Exercise: Joining, Correlating, and Visualizing

Table of Contents

[1 Class Practice 1](#_Toc162529241)

[2 Loading Libraries 1](#_Toc162529242)

[3 Reading Data 1](#_Toc162529243)

[4 Joining Data Frames 1](#_Toc162529244)

[5 (Optional): Pivoting to Wide based on Wave 3](#_Toc162529245)

[6 Summarizing a Variable (or Variables) by Group 3](#_Toc162529246)

[7 Correlating Variables 3](#_Toc162529247)

[8 Exploring Variables 4](#_Toc162529248)

# 1 Class Practice

Your project involves joining, summarizing, modeling, and visualizing data. You will need to join components created by team members in order to create data summaries and visualizations. The exercise provides practice doing just that and moving your toward those goals.

# 2 Loading Libraries

Load libraries needed. Add them as needed.

# 3 Reading Data

1. Read some of your project task data that has been cleaned as well as the demographic data that has been created by your teammate.
2. Verify that the files contain a unique identifier for joining the data frames.

# 4 Joining Data Frames

1. Join the two data frames into a single data frame.
2. Review your joined data frame to see that it looks like you expect it to look.
3. Examine your data frame for column variables with a .x or .y. Review such variable for errors. If you have them, consider adding suffix = c("", "")) to your join function. This will ensure that no *suffixes* should be added to the variable names in the returned data frame.
4. If you have multiple rows in one data frame (e.g., long format) but not the other data frame, you may receive a warning message like:

 Detected an unexpected many-to-many relationship between `x` and `y`.
ℹ Row 10 of `x` matches multiple rows in `y`.
ℹ Row 1 of `y` matches multiple rows in `x`.
ℹ If a many-to-many relationship is expected, set `relationship = "many-to-many"` to silence this warning.

If, and only if, your data frame looks appropriate for your goal, silence the warning by adding the recommended (in the warning message) parameter and argument to your join function. You won’t want to see this message when you build your files.

1. Once you are certain that your new data frame represents a combined data frame, assign it to an object name like JOINED or something else meaningful. If you are matching data with a long format (in any data frame), consider modifying the name to JOINED\_LONG so you can understand the file contents by the file name.
2. Options and Team Choices.
* *Script files*. There will be several tasks that involve using combined data frames (e.g., data summaries, models, visualizations) so your team should consider how best to address related issues. One recommendation is to create something like a "join\_data.R" or "merge\_data.R" script file performs all of your data frame joining operations needed for your project after the sub-components of the project are cleaned appropriately. Because certain project tasks like summarizing data, modeling data, and visualizing data will utilize the joined data, you could source this new script in the summarizing, modeling, and visualizing scripts before performing other operations. Moreover, depending on your goals, you may need to pivot data after joining or join data that have been pivoted to either long or wide. For example, in order to correlate data, the variables that you wish to correlate would need to be represented as **columns** (e.g., a single task data from different waves will need to be in separate columns). In such an instance, you could pivot your sub-task data by wave so that all participants are on one row and then join that data frame with others. Similarly, a regression model that uses data at time 1 to predict time 2 data will require a similar *wide* file structure. Visualizing data, however, will best be structured in a *long* format.
* *Data files*. Similarly, to facilitate access to the joined data, you may also wish to save the joined data file(s). In such instances, you might wish to save the joined data in a sub-directory of your /data directory, for example, /data/joined or /data/merged. If you decide to do this, you should also read in the joined data files after sourcing the code that produced them. In some instances you might want to create different joined files but the main goal would likely be to combine data across components of the project.
* Your team should decide how best to handle joined data so that everyone knows whether there are data files and if so, where they would be located. You would also want to make decisions that would be clear to the project liaison.

# 5 (Optional): Pivoting to Wide based on Wave

1. In order to correlate measures, you will need to have all variables you wish to correlate represented as columns. Use pivot\_wider() to pivot your data frame by wave.
2. Join this *wide* version of your file with a team members data frame.

# 6 Summarizing a Variable (or Variables) by Group

1. Using wave 1 data, create a data summary using summarize() to present the means, standard errors, upper and lower limit 95% confidence-interval thresholds, and sample size associate with a measurement/outcome variable of your choice. To assist with the standard error, confidence intervals, *n*, use the list provided below. Include a grouping variable in your summary.

summarize\_across\_list <- list(
 mean = ~mean(na.omit(.x)),
 se = ~sd(na.omit(.x)) / sqrt(length(na.omit(.x))),
 ci.95l = ~ifelse(length(na.omit(.x)) > 1, t.test(.x, conf.level = .95)$conf.int[1], NA),
 ci.95u = ~ifelse(length(na.omit(.x)), t.test(.x, conf.level = .95)$conf.int[2], NA),
 n = ~length(na.omit(.x))
 )

1. After you are sure that your summary table looks correct, pipe the data frame to gt() from the **{gt}** library (used in modules) to present the summary in a simple table.
2. To add a title to the table, pipe your data frame to tab\_header() from the **{gt}** library.

Example: tab\_header(title = "My Clear Title")

1. In preparation for your liaison meeting, you can repeat the process for another variable that you would want to present.

# 7 Correlating Variables

1. Create a Pearson’s *r* correlation table of numeric variables that are also grouped by some grouping variable. Use the gt() function from module to present the correlations in a simple table.
2. Extend the correlation table by adding a grouping variable that is relevant to the liaison’s proposal. Use the gt() function from module to present the correlations in a simple table.

# 8 Exploring Variables

1. Create a generalized pairs plot to explore the data and bivariate relationships among a set of numeric variables. Consider variable relationships that would be relevant for your liaison’s proposal.
2. Extend the generalized pairs plot by adding a grouping variable that is relevant to the liaison’s proposal.