K Walter Leiker

kwleiker@gmail.com

913-954-2380

I struggled with this assignment but by the end, I feel I had given the competency need to begin the assignment.

# The Bad

1. Iteration. I could not figure out how to get it to run iteratively, so I had it do batches.
2. Exceptions. I didn’t know how to handle exceptions at all – so when a batch hit a bad call, it failed. Over and over again.
3. File Shape. Obviously, running batches and having failures, the file and look in the wrong shape and I had difficulty reshaping the file.
4. Deletions. I somehow deleted outputs while appending, although results and just re-ran it to re-create it although that went a lot quicker.
5. Graphing. Although doing the output, the charts, isn’t hard at all, I was so in efficient with getting the data, I ran out of time and I didn’t graph. I took it to work hoping I’d be able to do things there, but I wasn’t able to work on my machine whatsoever – other than trying to turn it in.

# The Ugly

1. Messiness. I just had to keep repeating things and I got pages and pages of errors, and I never really cleaned it up.
2. When I was at work, even using the public Wi-Fi network, I couldn’t access GitHub. It was blocked. I called tech-support and she explained how to get around blocks on streaming music, downloading videos, accessing boardgame geek – things I don’t do it work at all. So, I ended up using my phone as a personal Wi-Fi hotspot. That wasn’t awesome because cell phones don’t work in our building – we’re a lockdown unit and our architecture blocks signals: iPhone is zero bars most places, Android is usually one bar. To make a phone call out, I have to stand by the window in the vestibule. I set my phone by that window in the vestibule and went back into my office and read my computer off the Wi-Fi Hotspot, and it was very very very very slow.

# The Good

1. Call List. When I had trouble calling cities and with file shape, I assumed it had to do with dups on the city names. I thought it would be better to pre-handle that, by matching to city# and then querying those. Thus I built a list of all 200k+ places, dropped the duplicates, then matched the returned list of cities to that so that I could run the queries using city\_number instead of city\_name. No effect.
2. Sampling! We’re trying to look at variation in temperatures over geographic space but are doing that indirectly: the sampling used is looking at variation in temperature across population-space, with pop-space distributed across geographic space. More so, we are looking at variation across N-S space, so essentially we’re after variation across (absolute) distance from the Equator. I started playing with this before even looking at the assignment.

I did a grid/net sampling procedure. I divided the globe into 24 E-W zones, each 15° wide (~time zones); and then divided into 18 N-S bands, each 10° high. I deprecated some of these 432 zones: there are only a handful of stations south of 55°S or north of 73°N. Each case in the dataframe was assigned to one of these zones. In each pass, an RNG pair was generated inside each of these zones. I then called a prox.min to select a station. Areas that were void (Antarctic, mid-Ocean) got dropped or catted.

The advantage is that some locations in sparsely inhabited areas are called more frequently than if it were on a population or populated-place basis. More important, since I was using METAR and NOAA navigation data, I pulled stations at unpopulated spaces (e.g. Buoy 62001, Buoy 46006, Toa Ridge). Alas, I didn’t have the skills to make me way through these data – I was only selecting locations, not the actual data. For the actual data retrieval, I used UWM. Just like everyone else.

I found a better way, but I didn’t implement it.