Lecture 13: Data wrangling

So far

- `select`: choose certain columns
- `filter`: choose certain rows
- `summarize`: calculate summary statistics
- `group_by`: group rows together
- `mutate`: create new columns
- `count`: count the number of rows
- `arrange`: re-order the rows

Do dogs help exam stress?

- Data collected on 284 students at a mid-size Canadian university
- Students randomly assigned to one of three treatment groups: handler-only contact, indirect contact, and direct contact
- Well-being and ill-being measures recorded before and after treatment for each student
- Approach: compare pre/post measures of well-being and ill-being

Recording well-being and ill-being measures

- Likert items for each well-being / ill-being measure
- Average the likert items to get a score for each measure
- E.g.:
  - Positive affect score is the average of 5 Likert items
  - Social connectedness is the average of 20 Likert items

Example Likert item for social connectedness

“I am able to relate to my peers.”

- Strongly disagree (1)
- Disagree
- Somewhat disagree
- Somewhat agree
- Agree
- Strongly agree (6)

The raw data

- 284 rows (one per student)
- 200+ columns

```r
raw_data |>
  select(starts_with("SC"))
```

```
    SC1_1 SC1_2 SC1_3 SC1_4 SC1_5 SC1_6 SC1_7 SC1_8 SC1_9 SC1_10 SC1_11
SC1_12
1       5     4     5     5     5     2     1     6     5      6      5
2
2       5     6     2     4     6     3     1     5     1      4      2
6
3       3     4     2     3     4     4     3     3     2      5      3
5
4       4     6     2     5     5     2     2     4     4      5      5
5
5       1     5     2     4     3     2     5     2     5      4      4
5
6       3     6     4     5     4     1     3     5     2      5      2
4
```

Our goal for today

- Calculate the pre- and post-treatment social connectedness scores for each participant
- **Question:** What do we want the final data to look like?

Initial data processing

- Social connectedness is the average of 20 Likert items
- These items should take values between 1 and 6
- However:

```r
raw_data |>
  select(starts_with("SC")) |>
  max(na.rm=T)
```

```
[1] 66
```

Handling errors

```r
cleaned_data <- raw_data |>
  mutate(SC1_1 = ifelse(SC1_1 > 6, NA, SC1_1),
         SC1_2 = ifelse(SC1_2 > 6, NA, SC1_2),
         ...)
```

Are there any issues with this approach?

across

```r
example_df
```

```
  x1 x2 x3 y1 y2
1  5  7  8  5 NA
2  2  4  4  2  4
3  4  8  7  5  5
```

```r
example_df |>
  summarize(across(c(x1, x2, x3, y1, y2), mean))
```

```
        x1       x2       x3 y1 y2
1 3.666667 6.333333 6.333333  4 NA
```

Question: What if I want to ignore NAs when computing the mean?

across

```r
example_df
```

```
  x1 x2 x3 y1 y2
1  5  7  8  5 NA
2  2  4  4  2  4
3  4  8  7  5  5
```

```r
example_df |>
  summarize(across(c(x1, x2, x3, y1, y2),
                   function(x) {mean(x, na.rm=T)} ))
```

```
        x1       x2       x3 y1  y2
1 3.666667 6.333333 6.333333  4 4.5
```

across

```r
example_df
```

```
  x1 x2 x3 y1 y2
1  5  7  8  5 NA
2  2  4  4  2  4
3  4  8  7  5  5
```

```r
example_df |>
  mutate(across(c(x1, x2, x3, y1, y2),
                function(x) {x + 1} ))
```

```
  x1 x2 x3 y1 y2
1  6  8  9  6 NA
2  3  5  5  3  5
3  5  9  8  6  6
```

Question: What if I want to replace values > 6 with NA?

across

```r
example_df
```

```
  x1 x2 x3 y1 y2
1  5  7  8  5 NA
2  2  4  4  2  4
3  4  8  7  5  5
```

```r
example_df |>
  mutate(across(c(x1, x2, x3, y1, y2),
                function(x) {ifelse(x > 6, NA, x)} ))
```

```
  x1 x2 x3 y1 y2
1  5 NA NA  5 NA
2  2  4  4  2  4
3  4 NA NA  5  5
```

Handling errors

```r
cleaned_data <- raw_data |>
  mutate(across(c(SC1_1, SC1_2, ... ),
                function(x) {ifelse(x > 6, NA, x)}))
```

Question: Are there any issues with this approach?

Handling errors

```r
cleaned_data <- raw_data |>
  mutate(across(starts_with("SC"),
                function(x) {ifelse(x > 6, NA, x)}))

cleaned_data |>
  select(starts_with("SC")) |>
  max(na.rm=T)
```

```
[1] 6
```

More data cleaning

- For some Social Connectedness items, “6” means “more connected”
  - e.g.: “I find myself actively involved in people’s lives.”
- For some Social Connectedness items, “6” means “less connected”
  - e.g.: “I feel like an outsider.”
- We want higher scores to always mean “more connected”

We need to reverse the scores for some Social Connectedness items!

More data cleaning

```r
example_df |>
  mutate(across(c(x1, x2, x3, y1, y2),
                function(x) {ifelse(x > 6, NA, x)} ))
```

```
  x1 x2 x3 y1 y2
1  5 NA NA  5 NA
2  2  4  4  2  4
3  4 NA NA  5  5
```

Suppose we want to reverse the scores for `x1` and `x3`

```r
example_df |>
  mutate(across(c(x1, x2, x3, y1, y2),
                function(x) {ifelse(x > 6, NA, x)} )) |>
  select(num_range("x", c(1, 3)))
```

```
  x1 x3
1  5 NA
2  2  4
3  4 NA
```

More data cleaning

```r
example_df |>
  mutate(across(c(x1, x2, x3, y1, y2),
                function(x) {ifelse(x > 6, NA, x)} ))
```

```
  x1 x2 x3 y1 y2
1  5 NA NA  5 NA
2  2  4  4  2  4
3  4 NA NA  5  5
```

Suppose we want to reverse the scores for `x1` and `x3`

```r
example_df |>
  mutate(across(c(x1, x2, x3, y1, y2),
                function(x) {ifelse(x > 6, NA, x)} ),
         across(num_range("x", c(1, 3)),
                function(x) {7 - x}))
```

```
  x1 x2 x3 y1 y2
1  2 NA NA  5 NA
2  5  4  3  2  4
3  3 NA NA  5  5
```

With the dog data

```r
cleaned_data <- raw_data |>
  mutate(across(starts_with("SC"),
                function(x) {ifelse(x > 6, NA, x)}),
         across(num_range("SC1_",
                          c(3, 6, 7, 9, 11, 13, 15, 17, 18, 20)),
                function(x) {7 - x}),
         across(num_range("SC2_",
                          c(3, 6, 7, 9, 11, 13, 15, 17, 18, 20)),
                function(x) {7 - x}))
```

Averaging columns

```r
example_df |>
  mutate(across(c(x1, x2, x3, y1, y2),
                function(x) {ifelse(x > 6, NA, x)}),
         across(num_range("x", c(1, 3)),
                function(x) {7 - x}))
```

```
  x1 x2 x3 y1 y2
1  2 NA NA  5 NA
2  5  4  3  2  4
3  3 NA NA  5  5
```

Question: What if I want to calculate the average of the `x` columns for each row?

Averaging columns

```r
example_df |>
  mutate(across(c(x1, x2, x3, y1, y2),
                function(x) {ifelse(x > 6, NA, x)} ))
```

```
  x1 x2 x3 y1 y2
1  5 NA NA  5 NA
2  2  4  4  2  4
3  4 NA NA  5  5
```

```r
example_df |>
  mutate(across(c(x1, x2, x3, y1, y2),
                function(x) {ifelse(x > 6, NA, x)} ),
         x_mean = (x1 + x2 + x3)/3)
```

```
  x1 x2 x3 y1 y2   x_mean
1  5 NA NA  5 NA       NA
2  2  4  4  2  4 3.333333
3  4 NA NA  5  5       NA
```

Averaging columns

```r
example_df |>
  mutate(across(c(x1, x2, x3, y1, y2),
                       function(x) {ifelse(x > 6, NA, x)} ))
```

```
  x1 x2 x3 y1 y2
1  5 NA NA  5 NA
2  2  4  4  2  4
3  4 NA NA  5  5
```

```r
example_df |>
  mutate(across(c(x1, x2, x3, y1, y2),
                       function(x) {ifelse(x > 6, NA, x)} ),
                x_mean = mean(c(x1, x2, x3), na.rm=T))
```

```
  x1 x2 x3 y1 y2 x_mean
1  5 NA NA  5 NA    3.8
2  2  4  4  2  4    3.8
3  4 NA NA  5  5    3.8
```

Averaging columns

```r
example_df |>
  mutate(across(c(x1, x2, x3, y1, y2),
                function(x) {ifelse(x > 6, NA, x)} ))
```

```
  x1 x2 x3 y1 y2
1  5 NA NA  5 NA
2  2  4  4  2  4
3  4 NA NA  5  5
```

```r
example_df |>
  mutate(across(c(x1, x2, x3, y1, y2),
                function(x) {ifelse(x > 6, NA, x)} )) |>
  rowwise() |>
  mutate(x_mean = mean(c(x1, x2, x3), na.rm=T))
```

```
# A tibble: 3 × 6
# Rowwise:
     x1    x2    x3    y1    y2 x_mean
  <int> <int> <int> <int> <int>  <dbl>
1     5    NA    NA     5    NA   5
2     2     4     4     2     4   3.33
3     4    NA    NA     5     5   4
```

Averaging columns

```r
example_df |>
  mutate(across(c(x1, x2, x3, y1, y2),
                function(x) {ifelse(x > 6, NA, x)} ))
```

```
  x1 x2 x3 y1 y2
1  5 NA NA  5 NA
2  2  4  4  2  4
3  4 NA NA  5  5
```

```r
example_df |>
  mutate(across(c(x1, x2, x3, y1, y2),
                function(x) {ifelse(x > 6, NA, x)} )) |>
  rowwise() |>
  mutate(x_mean = mean(c_across(starts_with("x")), na.rm=T))
```

```
# A tibble: 3 × 6
# Rowwise:
     x1    x2    x3    y1    y2 x_mean
  <int> <int> <int> <int> <int>  <dbl>
1     5    NA    NA     5    NA   5   
2     2     4     4     2     4   3.33
3     4    NA    NA     5     5   4   
```

With the dog data

```r
cleaned_data <- raw_data |>
  mutate(across(starts_with("SC"),
                function(x) {ifelse(x > 6, NA, x)}),
         across(num_range("SC1_",
                          c(3, 6, 7, 9, 11, 13, 15, 17, 18, 20)),
                function(x) {7 - x}),
         across(num_range("SC2_",
                          c(3, 6, 7, 9, 11, 13, 15, 17, 18, 20)),
                function(x) {7 - x})) |>
  rowwise() |>
  mutate(sc_pre = mean(c_across(starts_with("SC1_")), na.rm=T),
         sc_post = mean(c_across(starts_with("SC2_")), na.rm=T))
```

```r
cleaned_data |>
  select(sc_pre, sc_post)
```

```
# A tibble: 284 × 2
# Rowwise:
   sc_pre sc_post
    <dbl>   <dbl>
 1   3.9     3.8
 2   5.15    5.26
 3   4.1     4.15
 4   4.65    5.1
 5   3.65    3.6
 6   4.35    4.65
```

7	4.75	4.4
8	4.6	4.65
9	4.2	4.15
10	5.8	5.75

Class activity

https://sta279-f23.github.io/class_activities/ca_lecture_13.html