GEOG 4057 Final Project Report

Rorisang Kgoadi

ArcGIS is not easily able import JSON files to the ArcGIS interface using any of the provided tools in the toolboxes. The goal of this study was to provide a tool to convert the JSON file into a shapefile and visualize the output. To do so, I define a python script that will create the features class from given JSON data then use the script in a python toolbox to make an interactive tool.

Data for the study came from the 2018 Market Value Analysis for New Orleans, collected from data.gov as a JSON file. To create a shapefile from the JSON data, I used VSCode for script editing and ArcPy for data analysis and conversion.

The first step in the conversion process was writing a script to create an output shapefile. The Jupyter notebook ‘Project1” contains the code used to test creating a shapefile. The first section, titled ‘Explore the JSON file’, imports and reads in sections of the JSON data. The first block of code loads in the JSON file and shows some of the metadata within the file, while the next blocks look at the well-known text (WKT) within each of the columns of the file. Under the WKT, I change the geometry data in the 9th element (8th index) into a polygon object that will be used in later visualization.

In the next section (‘Create feature class and write fields’), I create the feature class and update the feature class field names. I do this by first creating a polygon feature class using arcpy management. Next, each of the field names for the feature class are defined with the appropriate data type and name. The field names are then added to the previously created feature class shapefile.

In the final section of code (‘Add data to feature class’), the data is defined for the feature class using a cursor to insert information for each of the fields. I also make sure to account for another set of geometry properties using the SHAPE@ token. Finally, the data from the JSON file is appended to the feature class using an insert cursor. The entire code is then checked over and run to ensure functionality.

After this Jupyter notebook is tested and confirmed working, parts of the code were taken and put into a python file (‘project1.py’) for later use in the python toolbox. I first defined a function called ‘importNoTaxJSON’ that runs all the essential lines of code from the Jupyter notebook to create the shapefile from the JSON data. Input arguments of the ‘workspace’, ‘json\_file’, and ‘output\_fc’ are defined in the function for later use in the code.

Once the python file has been defined, I then make the python toolbox that will be used with the previously defined python code. I first created a new python toolbox (‘project1\_json.pyt’) and changed the label of the tool to ‘Import No Tax JSON to Feature Class’ for easier identification. Then, I define the characteristics of the parameters to be input into the tool. These include the workspace name, json file, and output name. With the parameters defined, I then import the ‘importNoTaxJson’ function from the python script and use it under the execute section with definitions of input parameters. After the parameters for the toolbox are defined, the tool is ready to be used in the ArcGIS interface.

After the tool is used to create a feature class from the JSON file, the data is plotted on the map and made into a layout (‘Layout1.pdf’). The layout shows clusters of market values for different areas of New Orleans, organized by color.

The challenge with this type of work is it is only applicable to JSON files, so other formats that may need more work to be imported to ArcGIS do not have a simple way to do that. Future work would involve creating tools to make other file formats easier to import.

# Works Cited

data.nola.gov. (2023, June 17). *City of New Orleans - market value analysis 2018*. data.gov. https://catalog.data.gov/dataset/market-value-analysis-2018-990b2