Lab Report

Title: GIS 5571 – Lab 1
Notice: Dr. Bryan Runck
Author: Luke Zaruba
Date: October 5, 2022

Project Repository: https://github.com/lukezaruba/GIS5571/tree/main/Lab1
Time Spent: 6.0

Abstract

Data cleaning and preparation is one of the most vital components of (spatial) data science, because it ensures that datasets used in analyses are adequate and will provide us with useful results for whatever the purpose is. Extraction, transformation, and loading, or ETL, is the initial part of any data science pipeline where the data wrangling occurs. Arguably, ETL processes are some of the most important parts of any workflow since it affects all subsequent parts of the analysis. It is so important, that the entire field of data engineering revolves around creating and implementing ETL pipelines. In this project, geospatial ETL processes will be explored to allow for easy data wrangling, analysis, and visualization. Furthermore, the project will utilize a variety of different tools, ranging from open-source libraries like Pandas and GeoPandas, to commercial tools like the ArcGIS API for Python. This integration of different tools will allow for more flexibility in implementing ETL processes.

Problem Statement

Data ETL processes can become complex and messy, especially when dealing with large, poorly formatted datasets. When spatial data is brought into the picture, the process becomes even more complicated, due to the inherent difficulties of working with spatial data. The goal of this project is to perform basic spatial data ETL processes by extracting data via API requests, transforming the raw data into a more useable format, and then loading the data into platforms like ArcGIS Pro or ArcGIS Online to use for spatial analysis.

Table 1. Analysis Requirements for the Project

#	Requirement	Defined As	(Spatial) Data	Attribute Data	Dataset	Preparation
1	Raw data returned from APIs	Raw outputs from Google Places, MN Geospatial Commons, and NDAWN API requests	Point and polygon geometries	Various attributes	Google Places, MN Geospatial Commons, NDAWN	Parameterize the search, to return the desired results
2	Datasets converted to GDFs	Transformed datasets from various APIs, loaded into GeoPandas GeoDataFrame	Point and polygon geometries	Various attributes	Same data as above, reformatted	Converted JSON/CSV to GDF
3	Folium Maps	Loaded GDFs into Folium maps	Point and polygon geometries	Very little, other than for popups	Same data as above, visualized	Loaded GDFs into Folium maps

4	Feature Layers	GDFs converted to Feature Layers in ArcGIS Online	Polygon geometries	Very little	Same data as above, reformatted	Converted GDFs to SEDFs, then to FL

Input Data

In this project, ETL processes were built out for working with three unique data sources, the Google Places API, a REST Service from the Minnesota Geospatial Commons, and the NDAWN API. The input data was either formatted as a JSON (Google Places and MN Geospatial Commons) or as a CSV (NDAWN). Each API had to use a different workflow to accommodate for the different search parameters, output formats, and other transformation processes needed to clean the data. At the end of the process, data was all output in the standard format of a GeoPandas GeoDataFrame, a spatially enabled version of a Pandas DataFrame. This allows for easy integration and interoperability for performing spatial analyses, like a spatial join.

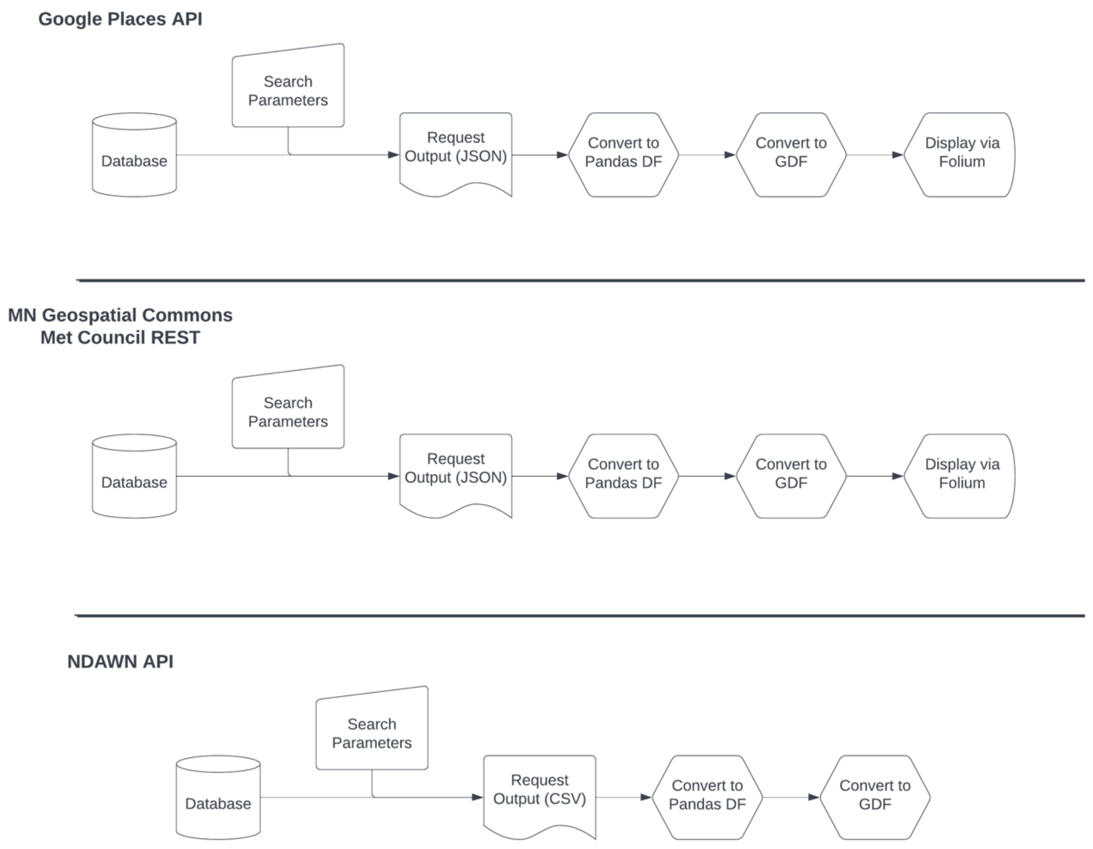
Table 2. Input Data Sources

#	**Title**	**Purpose in Analysis**	**Link to Source**
1	Google Places API	Global POI data from Google.	Google Places API
2	Met Council REST API	Boundaries dataset for city, townships, and unorganized territories in the Twin Cities.	MN Geospatial Commons
3	NDAWN API	Comprehensive weather data for North Dakota.	NDAWN

Methods

The general workflow for the different APIs was the same, however there were several differences based on output formats and geometry types. The workflow began with developing functions to take in desired search parameters and then request the data from the API and return the result as a JSON or CSV. After this, the JSON or CSV was read into a Pandas DataFrame, which could be done in many ways, depending on the input format. For example, the Google Places API JSON that was returned, was not an actual JSON, but rather a set of Python dictionaries and lists, therefore making it difficult to directly use Pandas' methods for reading JSONs, meaning that the data would either have to be converted to an actual JSON or it could be read into the DataFrame by iteratively searching through the lists and dictionaries and appending each value for a feature into the DataFrame, which is what I did. For the other two APIs, I simply used Pandas methods to read in the JSON and CSV that were returned by the APIs. With the data now in Pandas DataFrames, it was easy to convert the data to GeoPandas GeoDataFrames, with the simple use of a method. This then allows for the data to easily be visualized in Folium or analyzed through various Python packages, including GeoPandas itself. For this project, I spatially joined my output from the Google Places search, which was electric vehicle charging stations around Minneapolis, to my output from the Minnesota Geospatial Commons search, which was cities in the Twin Cities Metro Area. After the join, I then used the ArcGIS API for Python to convert the GeoDataFrame to an ArcGIS Spatially Enabled DataFrame (SEDF), which could then be converted to a Feature Layer in ArcGIS Online.

Figure 1. Data flow diagram.



Results

The result was a map containing the electric vehicle charging stations that were joined with the cities. The points included popups containing information about the charging station, along with the city that the station was in. The map can be accessed by going to the project's GitHub repository and opening the Jupyter Notebook within that repo. The map was created using Folium, a Python wrapper for Leaflet, a popular mapping library in JavaScript. Additionally, other results included the whole workflow itself, which could easily be implemented in future projects to process data more efficiently.

Results Verification

To verify the results of the analysis, I visually inspected the outputs of the DataFrames and the maps that were created to ensure that the geometries and attribute information were correctly extracted, transformed, and loaded. With the datasets being small, it was a pretty simple process to ensure that the results were correct and would be useful for any future analyses.

Discussion and Conclusion

Initially, it was difficult to develop a workflow without ever having done similar tasks before. As I continued to develop a better understanding of the requirements and needs of the workflow, it then became more apparent how I could go about doing it, which made it far easier, since I was able to use tools and packages that I was already familiar with, just in a different way and for different use-cases. After finishing up the first ETL process for the Google Places API, it also made it far easier to develop the other ETL processes since the general workflow is the same, with just a few minor changes being needed. I am confident that I could now replicate the process for other APIs in the future, if need be, which is a very valuable skill to have.

References

"API Reference." API reference - pandas 1.5.0 documentation. pandas via NumFOCUS, Inc., 2022. https://pandas.pydata.org/docs/reference/index.html.

"Documentation." Documentation - GeoPandas. GeoPandas Developers, 2022. https://geopandas.org/en/stable/docs.html.

"Folium." Folium - Folium 0.12.1 documentation. Rob Story, 2013. http://python-visualization.github.io/folium/.

"Introduction to the Spatially Enabled DataFrame." ArcGIS API for Python. Esri, 2022. https://developers.arcgis.com/python/guide/introduction-to-the-spatially-enabled-dataframe/.

Self-score

Category	Description	Points Possible	Score
Structural Elements	All elements of a lab report are included **(2 points each)**: Title, Notice: Dr. Bryan Runck, Author, Project Repository, Date, Abstract, Problem Statement, Input Data w/ tables, Methods w/ Data, Flow Diagrams, Results, Results Verification, Discussion and Conclusion, References in common format, Self-score	28	**28**
Clarity of Content	Each element above is executed at a professional level so that someone can understand the goal, data, methods, results, and their validity and implications in a 5 minute reading at a cursory-level, and in a 30 minute meeting at a deep level **(12 points)**. There is a clear connection from data to results to discussion and conclusion **(12 points)**.	24	**23**
Reproducibility	Results are completely reproducible by someone with basic GIS training. There is no ambiguity in data flow or rationale for data operations. Every step is documented and justified.	28	**27**
Verification	Results are correct in that they have been verified in comparison to some standard. The standard is clearly stated **(10 points)**, the method of comparison is clearly stated **(5 points)**, and the result of verification is clearly stated **(5 points)**.	20	**20**
		100	**98**