Homework 3

Due Wednesday September 18, 2019

Eric Bae

2019-09-16

For each assignment, turn in by the due date/time. Late assignments must be arranged prior to submission. In every case, assignments are to be typed neatly using proper English in Markdown.

This week, we spoke about R and version control, munging and 'tidying' data, good programming practice and finally some basic programming building blocs. To begin the homework, we will for the rest of the course, start by loading data and then creating tidy data sets.

Problem 1

Work through the "Getting and Cleaning Data" lesson parts 3 and 4.

From the R command prompt:

```
library(swirl)
swirl()
```

Problem 2

Create a new R Markdown file within your local GitHub repo folder (file–>new–>R Markdown–>save as).

The filename should be: HW3_lastname, i.e. for me it would be HW3_Settlage

You will use this new R Markdown file to solve the following problems.

Problem 3

Redo Problem 4 parts a-d from last time using the tidyverse functions and piping.

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

```
# Sensory Table
sensory.url <- "http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/Sensory.dat"

# Importing data as dataframe
sensory <- as.data.frame(fread(sensory.url, fill = TRUE, skip = 2))
N <- nrow(sensory) # Number of rows = 30
D <- ncol(sensory) - 1 # Number of objectives/variables = 5
```

```r
# Rearranging data so that NA goes to the first row
for (i in 1:N) {
  if (is.na(sensory$V6[i])) {
    sensory[i,-1] <- sensory[i,1:D]
    sensory[i,1] <- NA
  }
}

# Total number of items = 10
I <- length(sensory$V1[which(is.na(sensory$V1) == FALSE)])

# Reorganizes item
sensory$V1 <- rep(1:I, each = 3)

# Tidying up, one column by column
Item <- sort(rep(sensory$V1, each = D))
#Observation <- rep(rep(1:3, each = 5), I)
Operator <- rep(rep(1:D), 3*I)
Dat <- c(t(sensory[,-1]))

# Combining the columns
#sensory <- as_tibble(cbind(Item, Observation, Operator, Dat))
sensory <- as_tibble(cbind(Item, Operator, Dat))

# Summary statistics
summary(sensory)
```

```
##       Item         Operator      Dat
##  Min.   : 1.0   Min.   :1   Min.   :0.700
##  1st Qu.: 3.0   1st Qu.:2   1st Qu.:3.025
##  Median : 5.5   Median :3   Median :4.700
##  Mean   : 5.5   Mean   :3   Mean   :4.657
##  3rd Qu.: 8.0   3rd Qu.:4   3rd Qu.:6.000
##  Max.   :10.0   Max.   :5   Max.   :9.400
```

```r
sensory.operator <- sensory %>%
  group_by(Operator) %>%
  summarize(Mean = mean(Dat), SD = sd(Dat), Min = min(Dat), Max = max(Dat))
sensory.item <- sensory %>%
  group_by(Item) %>%
  summarize(Mean = mean(Dat), SD = sd(Dat), Min = min(Dat), Max = max(Dat))
sensory.itemoperator <- sensory %>%
  group_by(Item, Operator) %>%
  summarize(Mean = mean(Dat), SD = sd(Dat), Min = min(Dat), Max = max(Dat))
sensory.operator
```

```
## # A tibble: 5 x 5
##   Operator  Mean    SD   Min   Max
##      <dbl> <dbl> <dbl> <dbl> <dbl>
## 1       1.  4.59  2.24 0.900  9.00
## 2       2.  5.06  2.05 1.50   9.20
## 3       3.  4.17  2.10 0.800  9.00
## 4       4.  5.19  2.13 0.900  9.40
## 5       5.  4.27  2.14 0.700  8.80
```

```
sensory.item
```

```
## # A tibble: 10 x 5
##     Item  Mean    SD   Min   Max
##    <dbl> <dbl> <dbl> <dbl> <dbl>
##  1    1.  4.47 0.779 3.30   5.70
##  2    2.  5.31 0.713 4.20   6.30
##  3    3.  2.77 0.841 1.30   4.60
##  4    4.  6.88 0.665 5.90   8.20
##  5    5.  5.92 0.500 4.90   7.00
##  6    6.  2.39 0.818 1.10   4.00
##  7    7.  1.41 0.642 0.700  3.10
##  8    8.  4.43 0.574 3.00   4.90
##  9    9.  8.47 0.761 6.70   9.40
## 10   10.  4.52 0.825 2.80   5.50
```

```
sensory.itemoperator
```

```
## # A tibble: 50 x 6
## # Groups:   Item [?]
##     Item Operator  Mean    SD   Min   Max
##    <dbl>    <dbl> <dbl> <dbl> <dbl> <dbl>
##  1    1.       1.  4.23 0.115  4.10  4.30
##  2    1.       2.  4.90 0.400  4.50  5.30
##  3    1.       3.  3.57 0.379  3.30  4.00
##  4    1.       4.  5.50 0.200  5.30  5.70
##  5    1.       5.  4.13 0.737  3.30  4.70
##  6    2.       1.  5.63 0.635  4.90  6.00
##  7    2.       2.  5.83 0.503  5.30  6.30
##  8    2.       3.  4.47 0.252  4.20  4.70
##  9    2.       4.  5.90 0.400  5.50  6.30
## 10    2.       5.  4.73 0.153  4.60  4.90
## # ... with 40 more rows
```

The summary statistics shows that the all-time minimum sensory measurement (no idea what the unit is) is 0.700, the maximum is 9.400, and the mean is 4.657.
Grouping by operator, operator 3 had the minimum average measurement with the value of 4.17, and operator 4 had the maximum average measurement with the value of 5.19.
Grouping by item, item 7 had the minimum mean measurement at 1.41 while item 9 had the highest at 8.47.

```
# Kable
kable(sensory.operator, format = "markdown",
      caption = "Summary statistics, grouped by operator")
```

Operator	Mean	SD	Min	Max
1	4.593333	2.239140	0.9	9.0
2	5.063333	2.045429	1.5	9.2
3	4.166667	2.098494	0.8	9.0
4	5.193333	2.132334	0.9	9.4
5	4.266667	2.143206	0.7	8.8

```
kable(sensory.item, format = "markdown",
      caption = "Summary statistics, grouped by item")
```

Item	Mean	SD	Min	Max
1	4.466667	0.7788881	3.3	5.7
2	5.313333	0.7130084	4.2	6.3
3	2.773333	0.8413142	1.3	4.6
4	6.880000	0.6646159	5.9	8.2
5	5.920000	0.5002856	4.9	7.0
6	2.393333	0.8180697	1.1	4.0
7	1.406667	0.6419464	0.7	3.1
8	4.426667	0.5737927	3.0	4.9
9	8.466667	0.7612646	6.7	9.4
10	4.520000	0.8247943	2.8	5.5

```
kable(sensory.itemoperator, format = "markdown",
      caption = "Summary statistics, grouped by item & operator")
```

Item	Operator	Mean	SD	Min	Max
1	1	4.233333	0.1154701	4.1	4.3
1	2	4.900000	0.4000000	4.5	5.3
1	3	3.566667	0.3785939	3.3	4.0
1	4	5.500000	0.2000000	5.3	5.7
1	5	4.133333	0.7371115	3.3	4.7
2	1	5.633333	0.6350853	4.9	6.0
2	2	5.833333	0.5033223	5.3	6.3
2	3	4.466667	0.2516611	4.2	4.7
2	4	5.900000	0.4000000	5.5	6.3
2	5	4.733333	0.1527525	4.6	4.9
3	1	2.733333	1.0408330	1.9	3.9
3	2	3.133333	0.7094599	2.5	3.9
3	3	2.566667	0.2516611	2.3	2.8
3	4	3.466667	1.0016653	2.7	4.6
3	5	1.966667	0.5859465	1.3	2.4
4	1	6.966667	0.5131601	6.4	7.4
4	2	7.733333	0.5686241	7.1	8.2
4	3	6.400000	0.5000000	5.9	6.9
4	4	7.033333	0.2516611	6.8	7.3
4	5	6.266667	0.3785939	6.0	6.7
5	1	5.766667	0.0577350	5.7	5.8
5	2	6.000000	0.3000000	5.7	6.3
5	3	5.633333	0.4041452	5.4	6.1
5	4	6.433333	0.4932883	6.1	7.0
5	5	5.766667	0.8082904	4.9	6.5
6	1	2.433333	0.4932883	2.1	3.0
6	2	2.500000	0.7549834	1.8	3.3
6	3	1.633333	0.5033223	1.1	2.1
6	4	3.566667	0.3785939	3.3	4.0
6	5	1.833333	0.2309401	1.7	2.1
7	1	1.133333	0.2081666	0.9	1.3
7	2	2.333333	0.8020806	1.5	3.1
7	3	1.033333	0.2081666	0.8	1.2
7	4	1.333333	0.5131601	0.9	1.9
7	5	1.200000	0.4582576	0.7	1.6
8	1	4.000000	0.9165151	3.0	4.8

Item	Operator	Mean	SD	Min	Max
8	2	4.700000	0.1732051	4.5	4.8
8	3	4.633333	0.1154701	4.5	4.7
8	4	4.766667	0.1527525	4.6	4.9
8	5	4.033333	0.7371115	3.2	4.6
9	1	8.633333	0.5507571	8.0	9.0
9	2	8.500000	0.7549834	7.7	9.2
9	3	7.933333	1.1590226	6.7	9.0
9	4	9.166667	0.2081666	9.0	9.4
9	5	8.100000	0.6244998	7.6	8.8
10	1	4.400000	1.4000000	2.8	5.4
10	2	5.000000	0.2000000	4.8	5.2
10	3	3.800000	0.3605551	3.4	4.1
10	4	4.766667	0.8082904	3.9	5.5
10	5	4.633333	0.8504901	3.8	5.5

```
kable(sensory, format = "markdown",
      caption = "Sensory data, reformated")
```

Item	Operator	Dat
1	1	4.3
1	2	4.9
1	3	3.3
1	4	5.3
1	5	4.4
1	1	4.3
1	2	4.5
1	3	4.0
1	4	5.5
1	5	3.3
1	1	4.1
1	2	5.3
1	3	3.4
1	4	5.7
1	5	4.7
2	1	6.0
2	2	5.3
2	3	4.5
2	4	5.9
2	5	4.7
2	1	4.9
2	2	6.3
2	3	4.2
2	4	5.5
2	5	4.9
2	1	6.0
2	2	5.9
2	3	4.7
2	4	6.3
2	5	4.6
3	1	2.4
3	2	2.5

Item	Operator	Dat
3	3	2.3
3	4	3.1
3	5	2.4
3	1	3.9
3	2	3.0
3	3	2.8
3	4	2.7
3	5	1.3
3	1	1.9
3	2	3.9
3	3	2.6
3	4	4.6
3	5	2.2
4	1	7.4
4	2	8.2
4	3	6.4
4	4	6.8
4	5	6.0
4	1	7.1
4	2	7.9
4	3	5.9
4	4	7.3
4	5	6.1
4	1	6.4
4	2	7.1
4	3	6.9
4	4	7.0
4	5	6.7
5	1	5.7
5	2	6.3
5	3	5.4
5	4	6.1
5	5	5.9
5	1	5.8
5	2	5.7
5	3	5.4
5	4	6.2
5	5	6.5
5	1	5.8
5	2	6.0
5	3	6.1
5	4	7.0
5	5	4.9
6	1	2.2
6	2	2.4
6	3	1.7
6	4	3.4
6	5	1.7
6	1	3.0
6	2	1.8
6	3	2.1
6	4	4.0

Item	Operator	Dat
6	5	1.7
6	1	2.1
6	2	3.3
6	3	1.1
6	4	3.3
6	5	2.1
7	1	1.2
7	2	1.5
7	3	1.2
7	4	0.9
7	5	0.7
7	1	1.3
7	2	2.4
7	3	0.8
7	4	1.2
7	5	1.3
7	1	0.9
7	2	3.1
7	3	1.1
7	4	1.9
7	5	1.6
8	1	4.2
8	2	4.8
8	3	4.5
8	4	4.6
8	5	3.2
8	1	3.0
8	2	4.5
8	3	4.7
8	4	4.9
8	5	4.6
8	1	4.8
8	2	4.8
8	3	4.7
8	4	4.8
8	5	4.3
9	1	8.0
9	2	8.6
9	3	9.0
9	4	9.4
9	5	8.8
9	1	9.0
9	2	7.7
9	3	6.7
9	4	9.0
9	5	7.9
9	1	8.9
9	2	9.2
9	3	8.1
9	4	9.1
9	5	7.6
10	1	5.0

Item	Operator	Dat
10	2	4.8
10	3	3.9
10	4	5.5
10	5	3.8
10	1	5.4
10	2	5.0
10	3	3.4
10	4	4.9
10	5	4.6
10	1	2.8
10	2	5.2
10	3	4.1
10	4	3.9
10	5	5.5

The above is the tidy version of the data set. There is obviously a lot of issues with this data set. Starting with the fact that it is extremely long, it is hard to navigate and confusing to understand.
I also had to use a lot of hard-coding, which is not ideal.

```r
# Gold Medals Table
medals.url <- "http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/LongJumpData.dat"
medals <- as.data.frame(fread(medals.url, fill = TRUE, skip = 1))

# Assign names
colnames(medals) <- rep(c("Year", "Long Jump"), 4)

# Tibble
medals <- as_tibble(rbind(medals[,1:2], medals[,3:4], medals[,5:6], medals[,7:8]))

# Add rankings to my table
medals <- medals %>%
  mutate(Rank = round(rank(-`Long Jump`), 0)) # Added rankings
medals <- medals[-which(is.na(medals$Year)),] # Remove the two NAs

# Summary statistics
medals.summary <- medals %>%
  summarize(Mean = mean(`Long Jump`), SD = sd(`Long Jump`),
            Min = min(`Long Jump`), Max = max(`Long Jump`))

# Kable
kable(medals.summary, format = "markdown",
      caption = "Summary statistics of Gold Medals data")
```

Mean	SD	Min	Max
310.2873	24.36121	249.75	350.5

```r
kable(medals, format = "markdown", caption = "Gold Medals data, reformated")
```

Year	Long Jump	Rank
-4	249.75	22
0	282.88	20
4	289.00	19
8	294.50	17
12	299.25	15
20	281.50	21
24	293.13	18
28	304.75	13
32	300.75	14
36	317.31	10
48	308.00	12
52	298.00	16
56	308.25	11
60	319.75	8
64	317.75	9
68	350.50	1
72	324.50	7
76	328.50	6
80	336.25	4
84	336.25	4
88	343.25	2
92	342.50	3

The above table is the table of the record long jump of the male gold medalist by olympic year. The index year (0) is 1900. I also added the rankings as the third column.
The minimum record was 249.75, recorded at the 1896 Olympics and the maximum record was 350.50, recorded at the 1968 Olympics. The mean record across the years was 310.3.
This one went a little bit better than the Sensory data set, though some hard coding was used, especially for rbind().

```r
# Brain and Body Weight Table
brain.url <- "https://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/BrainandBodyWeight.dat"
brain <- as.data.frame(fread(brain.url, fill = TRUE, skip = 1))

# Assign names
colnames(brain) <- rep(c("Body Wt", "Brain Wt"), 3)

# As tibble
brain <- as_tibble(rbind.data.frame(brain[,1:2], brain[,3:4], brain[,5:6]))
brain <- brain[-which(is.na(brain$`Body Wt`)),]

# Add brain weight/body weight ratio
brain <- brain %>%
  arrange(`Body Wt`) %>%
  mutate(`Brain-to-Body Ratio` = `Brain Wt`/`Body Wt`)

# Summary statistics
brain.bodywt <- brain %>%
  summarize(`Body Wt Mean` = mean(`Body Wt`),
            `Body Wt SD` = sd(`Body Wt`),
            `Body Wt Min` = min(`Body Wt`),
```

```r
            `Body Wt Max` = max(`Body Wt`))
brain.brainwt <- brain %>%
  summarize(`Brain Wt Mean` = mean(`Brain Wt`),
            `Brain Wt SD` = sd(`Brain Wt`),
            `Brain Wt Min` = min(`Brain Wt`),
            `Brain Wt Max` = max(`Brain Wt`))

# Kable
kable(brain.brainwt, format = "markdown",
      caption = "Summary statistics of brain wt")
```

Brain Wt Mean	Brain Wt SD	Brain Wt Min	Brain Wt Max
283.1344	930.2789	0.1	5712

```r
kable(brain.bodywt, format = "markdown",
      caption = "Summary statistics of body wt")
```

Body Wt Mean	Body Wt SD	Body Wt Min	Body Wt Max
198.79	899.158	0.005	6654

```r
kable(brain, format = "markdown", caption = "Brain and Body Weight data, reformated")
```

Body Wt	Brain Wt	Brain-to-Body Ratio
0.005	0.10	20.0000000
0.010	0.30	30.0000000
0.023	0.30	13.0434783
0.023	0.40	17.3913043
0.048	0.33	6.8750000
0.060	1.00	16.6666667
0.075	1.20	16.0000000
0.101	4.00	39.6039604
0.104	2.50	24.0384615
0.120	1.00	8.3333333
0.122	3.00	24.5901639
0.200	5.00	25.0000000
0.280	1.90	6.7857143
0.425	6.40	15.0588235
0.480	15.50	32.2916667
0.550	2.40	4.3636364
0.750	12.30	16.4000000
0.785	3.50	4.4585987
0.900	2.60	2.8888889
0.920	5.70	6.1956522
1.000	6.60	6.6000000
1.040	5.50	5.2884615
1.350	8.10	6.0000000
1.400	12.50	8.9285714
1.410	17.50	12.4113475
1.620	11.40	7.0370370
1.700	6.30	3.7058824

Body Wt	Brain Wt	Brain-to-Body Ratio
2.000	12.30	6.1500000
2.500	12.10	4.8400000
3.000	25.00	8.3333333
3.300	25.60	7.7575758
3.385	44.50	13.1462334
3.500	10.80	3.0857143
3.500	3.90	1.1142857
3.600	21.00	5.8333333
4.050	17.00	4.1975309
4.190	58.00	13.8424821
4.235	50.40	11.9008264
4.288	39.20	9.1417910
6.800	179.00	26.3235294
10.000	115.00	11.5000000
10.550	179.50	17.0142180
14.830	98.20	6.6217127
27.660	115.00	4.1576283
35.000	56.00	1.6000000
36.330	119.50	3.2892926
52.160	440.00	8.4355828
55.500	175.00	3.1531532
60.000	81.00	1.3500000
62.000	1320.00	21.2903226
85.000	325.00	3.8235294
100.000	157.00	1.5700000
160.000	169.00	1.0562500
187.100	419.00	2.2394441
192.000	180.00	0.9375000
207.000	406.00	1.9613527
250.000	490.00	1.9600000
465.000	423.00	0.9096774
521.000	655.00	1.2571977
529.000	680.00	1.2854442
2547.000	4603.00	1.8072242
6654.000	5712.00	0.8584310

The average weight of all bodies observed was 198.790 kg whereas the average weight of all brains was 283.13 g. However, the median weight of bodies and brains were 3.342 kg and 17.25 g, respectively, suggesting that both weight were skewed heavily to the right. The weights ranged from 0.005 kg to 6,654.000 kg for body and 0.10 g to 5,712.00 g for brain.
I added the brain-to-body weight ratio (in 1000s) as the third column just for reference. Though I did not perform statistical analysis, simple eyeballing it seems to indicate that the larger the body weight the lower the ratio.
The issue here was more similar to the issue I faced with the medals dataset in that there was some hard coding with binding the three pairs of columns into one.

```
# Tomato Yield Table
tomato.url <- "http://www2.isye.gatech.edu/~jeffwu/wuhamadabook/data/tomato.dat"
tomato <- as.data.frame(fread(tomato.url, fill = TRUE, skip = 1))

# Assign names
```

```r
colnames(tomato) <- as.character(tomato[1, c(4, 1:3)])
colnames(tomato)[1] <- "Tomato Variety"
tomato <- tomato[-1,]
tomato <- tomato[rep(1:2, each = 3),]

new.tomato <- tomato
for (i in seq(1, nrow(tomato), by = 3)) {
  for (j in 2:ncol(tomato)) {
    new.tomato[seq(i, i+2),j] <- strsplit(as.character(tomato[i,j]), ",")[[1]]
  }
}

# Tidying up, one column by column
Tomato <- c(t(rep(new.tomato$`Tomato Variety`, each = ncol(new.tomato) - 1)))
Density <- rep(colnames(new.tomato)[-1], length(unique(Tomato)))
Yield <- as.numeric(c(t(new.tomato[,-1])))
tomato <- as_tibble(cbind.data.frame(Tomato, Density, Yield))

# Summary statistics
tomato.summary <- tomato %>%
  group_by(Tomato) %>%
  summarize(`Mean Tomato Yield` = mean(Yield),
            `Tomato Yield SD` = sd(Yield),
            `Min Tomato Yield` = min(Yield),
            `Max Tomato Yield` = max(Yield))

tomato.summary.den <- tomato %>%
  group_by(Density) %>%
  summarize(`Mean Tomato Yield` = mean(Yield),
            `Tomato Yield SD` = sd(Yield),
            `Min Tomato Yield` = min(Yield),
            `Max Tomato Yield` = max(Yield))

tomato.summary.both <- tomato %>%
  group_by(Tomato, Density) %>%
  summarize(`Mean Tomato Yield` = mean(Yield),
            `Tomato Yield SD` = sd(Yield),
            `Min Tomato Yield` = min(Yield),
            `Max Tomato Yield` = max(Yield))

# Kable
kable(tomato.summary, format = "markdown",
      caption = "Summary statistics of tomato yield by variety")
```

Tomato	Mean Tomato Yield	Tomato Yield SD	Min Tomato Yield	Max Tomato Yield
Ife#1	18.11111	1.985223	15.3	21.0
PusaEarlyDwarf	12.02222	2.603257	8.1	15.4

```r
kable(tomato.summary.den, format = "markdown",
      caption = "Summary statistics of tomato yield by density")
```

Density	Mean Tomato Yield	Tomato Yield SD	Min Tomato Yield	Max Tomato Yield
10000	12.61667	4.148453	8.1	17.5
20000	15.36667	3.189775	11.5	19.2
30000	17.21667	3.205256	13.7	21.0

```
kable(tomato.summary.both, format = "markdown",
      caption = "Summary statistics of tomato yield by variety and density")
```

Tomato	Density	Mean Tomato Yield	Tomato Yield SD	Min Tomato Yield	Max Tomato Yield
Ife#1	10000	16.300000	1.1135529	15.3	17.5
Ife#1	20000	18.100000	1.3453624	16.6	19.2
Ife#1	30000	19.933333	1.6772994	18.0	21.0
PusaEarlyDwarf	10000	8.933333	1.0408330	8.1	10.1
PusaEarlyDwarf	20000	12.633333	1.1015141	11.5	13.7
PusaEarlyDwarf	30000	14.500000	0.8544004	13.7	15.4

```
kable(tomato, format = "markdown", caption = "Tomato yield by variety, reformated")
```

Tomato	Density	Yield
Ife#1	10000	16.1
Ife#1	20000	16.6
Ife#1	30000	20.8
Ife#1	10000	15.3
Ife#1	20000	19.2
Ife#1	30000	18.0
Ife#1	10000	17.5
Ife#1	20000	18.5
Ife#1	30000	21.0
PusaEarlyDwarf	10000	8.1
PusaEarlyDwarf	20000	12.7
PusaEarlyDwarf	30000	14.4
PusaEarlyDwarf	10000	8.6
PusaEarlyDwarf	20000	13.7
PusaEarlyDwarf	30000	15.4
PusaEarlyDwarf	10000	10.1
PusaEarlyDwarf	20000	11.5
PusaEarlyDwarf	30000	13.7

The first column represents the tomato variety, the second the planting density, and the third the yield. The lowest yield was found in the tomato variety called "PusaEarlyDwarf" and planting density of 10,000, while the highest yield was found in "Ife#1" and planting density of 30,000.
I had a lot of trouble with this data set because the yield data was not numeric but actually some sort of characters, delineated by commas. I had to find a way to split them up by the commas, but unable to change the classes of the yields without causing the numbers to change. This is why summary table is not available here because I could not find a way to generate summary statistics for character outputs.

Problem 4

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

1. In terminal: git pull – to make sure you have the most recent local repo

2. In terminal: do some work

3. In terminal: git add – check files you want to commit

4. In terminal: git commit – make message INFORMATIVE and USEFUL

5. In terminal: 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.

Only submit the .Rmd and .pdf solution files. Names should be formatted HW3_lastname_firstname.Rmd and HW3_lastname_firstname.pdf

Optional preperation for next class:

TBD – could be something sent as a class message