HW2_jaeyounglee

Jaeyoung Lee

September 15, 2020

Problem 3

First of all, it is to handle a mistake. Also, one can handle various versions of a code and see the history of a code. Furthermore, using version control, it is easy to collaborate with others. This is because one can share a code and work on the cloud such as GitHub.

Problem 4

For each dataset, you should perform the cleaning 2x: first with base R functions (ie no dplyr, piping, etc), second using tidyverse function. Make sure you weave your code and text into a complete description of the process and end by creating a tidy dataset describing the variables, create a summary table of the data (summary, NOT full listing), note issues with the data, and include an informative plot.

a. Sensory data from five operators. http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/Sensory.dat

```
######## Sensory data ########
# Getting "https://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/Sensory.dat"
url_sensory <- "https://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/Sensory.dat"
sensory_rawdata <- fread(url_sensory, fill = TRUE, skip = 2, data.table = FALSE)
saveRDS(sensory_rawdata, 'sensory_rawdata.RDS')
sensory_rawdata <- readRDS('sensory_rawdata.RDS')
```

There are missing values in the raw data and the categories "Items" are in the data like oberservations. We need to remove missing values and extract the 'Item' numbers from the data.

```
# Using base R function only
# Convert data.frame to matrix and transpose the raw data
matrix_sensory <- t(as.matrix(sensory_rawdata))

# Find where the missing values are
na <- which(is.na(matrix_sensory==TRUE))

# The indexes where Item numbers are in the data
x <- 1
item <- x
for (i in 1:9){
  x <- x+18
  item <- c(item, x)
}

# Remove missing values and 'Item's from the data
sensory_data <- t(matrix(matrix_sensory[-c(na,item)], byrow = T, nrow = 10))
```

```r
sensory_data <- data.table(sensory_data)
colnames(sensory_data) <- paste('Item', 1:10) # Assign column names
Opr <- rep(paste('Opr', 1:5), 3)              # Operator names
sensory_data <- cbind(Opr,sensory_data)       # Bind Operator names and the data

# Re-order the rows by names of operators
sensory_data <- sensory_data[order(sensory_data$Opr)]
sensory_data_base <- sensory_data

# Final tidy data with base R functions
sensory_data_base
```

```
##       Opr Item 1 Item 2 Item 3 Item 4 Item 5 Item 6 Item 7 Item 8 Item 9
##  1: Opr 1    4.3    6.0    2.4    7.4    5.7    2.2    1.2    4.2    8.0
##  2: Opr 1    4.3    4.9    3.9    7.1    5.8    3.0    1.3    3.0    9.0
##  3: Opr 1    4.1    6.0    1.9    6.4    5.8    2.1    0.9    4.8    8.9
##  4: Opr 2    4.9    5.3    2.5    8.2    6.3    2.4    1.5    4.8    8.6
##  5: Opr 2    4.5    6.3    3.0    7.9    5.7    1.8    2.4    4.5    7.7
##  6: Opr 2    5.3    5.9    3.9    7.1    6.0    3.3    3.1    4.8    9.2
##  7: Opr 3    3.3    4.5    2.3    6.4    5.4    1.7    1.2    4.5    9.0
##  8: Opr 3    4.0    4.2    2.8    5.9    5.4    2.1    0.8    4.7    6.7
##  9: Opr 3    3.4    4.7    2.6    6.9    6.1    1.1    1.1    4.7    8.1
## 10: Opr 4    5.3    5.9    3.1    6.8    6.1    3.4    0.9    4.6    9.4
## 11: Opr 4    5.5    5.5    2.7    7.3    6.2    4.0    1.2    4.9    9.0
## 12: Opr 4    5.7    6.3    4.6    7.0    7.0    3.3    1.9    4.8    9.1
## 13: Opr 5    4.4    4.7    2.4    6.0    5.9    1.7    0.7    3.2    8.8
## 14: Opr 5    3.3    4.9    1.3    6.1    6.5    1.7    1.3    4.6    7.9
## 15: Opr 5    4.7    4.6    2.2    6.7    4.9    2.1    1.6    4.3    7.6
##     Item 10
##  1:     5.0
##  2:     5.4
##  3:     2.8
##  4:     4.8
##  5:     5.0
##  6:     5.2
##  7:     3.9
##  8:     3.4
##  9:     4.1
## 10:     5.5
## 11:     4.9
## 12:     3.9
## 13:     3.8
## 14:     4.6
## 15:     5.5
```

Above is the converted tidy data frames using the base R functions only. A summary of the data is as follows:

Opr	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6	Item 7	Item 8	Item 9	Item 10
Length:15	Min. :3.300	Min. :4.200	Min. :1.300	Min. :5.90	Min. :4.90	Min. :1.100	Min. :0.700	Min. :3.000	Min. :6.700	Min. :2.80
Class :character	1st Qu.:4.050	1st Qu.:4.700	1st Qu.:2.350	1st Qu.:6.40	1st Qu.:5.70	1st Qu.:1.750	1st Qu.:1.000	1st Qu.:4.400	1st Qu.:7.950	1st Qu.:3.90

Opr	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6	Item 7	Item 8	Item 9	Item 10
Mode :character	Median :4.400	Median :5.300	Median :2.600	Median :6.90	Median :5.90	Median :2.100	Median :1.200	Median :4.600	Median :8.800	Median :4.80
NA	Mean :4.467	Mean :5.313	Mean :2.773	Mean :6.88	Mean :5.92	Mean :2.393	Mean :1.407	Mean :4.427	Mean :8.467	Mean :4.52
NA	3rd Qu.:5.100	3rd Qu.:5.950	3rd Qu.:3.050	3rd Qu.:7.20	3rd Qu.:6.15	3rd Qu.:3.150	3rd Qu.:1.550	3rd Qu.:4.800	3rd Qu.:9.000	3rd Qu.:5.10
NA	Max. :5.700	Max. :6.300	Max. :4.600	Max. :8.20	Max. :7.00	Max. :4.000	Max. :3.100	Max. :4.900	Max. :9.400	Max. :5.50

Now, handle the same data with `tidyverse` package.

```
# Sensory data with tidyverse package
# Making matrix which is the same with base R function but using pipes.
matrix_sensory <- sensory_rawdata %>% as.matrix() %>% t()
na <- which(is.na(matrix_sensory==TRUE))      # Find missing values

# The indexes where Item numbers are in the data
x <- 1
item <- x
for (i in 1:9){
  x <- x+18
  item <- c(item, x)
}

# Remove missing values and Item numbers from the data
sensory_data <- matrix_sensory[-c(na,item)] %>% matrix(byrow = T, nrow = 10) %>% t()
sensory_data <- data.table(sensory_data)
Opr <- rep(paste('Opr', 1:5), 3)
sensory_data <- bind_cols(Opr,sensory_data) # bind operators and data
```

```
## New names:
## * NA -> ...1
```

```
colnames(sensory_data) <- c('Opr',paste('Item', 1:10))

# Re-order the rows by names of operators
sensory_data <- sensory_data[order(sensory_data$Opr)]
sensory_data_tidyverse <- sensory_data

# Final tidy data with tidyverse
sensory_data_tidyverse
```

```
##       Opr Item 1 Item 2 Item 3 Item 4 Item 5 Item 6 Item 7 Item 8 Item 9
##  1: Opr 1    4.3    6.0    2.4    7.4    5.7    2.2    1.2    4.2    8.0
##  2: Opr 1    4.3    4.9    3.9    7.1    5.8    3.0    1.3    3.0    9.0
##  3: Opr 1    4.1    6.0    1.9    6.4    5.8    2.1    0.9    4.8    8.9
##  4: Opr 2    4.9    5.3    2.5    8.2    6.3    2.4    1.5    4.8    8.6
##  5: Opr 2    4.5    6.3    3.0    7.9    5.7    1.8    2.4    4.5    7.7
##  6: Opr 2    5.3    5.9    3.9    7.1    6.0    3.3    3.1    4.8    9.2
##  7: Opr 3    3.3    4.5    2.3    6.4    5.4    1.7    1.2    4.5    9.0
##  8: Opr 3    4.0    4.2    2.8    5.9    5.4    2.1    0.8    4.7    6.7
##  9: Opr 3    3.4    4.7    2.6    6.9    6.1    1.1    1.1    4.7    8.1
## 10: Opr 4    5.3    5.9    3.1    6.8    6.1    3.4    0.9    4.6    9.4
## 11: Opr 4    5.5    5.5    2.7    7.3    6.2    4.0    1.2    4.9    9.0
## 12: Opr 4    5.7    6.3    4.6    7.0    7.0    3.3    1.9    4.8    9.1
## 13: Opr 5    4.4    4.7    2.4    6.0    5.9    1.7    0.7    3.2    8.8
## 14: Opr 5    3.3    4.9    1.3    6.1    6.5    1.7    1.3    4.6    7.9
## 15: Opr 5    4.7    4.6    2.2    6.7    4.9    2.1    1.6    4.3    7.6
##     Item 10
##  1:     5.0
##  2:     5.4
##  3:     2.8
##  4:     4.8
##  5:     5.0
##  6:     5.2
##  7:     3.9
##  8:     3.4
##  9:     4.1
## 10:     5.5
## 11:     4.9
## 12:     3.9
## 13:     3.8
## 14:     4.6
## 15:     5.5
```

The result by tidyverse is the same with the base R function. The summary of the data converted by tidyverse is as follows.

Opr	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6	Item 7	Item 8	Item 9	Item 10
Length:15	Min. :3.300	Min. :4.200	Min. :1.300	Min. :5.90	Min. :4.90	Min. :1.100	Min. :0.700	Min. :3.000	Min. :6.700	Min. :2.80
Class :character	1st Qu.:4.050	1st Qu.:4.700	1st Qu.:2.350	1st Qu.:6.40	1st Qu.:5.70	1st Qu.:1.750	1st Qu.:1.000	1st Qu.:4.400	1st Qu.:7.950	1st Qu.:3.90
Mode :character	Median :4.400	Median :5.300	Median :2.600	Median :6.90	Median :5.90	Median :2.100	Median :1.200	Median :4.600	Median :8.800	Median :4.80
NA	Mean :4.467	Mean :5.313	Mean :2.773	Mean :6.88	Mean :5.92	Mean :2.393	Mean :1.407	Mean :4.427	Mean :8.467	Mean :4.52
NA	3rd Qu.:5.100	3rd Qu.:5.950	3rd Qu.:3.050	3rd Qu.:7.20	3rd Qu.:6.15	3rd Qu.:3.150	3rd Qu.:1.550	3rd Qu.:4.800	3rd Qu.:9.000	3rd Qu.:5.10
NA	Max. :5.700	Max. :6.300	Max. :4.600	Max. :8.20	Max. :7.00	Max. :4.000	Max. :3.100	Max. :4.900	Max. :9.400	Max. :5.50

b. Gold Medal performance for Olympic Men's Long Jump, year is coded as 1900=0. http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/LongJumpData.dat

```
######## Long Jump data ########
# Getting "http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/LongJumpData.dat"
url_medal <- "http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/LongJumpData.dat"
medal_rawdata <- fread(url_medal)
```

```
## Warning in fread(url_medal): Detected 12 column names but the data has 8
## columns. Filling rows automatically. Set fill=TRUE explicitly to avoid this
## warning.
```

```
saveRDS(medal_rawdata, 'medal_rawdata.RDS')
medal_rawdata <- readRDS('medal_rawdata.RDS')
```

The raw data has missing values and wide type data. It is better to reshape the data. Also, we need two vectors : 'Year' and 'Long Jump'.

```
# Using base R function only
# Year is coded as 1900 = 0
# Extract Year and Long Jump vectors
year <- c(medal_rawdata[[1]], medal_rawdata[[3]],
          medal_rawdata[[5]], medal_rawdata[[7]]) + 1900
longjump <- c(medal_rawdata[[2]], medal_rawdata[[4]],
              medal_rawdata[[6]], medal_rawdata[[8]])

# Bind the vectors as a data table and rename the categories
medal_data <- data.table(year[1:(length(year)-2)],
                         longjump[1:(length(longjump)-2)])
colnames(medal_data) <- c('Year', 'Long Jump')
medal_data_base <- medal_data

# Final tidy data with base R functions
medal_data_base
```

```
##     Year Long Jump
##  1: 1896    249.75
##  2: 1900    282.88
##  3: 1904    289.00
##  4: 1908    294.50
##  5: 1912    299.25
##  6: 1920    281.50
##  7: 1924    293.13
##  8: 1928    304.75
##  9: 1932    300.75
## 10: 1936    317.31
## 11: 1948    308.00
## 12: 1952    298.00
## 13: 1956    308.25
## 14: 1960    319.75
## 15: 1964    317.75
## 16: 1968    350.50
## 17: 1972    324.50
## 18: 1976    328.50
## 19: 1980    336.25
## 20: 1984    336.25
## 21: 1988    343.25
## 22: 1992    342.50
##     Year Long Jump
```

Above is the converted tidy data frames using the base R functions. A summary of the data is as follows:

Year	Long Jump
Min. :1896	Min. :249.8
1st Qu.:1921	1st Qu.:295.4
Median :1950	Median :308.1
Mean :1945	Mean :310.3
3rd Qu.:1971	3rd Qu.:327.5
Max. :1992	Max. :350.5

Now, handle the same data with `tidyverse` package.

```
# Using tidyverse package
# Year is coded as 1900 = 0
medal_data <- medal_rawdata[,1:8] # remove missing values only columns

# Extracting 'Year' columns and 'Long Jump' columns
colnames(medal_data) <- paste(rep(c('Year', 'Jump'),4), rep(1:4,each = 2))
year <- medal_data[,c(1,3,5,7)] %>%
  gather(key = 'name1', value = 'Year', 1,2,3,4) %>% filter(Year != na)
```

```
## Warning in Year != na: longer object length is not a multiple of shorter object
## length
```

```
year[,2] <- year[,2] + 1900
jump <- medal_data[,c(2,4,6,8)] %>%
  gather(key = 'name2', value = 'LongJump', 1,2,3,4) %>% filter(LongJump != na)
```

```
## Warning in LongJump != na: longer object length is not a multiple of shorter
## object length
```

```
# Bind the vectors as a data table and rename the categories
medal_data <- bind_cols(year[,2], jump[,2])
```

```
## New names:
## * NA -> ...1
## * NA -> ...2
```

```
colnames(medal_data) <- c('Year', 'Long Jump')
medal_data_tidyverse <- medal_data

# Final tidy data with tidyverse
medal_data_tidyverse
```

```
## # A tibble: 22 x 2
##     Year `Long Jump`
##    <dbl>       <dbl>
##  1  1896        250.
##  2  1900        283.
##  3  1904        289
##  4  1908        294.
##  5  1912        299.
##  6  1920        282.
##  7  1924        293.
##  8  1928        305.
##  9  1932        301.
## 10  1936        317.
## # ... with 12 more rows
```

The result by tidyverse is the same with the base R function. The summary of the data converted by tidyverse is as follows.

Year	Long Jump
Min. :1896	Min. :249.8
1st Qu.:1921	1st Qu.:295.4
Median :1950	Median :308.1
Mean :1945	Mean :310.3

Year	Long Jump
3rd Qu.:1971	3rd Qu.:327.5
Max. :1992	Max. :350.5

c. Brain weight (g) and body weight (kg) for 62 species.
   http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/BrainandBodyWeight.dat

```
######## Brain weight data ########
# Getting "http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/BrainandBodyWeight.dat"
url_brain <- "http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/BrainandBodyWeight.dat"
brain_rawdata <- fread(url_brain)
```

```
## Warning in fread(url_brain): Detected 12 column names but the data has 6
## columns. Filling rows automatically. Set fill=TRUE explicitly to avoid this
## warning.
```

```
saveRDS(brain_rawdata, 'brain_rawdata.RDS')
brain_rawdata <- readRDS('brain_rawdata.RDS')
```

The data needs two columns which are 'Body Wt' and 'Brain Wt'.

```
# Using base R function only
# The method is the same with the data from part (b)
# Extract Body Wt and Brain Wt vectors
bodywt <- c(brain_rawdata[[1]], brain_rawdata[[3]], brain_rawdata[[5]])
brainwt <- c(brain_rawdata[[2]], brain_rawdata[[4]], brain_rawdata[[6]])

# Remove missing values
brain_data <- data.table(bodywt[-length(bodywt)], brainwt[-length(brainwt)])
colnames(brain_data) <- c('Body Wt', 'Brain Wt')
brain_data_base <- brain_data

# Final tidy data with base R functions
brain_data_base
```

```
##      Body Wt Brain Wt
##  1:    3.385    44.50
##  2:    0.480    15.50
##  3:    1.350     8.10
##  4:  465.000   423.00
##  5:   36.330   119.50
##  6:   27.660   115.00
##  7:   14.830    98.20
##  8:    1.040     5.50
##  9:    4.190    58.00
## 10:    0.425     6.40
## 11:    0.101     4.00
## 12:    0.920     5.70
## 13:    1.000     6.60
## 14:    0.005     0.10
## 15:    0.060     1.00
## 16:    3.500    10.80
## 17:    2.000    12.30
## 18:    1.700     6.30
## 19: 2547.000  4603.00
## 20:    0.023     0.30
## 21:  187.100   419.00
## 22:  521.000   655.00
## 23:    0.785     3.50
## 24:   10.000   115.00
## 25:    3.300    25.60
## 26:    0.200     5.00
## 27:    1.410    17.50
## 28:  529.000   680.00
## 29:  207.000   406.00
## 30:   85.000   325.00
## 31:    0.750    12.30
## 32:   62.000  1320.00
## 33: 6654.000  5712.00
## 34:    3.500     3.90
## 35:    6.800   179.00
## 36:   35.000    56.00
## 37:    4.050    17.00
## 38:    0.120     1.00
## 39:    0.023     0.40
## 40:    0.010     0.30
## 41:    1.400    12.50
## 42:  250.000   490.00
## 43:    2.500    12.10
## 44:   55.500   175.00
## 45:  100.000   157.00
## 46:   52.160   440.00
## 47:   10.550   179.50
## 48:    0.550     2.40
## 49:   60.000    81.00
## 50:    3.600    21.00
## 51:    4.288    39.20
## 52:    0.280     1.90
## 53:    0.075     1.20
## 54:    0.122     3.00
## 55:    0.048     0.33
## 56:  192.000   180.00
## 57:    3.000    25.00
## 58:  160.000   169.00
## 59:    0.900     2.60
## 60:    1.620    11.40
## 61:    0.104     2.50
## 62:    4.235    50.40
##      Body Wt Brain Wt
```

Above is the converted tidy data frames using the base R functions. A summary of the data is as follows:

Body Wt	Brain Wt
Min. : 0.005	Min. : 0.10
1st Qu.: 0.600	1st Qu.: 4.25
Median : 3.342	Median : 17.25
Mean : 198.790	Mean : 283.13
3rd Qu.: 48.202	3rd Qu.: 166.00
Max. :6654.000	Max. :5712.00

Now, handle the same data with `tidyverse` package.

```
# Tidy data with tidyverse
# Remove vectors which have only missing values
brain_data <- brain_rawdata[,1:6]

# Extracting 'Year' columns and 'Long Jump' columns
colnames(brain_data) <- paste(rep(c('bw', 'brw'),3), rep(1:3,each = 2))
bw <- brain_data[,c(1,3,5)] %>% gather(key = 'name1', value = 'BW', 1,2,3)
brw <- brain_data[,c(2,4,6)] %>% gather(key = 'name2', value = 'BRW', 1,2,3)

# Bind the vectors as a data table and rename the categories
brain_data <- bind_cols(bw[,2], brw[,2])
```

```
## New names:
## * NA -> ...1
## * NA -> ...2
```

```
colnames(brain_data) <- c('Body Wt', 'Brain Wt')
brain_data_tidyverse <- brain_data

# Final tidy data with tidyverse
brain_data_tidyverse
```

```
## # A tibble: 63 x 2
##    `Body Wt` `Brain Wt`
##        <dbl>      <dbl>
##  1     3.38        44.5
##  2     0.48        15.5
##  3     1.35         8.1
##  4   465          423
##  5    36.3        120.
##  6    27.7        115
##  7    14.8         98.2
##  8     1.04         5.5
##  9     4.19        58
## 10     0.425        6.4
## # ... with 53 more rows
```

The result by tidyverse is the same with the base R function. The summary of the data converted by tidyverse is as follows.

Body Wt	Brain Wt
Min. : 0.005	Min. : 0.10
1st Qu.: 0.600	1st Qu.: 4.25
Median : 3.342	Median : 17.25
Mean : 198.790	Mean : 283.13
3rd Qu.: 48.202	3rd Qu.: 166.00
Max. :6654.000	Max. :5712.00
NA's :1	NA's :1

d. Triplicate measurements of tomato yield for two varieties of tomatos at three planting densities.
http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/tomato.dat

```
######## Tomato data ########
# Getting "http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/tomato.dat"
```

```r
url_tomato <- "http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/tomato.dat"
tomato_rawdata <- fread(url_tomato, skip = 1)
```

```
## Warning in fread(url_tomato, skip = 1): Detected 3 column names but the data has
## 4 columns (i.e. invalid file). Added 1 extra default column name for the first
## column which is guessed to be row names or an index. Use setnames() afterwards
## if this guess is not correct, or fix the file write command that created the
## file to create a valid file.
```

```r
saveRDS(tomato_rawdata, 'tomato_rawdata.RDS')
tomato_rawdata <- readRDS('tomato_rawdata.RDS')
```

The values are grouped in the cells of the data above. Therefore, we should split the cells into single values.

```r
# Using base R function only
# Need to split the values
cells <- strsplit(unlist(tomato_rawdata), split = ',', fixed = T) # split the data
categories <- unlist(c(cells[1],cells[2]))  # two categories
values <- as.numeric(unlist(c(cells[3:8]))) # numerical data

# Combine the split values into data frame
tomato_matrix <- matrix(values, byrow = T, ncol = 3)
tomato_matrix <- t(cbind(tomato_matrix[1:2,], tomato_matrix[3:4,], tomato_matrix[5:6,]))

# Bind the data with the densities (categories)
tomato_data <- data.frame(tomato_matrix, as.character(rep(c(10000,20000,30000), each=3)))
colnames(tomato_data) <- c(categories, 'Density')
tomato_data_base <- tomato_data

# Final tidy data with base R functions
tomato_data_base
```

```
##   Ife\\#1 PusaEarlyDwarf Density
## 1    16.1            8.1   10000
## 2    15.3            8.6   10000
## 3    17.5           10.1   10000
## 4    16.6           12.7   20000
## 5    19.2           13.7   20000
## 6    18.5           11.5   20000
## 7    20.8           14.4   30000
## 8    18.0           15.4   30000
## 9    21.0           13.7   30000
```

Above is the converted tidy data frames using the base R functions. A summary of the data is as follows:

Ife#1	PusaEarlyDwarf	Density
Min. :15.30	Min. : 8.10	Length:9
1st Qu.:16.60	1st Qu.:10.10	Class :character
Median :18.00	Median :12.70	Mode :character
Mean :18.11	Mean :12.02	NA
3rd Qu.:19.20	3rd Qu.:13.70	NA
Max. :21.00	Max. :15.40	NA

Now, handle the same data with `tidyverse` package.

```
# Using tidyverse package
# Need to split the values
tomato_data <- tomato_rawdata[,-1] %>%
  separate(col = '10000', into = c("1","2","3"), sep = ",", convert = T) %>%
  separate(col = '20000', into = c("4","5","6"), sep = ",", convert = T) %>%
  separate(col = '30000', into = c("7","8","9"), sep = ",", convert = T) %>%
  as.matrix() %>% t()
```

```
## Warning: Expected 3 pieces. Additional pieces discarded in 1 rows [2].
```

```
dens <- rep(c(10000,20000,30000), each = 3) %>% as.character() # Densities

# Bind the data with the densities vector
tomato_data <- tomato_data %>% cbind(dens) %>% as.data.table()
colnames(tomato_data) <- tomato_rawdata[,1] %>% unlist() %>% c("Density")
tomato_data_tidyverse <- tomato_data

# Final tidy data with tidyverse
tomato_data_tidyverse
```

```
##    Ife\\#1 PusaEarlyDwarf Density
## 1:    16.1            8.1   10000
## 2:    15.3            8.6   10000
## 3:    17.5           10.1   10000
## 4:    16.6           12.7   20000
## 5:    19.2           13.7   20000
## 6:    18.5           11.5   20000
## 7:    20.8           14.4   30000
## 8:      18           15.4   30000
## 9:      21           13.7   30000
```

Ife#1	PusaEarlyDwarf	Density
Length:9 Class :character Mode :character	Length:9 Class :character Mode :character	Length:9 Class :character Mode :character

Problem 5

Finish this homework by pushing your changes to your repo. In general, your workflow for this should be:

1. git pull – to make sure you have the most recent repo

2. In R: do some work

3. git add – this tells git to track new files

4. git commit – make message INFORMATIVE and USEFUL

5. git push – this pushes your local changes to the repo

If you have difficulty with steps 1-5, git is not correctly or completely setup. See me for help.

Only submit the .Rmd and .pdf solution files. Names should be formatted HW2_lastname.Rmd and HW2_lastname.pdf

Optional preperation for next class:

TBD

Appendix

```r
knitr::opts_chunk$set(echo = TRUE)
library(data.table)
library(tidyverse)
######## Sensory data ########
# Getting "https://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/Sensory.dat"
url_sensory <- "https://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/Sensory.dat"
sensory_rawdata <- fread(url_sensory, fill = TRUE, skip = 2, data.table = FALSE)
saveRDS(sensory_rawdata, 'sensory_rawdata.RDS')
sensory_rawdata <- readRDS('sensory_rawdata.RDS')

# Using base R function only
# Convert data.frame to matrix and transpose the raw data
matrix_sensory <- t(as.matrix(sensory_rawdata))

# Find where the missing values are
na <- which(is.na(matrix_sensory==TRUE))

# The indexes where Item numbers are in the data
x <- 1
item <- x
for (i in 1:9){
  x <- x+18
  item <- c(item, x)
}

# Remove missing values and 'Item's from the data
sensory_data <- t(matrix(matrix_sensory[-c(na,item)], byrow = T, nrow = 10))
sensory_data <- data.table(sensory_data)
colnames(sensory_data) <- paste('Item', 1:10) # Assign column names
Opr <- rep(paste('Opr', 1:5), 3)              # Operator names
sensory_data <- cbind(Opr,sensory_data)       # Bind Operator names and the data

# Re-order the rows by names of operators
sensory_data <- sensory_data[order(sensory_data$Opr)]
sensory_data_base <- sensory_data

# Final tidy data with base R functions
sensory_data_base

knitr::kable(summary(sensory_data_base))
# Sensory data with tidyverse package
# Making matrix which is the same with base R function but using pipes.
matrix_sensory <- sensory_rawdata %>% as.matrix() %>% t()
na <- which(is.na(matrix_sensory==TRUE))     # Find missing values

# The indexes where Item numbers are in the data
```

```r
x <- 1
item <- x
for (i in 1:9){
  x <- x+18
  item <- c(item, x)
}

# Remove missing values and Item numbers from the data
sensory_data <- matrix_sensory[-c(na,item)] %>% matrix(byrow = T, nrow = 10) %>% t()
sensory_data <- data.table(sensory_data)
Opr <- rep(paste('Opr', 1:5), 3)
sensory_data <- bind_cols(Opr,sensory_data) # bind operators and data
colnames(sensory_data) <- c('Opr',paste('Item', 1:10))

# Re-order the rows by names of operators
sensory_data <- sensory_data[order(sensory_data$Opr)]
sensory_data_tidyverse <- sensory_data

# Final tidy data with tidyverse
sensory_data_tidyverse

knitr::kable(summary(sensory_data_tidyverse))
######## Long Jump data ########
# Getting "http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/LongJumpData.dat"
url_medal <- "http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/LongJumpData.dat"
medal_rawdata <- fread(url_medal)
saveRDS(medal_rawdata, 'medal_rawdata.RDS')
medal_rawdata <- readRDS('medal_rawdata.RDS')

# Using base R function only
# Year is coded as 1900 = 0
# Extract Year and Long Jump vectors
year <- c(medal_rawdata[[1]], medal_rawdata[[3]],
          medal_rawdata[[5]], medal_rawdata[[7]]) + 1900
longjump <- c(medal_rawdata[[2]], medal_rawdata[[4]],
              medal_rawdata[[6]], medal_rawdata[[8]])

# Bind the vectors as a data table and rename the categories
medal_data <- data.table(year[1:(length(year)-2)],
                         longjump[1:(length(longjump)-2)])
colnames(medal_data) <- c('Year', 'Long Jump')
medal_data_base <- medal_data

# Final tidy data with base R functions
medal_data_base

knitr::kable(summary(medal_data_base))
# Using tidyverse package
# Year is coded as 1900 = 0
medal_data <- medal_rawdata[,1:8] # remove missing values only columns

# Extracting 'Year' columns and 'Long Jump' columns
colnames(medal_data) <- paste(rep(c('Year', 'Jump'),4), rep(1:4,each = 2))
```

```r
year <- medal_data[,c(1,3,5,7)] %>%
  gather(key = 'name1', value = 'Year', 1,2,3,4) %>% filter(Year != na)
year[,2] <- year[,2] + 1900
jump <- medal_data[,c(2,4,6,8)] %>%
  gather(key = 'name2', value = 'LongJump', 1,2,3,4) %>% filter(LongJump != na)

# Bind the vectors as a data table and rename the categories
medal_data <- bind_cols(year[,2], jump[,2])
colnames(medal_data) <- c('Year', 'Long Jump')
medal_data_tidyverse <- medal_data

# Final tidy data with tidyverse
medal_data_tidyverse


######## Brain weight data ########
# Getting "http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/BrainandBodyWeight.dat"
url_brain <- "http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/BrainandBodyWeight.dat"
brain_rawdata <- fread(url_brain)
saveRDS(brain_rawdata, 'brain_rawdata.RDS')
brain_rawdata <- readRDS('brain_rawdata.RDS')
# Using base R function only
# The method is the same with the data from part (b)
# Extract Body Wt and Brain Wt vectors
bodywt <- c(brain_rawdata[[1]], brain_rawdata[[3]], brain_rawdata[[5]])
brainwt <- c(brain_rawdata[[2]], brain_rawdata[[4]], brain_rawdata[[6]])

# Remove missing values
brain_data <- data.table(bodywt[-length(bodywt)], brainwt[-length(brainwt)])
colnames(brain_data) <- c('Body Wt', 'Brain Wt')
brain_data_base <- brain_data

# Final tidy data with base R functions
brain_data_base

knitr::kable(summary(brain_data_base))
# Tidy data with tidyverse
# Remove vectors which have only missing values
brain_data <- brain_rawdata[,1:6]

# Extracting 'Year' columns and 'Long Jump' columns
colnames(brain_data) <- paste(rep(c('bw', 'brw'),3), rep(1:3,each = 2))
bw <- brain_data[,c(1,3,5)] %>% gather(key = 'name1', value = 'BW', 1,2,3)
brw <- brain_data[,c(2,4,6)] %>% gather(key = 'name2', value = 'BRW', 1,2,3)

# Bind the vectors as a data table and rename the categories
brain_data <- bind_cols(bw[,2], brw[,2])
colnames(brain_data) <- c('Body Wt', 'Brain Wt')
brain_data_tidyverse <- brain_data

# Final tidy data with tidyverse
brain_data_tidyverse
```

```r
######## Tomato data ########
# Getting "http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/tomato.dat"
url_tomato <- "http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/tomato.dat"
tomato_rawdata <- fread(url_tomato, skip = 1)
saveRDS(tomato_rawdata, 'tomato_rawdata.RDS')
tomato_rawdata <- readRDS('tomato_rawdata.RDS')

# Using base R function only
# Need to split the values
cells <- strsplit(unlist(tomato_rawdata), split = ',', fixed = T) # split the data
categories <- unlist(c(cells[1],cells[2]))  # two categories
values <- as.numeric(unlist(c(cells[3:8]))) # numerical data

# Combine the split values into data frame
tomato_matrix <- matrix(values, byrow = T, ncol = 3)
tomato_matrix <- t(cbind(tomato_matrix[1:2,], tomato_matrix[3:4,], tomato_matrix[5:6,]))

# Bind the data with the densities (categories)
tomato_data <- data.frame(tomato_matrix, as.character(rep(c(10000,20000,30000), each=3)))
colnames(tomato_data) <- c(categories, 'Density')
tomato_data_base <- tomato_data

# Final tidy data with base R functions
tomato_data_base

knitr::kable(summary(tomato_data_base))
# Using tidyverse package
# Need to split the values
tomato_data <- tomato_rawdata[,-1] %>%
  separate(col = '10000', into = c("1","2","3"), sep = ",", convert = T) %>%
  separate(col = '20000', into = c("4","5","6"), sep = ",", convert = T) %>%
  separate(col = '30000', into = c("7","8","9"), sep = ",", convert = T) %>%
  as.matrix() %>% t()
dens <- rep(c(10000,20000,30000), each = 3) %>% as.character() # Densities

# Bind the data with the densities vector
tomato_data <- tomato_data %>% cbind(dens) %>% as.data.table()
colnames(tomato_data) <- tomato_rawdata[,1] %>% unlist() %>% c("Density")
tomato_data_tidyverse <- tomato_data

# Final tidy data with tidyverse
tomato_data_tidyverse
```