The Power of Efficiency

Efficiency means completing a task without spending effort on unnecessary or extraneous tasks. With regards to data science, this permeates every level of a project from organizational efficiency (high level) to coding efficiency (low level). Organizational efficiency at the highest level makes file navigation easy. I spent a good bit of the essay "Growth as a Data Scientist" on the topic of file organization, so I don't want to spend too much time on it here as well, but it is worth noting that the time saved by well-organized files makes them inherently efficient for the user. It also makes file sharing efficient because time is neither wasted by the sender in processing their own files nor by the receiver because they do not have to spend time trying to determine the location of the files they need. At a lower level, efficiency inside a coding document requires a consistent naming convention and simple code chunks. Scripts become much easier to work with when it is easy to understand what each portion of the script is accomplishing. At a coding level, efficient code runs faster on a computer and uses less memory space because it does not force the computer to perform tasks which are not necessary for the analysis.

Efficiency is important because it saves time and opens the door to more complex tasks. Well organized and written code can be shared more commonly and updated easily to perform other tasks. It can also be run on computers which might not be able to perform less well-written analyses, which means efficient code allows the writer to perform more task on their computer than they might be able to do if their code was less efficient. From a timing perspective, if a script takes a long time to run, it might not be possible to run multiple times in a day. A script which accomplishes the same task but runs in only a few minutes, on the other hand, could be run multiple times in an hour.

I encountered efficiency at almost every level of this course from organizing our file structures and code chunks to using thoughtfully written functions. My first “a-ha” moment was the use of the “here” package, which can be found in any artifact in this portfolio (lab 8, for example). I had not realized the importance of a working directory until we began using that package. My next “a-ha” moment came in Lab 4_Revised when we used the anti_join function to filter extraneous information out of the dataset we were using. I did not realize how much easier it was to work with reduced datasets until we began regularly simplifying the datasets we imported before beginning our analysis. Finally, the last “a-ha” moment I will discuss came in my revision of lab 4 (Lab 4_Revised) when I updated my code to use one function to do the work of several lines I wrote on the first attempt. This made my code much more readable, and also probably made it run faster, which made me realize hoe beneficial it can be to use the right functions to do the right jobs.