Task 1 - Install the geopandas package.

Importing geopandas in Jupyter Notebook is impossible (it will show error) without installing it first through my new geodatabase environment.

Instead, I opened Anaconda command prompt and created a new environment called “geo_env” by using the command given below:

conda create -n geo_env

This new environment I will use for my geospatial packages.

Next, I will activate this environment “geo_env”:

conda activate geo_env

After that I installed Geopandas in the environment just created (geo_env):

conda install geopandas

Now I have verify that geopandas is installed by running python in my terminal and then running the following code (no

Loading [MathJax]/extensions/Safe.js

errors means geopandas is successfully installed):

import geopandas

The following step includes installation of Jupyter Notebook in my geo_env, as by default it is installed only in “base” environment. This step takes a while.

conda install jupyter notebook

In this step I will add my environment (geo_env) to Jupyter Notebook:

python -m ipykernel install –name geo_env

Lastly, I opened Anaconda Navigator, went to Environments section, selected my environment (geo_env). Then I clicked on “play” button and selected “Open with Jupyter Notebook”. In this way I will continue working in Jupyter Notebook within my newly created geospatial environment (geo_env).

Task 2 - Find at least three sets of spatial data, in Shapefile, zipped Shapefile, GeoJSON, or other supported formats. Import those data into GeoDataFrame using geopandas . Take one dataset as an example, show the specific types of the [illegible]ataframe, the geometry column,

Loading [MathJax]/extensions/Safe.js

and individual geometry object. Export all three into a SINGLE geopackage file (geodatabase). Note that the single geopackage file should have three layers inside, not three layers in three separate geopackage files.

In [94]:
```
import geopandas as gpd
import fiona
```

In [95]:
```
zipfile_1 = r"C:\Users\Toshiba\Downloads\SEN22_June_03_2022.zip"
data_1 = gpd.read_file(zipfile_1)
```

In [96]:
```
type(data_1)
```

Out[96]:
```
geopandas.geodataframe.GeoDataFrame
```

With the step above I am sure that my first dataset is a GeoDataFrame.

Now I will print results for each of my 3 zip files, by using the function head.

In [97]:
```
data_1.head
```

Loading [MathJax]/extensions/Safe.js

Out[97]:
```
<bound method NDFrame.head of     OBJECTID     Shape_Leng    Shape_Area  District
\
0          1  322853.258786  3.954679e+09         1
1          2  114388.392696  6.435845e+08         2
2          3  160278.163328  8.341962e+08         3
3          4   78436.523765  1.821229e+08         4
4          5  102079.122816  2.509553e+08         5
..       ...            ...           ...       ...
60        61   29518.515798  1.898123e+07        59
61        62  320334.149759  2.664090e+09        60
62        63  104927.049299  3.134754e+08        61
63        64  377675.712575  6.184955e+09        62
64        65   86918.711112  2.049354e+08        63

                                             geometry
0   POLYGON ((742169.526 4532898.369, 736334.250 4...
1   POLYGON ((655759.831 4532072.892, 655703.520 4...
2   POLYGON ((693595.108 4531758.179, 693510.421 4...
3   POLYGON ((656117.458 4521399.761, 656154.066 4...
4   POLYGON ((628387.043 4516930.000, 628425.993 4...
..                                                ...
60  POLYGON ((591480.854 4514514.411, 591297.081 4...
61  POLYGON ((204065.058 4777592.459, 204087.481 4...
62  POLYGON ((198610.415 4777088.980, 198633.906 4...
63  POLYGON ((248485.354 4835634.524, 254869.353 4...
64  POLYGON ((189409.711 4763292.965, 189408.169 4...

[65 rows x 5 columns]>
```

In [98]:
```
zipfile_2 = r"C:\Users\Toshiba\Downloads\CON22_June_03_2022.zip"
data_2 = gpd.read_file(zipfile_2)
```

In [99]:
```
data_2.head
```

Loading [MathJax]/extensions/Safe.js

Out[99]: <bound method NDFrame.head of OBJECTID Shape_Leng Shape_Area DISTRICT
\

	OBJECTID	Shape_Leng	Shape_Area	DISTRICT
0	1	3.982402e+05	4.747143e+09	1
1	2	2.076729e+05	1.482939e+09	2
2	3	1.468134e+05	6.453502e+08	3
3	4	1.004443e+05	4.891851e+08	4
4	5	1.130366e+05	2.913121e+08	5
5	6	6.043320e+04	6.716070e+07	6
6	7	5.535552e+04	5.791640e+07	7
7	8	7.345937e+04	1.158570e+08	8
8	9	3.511608e+04	3.924811e+07	9
9	10	9.533722e+02	5.740103e+04	10
10	11	7.132348e+02	1.893861e+04	10
11	12	4.488322e+04	6.057440e+07	10
12	13	8.055368e+04	2.962222e+08	11
13	14	3.312166e+04	3.514574e+07	12
14	15	4.217439e+04	3.770423e+07	13
15	16	6.898629e+04	1.219229e+08	14
16	17	5.277969e+04	5.163914e+07	15
17	18	1.014937e+05	4.066207e+08	16
18	19	2.778405e+05	2.341177e+09	17
19	20	4.717750e+05	5.308040e+09	18
20	21	9.971849e+05	2.067891e+10	19
21	22	3.718790e+05	4.168971e+09	20
22	23	1.473855e+06	4.435027e+10	21
23	24	4.668402e+05	7.162431e+09	22
24	25	8.336454e+05	1.825681e+10	23
25	26	1.288241e+06	2.369059e+10	24
26	27	3.442252e+05	5.131726e+09	25
27	28	1.835296e+05	1.241497e+09	26

	geometry
0	POLYGON ((742169.526 4532898.369, 736334.250 4...
1	POLYGON ((693595.108 4531758.179, 693510.421 4...
2	POLYGON ((632044.962 4535980.849, 632198.219 4...
3	POLYGON ((622128.271 4510545.013, 622090.862 4...
4	POLYGON ((603301.333 4509330.502, 603368.523 4...
5	POLYGON ((599159.460 4514856.890, 599250.363 4...
6	POLYGON ((589781.751 4513091.815, 589853.354 4...
7	POLYGON ((590360.641 4505374.163, 590429.692 4...
8	POLYGON ((588225.000 4503687.809, 588257.247 4...
9	POLYGON ((580767.765 4504622.017, 580749.465 4...
10	POLYGON ((581071.108 4505910.259, 581075.092 4...
11	POLYGON ((583816.308 4510754.392, 584056.059 4...
12	POLYGON ((580038.399 4500963.725, 581793.122 4...
13	POLYGON ((587070.226 4517312.298, 587063.824 4...
14	POLYGON ((592803.385 4526500.112, 592845.748 4...
15	POLYGON ((598983.979 4527230.350, 599034.739 4...
16	POLYGON ((593809.645 4529143.221, 593819.298 4...
17	POLYGON ((596971.676 4548919.423, 597034.229 4...
18	POLYGON ((600717.984 4609432.925, 600849.402 4...
19	POLYGON ((590207.336 4659091.173, 590273.940 4...
20	POLYGON ((426714.738 4737991.449, 426842.162 4...
21	POLYGON ((590414.922 4805623.861, 590431.122 4...
22	POLYGON ((629790.775 4982375.767, 629861.519 4...
23	POLYGON ((491737.397 4821550.774, 491786.215 4...

Loading [MathJax]/extensions/Safe.js

```
24  POLYGON ((204065.058 4777592.459, 204087.481 4...
25  POLYGON ((421116.565 4910906.118, 421233.350 4...
26  POLYGON ((248485.354 4835634.524, 254869.353 4...
27  POLYGON ((194740.732 4785240.287, 194737.334 4...  >
```

In [100…
```
zipfile_3 = r"C:\Users\Toshiba\Downloads\2022assembly_shape_file.zip"
data_3 = gpd.read_file(zipfile_3)
```

In [101…
```
data_3.head
```

Out[101]:
```
<bound method NDFrame.head of       ID                                                 g
eometry
0      1  POLYGON ((-72.74444 40.82515, -72.74454 40.825...
1      2  POLYGON ((-73.03845 40.93770, -73.03773 40.938...
2      3  POLYGON ((-72.93671 40.84202, -72.93681 40.841...
3      4  POLYGON ((-73.16294 40.97559, -73.16429 40.981...
4      5  POLYGON ((-73.13736 40.81197, -73.13747 40.812...
..   ...                                                ...
145  146  POLYGON ((-78.82384 42.95813, -78.82383 42.958...
146  147  POLYGON ((-77.95538 42.71371, -77.95537 42.713...
147  148  POLYGON ((-77.73681 42.28153, -77.73328 42.281...
148  149  POLYGON ((-79.13708 42.56992, -79.13787 42.570...
149  150  POLYGON ((-79.06126 41.99941, -79.06479 41.999...

[150 rows x 2 columns]>
```

Based on above presented results:

- zipfile_1 (SEN22_June_03_2022) has 65 rows and 5 columns
- zipfile_2 (CON22_June_03_2022) has 28 rows and 5 columns
- zipfile_3 (2022assembly_shape_file) has 150 rows and 2 columns

In [102…
```
print("Simple Print\n", zipfile_1)
print("\nType of the data is: {}".format(type(zipfile_1)))
```

```
Simple Print
 C:\Users\Toshiba\Downloads\SEN22_June_03_2022.zip

Type of the data is: <class 'str'>
```

Now I have to save three layers into one single geopackage file/geodatabase.

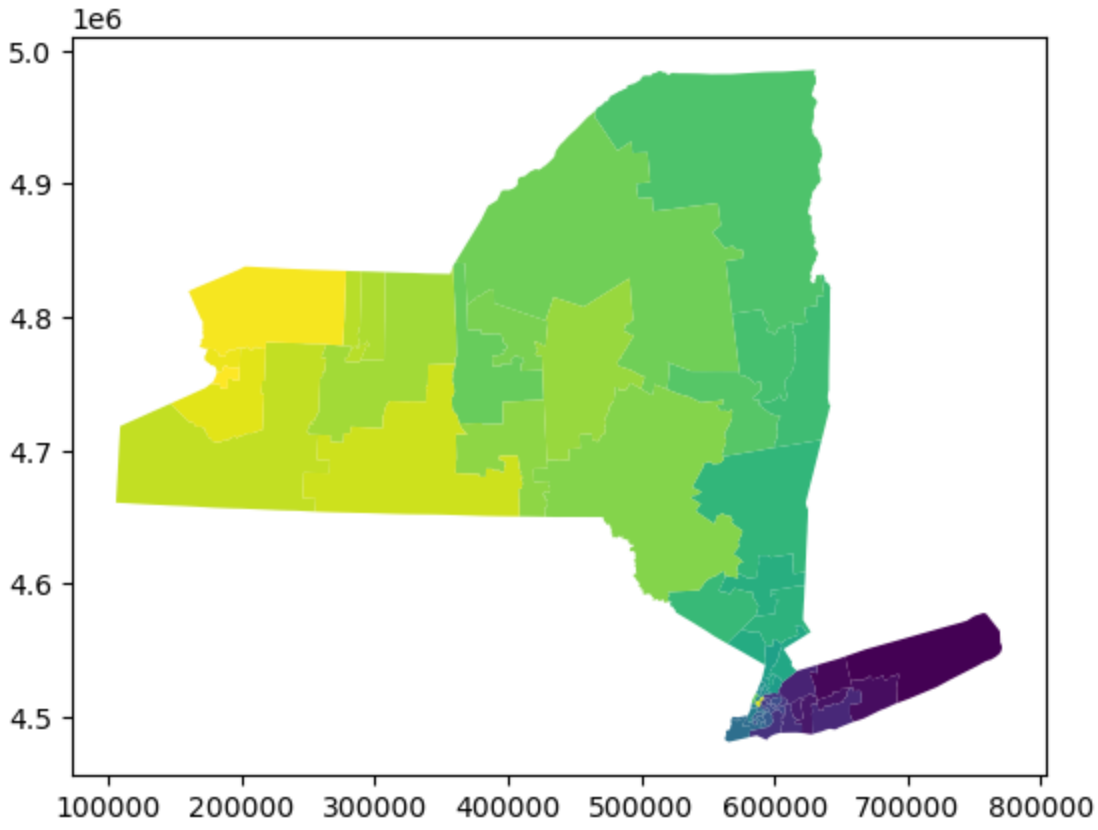
In [103…
```
data_1.to_file('file_1.gpkg', layer='newyork_1', driver='GPKG')
data_2.to_file('file_1.gpkg', layer='newyork_2', driver='GPKG')
data_3.to_file('file_1.gpkg', layer='newyork_3', driver='GPKG')
```

Loading [MathJax]/extensions/Safe.js

Task 3 - Use basic web mapping methods in geopandas to visualize the data.

In [104…
```
data_1.plot(column='OBJECTID')
```

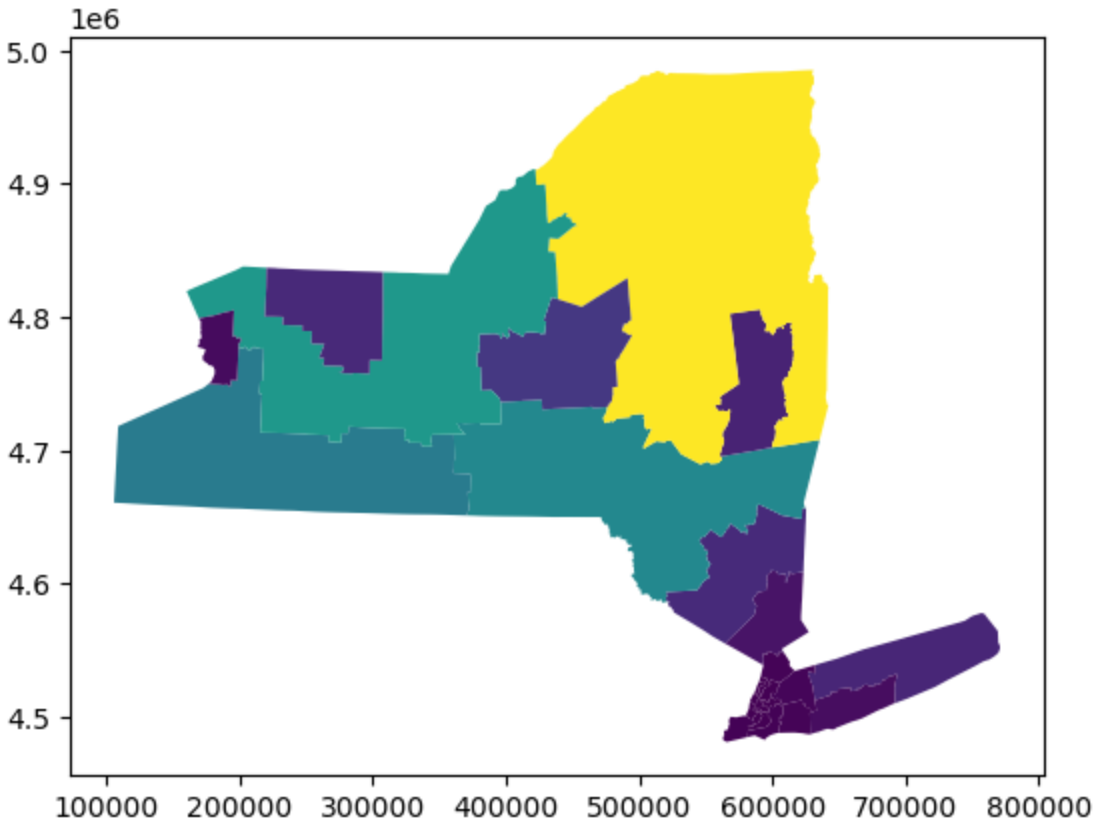
Out[104]: <Axes: >



In [105…
```
data_2.plot(column='Shape_Area')
```

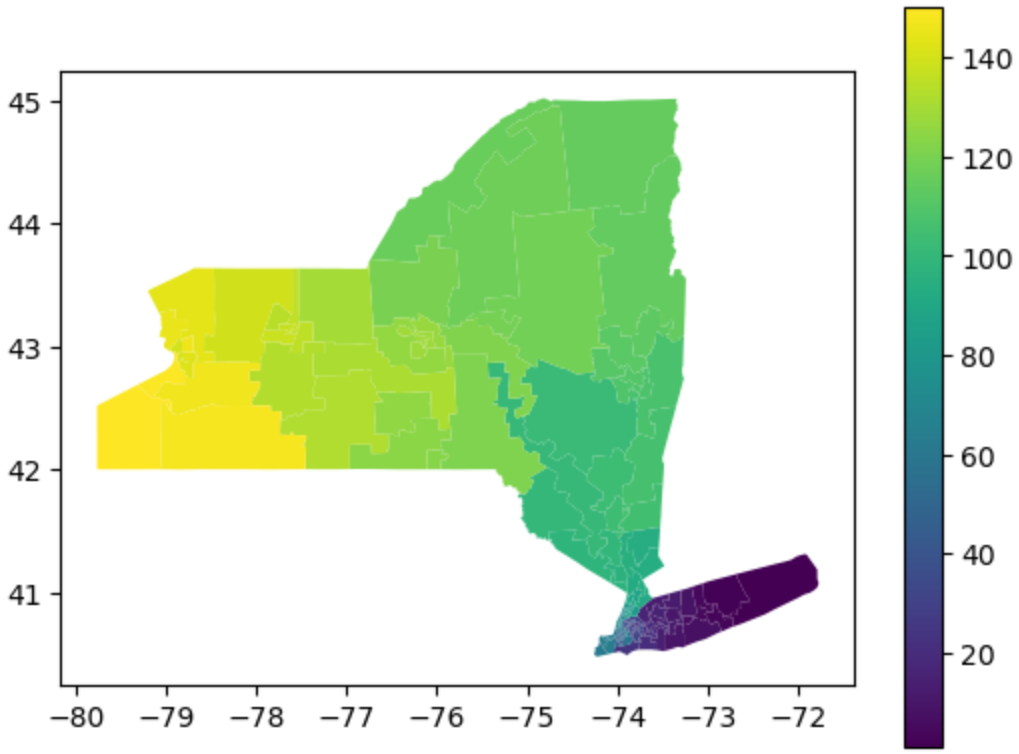
Out[105]: <Axes: >

Loading [MathJax]/extensions/Safe.js



In [106…
```
data_3.plot('ID', legend=True)
```

Out[106]: <Axes: >



Loading [MathJax]/extensions/Safe.js