Notes on Pset 1 #2

S&DS 361

2024-01-25

There was some confusion, and a typo by me, for #2, so here are some notes. First let's load the data.

```
d = readRDS('data/water.usage.rds')
head(d)
```

```
## # A tibble: 6 x 5
##   GEOID crop        src   year  value
##   <int> <chr>       <chr> <chr> <dbl>
## 1  1001 barley      gwd   2008      0
## 2  1001 corn        gwd   2008      0
## 3  1001 cotton      gwd   2008      0
## 4  1001 millet      gwd   2008      0
## 5  1001 oats        gwd   2008      0
## 6  1001 other_sctg2 gwd   2008      0
```

We want average water usage by year for each crop and source. The first incorrect thing we did is this

```
dd1 = d %>%
  group_by(crop, src, year) %>%
  summarise(value = mean(value))
```

```
## `summarise()` has grouped output by 'crop', 'src'. You can override using the
## `.groups` argument.
```

```
dd1
```

```
## # A tibble: 780 x 4
## # Groups:   crop, src [60]
##    crop   src   year     value
##    <chr>  <chr> <chr>    <dbl>
##  1 barley gwa   2008  0.000376
##  2 barley gwa   2009  0.000368
##  3 barley gwa   2010  0.000357
##  4 barley gwa   2011  0.000476
##  5 barley gwa   2012  0.000455
##  6 barley gwa   2013  0.000395
##  7 barley gwa   2014  0.000266
##  8 barley gwa   2015  0.000428
##  9 barley gwa   2016  0.000347
## 10 barley gwa   2017  0.000359
## # i 770 more rows
```

```r
dd2 = dd1 %>%
  group_by(crop, src) %>%
  summarise(value = mean(value))
```

```
## `summarise()` has grouped output by 'crop'. You can override using the
## `.groups` argument.
```

```r
dd2
```

```
## # A tibble: 60 x 3
## # Groups:   crop [20]
##    crop   src       value
##    <chr>  <chr>     <dbl>
##  1 barley gwa   0.000372
##  2 barley gwd   0.000222
##  3 barley sw    0.000688
##  4 corn   gwa   0.00176
##  5 corn   gwd   0.00110
##  6 corn   sw    0.00171
##  7 cotton gwa   0.000623
##  8 cotton gwd   0.000442
##  9 cotton sw    0.000517
## 10 millet gwa   0.0000231
## # i 50 more rows
```

I meant to say `sum` here. What I meant was

```r
dd3 = d %>%
  group_by(crop, src, year) %>%
  summarise(value = sum(value)) ## sum not mean
```

```
## `summarise()` has grouped output by 'crop', 'src'. You can override using the
## `.groups` argument.
```

```r
dd3
```

```
## # A tibble: 780 x 4
## # Groups:   crop, src [60]
##    crop   src   year  value
##    <chr>  <chr> <chr> <dbl>
##  1 barley gwa   2008  1.21
##  2 barley gwa   2009  1.19
##  3 barley gwa   2010  1.11
##  4 barley gwa   2011  1.53
##  5 barley gwa   2012  1.46
##  6 barley gwa   2013  1.27
##  7 barley gwa   2014  0.857
##  8 barley gwa   2015  1.33
##  9 barley gwa   2016  1.12
## 10 barley gwa   2017  1.16
## # i 770 more rows
```

```r
dd4 = dd3 %>%
  group_by(crop, src) %>%
  mutate(mean = mean(value))

dd4
```

```
## # A tibble: 780 x 5
## # Groups:   crop, src [60]
##    crop   src   year  value  mean
##    <chr>  <chr> <chr> <dbl> <dbl>
##  1 barley gwa   2008  1.21   1.19
##  2 barley gwa   2009  1.19   1.19
##  3 barley gwa   2010  1.11   1.19
##  4 barley gwa   2011  1.53   1.19
##  5 barley gwa   2012  1.46   1.19
##  6 barley gwa   2013  1.27   1.19
##  7 barley gwa   2014  0.857  1.19
##  8 barley gwa   2015  1.33   1.19
##  9 barley gwa   2016  1.12   1.19
## 10 barley gwa   2017  1.16   1.19
## # i 770 more rows
```

The data frame `dd3` has the total water usage for each crop, source, and year, for all census tracts in the US, and `dd4` is the average annual water usage for each crop and source. For example, for `barley` and `gwa`, the average annual water usage in the US is 1.19 kmˆ3. These numbers match the table in the article, except that the surface water numbers are slightly off for some reason.

I'll leave it to you to compute the change and percent change. Hint: check out `pivot_wider`, the inverse of our friend `pivot_longer`.

The two means we computed in class are not equal

I wanted to return to the (incorrect) code we were working on in class, where many people thought that two different ways of computing the mean that we tried should have been equal. Note that `dd2`, the mean across years of the values in `dd1`, is **not** exactly the same as simply finding the mean of the original data frame with all location-year combinations:

```r
dd5 = d %>%
  group_by(crop, src) %>%
  summarise(value = mean(value)) %>%
  ungroup()
```

```
## `summarise()` has grouped output by 'crop'. You can override using the
## `.groups` argument.
```

```r
dd5
```

```
## # A tibble: 60 x 3
##    crop   src       value
##    <chr>  <chr>     <dbl>
##  1 barley gwa   0.000372
```

```
##  2 barley gwd   0.000222
##  3 barley sw    0.000688
##  4 corn   gwa   0.00176
##  5 corn   gwd   0.00110
##  6 corn   sw    0.00172
##  7 cotton gwa   0.000623
##  8 cotton gwd   0.000442
##  9 cotton sw    0.000517
## 10 millet gwa   0.0000231
## # i 50 more rows
```

```
dd5 %>%
  left_join(dd2,
            by = c('crop', 'src'),
            suffix = c('.dd5', '.dd2')) %>%
  mutate(diff = value.dd5 - value.dd2) ## difference is not 0
```

```
## # A tibble: 60 x 5
##    crop   src   value.dd5 value.dd2          diff
##    <chr>  <chr>     <dbl>     <dbl>         <dbl>
##  1 barley gwa   0.000372  0.000372  -0.000000118
##  2 barley gwd   0.000222  0.000222  -0.000000293
##  3 barley sw    0.000688  0.000688  -0.000000603
##  4 corn   gwa   0.00176   0.00176    0.00000175
##  5 corn   gwd   0.00110   0.00110    0.00000133
##  6 corn   sw    0.00172   0.00171    0.00000117
##  7 cotton gwa   0.000623  0.000623  -0.000000325
##  8 cotton gwd   0.000442  0.000442   0.000000299
##  9 cotton sw    0.000517  0.000517   0.0000000906
## 10 millet gwa   0.0000231 0.0000231  0.0000000173
## # i 50 more rows
```

The differences are small but not zero. To see why this is, let's simplify let's look at just `barley` and `gwa` and find the yearly averages. Let's also include a column showing the number of locations.

```
dd6 = d %>%
  filter(crop == 'barley',
         src  == 'gwa') %>%
  group_by(crop, src, year) %>%
  summarise(mean  = mean(value),
            count = n(),
            sum   = sum(value))
```

```
## `summarise()` has grouped output by 'crop', 'src'. You can override using the
## `.groups` argument.
```

```
dd6
```

```
## # A tibble: 13 x 6
## # Groups:   crop, src [1]
##    crop   src   year      mean count   sum
##    <chr>  <chr> <chr>    <dbl> <int> <dbl>
```

```
##  1 barley gwa   2008  0.000376  3222 1.21
##  2 barley gwa   2009  0.000368  3222 1.19
##  3 barley gwa   2010  0.000357  3108 1.11
##  4 barley gwa   2011  0.000476  3222 1.53
##  5 barley gwa   2012  0.000455  3222 1.46
##  6 barley gwa   2013  0.000395  3222 1.27
##  7 barley gwa   2014  0.000266  3222 0.857
##  8 barley gwa   2015  0.000428  3108 1.33
##  9 barley gwa   2016  0.000347  3223 1.12
## 10 barley gwa   2017  0.000359  3223 1.16
## 11 barley gwa   2018  0.000334  3223 1.08
## 12 barley gwa   2019  0.000277  3223 0.892
## 13 barley gwa   2020  0.000396  3223 1.28
```

Note that the count column is not the same for every year. Uh oh. So averaging over locations first, and then averaging over years, is not the same as the average over all location-year combinations:

```
mean(dd6$mean) ## this weights each year the same
```

```
## [1] 0.0003718526
```

```
dd5 %>%
  filter(crop == 'barley',
         src  == 'gwa') %>%
  select(value) %>%
  as.numeric()
```

```
## [1] 0.0003717348
```

We would have to use a weighted mean to get the same result

```
weighted.mean(dd6$mean, w = dd6$count) ## this weights each location-year the same
```

```
## [1] 0.0003717348
```

```
sum(dd6$sum)/sum(dd6$count) ## same as above
```

```
## [1] 0.0003717348
```

Let's use mathematical notation. We'll again limit ourselves to `barley` and `gwa` for simplicity. Let x_{jk} be the amount of groundwater `gwa` used by `barley` at location j in year k. Let n_k be the number of locations for which there is data in year k. Note that there are 13 years.

```
length(unique(d$year ))
```

```
## [1] 13
```

Then the yearly average water usage in year k is

$$\frac{1}{n_k}\sum_{j=1}^{n_k} x_{jk}$$

The average of that across years is

$$\frac{1}{13}\sum_{k=2008}^{2020}\frac{1}{n_k}\sum_{j=1}^{n_k} x_{jk}$$

Since n_k depends on k, we can't pull it out of the sum.