



PUBPOL 2130/ INFO 3130

Lab 1





Logistics!

- Main GitHub Repository [here](#)
- We suggest going through the [gentle intro notebook](#) if you do not have experience with programming concepts
- You can reach out to us at
 - jrg377@cornell.edu
 - tp399@cornell.edu
- You can use Colab, Jupyter, VS code, etc.
- We're not going to do installations today, let's work on Colab if you don't have Jupyter, VS Code, etc. installed



Jan 24: Pandas

Python library used for data manipulation, analysis, and cleaning.

```
instructors = pd.Series(["Laura Tach", "Moon Duchin", "Rachel Riedl", "Benjamin Soltoff"], index=["PUBPOL 2301", "PUBPOL 2130", "PUBPOL 2320", "INFO 2951"])

print("\nPandas Series Example")
print(instructors)
```

Pandas Series Example
 PUBPOL 2301 Laura Tach
 PUBPOL 2130 Moon Duchin
 PUBPOL 2320 Rachel Riedl
 INFO 2951 Benjamin Soltoff
 dtype: object

Series

```
df = pd.DataFrame({
    "id": [
        "PUBPOL 2301",
        "PUBPOL 2130",
        "PUBPOL 2320",
        "INFO 2951",
    ],
    "name": [
        "Introduction to Public Policy",
        "Data and the State: How Governments See People and Places",
        "Global Democracy and Public Policy",
        "Introduction to Data Science with R",
    ],
    "instructor": ["Laura Tach", "Moon Duchin", "Rachel Riedl", "Benjamin Soltoff"],
    "credits": [4., 4., 3., 4.],
})
df
```

Dataframe

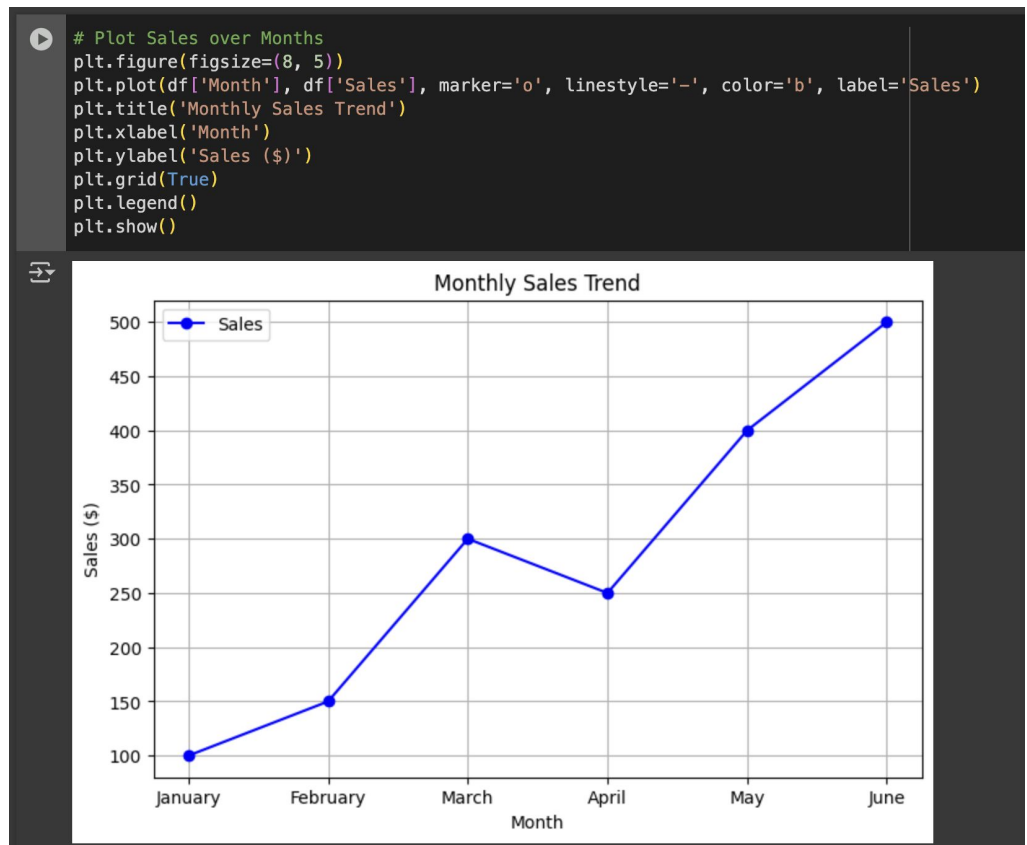
	id	name	instructor	credits
0	PUBPOL 2301	Introduction to Public Policy	Laura Tach	4.0
1	PUBPOL 2130	Data and the State: How Governments See People...	Moon Duchin	4.0
2	PUBPOL 2320	Global Democracy and Public Policy	Rachel Riedl	3.0
3	INFO 2951	Introduction to Data Science with R	Benjamin Soltoff	4.0



Jan 24: Matplotlib Theory

Python library used for creating static, interactive, and animated visualizations.

- Versatility
- Customization
- Integration
- Interactive Capabilities
- Export Options





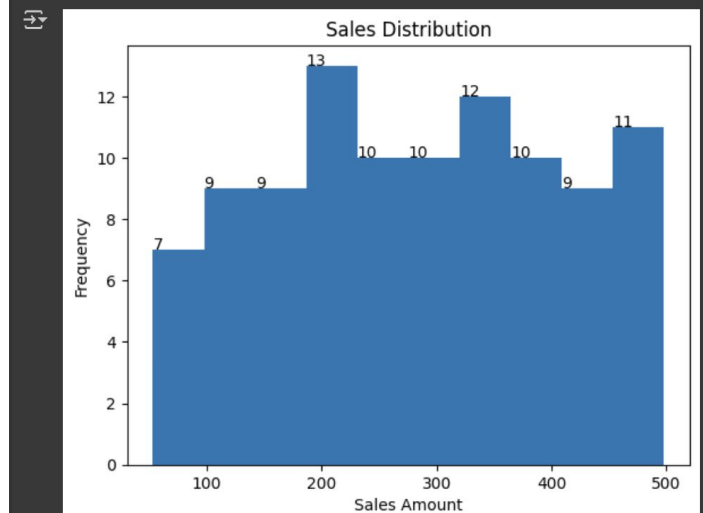
Jan 24: Matplotlib Disadvantages

- Complexity
- Verbose Syntax
- Limited Interactivity
- Performance Issues
- Default Aesthetics are Outdated

```
[8] print(df.head())
```

```
  Sales  
0     99  
1    406  
2    319  
3    206  
4    187
```

```
counts, bins, _ = plt.hist(df['Sales'], bins=10) # 10 bins by default  
plt.title('Sales Distribution')  
plt.xlabel('Sales Amount')  
plt.ylabel('Frequency')  
  
# Add default placement of data labels  
for i in range(len(counts)):  
    plt.text(bins[i], counts[i], str(int(counts[i])))  
  
# Show the plot  
plt.show()
```





Let's start executing week1.ipynb together!



PUBPOL 2130/ INFO 3130

Lab 2





Announcements!

Weekly homework assignments:

- Will be due in 11 days
- New homework assigned on Fridays during lab
- Turn in on Gradescope

Upcoming exam on Feb. 13th:

- Will be 40 minutes, in class
- Lecture on Feb. 11th – likely exam review or makeup



Announcements!

Homework Reminders:

- Don't give us code unless we ask for it!
 - *Don't turn in an .ipynb file*
 - *Turn in exports, not screenshots*
- Make sure axis labels are clear
- Include information on parameters that don't change
- Default parameters in matplotlib may not be optimal – experiment with different ones
 - *E.g., binning with histograms*



Jan 31: Census Data

United States[®] Census Bureau

Search Advanced Search

All **Tables** Maps Charts Profiles Pages

1 Filter 1 New York 4109 Results Clear all filters Search for a filter or table

Geographies

- Nation
- State
- County
- County Subdivision
- Place
- ZIP Code Tabulation Area
- Metropolitan/Micropolitan Statistical Area
- Census Tract
- Block
- Block Group
- All Geographies

Topics

- Business and Economy
- Education
- Employment
- Families and Living Arrangements

4109 Results View: 10 | 25 | 50 Download Table Data

P1 | RACE

Decennial Census Universe: Total population 2020: DEC Redistricting Data (PL 94-171)

P1 | TOTAL POPULATION View All 10 Products

Population Estimates
PEPANNRES | Annual Estimates of the Resident Population: April 1, 2010 to July 1, 2019: PEP Population Estimates

American Community Survey
DPO5 | ACS Demographic and Housing Estimates View All 30 Products

Household Pulse Survey
HPS01 | All HPS Indicators for Phase 4.0 and Later
HPS High Frequency Social and Economic Data

American Community Survey
S0101 | Age and Sex View All 27 Products

American Community Survey
S0102 | Population 60 Years and Over in the United States View All 27 Products

American Community Survey
S0103 | Population 65 Years and Over in the United States View All 27 Products

Table Data

	New York
Total:	20,201,249
Population of one race:	18,433,786
White alone	11,143,349
Black or African American alone	2,986,172
American Indian and Alaska Native alone	149,690
Asian alone	1,933,127
Native Hawaiian and Other Pacific Islander alone	10,815
Some Other Race alone	2,210,633
Population of two or more races:	1,767,463
Population of two races:	1,649,229
White; Black or African American	175,686
White; American Indian and Alaska Native	113,950
White; Asian	148,927
White; Native Hawaiian and Other Pacific Islander	3,890
White; Some Other Race	840,481
Black or African American; American Indian and Alaska Native	31,562
Black or African American; Asian	21,450
Black or African American; Native Hawaiian and Other Pacific Islander	3,274
Black or African American; Some Other Race	226,733
American Indian and Alaska Native; Asian	5,958
American Indian and Alaska Native; Native Hawaiian and Other Pacific Islander	564

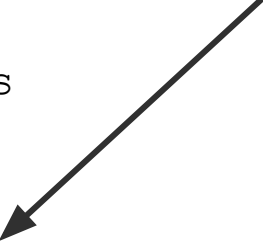


Jan 31: census Python Package

- **Wrapper** for the United States Census Bureau's API
 - More information [here](#)
- Information on the Census Bureau API is [here](#) and [here](#)
 - You can request an API key [here](#)

```
from census import Census
from us import states
```

```
c = Census("MY_API_KEY")
c.acs5.get(('NAME', 'B25034_010E'),
          {'for': 'state:{}'.format(states.MD.fips)})
```



Note: you do not need an API key for querying small quantities of data, with minimal restrictions (e.g. <500 queries/day per IP)



Jan 31: Exporting plots

- Tricky in Colab vs. VSCode/Jupyter
- In **matplotlib**: `plt.savefig("file_name.jpg")`
- In **Colab**:

```
from google.colab import files  
plt.savefig("file_name.jpg")  
files.download("file_name.jpg")
```

- Alternatively, you can use simple scripts in Colab to save exports to your **temporary** Colab environment

```
plt.savefig("file_name.jpg", format="jpeg", dpi=95)
```



Let's start executing Week2.ipynb together!



PUBPOL 2130/ INFO 3130

Lab 3

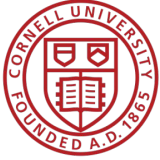




Feb 07: What Are Shapefiles?

A shapefile is a widely used **geospatial data format** for mapping locations, boundaries, and spatial relationships.

- It represents **geographic features** as points, lines, or polygons.
- **Common Uses:** Political boundaries, census tracts, roads, environmental features.
- Shapefile **Components:**
 - .shp – Stores geometry (the actual shapes).
 - .shx – Index for quick lookup.
 - .dbf – Attribute data (tabular information).



Feb 07: Census Shapefiles

Some examples of Census shapefiles

- States
- Counties and county equivalents
- County subdivisions
- Census tracts
- American Indian, Alaska Native, Native Hawaiian areas
- Tribal subdivisions
- Roads, rails, rivers
- School districts, etc.



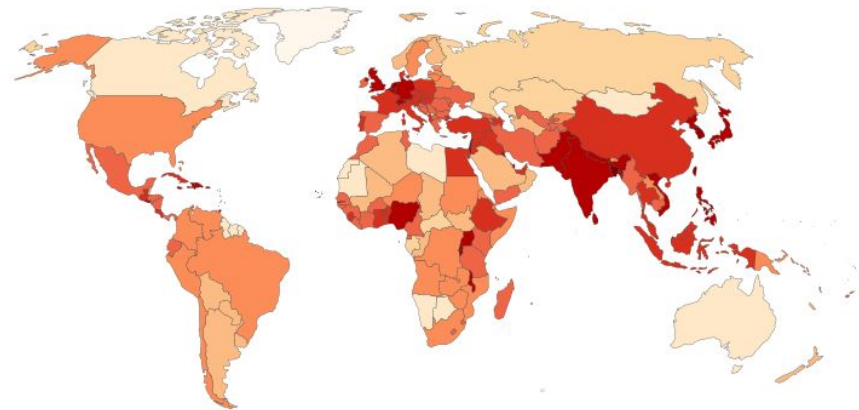
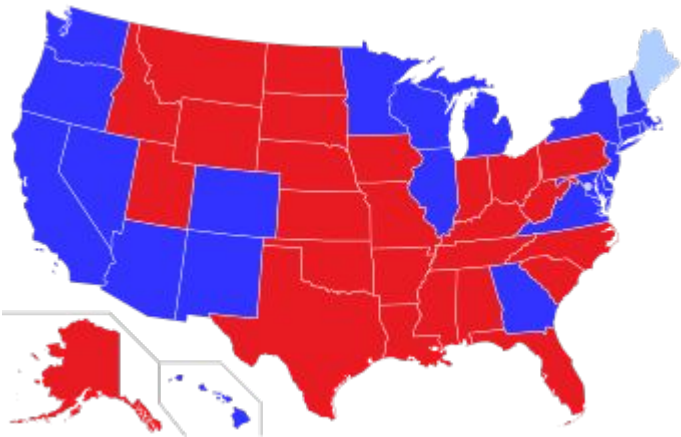
Feb 07: What Is a Choropleth Map?

A choropleth map is a thematic map where areas are shaded or colored based on data values.

- Each region (e.g., state, county) is filled with a color corresponding to a data variable (e.g., population, unemployment rate).

Population density, 2022
The number of people per km² of land area

Our World
in Data



No data 0 2 5 10 20 50 100 200 500 1,000

Data source: HYDE (2017); Gapminder (2022); UN WPP (2022); UN FAO (2022)



Let's start executing Week3.ipynb together!



PUBPOL 2130/ INFO 3130

Lab 4





Feb 14: Announcements

- No new homework this week
- Seems you all did great on the test (and particularly on the Python problems)
- Python literacy is a learning objective -- understanding basic syntax is important
 - Spend time on the notebooks outside of class, and ask questions!
- We plan to offer a Python learning session some time next week if you need more support
 - Please fill out this google form:
<https://tinyurl.com/2130-poll>



Feb 14: Census Blocks

Blocks:

- Statistical areas with natural boundaries (e.g., roads)
- Cover the entire U.S.
- Smallest geographic unit for demographic data

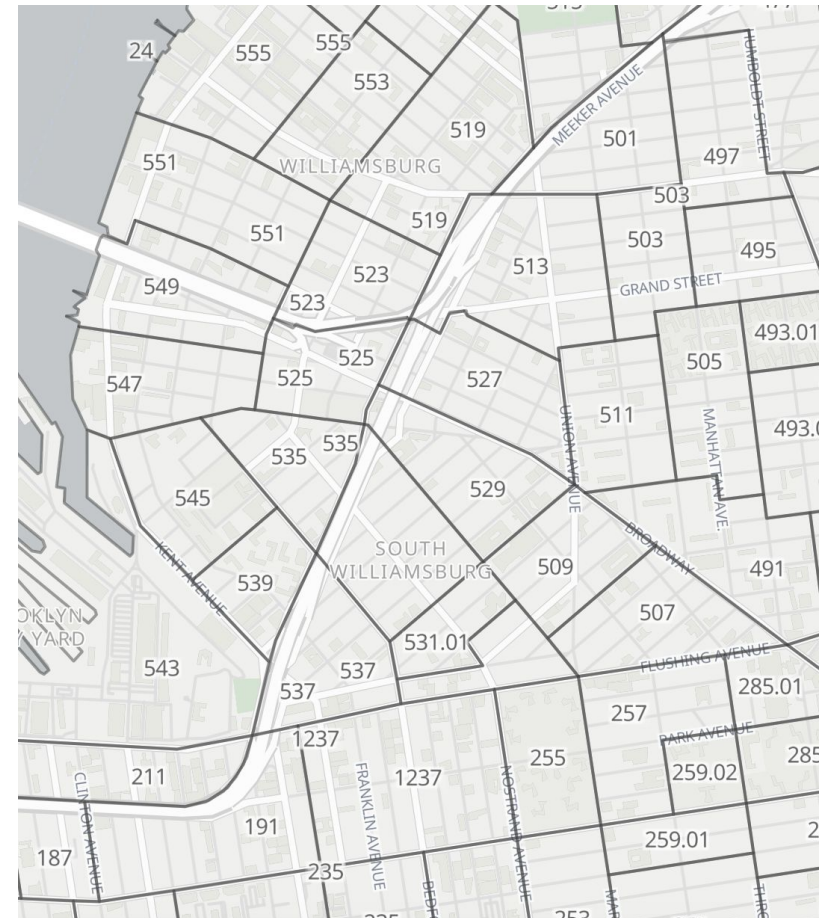




Feb 14: Census Tracts

Tracts:

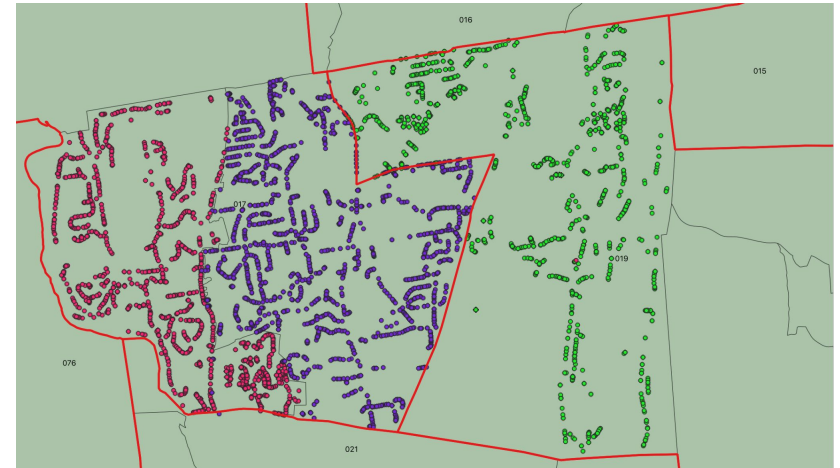
- Small, statistical subdivisions of counties
- Population between 1,200 and 8,000
- Spatial size varies widely





Feb 14: Precincts

- Finest resolution of election data
- Not consistently maintained by states!
- mggg contains an open repository of precincts data





Feb 14: maup

- A geospatial toolkit for redistricting data
- Helpful for:
 - Aggregating from blocks to precincts
 - Disaggregating from precincts to blocks
 - “Prorating” data when there is no clean overlap



Feb 14: Assigning precincts to districts

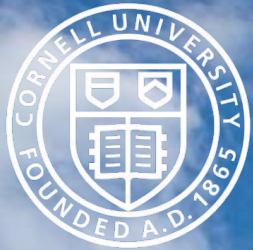
Assigning precincts to districts

The `assign` function in `maup` takes two sets of geometries called `sources` and `targets` and returns a pandas `Series`. The Series maps each geometry in `sources` to the geometry in `targets` that covers it. (Here, geometry *A* covers geometry *B* if every point of *A* and its boundary lies in *B* or its boundary.) If a source geometry is not covered by one single target geometry, it is assigned to the target geometry that covers the largest portion of its area.

```
>>> import maup
>>>
>>> precinct_to_district_assignment = maup.assign(precincts, districts)
>>> # Add the assigned districts as a column of the `precincts` GeoDataFrame:
>>> precincts["DISTRICT"] = precinct_to_district_assignment
>>> precinct_to_district_assignment.head()
0      7
1      5
2     13
3      6
4      1
dtype: int64
```



Let's start executing Week4.ipynb together!



PUBPOL 2130/ INFO 3130

Lab 5





Logistics!

- Created a slide to guide you through joins - please refer to it
- Today's notebook requires creation of a Google sheet, and map creation in flowmap.blue using this Google sheet; in CASE your sheet creation does not work, you can use these sheet IDs instead
 - [yAnUt3bQcGpokOzteRQ2iGQUK3Lyw5S0OMwOOCa0wLQ](#)
 - [1ffUAGYyzzPn3yY-0HehKnVsLayonauOLFtSPT3cPd0A](#)
 - [1DoFxzh7 TKj2hbW7WV7fxbt0 MjjliKhAsGY7TDv4TE](#)



Feb 21: Flowmaps

A flow map is a type of thematic map that visualizes movement or flow of objects, people, or data from one location to another.

Uses lines/arrows to show direction and magnitude of movement.

- **Lines/Arrows:** Represent movement direction.
- **Line Thickness/Color:** Can indicate volume/intensity of movement.
- **Nodes (Start/End Points):** Origin and destination of movement.
- **Base Map:** Provides spatial context (e.g., roads, cities, regions).

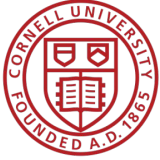
What we're using for Flowmaps: [flowmap.blue](#)



Feb 21: Flowmaps Applications

Flowmaps are commonly used to visualise these patterns

- **Migration Patterns:** People moving between cities or countries.
- **Trade Flows:** Import/export routes between regions.
- **Transportation & Traffic:** Airline routes, shipping lanes, or road traffic.
- **Energy Distribution:** Power grids, oil pipelines.
- **Internet Data Flow:** Digital connectivity between locations.



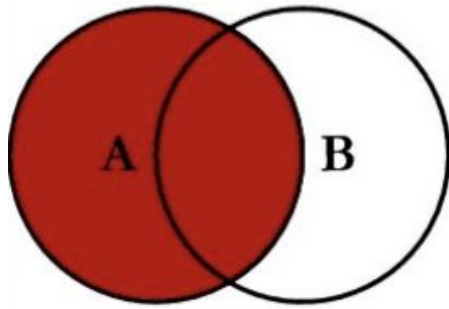
Feb 21: Flowmap example

Visualization of Bike Sharing System movements in Helsinki



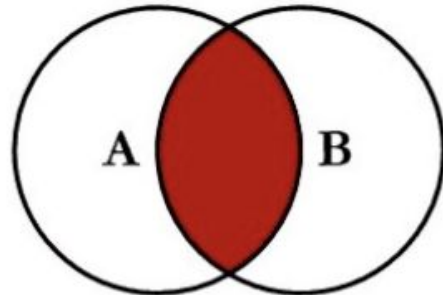


Feb 21: Joins (reference slide)



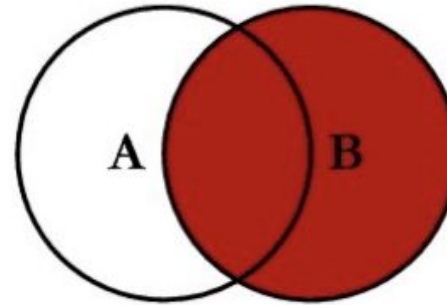
Left Join

Keeps all elements in the left table, and common elements from the right table ONLY



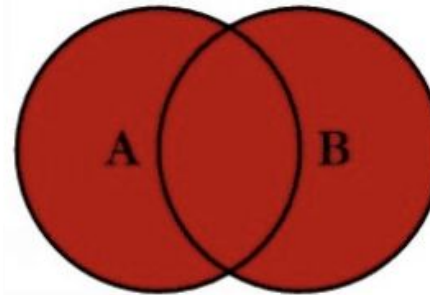
Inner Join

Keeps all elements common common to BOTH tables



Right Join

Keeps all elements in the right table, and common elements from the left table ONLY



Full Outer Join

Keeps ALL elements in from both tables.

```
df_joined = left_table.join(right_table, how = "outer")
```



Let's start executing Week5.ipynb together!



PUBPOL 2130/ INFO 3130

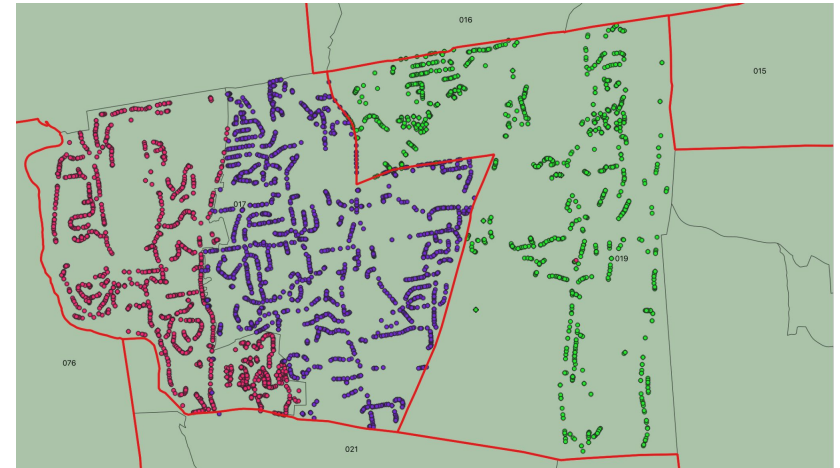
Lab 6






Feb 27: Precincts

- Finest resolution of election data
- Not consistently maintained by states!
- mggg contains an open repository of precincts data





Feb 27: Precincts

A banner image with a light blue background. On the left, there is a partial view of a circular object with an American flag pattern (stars and stripes). The text "2022 NY General Election Shapefile and Results by Election District" is written in a light blue, sans-serif font, centered on the right side of the banner.

2022 NY General Election Shapefile and Results by Election District

Credit and Thank You

This data would not have been possible to create and provide without the assistance of each of New York's county Boards of Elections commissioners and staffers, who were often very eager to help provide the data I was looking for; county GIS and Planning Department workers, who filled in major gaps; Derek Willis and the volunteers at Open Elections who worked diligently to clean up and standardize the election results data files I compiled for the entire state; the folks at the Redistricting Data Hub who helped verify and correct shapefile errors; and the wonderful folks at the Census Bureau and the U.S. Department of Agriculture National Resources Conservation Service and the U.S. Geological Survey, without whom the shapefile would have been impossible to create.



Feb 27: maup

- A geospatial toolkit for redistricting data
- Potential uses:
 - Assigning precincts to districts,
 - Aggregating block data to precincts,
 - Disaggregating data from precincts down to blocks,
 - Prorating data when units do not nest neatly, and
 - Fixing topological issues, overlaps, and gaps



Feb 27: maup.assign()

```
precinct_to_district_assignment = maup.assign(blocks, pcts)
```

- Takes in two sets of geometries
- Returns a Pandas series with the assignments
- Use cases:
 - Assign blocks to precincts
 - Assign precincts to districts



Feb 27: maup.prorate()

`maup.prorate(overlapping_blocks, data_to_prorate, weights)`

- Problem:
 - You have precincts with some election results data
 - You also have different precincts (e.g., redistricting)
- We can **prorate** data from the old precincts to the new precincts by weighting the data proportional to the overlapping population.
 - Disaggregate to Census blocks
 - Reaggregate to new precincts



Let's start executing Week6.ipynb together!