Final Project

Sam Loyd

January 2020

This project on data wrangling focuses on using an API with a wrapper program to collect data from NASA on exoplanets. I selected the Nasapy wrapper library for its simplicity and usability. Nasapy was written by Aaron Schlegel (2020). I had to register for a login with NASA from api.nasa.gov to gain access to the selected data set. Unlike the same process with twitter, this was accomplished rather quickly. I was provided with a key that I store locally in a config file on my laptop and use the config.read function to keep that information obfuscated from my code.

The first opportunity for improving the data set came from the titles. They were not descriptive. I found a code book on NASA's website that was helpful, but it needed a lot of work to address my problem. Several of the fields were collapsed into one entry so I decided to manually copy them into a CSV file with many manual modifications that I could later load into my Jupyter notebook and change the matching column headers. This also gave me a chance to show loading data from flat files that we covered.

The next challenge came from the join portion of the project. I wanted to cover that key concept from class. I decided to include a country of origin or location for each facility that discovered an exoplanet. I had to create another csv file listing each facility and manually searched google for that information. This data was loaded into Python and joined with the existing data extending its usefulness.

The original API call resulted in 82 columns with primarily numerical data. I decided to look at missingness as well. I used code modified from the text and verified with the missingno library. In order to preserve the original data, I loaded it without modification into a Postgres database housed locally. At the start, I loaded in 4,116 rows of data. Interestingly, over the time it took me to complete my project, NASA loaded in ten newly discovered exoplanets. I tested the Planet Name field for uniqueness and discovered that it was.

I then decided to look at the facility information and created a subset uniquely identifying those. I noticed that several seemed to be related so I used fuzzy logic to quickly match up observatories that were likely part of the same organization. I concluded with a quick EDA analysis using panda-profiling. Interestingly, this library was broken due to an update to the panda's package when I started my project. However, over the course of a few days, contributors to the package had corrected the issue and had it back in working order.

In conclusion, I used an API call with a wrapper program to pull data from NASA. I had to manually clean up codebook information and search for facilities on google. I merged this data with what I had pulled using the API. I performed several transformations and illustrated key learnings from this class. I may use this dataset for a future project as it is in much better shape than my initial pull from the API.

References:

Schlegel, A. (2020, January 20). Analyzing the Next Decade of Earth Close-Approaching Objects with nasapy. Retrieved from https://medium.com/@AaronSchlegel/analyzing-the-next-decade-of-earth-close-approaching-objects-with-nasapy-8a6194c4a493