mean   :-0.1305   Mean   :  -5.595   Mean   : 30.15
##  3rd Qu.: 0.11000   3rd Qu.:-0.0200   3rd Qu.:  -5.000   3rd Qu.: 61.00
##  Max.   : 0.64000   Max.   : 1.6200   Max.   :  85.000   Max.   :164.00
##   accel_belt_z     magnet_belt_x   magnet_belt_y   magnet_belt_z
##  Min.   :-275.00   Min.   :-52.0   Min.   :354.0   Min.   :-623.0
##  1st Qu.:-162.00   1st Qu.:  9.0   1st Qu.:581.0   1st Qu.:-375.0
##  Median :-152.00   Median : 35.0   Median :601.0   Median :-320.0
##  Mean   : -72.59   Mean   : 55.6   Mean   :593.7   Mean   :-345.5
##  3rd Qu.:  27.00   3rd Qu.: 59.0   3rd Qu.:610.0   3rd Qu.:-306.0
##  Max.   : 105.00   Max.   :485.0   Max.   :673.0   Max.   : 293.0
##     roll_arm         pitch_arm          yaw_arm          total_accel_arm
##  Min.   :-180.00   Min.   :-88.800   Min.   :-180.0000   Min.   : 1.00
##  1st Qu.: -31.77   1st Qu.:-25.900   1st Qu.: -43.1000   1st Qu.:17.00
##  Median :   0.00   Median :  0.000   Median :   0.0000   Median :27.00
##  Mean   :  17.83   Mean   : -4.612   Mean   :  -0.6188   Mean   :25.51
##  3rd Qu.:  77.30   3rd Qu.: 11.200   3rd Qu.:  45.8750   3rd Qu.:33.00
##  Max.   : 180.00   Max.   : 88.500   Max.   : 180.0000   Max.   :66.00
##   gyros_arm_x        gyros_arm_y       gyros_arm_z       accel_arm_x
##  Min.   :-6.37000   Min.   :-3.4400   Min.   :-2.3300   Min.   :-404.00
##  1st Qu.:-1.33000   1st Qu.:-0.8000   1st Qu.:-0.0700   1st Qu.:-242.00
##  Median : 0.08000   Median :-0.2400   Median : 0.2300   Median : -44.00
##  Mean   : 0.04277   Mean   :-0.2571   Mean   : 0.2695   Mean   : -60.24
##  3rd Qu.: 1.57000   3rd Qu.: 0.1400   3rd Qu.: 0.7200   3rd Qu.:  84.00
##  Max.   : 4.87000   Max.   : 2.8400   Max.   : 3.0200   Max.   : 437.00
##   accel_arm_y      accel_arm_z       magnet_arm_x     magnet_arm_y
##  Min.   :-318.0   Min.   :-636.00   Min.   :-584.0   Min.   :-392.0
##  1st Qu.: -54.0   1st Qu.:-143.00   1st Qu.:-300.0   1st Qu.:  -9.0
##  Median :  14.0   Median : -47.00   Median : 289.0   Median : 202.0
##  Mean   :  32.6   Mean   : -71.25   Mean   : 191.7   Mean   : 156.6
##  3rd Qu.: 139.0   3rd Qu.:  23.00   3rd Qu.: 637.0   3rd Qu.: 323.0
##  Max.   : 308.0   Max.   : 292.00   Max.   : 782.0   Max.   : 583.0
##   magnet_arm_z    roll_dumbbell     pitch_dumbbell     yaw_dumbbell
##  Min.   :-597.0   Min.   :-153.71   Min.   :-149.59   Min.   :-150.871
##  1st Qu.: 131.2   1st Qu.: -18.49   1st Qu.: -40.89   1st Qu.: -77.644
##  Median : 444.0   Median :  48.17   Median : -20.96   Median :  -3.324
##  Mean   : 306.5   Mean   :  23.84   Mean   : -10.78   Mean   :   1.674
##  3rd Qu.: 545.0   3rd Qu.:  67.61   3rd Qu.:  17.50   3rd Qu.:  79.643
##  Max.   : 694.0   Max.   : 153.55   Max.   : 149.40   Max.   : 154.952
##  total_accel_dumbbell gyros_dumbbell_x    gyros_dumbbell_y
##  Min.   : 0.00        Min.   :-204.0000   Min.   :-2.10000
##  1st Qu.: 4.00        1st Qu.:  -0.0300   1st Qu.:-0.14000
##  Median :10.00        Median :   0.1300   Median : 0.03000
##  Mean   :13.72        Mean   :   0.1611   Mean   : 0.04606
##  3rd Qu.:19.00        3rd Qu.:   0.3500   3rd Qu.: 0.21000
```

```
##  Max.   :58.00        Max.   :   2.2200   Max.   :52.00000
##  gyros_dumbbell_z  accel_dumbbell_x  accel_dumbbell_y  accel_dumbbell_z
##  Min.   : -2.380   Min.   :-419.00   Min.   :-189.00   Min.   :-334.00
##  1st Qu.: -0.310   1st Qu.: -50.00   1st Qu.:  -8.00   1st Qu.:-142.00
##  Median : -0.130   Median :  -8.00   Median :  41.50   Median :  -1.00
##  Mean   : -0.129   Mean   : -28.62   Mean   :  52.63   Mean   : -38.32
##  3rd Qu.:  0.030   3rd Qu.:  11.00   3rd Qu.: 111.00   3rd Qu.:  38.00
##  Max.   :317.000   Max.   : 235.00   Max.   : 315.00   Max.   : 318.00
##  magnet_dumbbell_x magnet_dumbbell_y magnet_dumbbell_z  roll_forearm
##  Min.   :-643.0    Min.   :-3600     Min.   :-262.00   Min.   :-180.0000
##  1st Qu.:-535.0    1st Qu.:  231     1st Qu.: -45.00   1st Qu.:  -0.7375
##  Median :-479.0    Median :  311     Median :  13.00   Median :  21.7000
##  Mean   :-328.5    Mean   :  221     Mean   :  46.05   Mean   :  33.8265
##  3rd Qu.:-304.0    3rd Qu.:  390     3rd Qu.:  95.00   3rd Qu.: 140.0000
##  Max.   : 592.0    Max.   :  633     Max.   : 452.00   Max.   : 180.0000
##  pitch_forearm     yaw_forearm      total_accel_forearm gyros_forearm_x
##  Min.   :-72.50   Min.   :-180.00   Min.   :  0.00      Min.   :-22.000
##  1st Qu.:  0.00   1st Qu.: -68.60   1st Qu.: 29.00      1st Qu.: -0.220
##  Median :  9.24   Median :   0.00   Median : 36.00      Median :  0.050
##  Mean   : 10.71   Mean   :  19.21   Mean   : 34.72      Mean   :  0.158
##  3rd Qu.: 28.40   3rd Qu.: 110.00   3rd Qu.: 41.00      3rd Qu.:  0.560
##  Max.   : 89.80   Max.   : 180.00   Max.   :108.00      Max.   :  3.970
##  gyros_forearm_y     gyros_forearm_z    accel_forearm_x   accel_forearm_y
##  Min.   : -7.02000   Min.   : -8.0900   Min.   :-498.00   Min.   :-632.0
##  1st Qu.: -1.46000   1st Qu.: -0.1800   1st Qu.:-178.00   1st Qu.:  57.0
##  Median :  0.03000   Median :  0.0800   Median : -57.00   Median : 201.0
##  Mean   :  0.07517   Mean   :  0.1512   Mean   : -61.65   Mean   : 163.7
##  3rd Qu.:  1.62000   3rd Qu.:  0.4900   3rd Qu.:  76.00   3rd Qu.: 312.0
##  Max.   :311.00000   Max.   :231.0000   Max.   : 477.00   Max.   : 923.0
##  accel_forearm_z   magnet_forearm_x  magnet_forearm_y magnet_forearm_z
##  Min.   :-446.00   Min.   :-1280.0   Min.   :-896.0   Min.   :-973.0
##  1st Qu.:-182.00   1st Qu.: -616.0   1st Qu.:   2.0   1st Qu.: 191.0
##  Median : -39.00   Median : -378.0   Median : 591.0   Median : 511.0
##  Mean   : -55.29   Mean   : -312.6   Mean   : 380.1   Mean   : 393.6
##  3rd Qu.:  26.00   3rd Qu.:  -73.0   3rd Qu.: 737.0   3rd Qu.: 653.0
##  Max.   : 291.00   Max.   :  672.0   Max.   :1480.0   Max.   :1090.0
##  classe
##  A:5580
##  B:3797
##  C:3422
##  D:3216
##  E:3607
##
```

```
library(lattice)
library(ggplot2)
library(caret)
summary(training$classe)
```

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

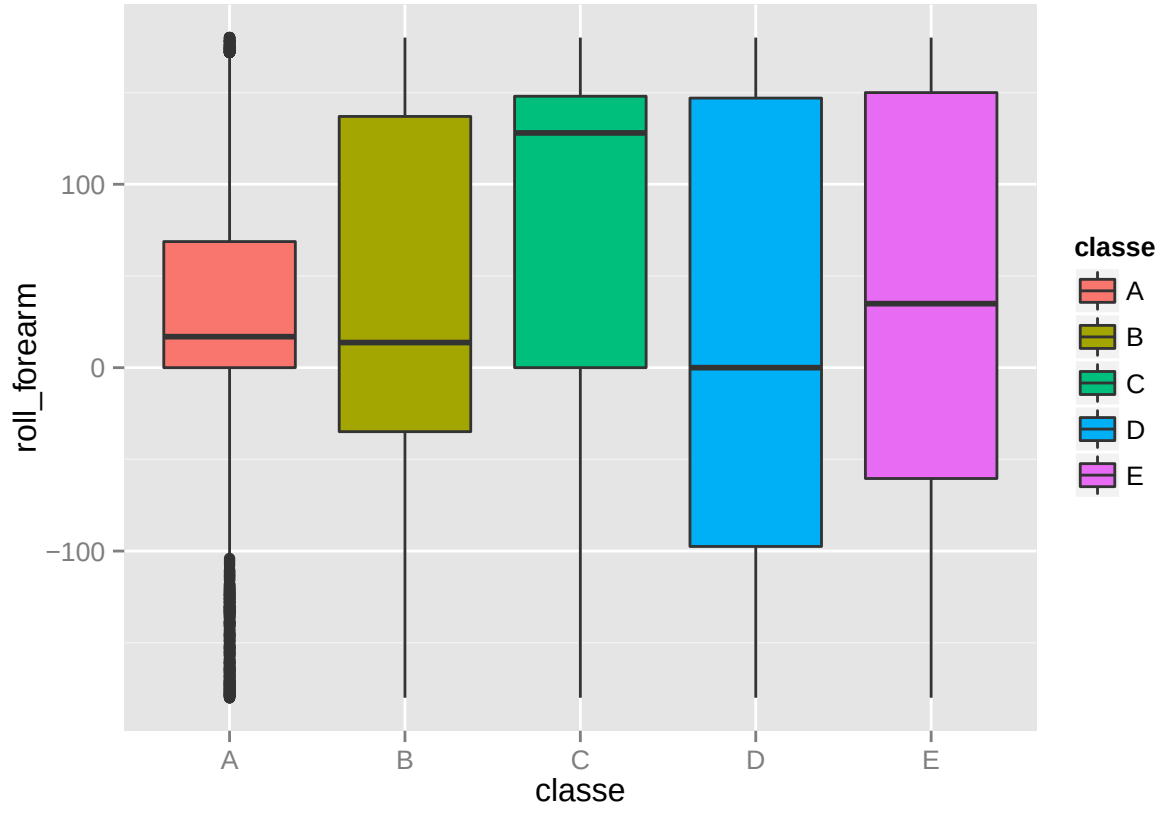
```
ClassA <- training$classe == "A"
ClassB <- training$classe == "B"
summary(ClassA)
```

```
##    Mode   FALSE    TRUE    NA's
## logical   14042    5580       0
```

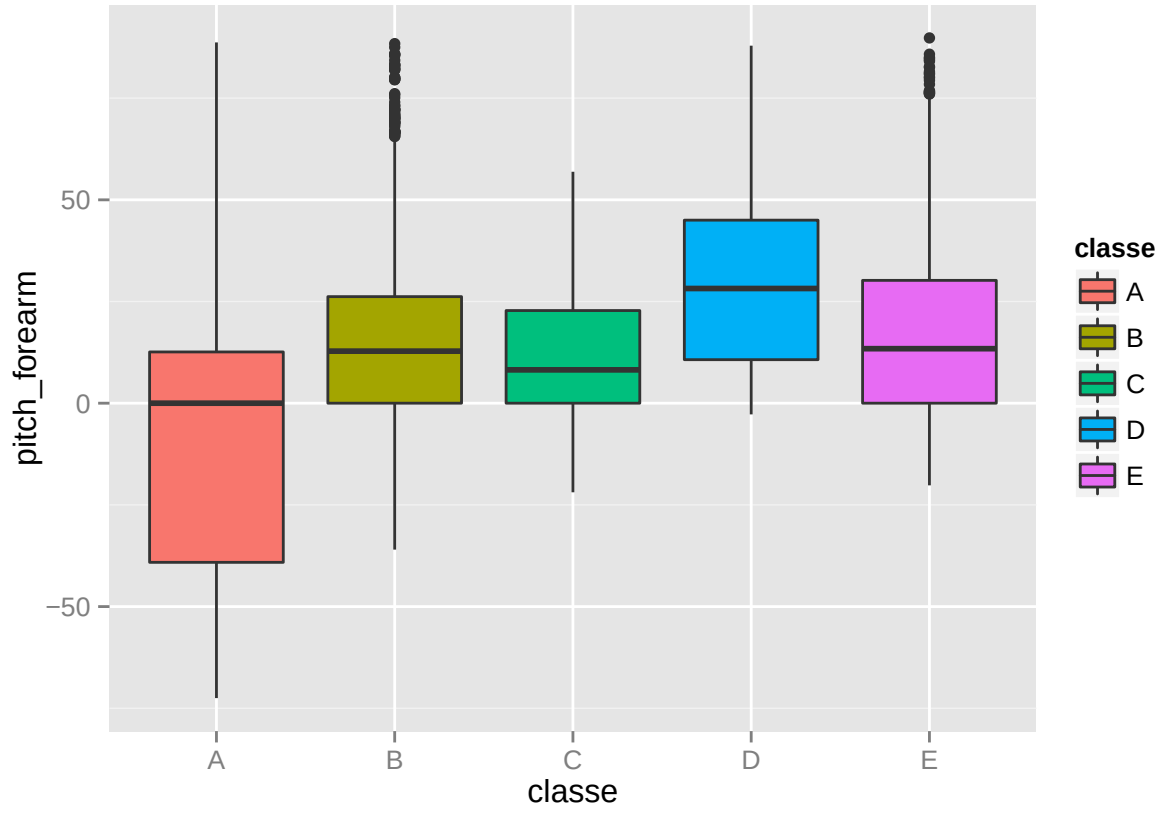
```
summary(ClassB)
```

```
##    Mode   FALSE    TRUE    NA's
## logical   15825    3797       0
```

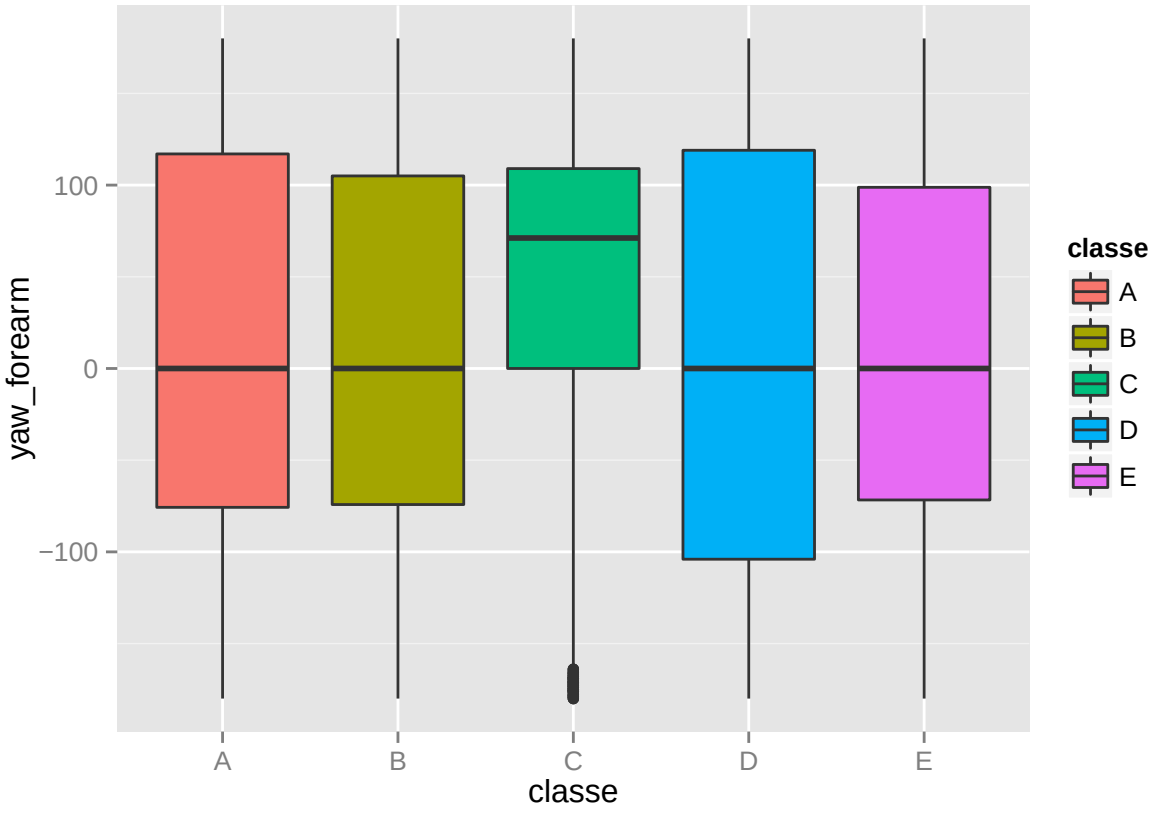
```
# featurePlot(x=training[ClassA, c("roll_forearm","pitch_forearm","yaw_forearm")],
# y = training$classe[ClassA],
# plot="pairs")

# featurePlot(x=training[ClassB, c("roll_forearm","pitch_forearm","yaw_forearm")],
# y = training$classe[ClassB],
# plot="pairs")

p1 <- qplot(classe, roll_forearm, data=training, fill=classe,
geom=c("boxplot"))
p1
```



```
p2 <- qplot(classe, pitch_forearm, data=training, fill=classe,
geom=c("boxplot"))
p2
```




```
p3 <- qplot(classe, yaw_forearm, data=training, fill=classe,
geom=c("boxplot"))
p3
```



A paper describing the experiment suggested that there were "96 features" (variables).

> "a wearable sensor-oriented classification approach for the detection of mistakes"
>
> "mounted the sensors in the users' glove, armband, lumbar belt and dumbbell (see Figure 1)."

Sensors (from Figure 1)

- ArmBand (on body)
- Belt (on body)
- Glove (on body)
- Dumbbell (on weight)

"For data recording we used four 9 degrees of freedom Razor inertial measurement units (IMU), which provide three-axes acceleration, gyroscope and magnetometer data at a joint sampling rate of 45 Hz."

Class
> "**Class A** corresponds to the [correct] specified execution of the exercise,
> while the other 4 classes correspond to common mistakes."

A. Correct – "exactly according to the specification"
B. Incorrect – "throwing the elbows to the front"
C. Incorrect – "lifting the dumbbell only halfway"
D. Incorrect – "lowering the dumbbell only halfway"
E. Incorrect – "throwing the hips to the front"

Window

Euler Angles
> "Euler angles (roll, pitch and yaw)"

-Roll
-Pitch
-Yaw

Calculated 8 Features
> "For the [**three**] **Euler angles** of each of the **four sensors**
> we **calculated eight features**:
> mean, variance, standard deviation, max, min, amplitude, kurtosis and skewness,
> generating in total **96 derived feature sets.**"

3 Euler angles
4 Sensors
8 Calculated Features

```
3*4*8
```

```
## [1] 96
```

Quality > "if we can specify how an activity has to be performed we can measure the quality
> by comparing its execution against this specification.
>
> From this, we define quality as the adherence of the execution of an activity
> to its specification.
>
> From this, we define a qualitative activity recognition system as a
> software artefact that observes the user's execution of an activity and > compares it to a specification."

Drop "X" the ID number to prevent "data leakage"
The variable "**X**" appears to be a sequence number and will have to be discarded prior to training to avoid "data leakage".

```
summary(rawtraining$X)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       1    4906    9812    9812   14720   19620
```

The sequence number should be non-informative (useless) but if it turns out to be predictive it would be an example of "**data leakage**".

Kaggle defines **data leakage** as, "the creation of unexpected additional information in the training data, allowing a model or machine learning algorithm to make unrealistically good predictions."
https://www.kaggle.com/wiki/Leakage

An similar example of data leakage would be:
"You're trying to study who has breast cancer. The **patient ID**, which seemed innocent, actually has predictive power. What happened? ... This is probably a consequence of using multiple databases [each from

different cancer centers], some of which correspond to [specialize in] sicker patients are more likely to be sick."
This blog post corresponds to pages 310-311 in the book, "**Doing Data Science**".
http://mathbabe.org/2012/11/20/columbia-data-science-course-week-12-predictive-modeling-data-leakage-model-evaluation/

> "For the [**three**] **Euler angles** of each of the **four sensors** we calculated **eight features**:
> mean, variance, standard deviation, max, min, amplitude, kurtosis and skewness,
> generating in total **96 derived feature sets.**"

3 Euler angles
- roll, pitch and yaw
4 Sensors
- belt, arm ("armband"), dumbell and forearm ("glove")
8 Calculated Features
- mean, variance, standard deviation, max, min, amplitude, kurtosis and skewness

```
3*4*8
```

```
## [1] 96
```

Preliminary Analysis

1. Fix converted timestamp
2. Normalize – preprocess – scale
3. Principal Components?
4. 10 fold cross-validation? ### Model Building ###
5. Naive Bayes / library(klaR) / nb() – did well in "Doing Data Science" NYT
6. KNN
7. Recursive Partitioning / libary(party) / ctree() – blog post http://www.r-bloggers.com/party-with-the-first-tribe/
8. Random Forest / library(randomForest) / rf() -or- cforest() a fancy method

"RF [Random Forests] thrives on variables–the more the better. There is no need for variable selection ,On a sonar data set with 208 cases and 60 variables, the RF error rate is 14%. Logistic Regression has a 50% error rate." Leo Breiman http://www.stat.berkeley.edu/~breiman/wald2002-2.pdf

Model Validation

1. confusion matrix – both training and test
2. AUC
3. Picture of Tree
4. 10 fold cross-validation
5. 20 predictions
6. Short paper

Conclusion

1. Subset the columns
2. Subset the rows (apparently not necessary in this project – I wasted a lot of time trying to understand the windows variables)
3. Check for NAs and impute values if necessary
4. Check whether numbers are of similar magnitude (Principal Components and KNN let biggest number dominate)
5. Split data for cross-validation (even though we have training and test data; I believe the peer evaluation asks about "cross validation")
6. Run the machine learning algorithm (the literature says the exact algorithm doesn't make much difference as long as your algorithm is appropriate to the task – supervised classification vs. unsupervised clustering, etc)
7. Generate the confusion matrix
8. Generate AUC curve
9. Generate graphics (picture of tree if you did a tree algorithm – data graph otherwise)
10. Do the 20 predictions
11. Write a very short paper describing what you did and the reasons for the choices you made.

Bibliography

DATA SOURCE:
Velloso, E.; Bulling, A.; Gellersen, H.; Ugulino W.; Fuks, H.
"Qualitative Activity Recognition of Weight Lifting Exercises"
Proceedings of the 4th International Conference in Cooperation with SIGCHI
(Augmented Human '13), Stuttgart, Germany
ACM SIGCHI, 2013.
Available at:
http://groupware.les.inf.puc-rio.br/public/papers/2013.Velloso.QAR-WLE.pdf

SOFTWARE:

R
R Core Team (2015).
"**R: A language and environment for statistical computing**".
R Foundation for Statistical Computing, Vienna, Austria.
https://www.R-project.org/

Caret (R package)
by Max Kuhn. Contributions from Jed Wing, Steve Weston, Andre Williams,
Chris Keefer, Allan Engelhardt, Tony Cooper, Zachary Mayer, Brenton
Kenkel, the R Core Team, Michael Benesty, Reynald Lescarbeau, Andrew
Ziem, Luca Scrucca, Yuan Tang and Can Candan. (2015).
"**caret: Classification and Regression Training**".
R package version 6.0-57.
"**Building Predictive Models in R Using the caret Package**" Journal of Statistical Software 28(5), 1-26. http://www.jstatsoft.org/v28/i05/paper
http://CRAN.R-project.org/package=caret http://topepo.github.io/caret/index.html

kernlab (R package)
by Alexandros Karatzoglou, Alex Smola, Kurt Hornik, Achim Zeileis (2004).
"**kernlab - An S4 Package for Kernel Methods in R**".
Journal of Statistical Software 11(9), 1-20.
http://www.jstatsoft.org/v11/i09/
http://CRAN.R-project.org/package=kernlab

"**ggplot2 (R package)**"
by H. Wickham.
"**ggplot2: elegant graphics for data analysis**". Springer New York, 2009.
https://cran.r-project.org/package=ggplot2
http://ggplot2.org/book/

"**R Graphics Cookbook**"
by Winston Chang (O'Reilly).
Copyright 2013 Winston Chang, ISBN 978-1-449-31695-2.
http://oreil.ly/R_Graphics_Cookbook
http://www.cookbook-r.com/Graphs/

"**Doing Data Science**"
by Cathy O'Neil and Rachel Schutt (O'Reilly).
Copyright 2014 Cathy O'Neil and Rachel Schutt, ISBN 978-1-449-35865-5
http://oreil.ly/doing_data_science
http://mathbabe.org/

"**Data analysis with Open Source Tools**"
by Phillip K. Janert (O'Reilly).
Copyright 2011 Phillip K. Janert, ISBN 978-0-596-80235-6.
http://shop.oreilly.com/product/9780596802363.do
http://www.beyondcode.org/

"**DISTRIBUTION BASED TREES ARE MORE ACCURATE**" by Nong Shang and Leo Breiman ?
https://www.stat.berkeley.edu/~breiman/DB-CART.pdf

"**OUT-OF-BAG ESTIMATION**" by Leo Breiman

"**WALD LECTURE II: LOOKING INSIDE THE BLACK BOX**" by Leo Breiman http://www.stat.berkeley.edu/~breiman/wald2002-2.pdf

"**Boosting Tutorial**" by Ron Meir Machine Learning Summer School 2002 Technion Univerity, Israel http://webee.technion.ac.il/people/rmeir/BoostingTutorial.pdf

Wikipedia
https://en.wikipedia.org/wiki/AdaBoost
https://en.wikipedia.org/wiki/Autoencoder
https://en.wikipedia.org/wiki/Bootstrap_aggregating
https://en.wikipedia.org/wiki/Decision_tree_learning
https://en.wikipedia.org/wiki/Delphi_method
https://en.wikipedia.org/wiki/Design_matrix
https://en.wikipedia.org/wiki/Ensemble_learning
https://en.wikipedia.org/wiki/Linear_discriminant_analysis
https://en.wikipedia.org/wiki/Probably_approximately_correct_learning
https://en.wikipedia.org/wiki/Recommender_system
https://en.wikipedia.org/wiki/Quadratic_classifier
https://en.wikipedia.org/wiki/Random_forest