This experiment was an interesting experiment in many ways, from how copilot writes or suggests codes to maintaining and writing code to github. Some of the more challenging parts were identifying issues to create and metrics to measure. This is due to many reasons, and different for each challenge. For metrics, time is an obvious choice but outside of that, I was not sure. Memory was a bit hard due to the size of the data and can be inaccurate because of how garbage collection can work. From what I assume, Python has some level of memory management and the size of the data did not make any difference in it. Time, on the other hand, was easier to manipulate due to the usage of for-loops in the process.

Speaking of for-loops, copilot seems to suggest vectorized code so it seems any suggestion for code creation creates, what seems to be, an efficient part. I find copilot to be helpful, especially in creating the issue.md where it lists potential issues that I had forgotten about, like ensuring file exist and size limits to prevent growing file size, etc.

I will say I did not learn much from debugging since most solutions for debugging issues are something that I had previously done. In my Ai course, we had to clean data for the model, and I would say that it is due to this that most of the issue was solved as an unconscious thought, like removing or filling in null values in the dataframe. The Optimization on the other hand was more insightful, even when it went from 0.01 to 0.001, but getting the differences to show was harder then I thought since, it seems like, the data, or process, was just too small and did not cost much processing power.