The Power of Efficiency

Caleb Jensen

Efficiency is key in statistical computing. It means using few, well written, understandable lines of code to accomplish a task. So for instance, if we are applying multiple filters we could create a second data set and apply a filtering join. This uses far fewer lines of code than applying many filters, is more understandable and very easy to write. Another example would be to use an across statement to iterate a single function through many columns or variables. Using these simple statements in our code can make it much easier to write, read, and faster for the computer to evaluate. We look at efficiency frequently in this class, learning to apply filters in one line or filter statement, learning to use filtering joins, sometimes with multiple variables to match to make filtering faster and easier for future readers to understand or replicate. I have had a few "a-ha" moments this quarter going through this course. One was using the across() function. In the lab 3 challenge I wrote about eight statements to apply one function across several variables but after revisions I was able to accomplish the same thing with a single across statement. My code was much less cluttered, it was easier to read and understand and the output was just as good. Another "a-ha" moment was the use of pipelines to create exactly the output I want step by step through the data. Rather than applying multiple changes to the data set and saving each one I can make step by step transformations to the data to end at exactly the end product I want. I can save the changes or just look at the output to answer questions or gain insight into the data. I really liked the ability to use slice_min(), slice_max(), or arrange() filter down the data at the final step to exactly what I want, making viewing the final output simple and understandable. Finally, the ability to use if_else() and case_when() to transform data to exactly the type or value of a variable was a very important discovery for me. I used this especially before I was comfortable with string manipulation. I used this in lab 4 to manipulate city names to be in the form I want. Now I know that could have been accomplished more efficiently with stringr functions, but there are many other application of case_when() that would still be efficient.