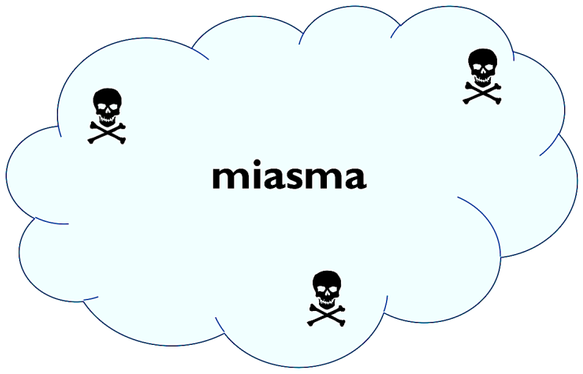
VISUALIZING GEOSPATIAL DATA IN PYTHON

Introduction

Mary van Valkenburg
Data Science Program Manager, Nashville Software School

Location

- 1854 cholera outbreak in London
- 600+ deaths



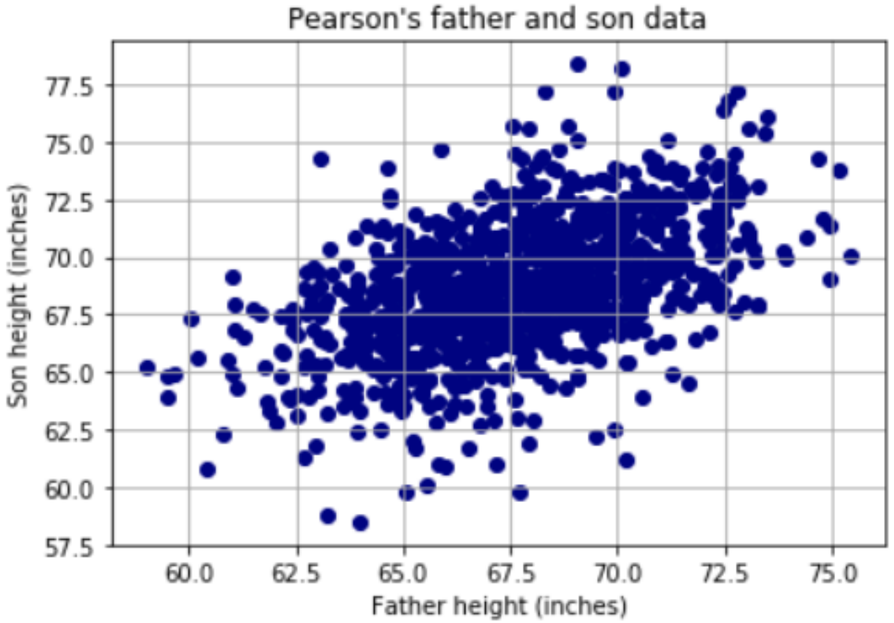
Snow's dot map

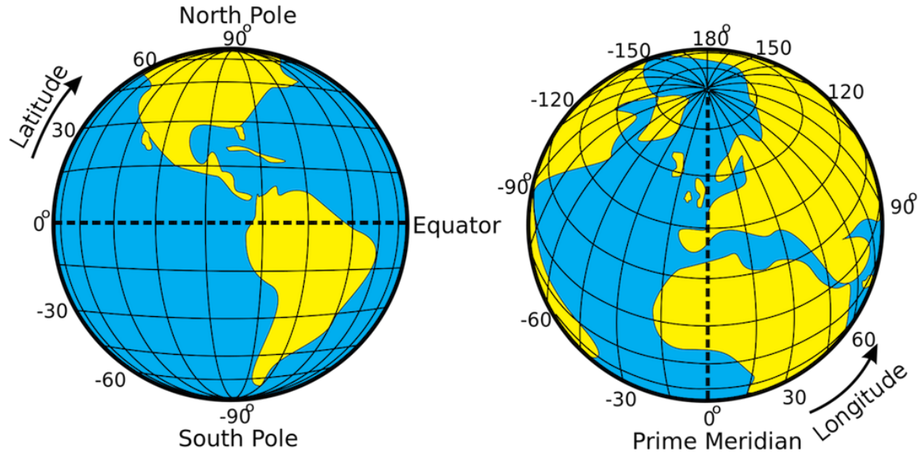
https://myweb.rollins.edu/jsiry/London-Cholera-John_Snow-copy.jpg

What you will learn in this course

- How to plot geospatial points as scatterplots
- How to plot geometries using `geopandas`
- How to construct a `GeoDataFrame` from a `DataFrame`
- How to spatially join datasets
- How to add a street map to your plots
- When and how to create a choropleth

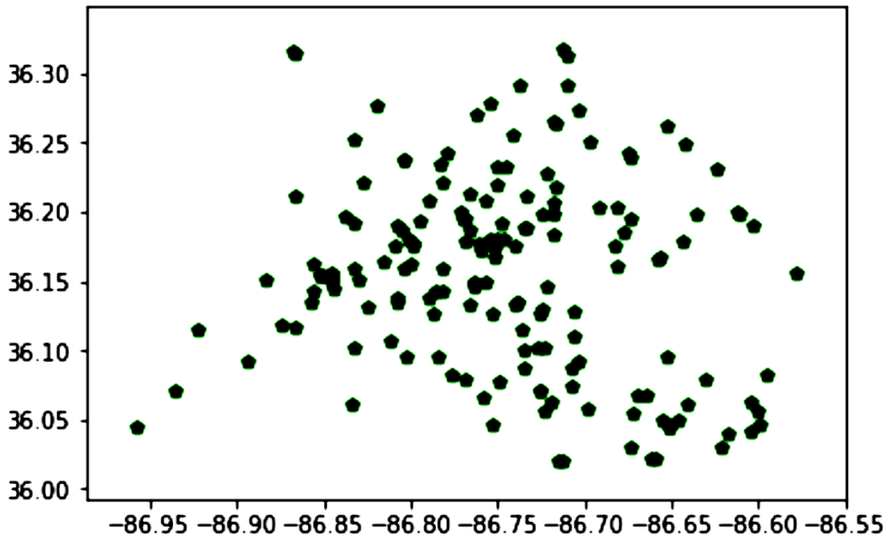
Longitude and latitude



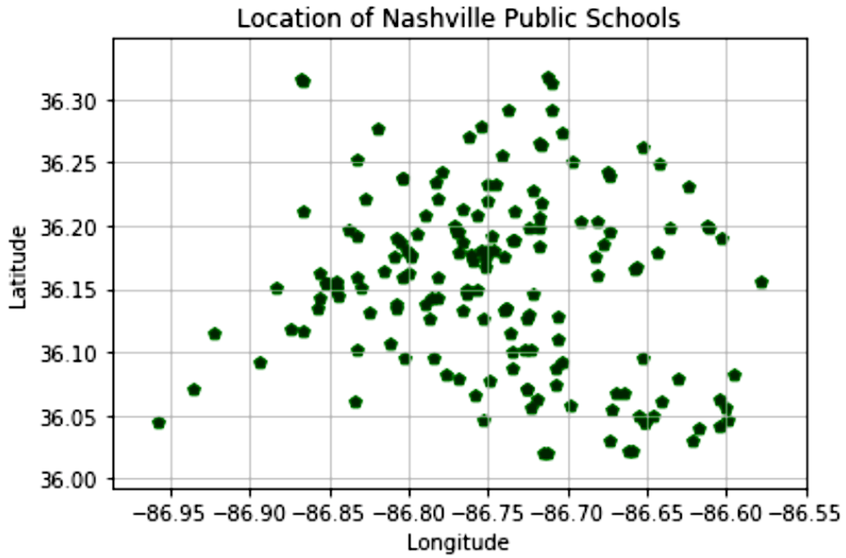


Scatterplot styling

```
plt.scatter(schools.Longitude,
            schools.Latitude,
            c = 'darkgreen',
            marker = 'p')
plt.show()
```




```
plt.scatter(schools.Longitude,
            schools.Latitude,
            c = 'darkgreen',
            marker = 'p')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.title('Nashville Public Schools')
plt.grid()
plt.show()
```



Extracting longitude and latitude

```
bus_stops.head()
```

```
Stop ID    StopName        Location
4431       MCC5_11     (36.16659, -86.781996)
588        CHA7AWN     (36.165, -86.78406)
590        CHA8AWN     (36.164393, -86.785451)
541        CXONGULC    (36.162249, -86.790464)
5231       7AVUNISM    (36.163822, -86.783791)
```

Extracting longitude and latitude

```
bus_stops['lat'] = [loc[0] for loc in bus_stops.Location]
bus_stops['lng'] = [loc[1] for loc in bus_stops.Location]
bus_stops.head()
```

```
Stop ID    StopName  Location                  lat         lng
4431       MCC5_11   (36.16659, -86.781996)    36.16659    -86.781996
588        CHA7AWN   (36.165, -86.78406)       36.165      -86.78406
590        CHA8AWN   (36.164393, -86.785451)   36.164393   -86.785451
541        CXONGULC  (36.162249, -86.790464)   36.162249   -86.790464
5231       7AVUNISM  (36.163822, -86.783791)   36.163822   -86.783791
```

Extracting longitude and latitude with regular expressions

```
bus_stops2.head()
```

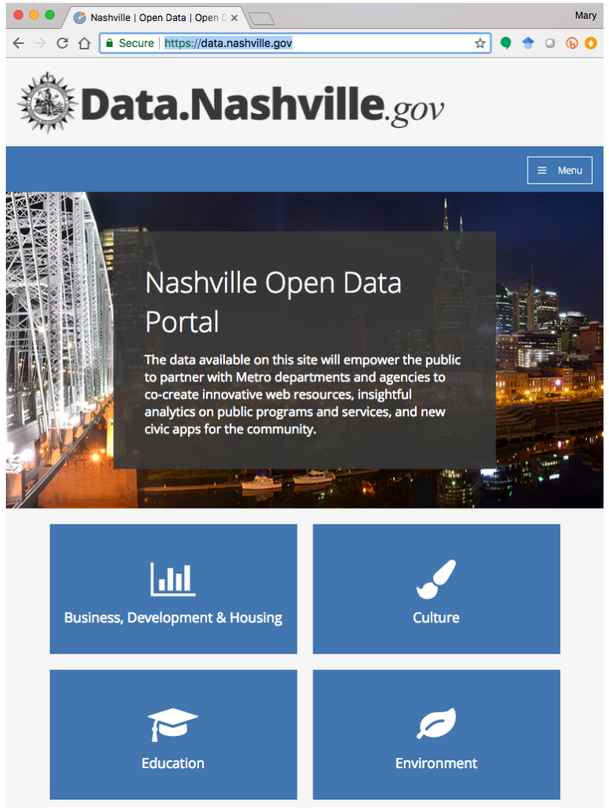
```
Stop ID         Location
4431       MCC - BAY 11\nNashville, TN\n(36.16659, -86.78199)
588        CHARLOTTE AVE\nNashville, TN\n(36.165, -86.78406)
590        CHARLOTTE AV\nNashville, TN\n(36.164393, -86.785451)
541        CHARLOTTE\nNashville, TN\n(36.162249, -86.790464)
5231       Nashville, TN\n(36.163822, -86.783791)
```

Extracting longitude and latitude with regular expressions

```
lat_lng_pattern = re.compile(r'\((.*),\s*(.*)\)', flags=re.MULTILINE)
def extract_lat_lng(address):
    try:
        lat_lng_match = lat_lng_pattern.search(address)
        lat = float(lat_lng_match.group(1))
        lng = float(lat_lng_match.group(2))
        return (lat, lng)
    except:
        return (np.NaN, np.NaN)
```

```
lat_lngs = [extract_lat_lng(location)for location in \
           bus_stops2.loc[:, 'Location']]
bus_stops2['lat'] = [lat for lat, lng in lat_lngs]
bus_stops2['lng'] = [lng for lat, lng in lat_lngs]
```

Nashville open data



VISUALIZING GEOSPATIAL DATA IN PYTHON

Let's practice!

VISUALIZING GEOSPATIAL DATA IN PYTHON

Geometries and shapefiles

Mary van Valkenburg
Data Science Program Manager, Nashville Software School

Shapefiles

Shapefiles store a special type of data known as *geometry*.

Shapefile components

KEEP ALL THE FILES TOGETHER!

```
$ ls my_map_files/

my_map.dbf
my_map.shp
my_map.shx
```

- my_map.shp (contains the geometry)
- my_map.dbf (holds attributes for each geometry)
- my_map.shx (links the attributes to the geometry)

geopandas

This code reads a shapefile into a GeoDataFrame and looks at the first few rows.

```
import geopandas as gpd

geo_df = gpd.read_file('My_Map_Files/my_map.shp')
geo_df.head()
```


KEEP NASHVILLE CLEAN

Viewing a geometry

```
service_district.loc[0, 'geometry']
```

Printing a geometry

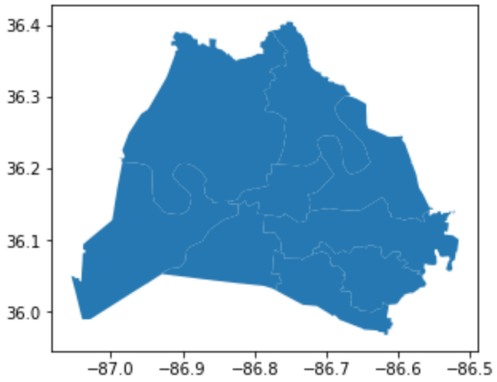
```
print(service_district.loc[0, 'geometry'])
```

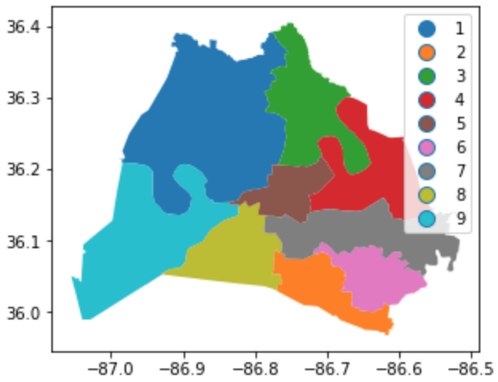
```
POLYGON ((-86.68680500011935 36.28670500013504,
-86.68706099969657 36.28550299967364, -86.68709498823965 36.28511683351293,
-86.68712691935902 36.28475404474551, -86.6871549990252 36.28443499969863,
-86.68715025108719 36.28438104319917, -86.68708600011215 36.2836510002216,
-86.6870599998375 36.28335400009232, -86.68683200030846 36.28073200026927,
-86.68678671280243 36.2804916722591, -86.68668199966068 36.27993600019391,
-86.686543000303 36.27920000021985, -86.68641799989246 36.27853199938513,
-86.68600744248923 36.27759483150202, -86.68579942352289 36.27711998225582,
-86.68482299948184 36.2748910007355, -86.68476799897849 36.27478700083996,
-86.68372700043393 36.27281799971492, -86.6832880000829 36.27208000018629,
-86.68313199902317 36.27181700012145, -86.68278700024624 36.27108100075766,
-86.68257822861736 36.27077209799597, -86.68177585777893 36.2694062861527,
-86.68129400001521 36.2685690000872, -86.68085800015712 36.26798600010722,
-86.68029000024265....
```

Plotting a GeoDataFrame

```
school_districts.plot()
plt.show()
```

```
school_districts.plot(column =
                      'district',
                      legend = True)
plt.show()
```





VISUALIZING GEOSPATIAL DATA IN PYTHON

Let's practice!

VISUALIZING GEOSPATIAL DATA IN PYTHON

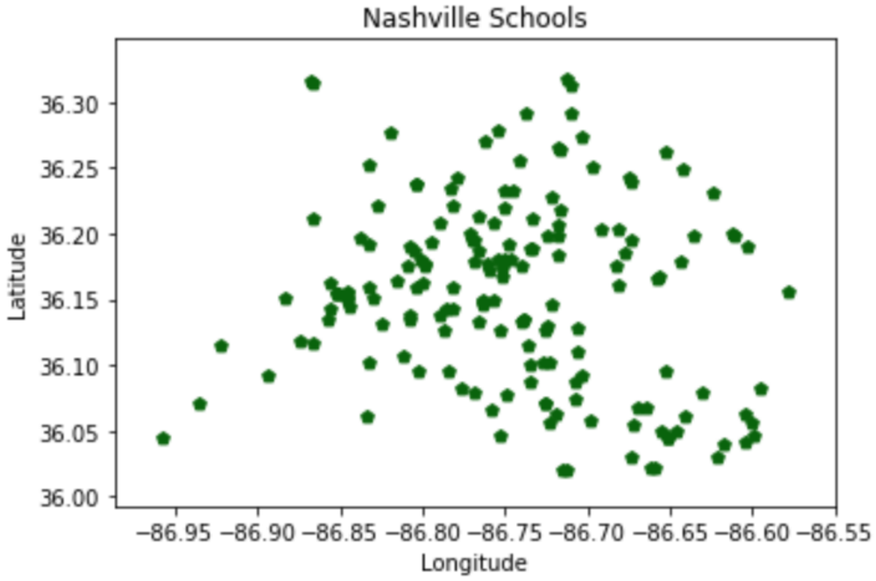
Putting it all together

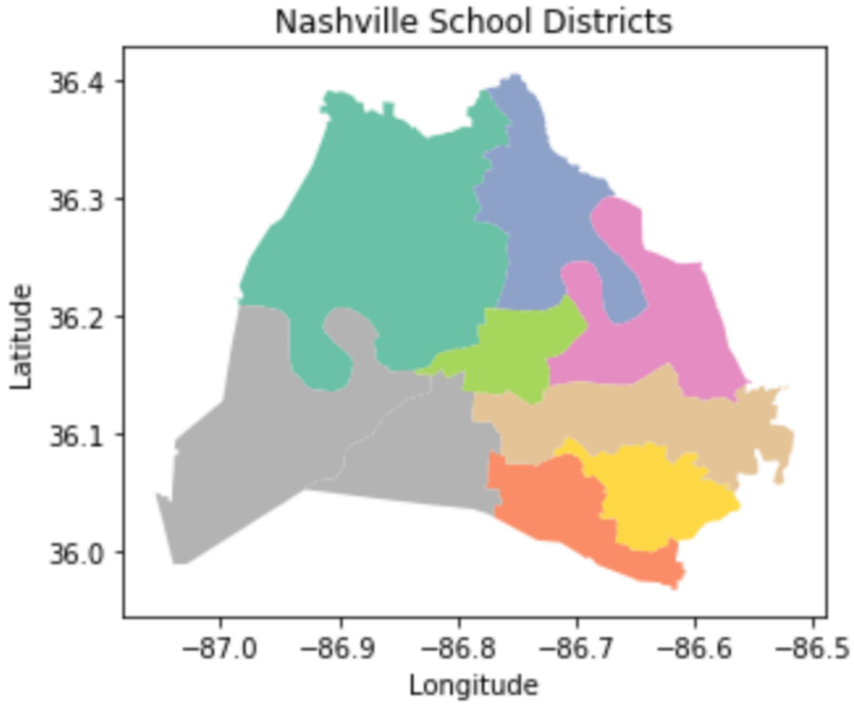
Mary van Valkenburg
Data Science Program Manager, Nashville Software School

Skills list

- Understanding longitude and latitude
- Extracting longitude and latitude
- Plotting points on scatterplot using longitude and latitude
- Styling scatterplots for better aesthetics and insight
- Plotting polygons from shapefiles

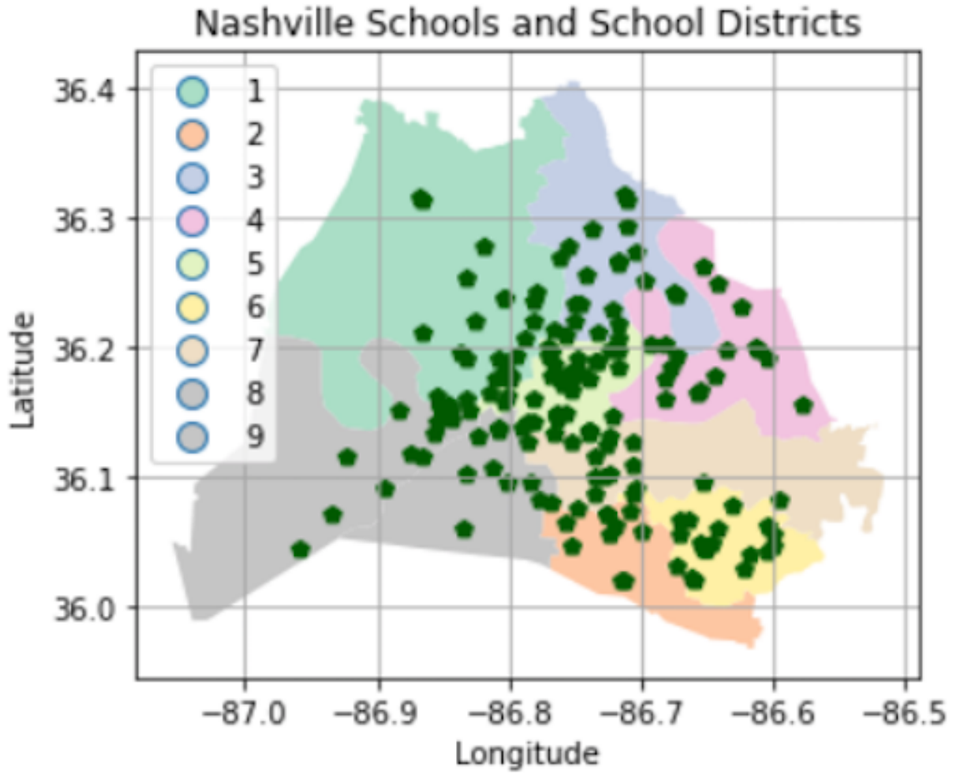
Combining scatterplots and polygons





Combining scatterplots and polygons

```
school_districts.plot(column = 'district', legend = True, cmap = 'Set2')
plt.scatter(schools.lng, schools.lat, marker = 'p', c = 'darkgreen')
plt.title('Nashville Schools and School Districts')
plt.show();
```



VISUALIZING GEOSPATIAL DATA IN PYTHON

Let's practice!

VISUALIZING GEOSPATIAL DATA IN PYTHON

Plotting with GeoJSON

Mary van Valkenburg
Data Science Program Manager, Nashville Software School

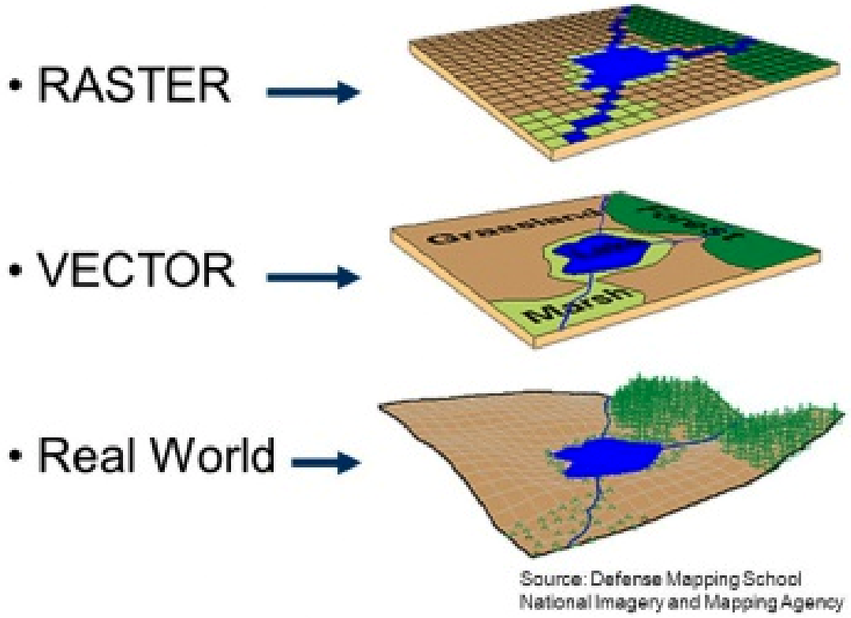
Neighborhoods GeoJSON

```
{
  "type": "FeatureCollection",
  "features": [
    {
      "type":"Feature",
      "properties":{
    "name":"Historic Buena Vista"
    },"geometry":{
        "type":"MultiPolygon","coordinates":[[[[-86.79511056795417,36.17575
        964963348],[-86.79403325521203,36.176723819622765],[-86.79395847673
        587,36.176734201205555],[-86.79373059621346,36.17641850227536],
```

```
neighborhoods = gpd.read_file('./data/neighborhood_boundaries.geojson')
neighborhoods.head(1)
```

```
        name                geometry
Historic Buena Vista    (POLYGON ((-86.79511056795417 36.17575964963348)))
```

Geopandas dependencies

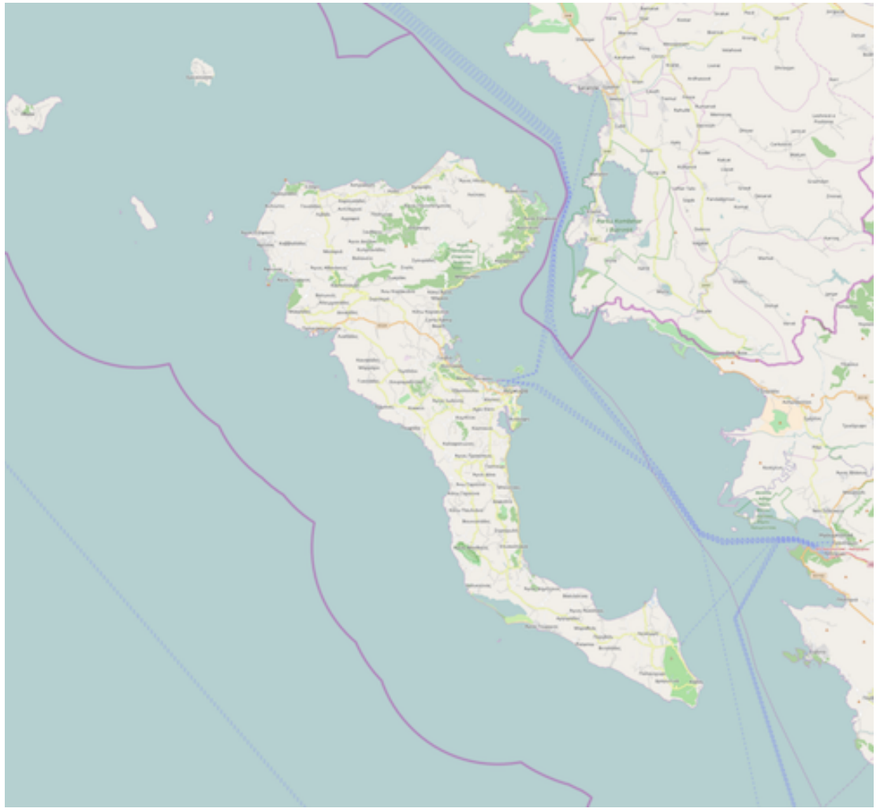


- Fiona
  - provides an python API for OGR
- GDAL/OGR
  - GDAL for translating raster data
  - OGR for translating vector data

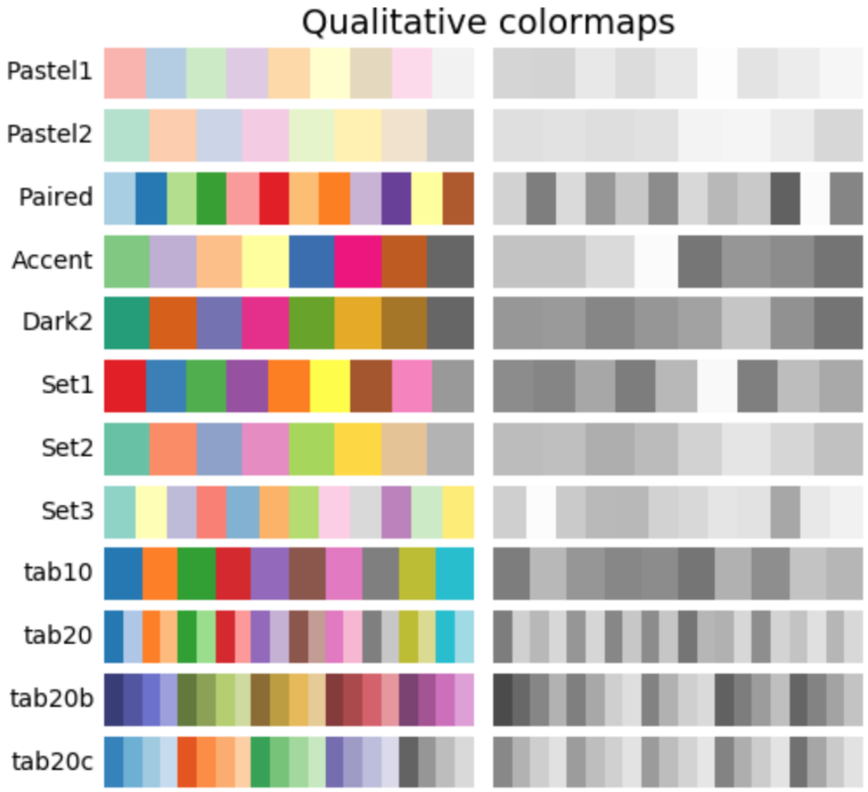
Comparing raster and vector graphics

raster image of Corfu

vector image of Corfu



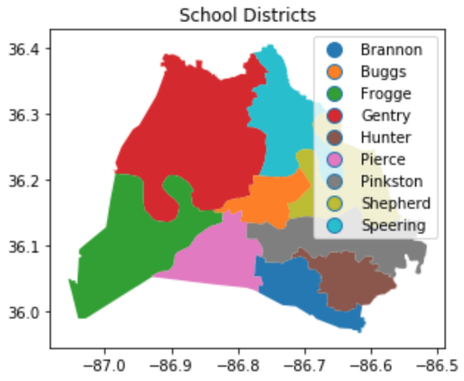
Colormaps

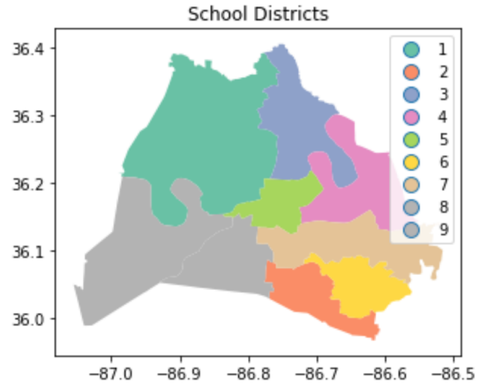


https://matplotlib.org/users/colormaps.html

Plotting with color

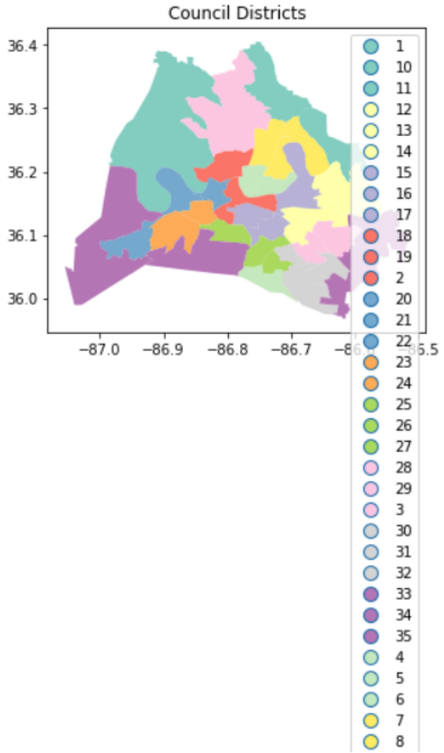
```
school_districts.head(3)

first_name      last_name     position     district      geometry
Sharon          Gentry        Member        1     (POLYGON ((-86.771 36.383)))
Jill            Speering      Vice-Chair    3     (POLYGON ((-86.753 36.404)))
Jo Ann          Brannon       Member        2     (POLYGON ((-86.766 36.083)))
```



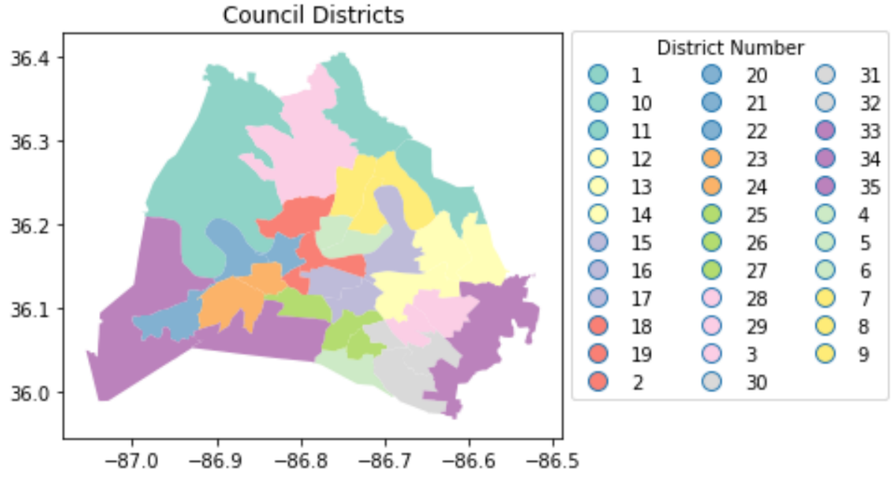


The legend_kwds argument to .plot()

```
council_dists.plot(
                    column='district',
                    cmap='Set3',
                    legend=True)
plt.title('Council Districts')
plt.show();
```




```
leg_kwds={'title':'District Number',
          'loc': 'upper left',
          'bbox_to_anchor':(1, 1.03),
          'ncol':3}
council_dists.plot(column='district',
                   cmap='Set3',
                   legend=True,
              legend_kwds=leg_kwds)
plt.title('Council Districts')
plt.show();
```



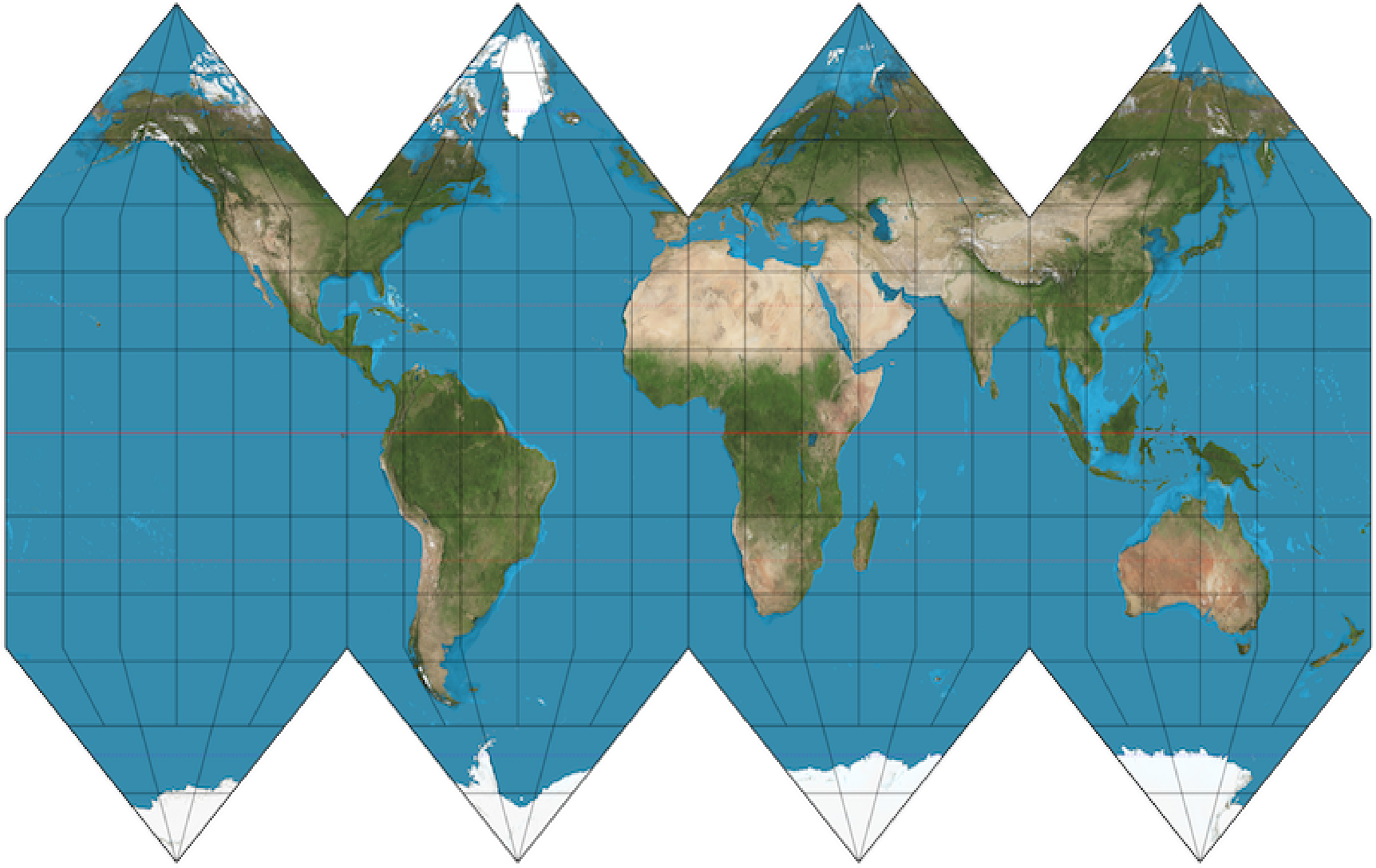
VISUALIZING GEOSPATIAL DATA IN PYTHON

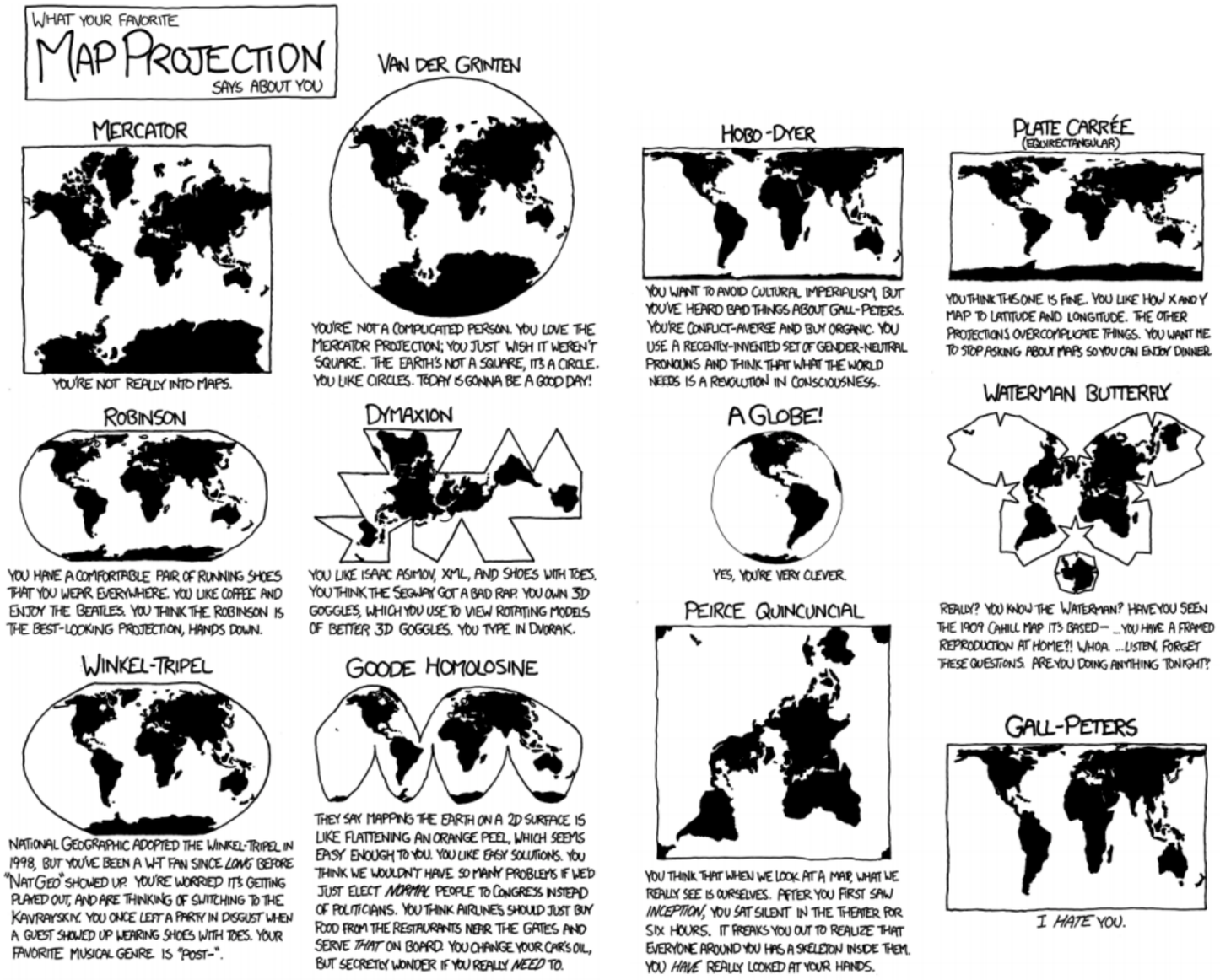
Let's practice!

VISUALIZING GEOSPATIAL DATA IN PYTHON

Projections and Coordinate Reference Systems

Mary van Valkenburg
Data Science Program Manager, Nashville Software School





*What's that? You think I don't like the Peters map because I'm uncomfortable with having my cultural assumptions challenged? Are you sure you're not... ::puts on sunglasses:: ... **projecting**?* http://xkcd.com/977/ - http://bit.ly/explainxkcd-977

Coordinate Reference Systems

EPSG:4326

- used by Google Earth
- units are decimal degrees

EPSG:3857

- used by Google Maps, Bing Maps, Open Street Maps
- units are meters

Creating a geometry column

```
School Name                Latitude     Longitude
A. Z. Kelley Elementary    36.021       -86.658
Alex Green Elementary      36.252       -86.832
Amqui Elementary           36.273       -86.703
Andrew Jackson Elementary  36.231       -86.623
```

```
# create a point geometry column
from shapely.geometry import Point

schools['geometry'] = schools.apply(
    lambda x: Point((x.Longitude, x.Latitude)),
    axis = 1)

schools.head()
```

```
School Name                Latitude     Longitude     geometry
A. Z. Kelley Elementary    36.021    -86.658       POINT (-86.658 36.021)
Alex Green Elementary      36.252    -86.832       POINT (-86.832 36.252)
Amqui Elementary           36.273    -86.703       POINT (-86.703 36.273)
Andrew Jackson Elementary  36.231    -86.623       POINT (-86.623 36.231)
```

Creating a GeoDataFrame from a DataFrame

```
import geopandas as gpd

schools_crs = {'init':  'epsg:4326'}
schools_geo = gpd.GeoDataFrame(schools,
                               crs = schools_crs,
                               geometry = schools.geometry)
```

```
schools_geo.head(4)
```

```
School Name                Latitude     Longitude     geometry
A. Z. Kelley Elementary    36.021    -86.658       POINT (-86.658 36.021)
Alex Green Elementary      36.252    -86.832       POINT (-86.832 36.252)
Amqui Elementary           36.273    -86.703       POINT (-86.703 36.273)
Andrew Jackson Elementary  36.231    -86.623       POINT (-86.623 36.231)
```

Changing from one CRS to another

```
schools_geo.head(2)
```

```
School Name                Latitude     Longitude     geometry
A. Z. Kelley Elementary    36.021    -86.658        POINT (-86.658 36.021)
Alex Green Elementary      36.252    -86.832        POINT (-86.832 36.252)
```

```
# convert geometry from decimal degrees to meters
schools_geo.geometry = schools_geo.geometry.to_crs(epsg = 3857)
schools_geo.head(2)
```

```
School Name                Latitude    Longitude    geometry
A. Z. Kelley Elementary    36.021    -86.658    POINT (-9646818.8 4303623.8)
Alex Green Elementary      36.252    -86.832    POINT (-9666119.5 4335484.4)
```

VISUALIZING GEOSPATIAL DATA IN PYTHON

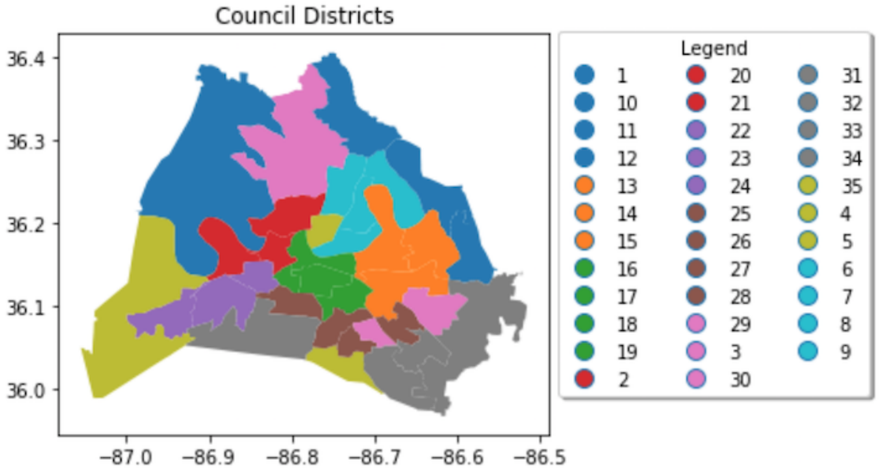
Let's practice!

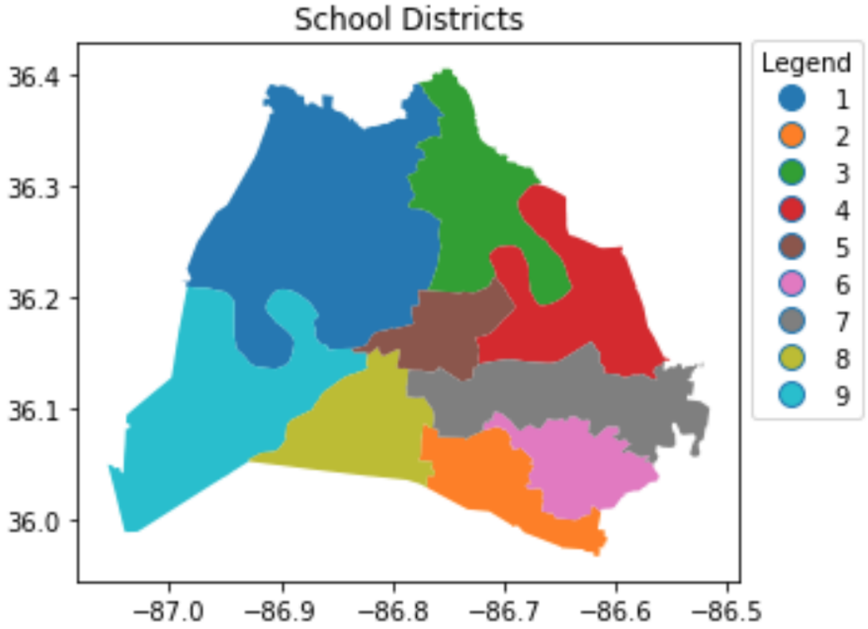
VISUALIZING GEOSPATIAL DATA IN PYTHON

Spatial joins

Mary van Valkenburg
Data Science Program Manager, Nashville Software School

Council districts and school districts



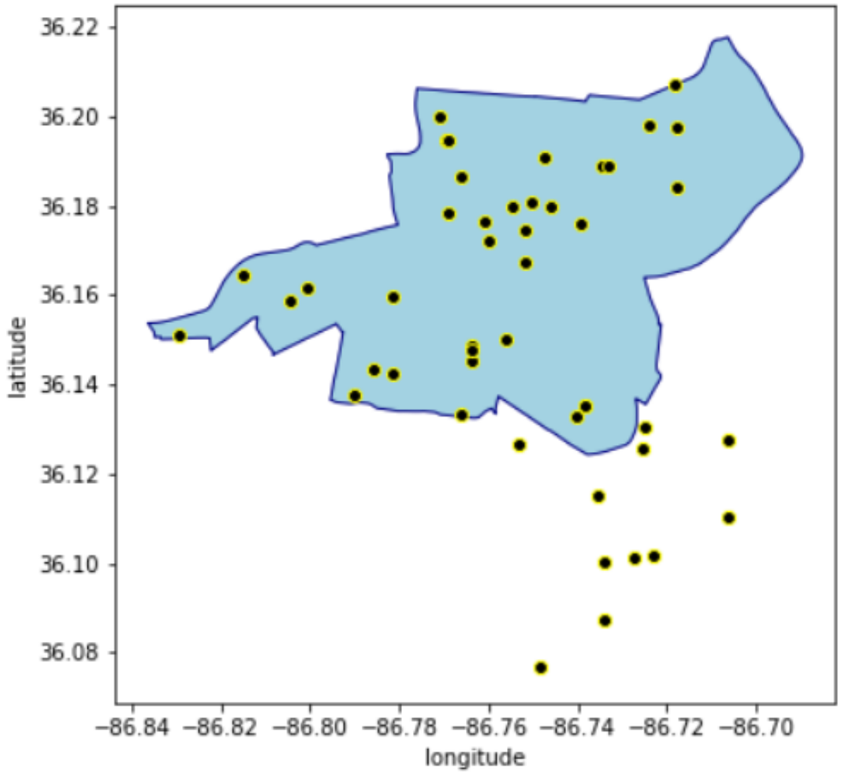


The .sjoin() op argument

```
import geopandas as gpd

gpd.sjoin(blue_region_gdf, black_point_gdf, op = <operation>)
```

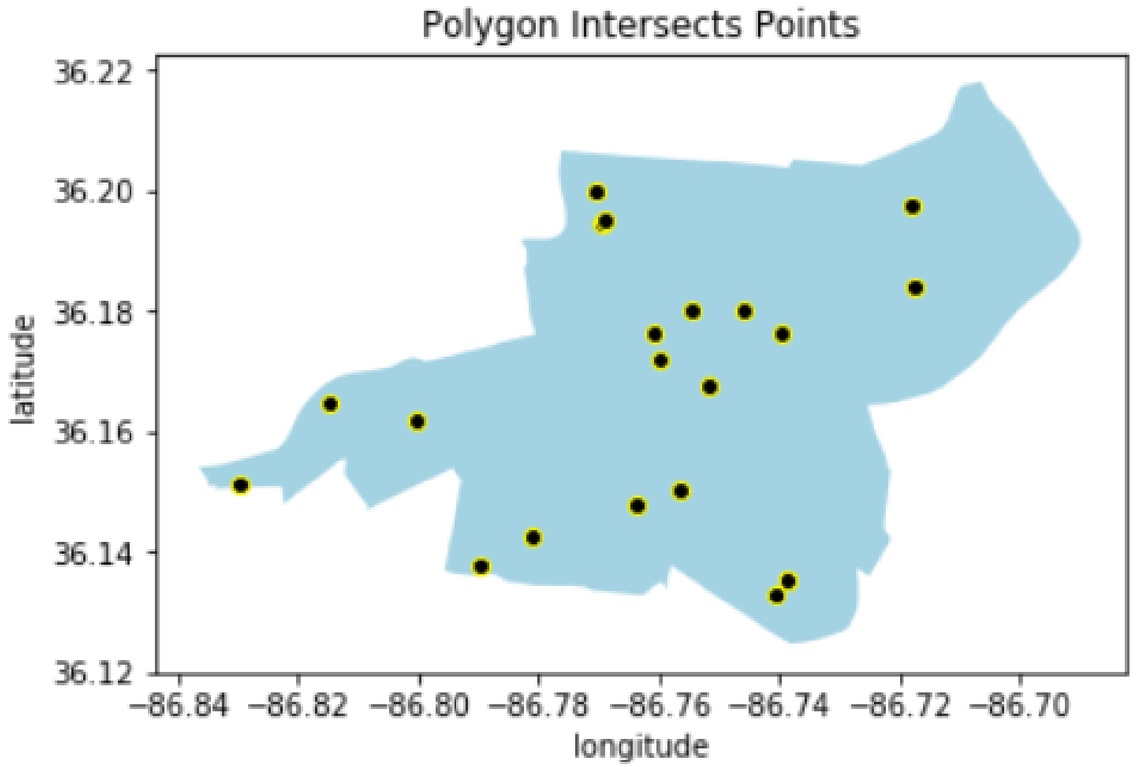
operation can be ***intersects***, ***contains***, or ***within***

Using .sjoin()


36.22
36.20
36.18
36.16
36.14
36.12
36.10
36.08
latitude
−86.84
−86.82
−86.80
−86.78
−86.76
−86.74
−86.72
−86.70
longitude

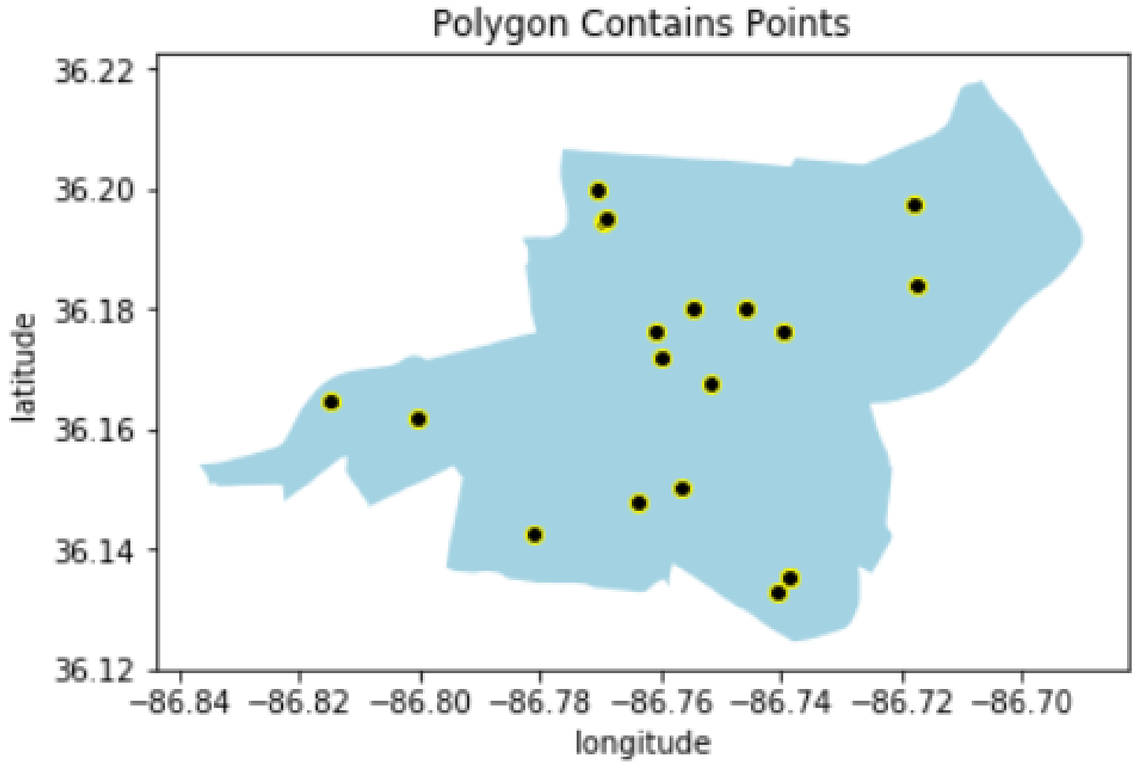
op = 'intersects'

```
gpd.sjoin(blue_region_gdf, black_point_gdf, op = 'intersects')
```



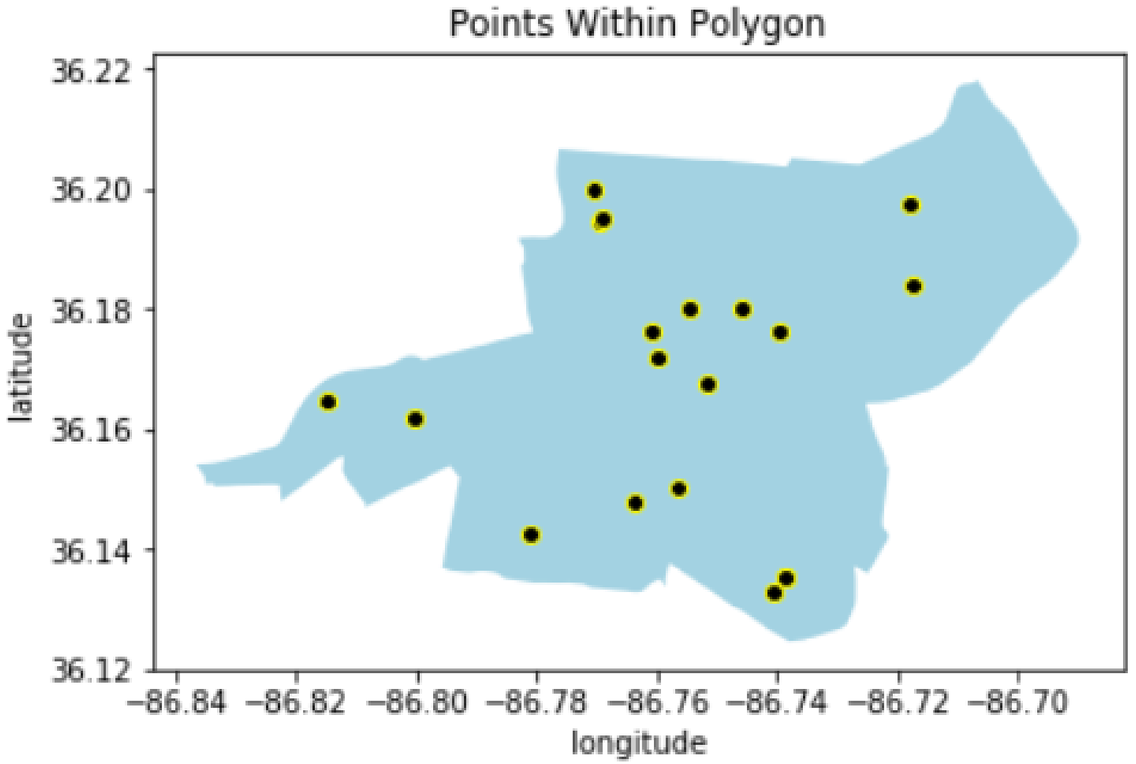
op = 'contains'

```
gpd.sjoin(blue_region_gdf, black_point_gdf, op = 'contains')
```



op = 'within'

```
gpd.sjoin(black_point_gdf, blue_region_gdf, op = 'within')
```



The sjoin.() op argument - within

```
# find council districts within school districts

within_gdf =gpd.sjoin(council_districts, school_districts, op='within')
print('council districts within school districts: ', within_gdf.shape[0])
```

```
council districts within school districts:  11
```

The sjoin.() op argument - contains

```
# find school districts that contain council districts

contains_gdf=pd.sjoin(school_districts, council_districts, op='contains')
print('school districts contain council districts: ', contains_gdf.shape[0])
```

```
school districts contain council districts:  11
```

The sjoin.() op argument - intersects

```
# find council districts that intersect with school districts

intersect_gdf=gpd.sjoin(council_districts, school_districts, op='intersects')
print('council districts intersect school districts: ', intersect.shape[0])
```

```
council districts intersect school districts:  100
```

Columns in a spatially joined GeoDataFrame

```
within_gdf=gpd.sjoin(council_districts, school_districts, op = 'within')
within_gdf.head()
```

```
  first_name_left  last_name_left district_left  index_right
0 Nick             Leonardo              1              0
1 DeCosta          Hastings              2              0
2 Nancy            VanReece              8              1
3 Bill             Pridemore             9              1
9 Doug             Pardue               10              1
```

Aggregating spatially joined data

```
# Aggregate council districts by school district
# to see how many council districts are within each school district.

# first rename district_left and district_right
within_gdf.district_left = council_district
within_gdf.district_right = school_district

within_gdf[['council_district', 'school_district']
          ].groupby('school_district'
                    ).agg('count'
                          ).sort_values('council_district', ascending = False)
```

```
                 council_district
school_district
              3                3
              1                2
              9                2
              2                1
              5                1
              6                1
              8                1
```

VISUALIZING GEOSPATIAL DATA IN PYTHON

Let's Practice!

VISUALIZING GEOSPATIAL DATA IN PYTHON

GeoSeries attributes and methods I

Mary van Valkenburg
Data Science Program Manager, Nashville Software School

Shapely attributes and methods

```
# the geometry column is a GeoSeries
type(school_districts.geometry)
```

```
geopandas.geoseries.GeoSeries
```

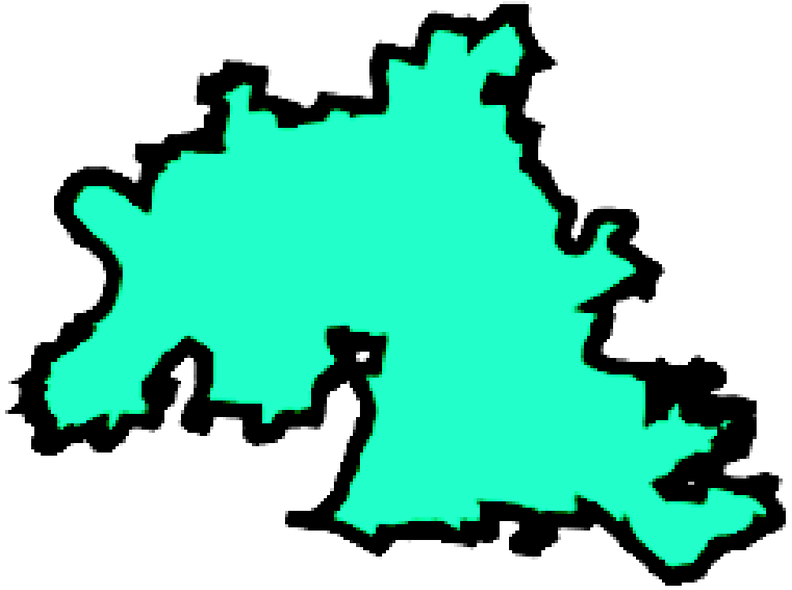
- `GeoSeries.area` - returns the area of each geometry in a GeoSeries
- `GeoSeries.centroid` - returns the center point of each geometry in a GeoSeries
- `GeoSeries.distance(other)` - returns the minimum distance to `other`

GeoSeries.area

- returns the area of each geometry in a GeoSeries

```
# area of first polygon in districts
print(districts.geometry[0].area)
```

```
325.78
```



School district areas

GEOSERIES.AREA

```
# print the first 5 rows of school districts and the total number of rows

print(school_districts.head())
print('There are ', school_districts.shape[0], ' school districts.' )
```

```
first_name  last_name    position  district  geometry
Sharon      Gentry       Member       1      (POLYGON ((-86.771 36.383)))
Jill        Speering     Vice-Chair   3      (POLYGON ((-86.753 36.404)))
Jo Ann      Brannon      Member       2      (POLYGON ((-86.766 36.083)))
Anna        Shepherd     Chair        4      (POLYGON ((-86.580 36.209)))
Amy         Frogge       Member       9      (POLYGON ((-86.972 36.208)))
There are  9  school districts.
```

School district areas

```
# calculate area of each school district
district_area = school_districts.geometry.area

# print the areas and crs used
print(district_area.sort_values(ascending = False))
print(school_districts.crs)
```

```
0    0.036641
4    0.023030
8    0.015004
1    0.014205
3    0.014123
5    0.010704
2    0.008328
7    0.007813
6    0.006415
dtype: float64
{'init': 'epsg:4326'}
```

School district areas

```
# create a copy of school_districts that uses EPSG:3857
school_districts_3857 = school_districts.to_crs(epsg = 3857)
```

```
# define a variable for m^2 to km^2
sqm_to_sqkm = 10**6

# get area in kilometers squared
district_area_km = school_districts_3857.geometry.area / sqkm_to_sqm
print(district_area_km.sort_values(ascending = False))
print(school_districts_3857.crs)
```

```
0    563.134380
4    353.232132
8    230.135653
1    218.369949
3    216.871511
5    164.137548
2    127.615396
7    119.742279
6     98.469632
dtype: float64
{'init': 'epsg:3857'}
```

VISUALIZING GEOSPATIAL DATA IN PYTHON

Let's Practice!

VISUALIZING GEOSPATIAL DATA IN PYTHON

GeoSeries attributes and methods II

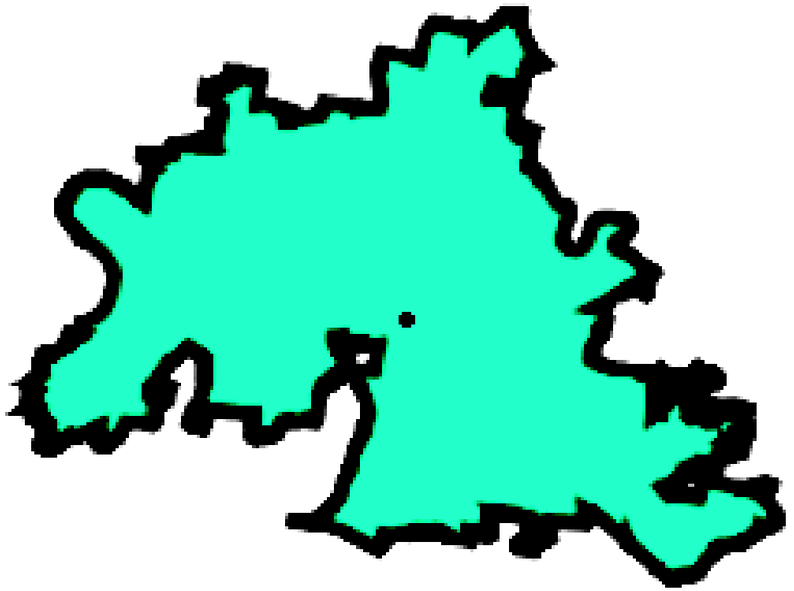
Mary van Valkenburg
Data Science Program Manager, Nashville Software School

GeoSeries.centroid

- returns the point at the center of each geometry in a GeoSeries

```
# centroid of first polygon
print(districts.geometry.centroid[0])
```

```
Point(-87.256 36.193)
```



School district centroids

GEOSERIES.CENTROID

```
# print the first 5 rows of school districts
print(school_districts.head())
```

```
first_name  last_name  district  geometry
Sharon      Gentry        1      (POLYGON ((-86.771 36.383...
Jill        Speering      3      (POLYGON ((-86.753 36.404...
Jo Ann      Brannon       2      (POLYGON ((-86.766 36.083...
Anna        Shepherd      4      (POLYGON ((-86.580 36.209...
Amy         Frogge        9      (POLYGON ((-86.972 36.208...
```

School district centroids

```
# create 'center` column from the centroid
school_districts['center'] = school_districts.geometry.centroid
```

```
# create GeoDataFrame with districts and centers
part = ['district', 'center']
school_district_centers = school_districts[part]
school_district_centers.head()
```

```
district          center
1                 POINT (-86.86086595994405 36.2628221811899)
3                 POINT (-86.72361421487962 36.28515517790142)
2                 POINT (-86.70156420691957 36.03021153030475)
4                 POINT (-86.63964402189863 36.19696692376599)
9                 POINT (-86.95428425398846 36.10392411644131)
```

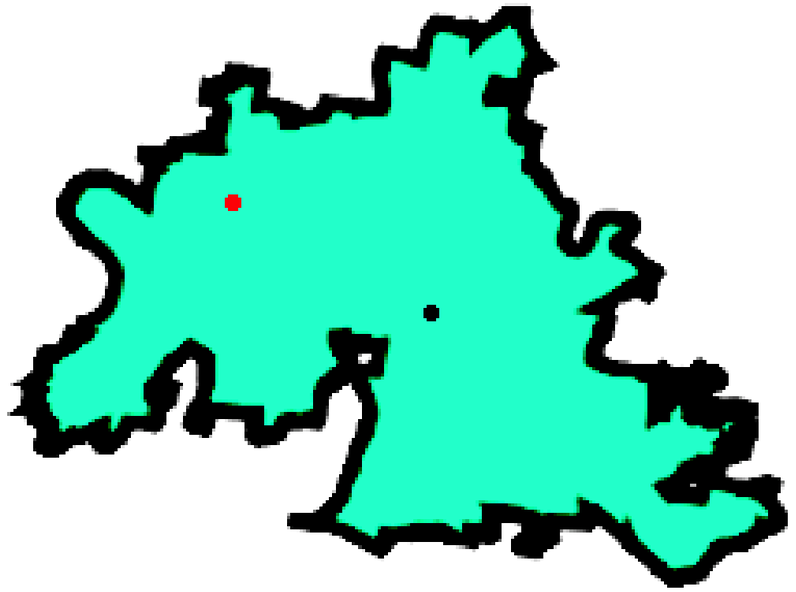
GeoSeries.distance()

- `GeoSeries.distance(other)` -

  returns minimum distance to `other`

```
# distance from red_pt to centroid
cen = districts.geometry.centroid[0]
print(red_pt.distance(other = cen))
```

```
24.273
```



Distance between two points

GEOSERIES.DISTANCE(OTHER)

```
district_one = school_districts.loc[school_districts.district == '1']
district_one.head()
```

```
first_name  last_name  district center                    geometry
Sharon      Gentry     1        POINT (-86.860 36.262)   (POLYGON ((-86.771...
```

Distance between two points

```
schools.head()
```

```
name                  lat     lng
AZ Kelley Elem        36.021  -86.658
Alex Green Elem       36.252  -86.832
Amqui Elem            36.27   -86.703
Andrew Jackson Elem   36.231  -86.623
Antioch High School   36.04   -86.599
```

```
# create geometry in schools
schools['geometry']=schools.apply(lambda x: Point((x.lng, x.lat)), axis=1)

# define crs
s_crs=district_one.crs

#construct schools GeoDataFrame
school_geo=gpd.GeoDataFrame(schools,crs = s_crs,geometry = schools.geometry)
```

Distance between two points

```
# spatial join schools within dist 1
schools_in_dist1 = gpd.sjoin(schools_geo, district_one, op = 'within')
schools_in_dist1.shape
```

```
(30, 8)
```

Distance between two points

```
# import pprint to format dictionary output
import pprint

distances = {}
for row in schools_in_dist1.iterrows():
    vals = row[1]
    key = vals['name']
    ctr = vals['center']
    distances[key] = vals['geometry'].distance(ctr)

pprint.pprint(distances)
```

```
{'Alex Green Elementary': 0.03028717271968277 3,
 'Bellshire Elementary': 0.0988045140909651,
 'Brick Church College Prep': 0.08961013862715599,
 'Buena Vista Elementary': 0.10570511270825833,
 'Cockrill Elementary': 0.1077685612196105,
.....
```

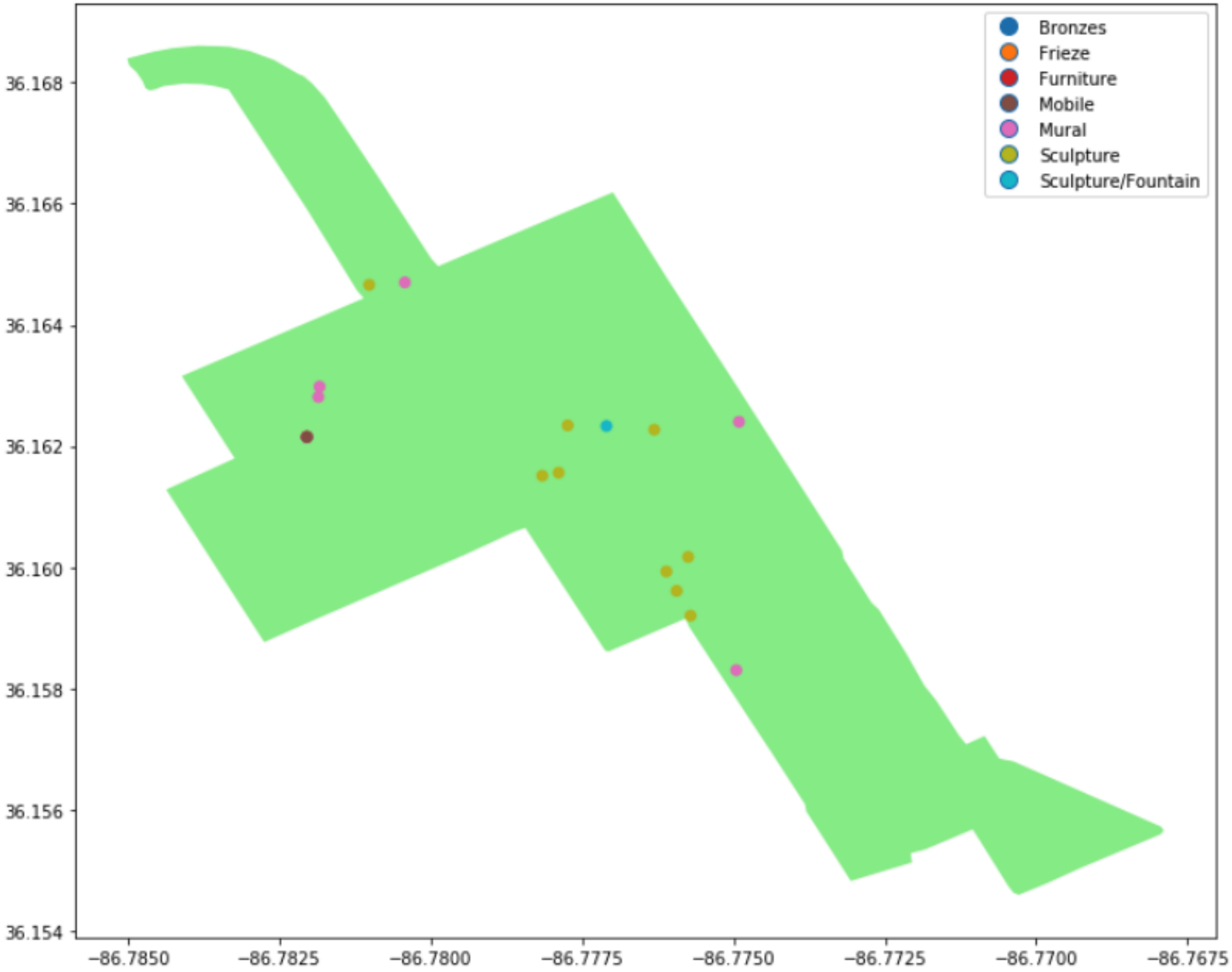
VISUALIZING GEOSPATIAL DATA IN PYTHON

Let's Practice!

VISUALIZING GEOSPATIAL DATA IN PYTHON

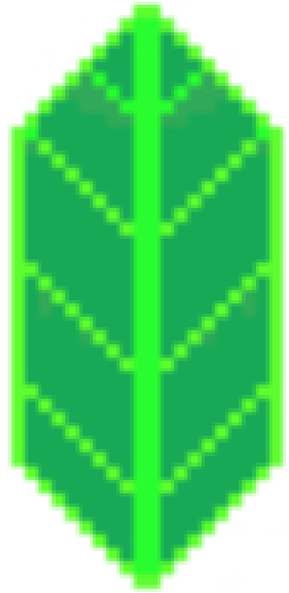
Working with Folium

Mary van Valkenburg
Data Science Program Manager, Nashville Software School


Bronzes
Frieze
Furniture
Mobile
Mural
Sculpture
Sculpture/Fountain
36.168
36.166
36.164
36.162
36.160
36.158
36.156
36.154
−86.7850
−86.7825
−86.7800
−86.7775
−86.7750
−86.7725
−86.7700
−86.7675

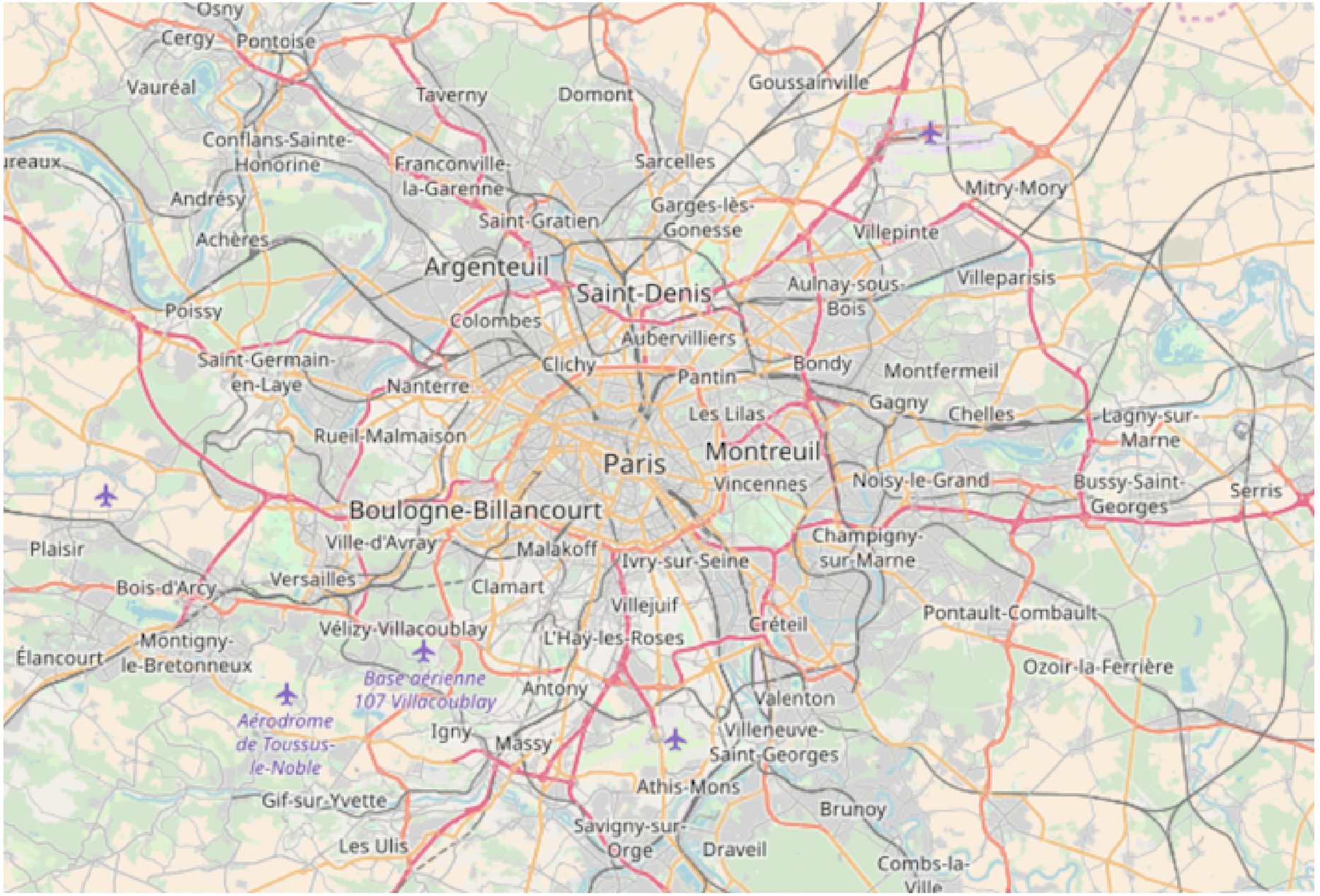
Folium

- python package
- interactive maps
- built upon Leaflet.js



folium.Map()

```
import folium
# construct a map centered at the Eiffel Tower
eiffel_tower = folium.Map(location = [48.8583736,2.2922926])

# display the map
display(eiffel_tower)
```


Osny
Cergy
Pontoise
Vauréal
Taverny
Domont
Goussainville
Conflans-Sainte-Honorine
Franconville-la-Garenne
Sarcelles
Mitry-Mory
Andrésy
Garges-lès-Gonesse
Saint-Gratien
Villepinte
Achères
Argenteuil
Saint-Denis
Aulnay-sous-Bois
Villeparisis
Poissy
Colombes
Aubervilliers
Saint-Germain-en-Laye
Clichy
Pantin
Bondy
Montfermeil
Nanterre
Gagny
Chelles
Les Lilas
Lagny-sur-Marne
Rueil-Malmaison
Paris
Montreuil
Vincennes
Noisy-le-Grand
Bussy-Saint-Georges
Serris
Boulogne-Billancourt
Champigny-sur-Marne
Plaisir
Ville-d'Avray
Malakoff
Ivry-sur-Seine
Bois-d'Arcy
Versailles
Clamart
Villejuif
Pontault-Combault
Vélizy-Villacoublay
Créteil
L'Haÿ-les-Roses
Montigny-le-Bretonneux
Élancourt
Base aérienne 107 Villacoublay
Ozoir-la-Ferrière
Antony
Valenton
Aérodrome de Toussus-le-Noble
Igny
Massy
Villeneuve-Saint-Georges
Athis-Mons
Gif-sur-Yvette
Brunoy
Savigny-sur-Orge
Les Ulis
Draveil
Combs-la-Ville

Setting the zoom level

```
import folium
# construct a map centered at the Eiffel Tower
eiffel_tower = folium.Map([location = 48.8583736,2.2922926], zoom_start = 12)

# display the map
display(eiffel_tower)
```


Avenue Georges Mandel
Rue Scheffer
Trocadéro
Avenue de New York
Passerelle Debilly
La Seine
Pont de l'Alma
Bateaux-Mouches
Quai d'Orsay
Quai Branly
Pont de l'Alma
Jardins du Trocadéro
Rue Vineuse
Rue de la Tour
Rue de l'Université
Rue Malar
Rue Cognacq-Jay
Gros-Caillou
Avenue Rapp
Rue Saint-Dominique
Rue Augereau
Avenue Bosquet
Rue Cler
Rue de Grenelle
Rue Amélie
Rue Dupleix
Passy
Rue Raynouard
Avenue du Président Kennedy
Promenade d'Australie
Champ de Mars -Tour Eiffel
Avenue de Suffren
Rue Jean Rey
Avenue Pierre Lou
Allée Adrienne Lecouvreur
Avenue Emile Deschanel
Allée Thomy Th
Avenue Anatole Fra
Rue d'Ankara

Folium location from centroid

```
district_one.head()
```

```
district  center                    geometry
1         POINT (-86.860 36.262)    (POLYGON ((-86.771 36.383...
```

```
center_point = district_one.center[0]
type(center_point)
```

```
<class 'shapely.geometry.point.Point'>
```

Folium location from centroid

```
# reverse the order for folium location array
district_center = [center_point.y, center_point.x]

# print center point and district_center
print(center_point)
print(district_center)
```

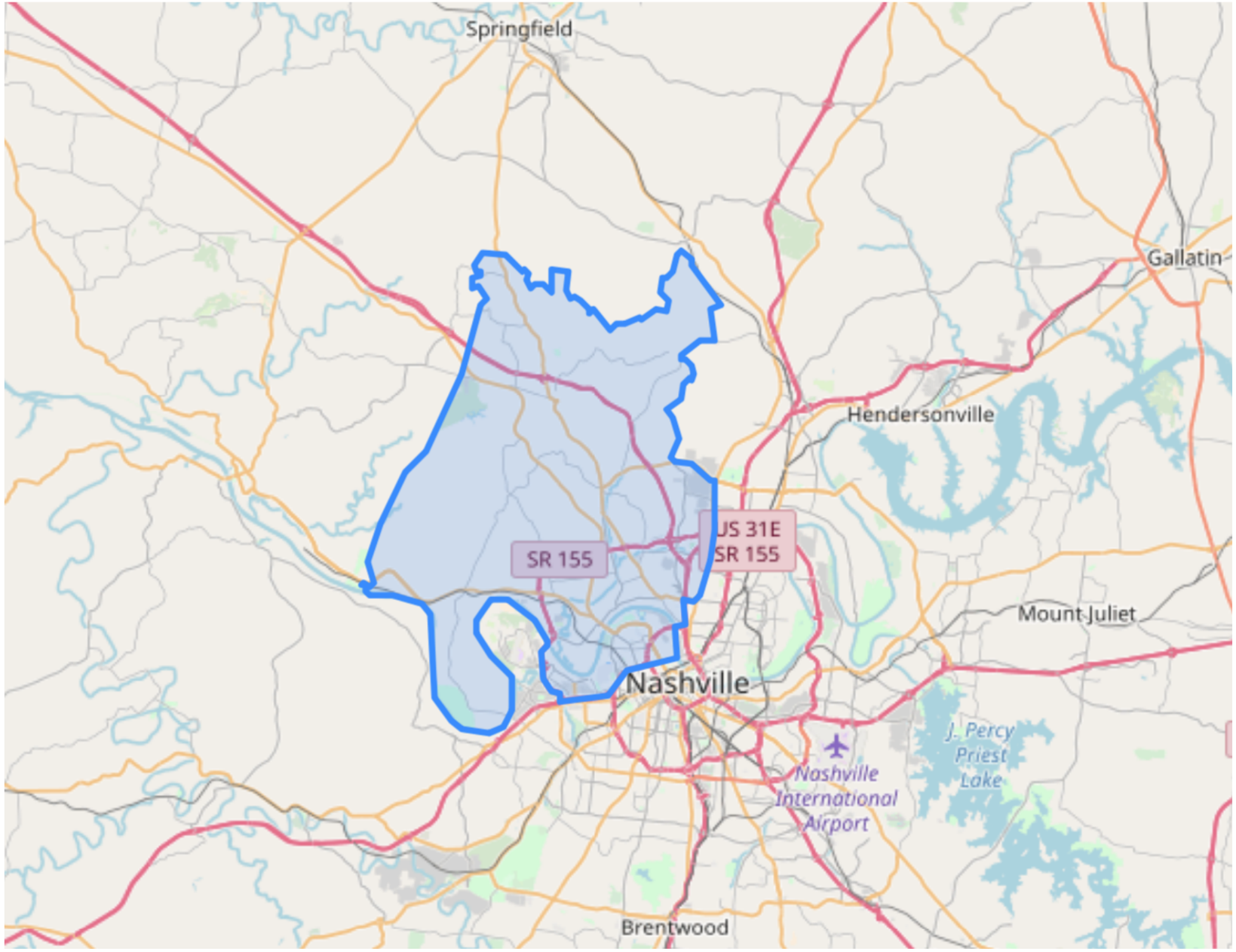
```
POINT (-86.86086595994405 36.2628221811899)
[36.262822181189904, -86.86086595994405]
```

Adding a polygon to a folium map

```
# create a folium map centered on district 1
district1_map = folium.Map(location = district_center)
```

```
# add the outline of district one
folium.GeoJson(district_one.geometry).add_to(district1_map)
```

```
# display the resulting map
display(district1_map)
```


Springfield
Gallatin
Hendersonville
US 31E
SR 155
SR 155
Mount Juliet
Nashville
J. Percy
Priest
Lake
Nashville
International
Airport
Brentwood

VISUALIZING GEOSPATIAL DATA IN PYTHON

Let's practice!

VISUALIZING GEOSPATIAL DATA IN PYTHON

Creating markers and popups in folium

Mary Van Valkenburg
Data Science Program Manager, Nashville Software School

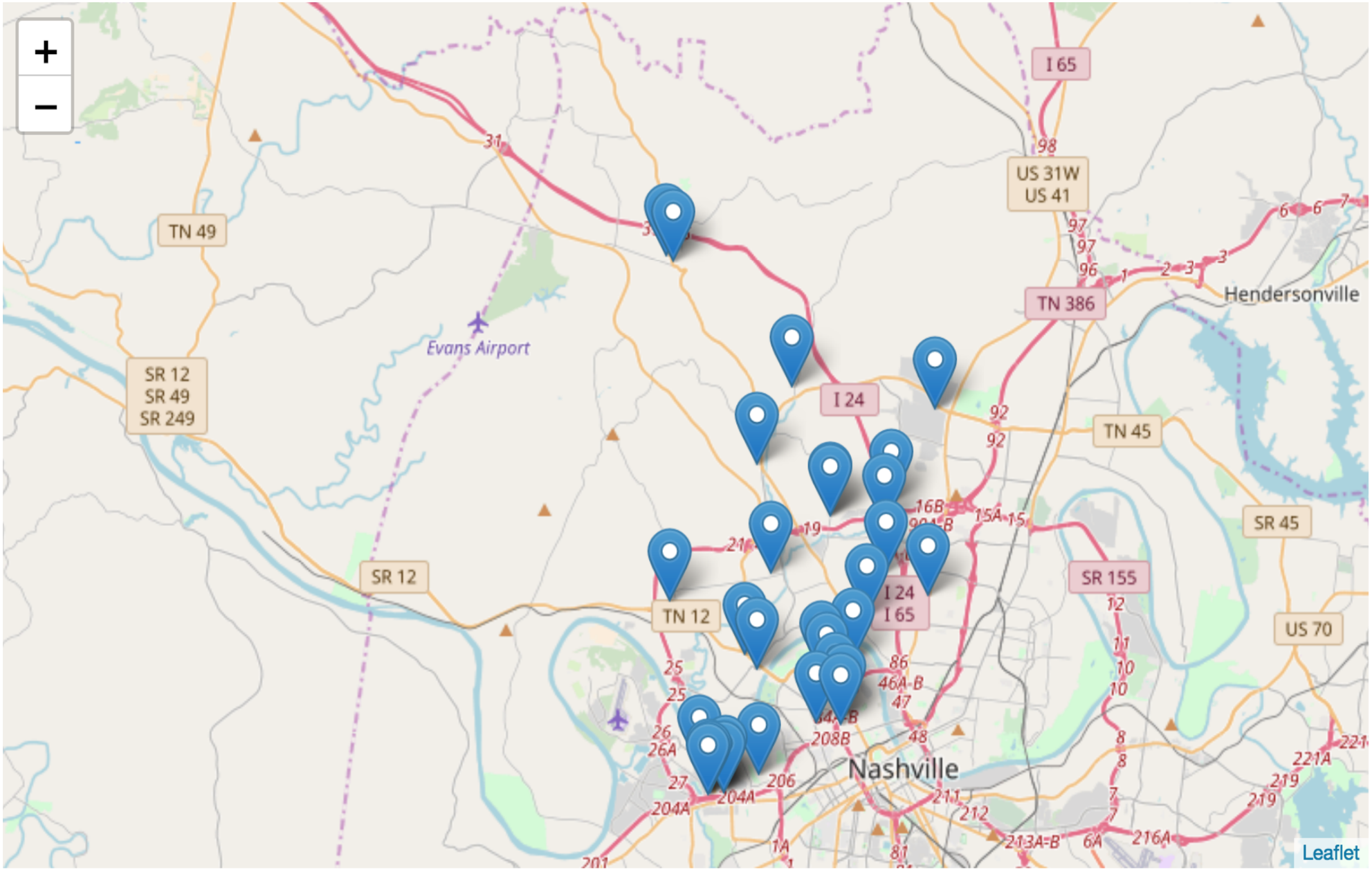
Using iterrows()

```
for row in schools_in_dist1.iterrows():
    row_values = row[1]
    print(row_values)
```

```
name                                 Alex Green Elementary
lat                                                   36.253
lng                                                 -86.8322
geometry                  POINT (-86.8322292 36.2529607)
district                                                 1
center      POINT (-86.86086595994405 36.2628221811899)
Name: 1, dtype: object
name                                  Bellshire Elementary
lat                                                  36.2697
lng                                                 -86.7623
geometry                POINT (-86.76230026 36.26968766)
district                                                 1
center      POINT (-86.86086595994405 36.2628221811899)
Name: 8, dtype: object
```

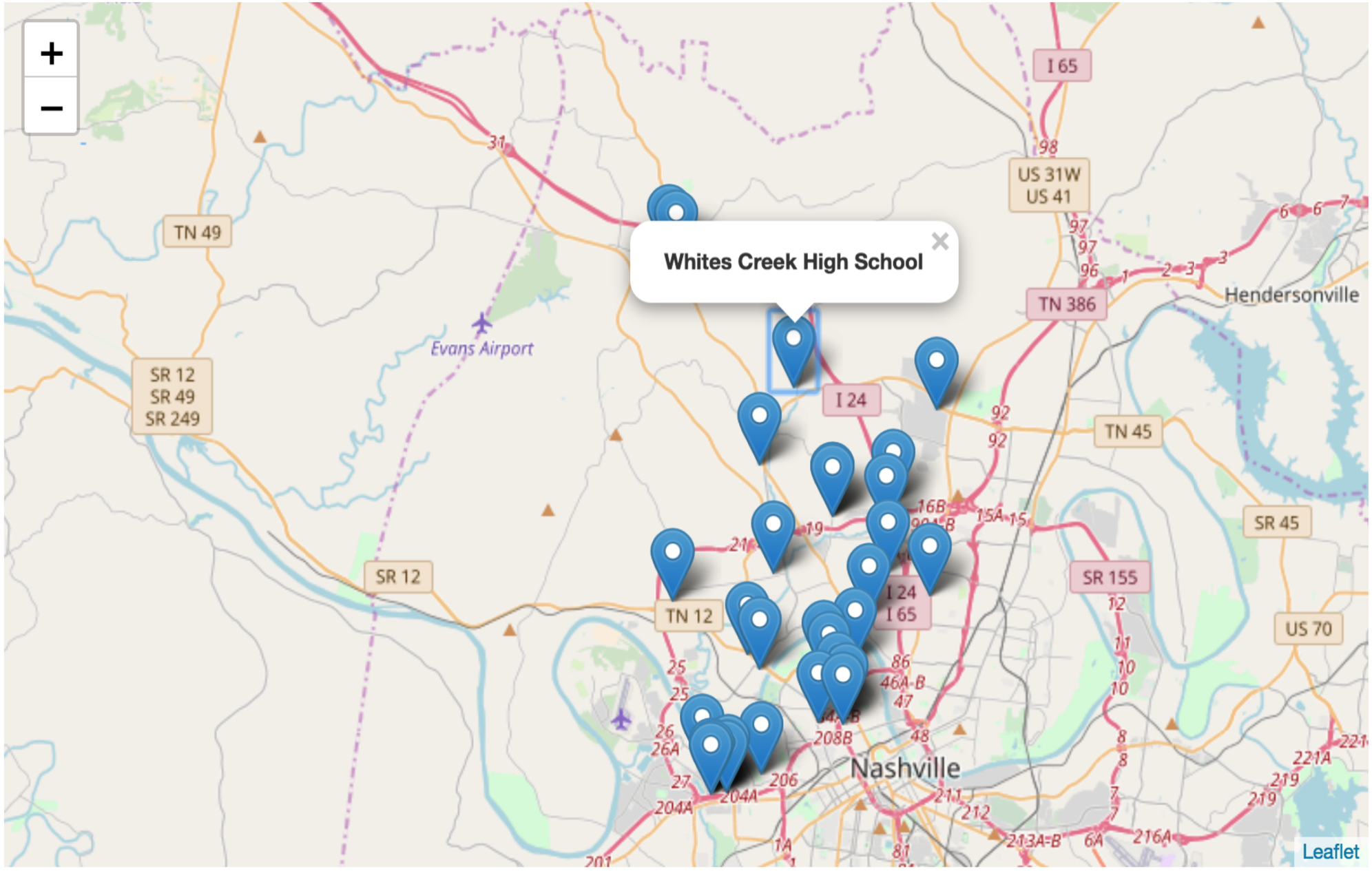
Building marker locations

```
# Construct a folium map for school district 1
district1_map = folium.Map(location = district_center, zoom_start = 11)

#create a marker for each school
for row in schools_in_dist1.iterrows():
    row_values = row[1]
    location = [row_values['lat'], row_values['lng']]
    marker = folium.Marker(location = location)
    marker.add_to(district1_map)

display(district1_map)
```

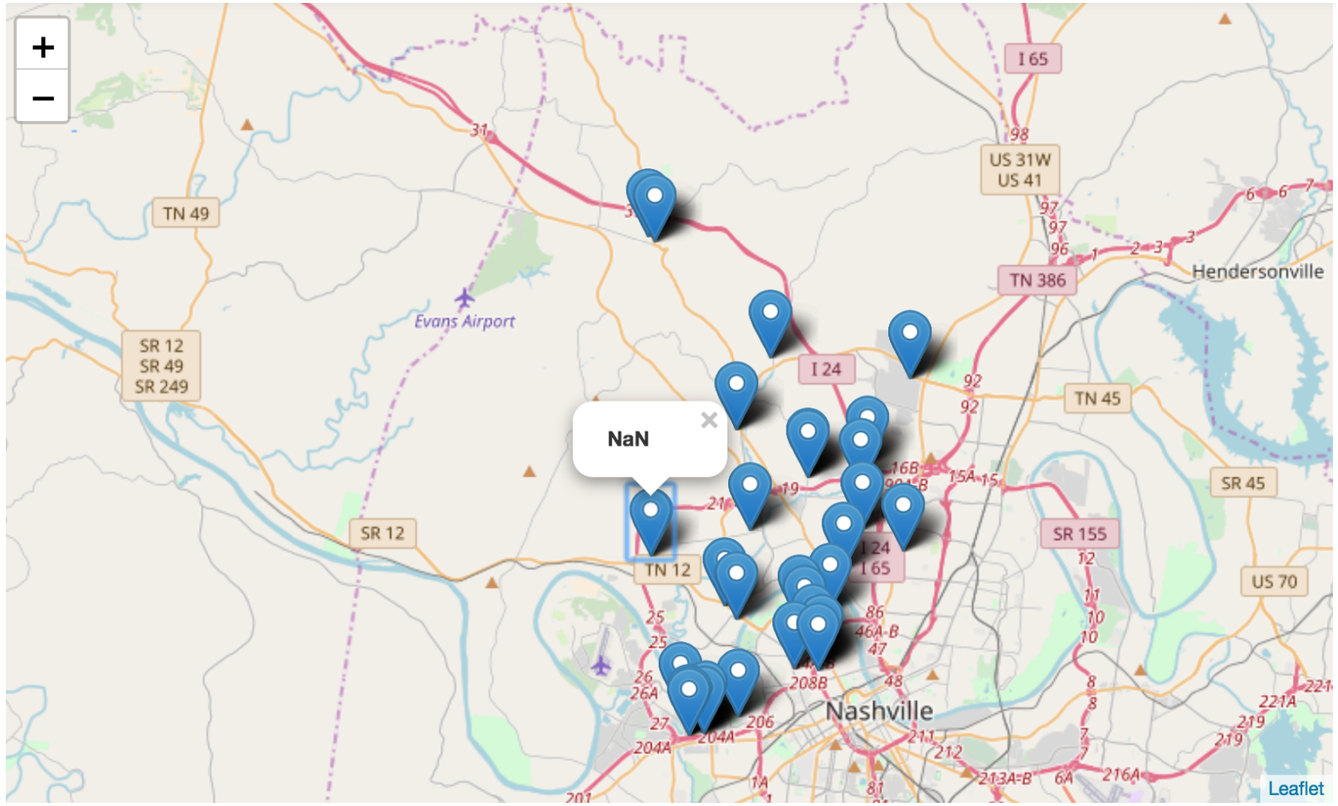

+
−
I 65
US 31W
US 41
TN 49
TN 386
Hendersonville
Evans Airport
SR 12
SR 49
SR 249
I 24
TN 45
SR 45
SR 12
SR 155
TN 12
I 24
I 65
US 70
Nashville
Leaflet

Creating popups from data in a DataFrame

```
# Construct a folium map for school district 1
district1_map = folium.Map(location = district_center, zoom_start = 11)

# Create a marker for each school
for row in schools_in_dist1.iterrows():
    row_values = row[1]
    location = [row_values['lat'], row_values['lng']]
    popup = popup = '<strong>' + row_values['name'] + '</strong>'
    marker = folium.Marker(location = location, popup = popup)
    marker.add_to(district1_map)

display(district1_map)
```


Whites Creek High School
Evans Airport
Nashville
Hendersonville
I 65
US 31W
US 41
TN 386
TN 49
SR 12
SR 49
SR 249
I 24
TN 45
SR 45
SR 155
TN 12
US 70
Leaflet

Troubleshooting popups



VISUALIZING GEOSPATIAL DATA IN PYTHON

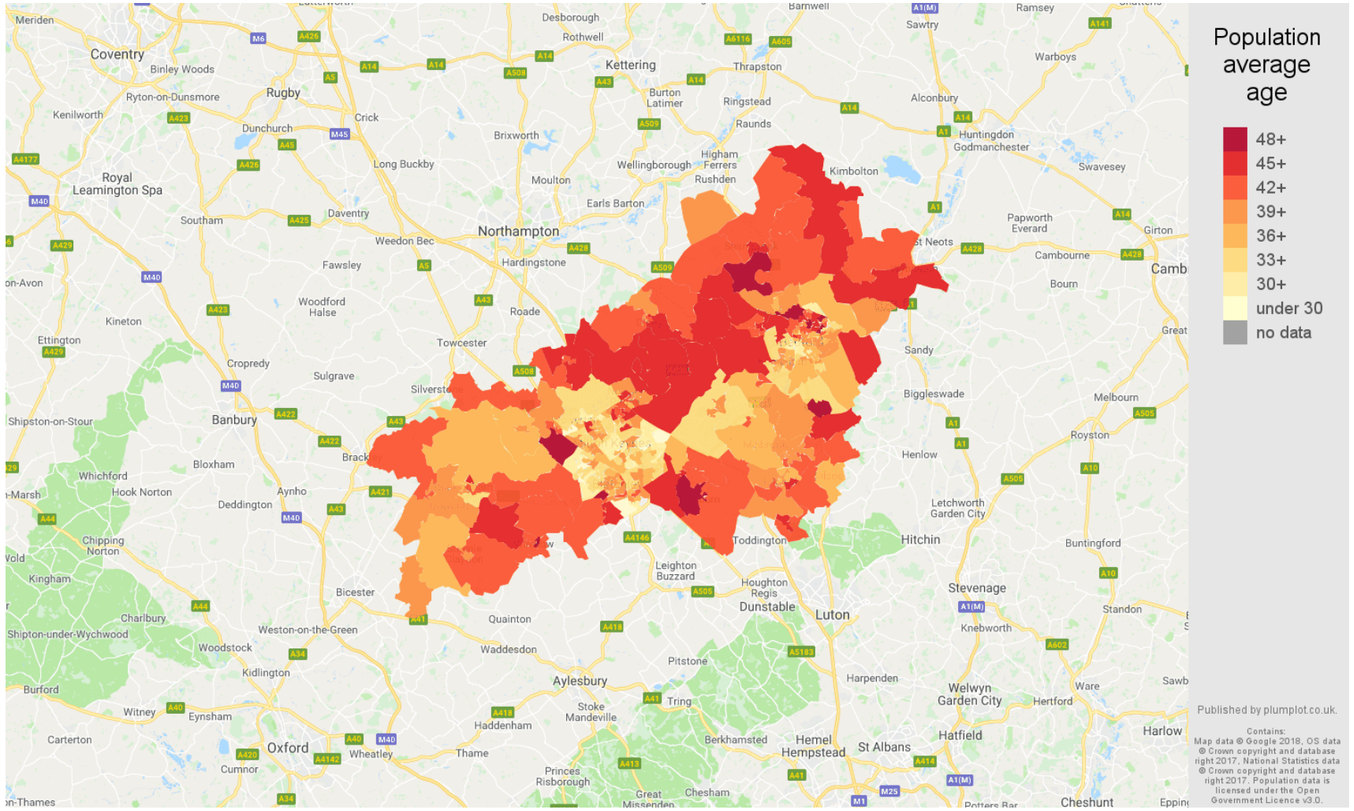
Let's practice!

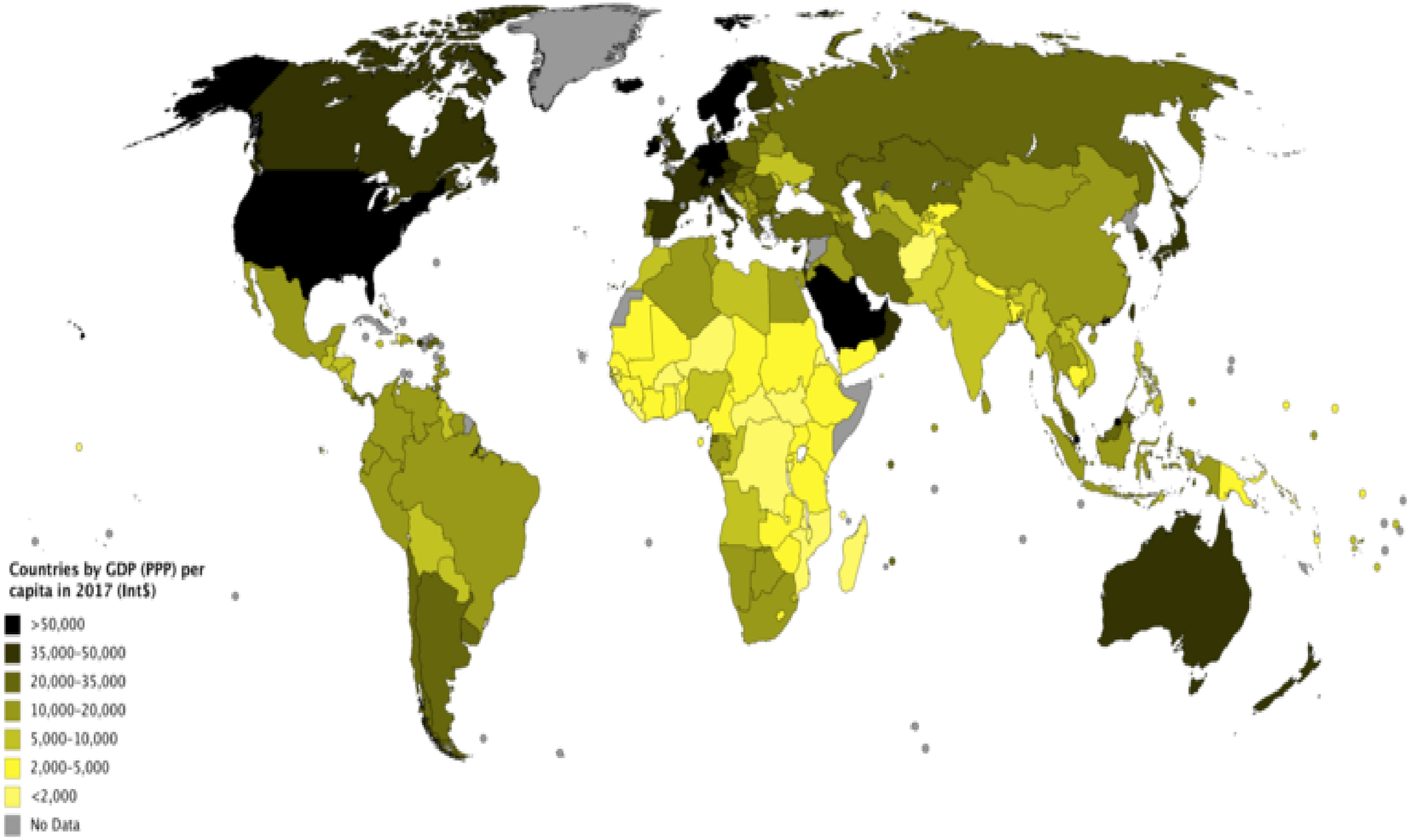
VISUALIZING GEOSPATIAL DATA IN PYTHON

What is a choropleth?

Mary van Valkenburg
Data Science Program Manager, Nashville Software School

Definition of a choropleth




Countries by GDP (PPP) per capita in 2017 (Int$)
>50,000
35,000-50,000
20,000-35,000
10,000-20,000
5,000-10,000
2,000-5,000
<2,000
No Data

Density

```
schools_in_districts.head(2)
```

```
district  geometry                    name                lat    lng
1         (POLYGON ((-86.77 36.38... Nashville Prep      36.16 -86.85
1         (POLYGON ((-86.77 36.38... Rocketship Prep     36.17 -86.79
```

Get counts

```
school_counts = schools_in_districts.groupby(['district']).size()
print(school_counts)
```

```
district
1    30
2    11
3    19
4    18
5    36
6    21
7    13
8    10
9    12
dtype: int64
```

Add counts

```
school_counts_df = school_counts.to_frame()
school_counts_df.reset_index(inplace=True)
school_counts_df.columns = ['district', 'school_count']
```

```
districts_with_counts = pd.merge(school_districts, school_counts_df,
                                 on = 'district')
districts_with_counts.head(2)
```

```
district geometry            school_count
1        (POLYGON ((-86.77 36.38...  30
3        (POLYGON ((-86.75 36.40...  19
```

Divide counts by area

```
districts_with_counts['area'] = districts_with_counts.geometry.area
```

```
districts_with_counts['school_density'] = districts_with_counts.apply(
    lambda row: row.school_count/row.area, axis = 1)

districts_with_counts.head(2)
```

```
district geometry            school_count   area     school_density
1        (POLYGON ((-86.77 36.38...    30     0.036641    818.745403
3        (POLYGON ((-86.75 36.40...    19     0.014205    1337.594495
```

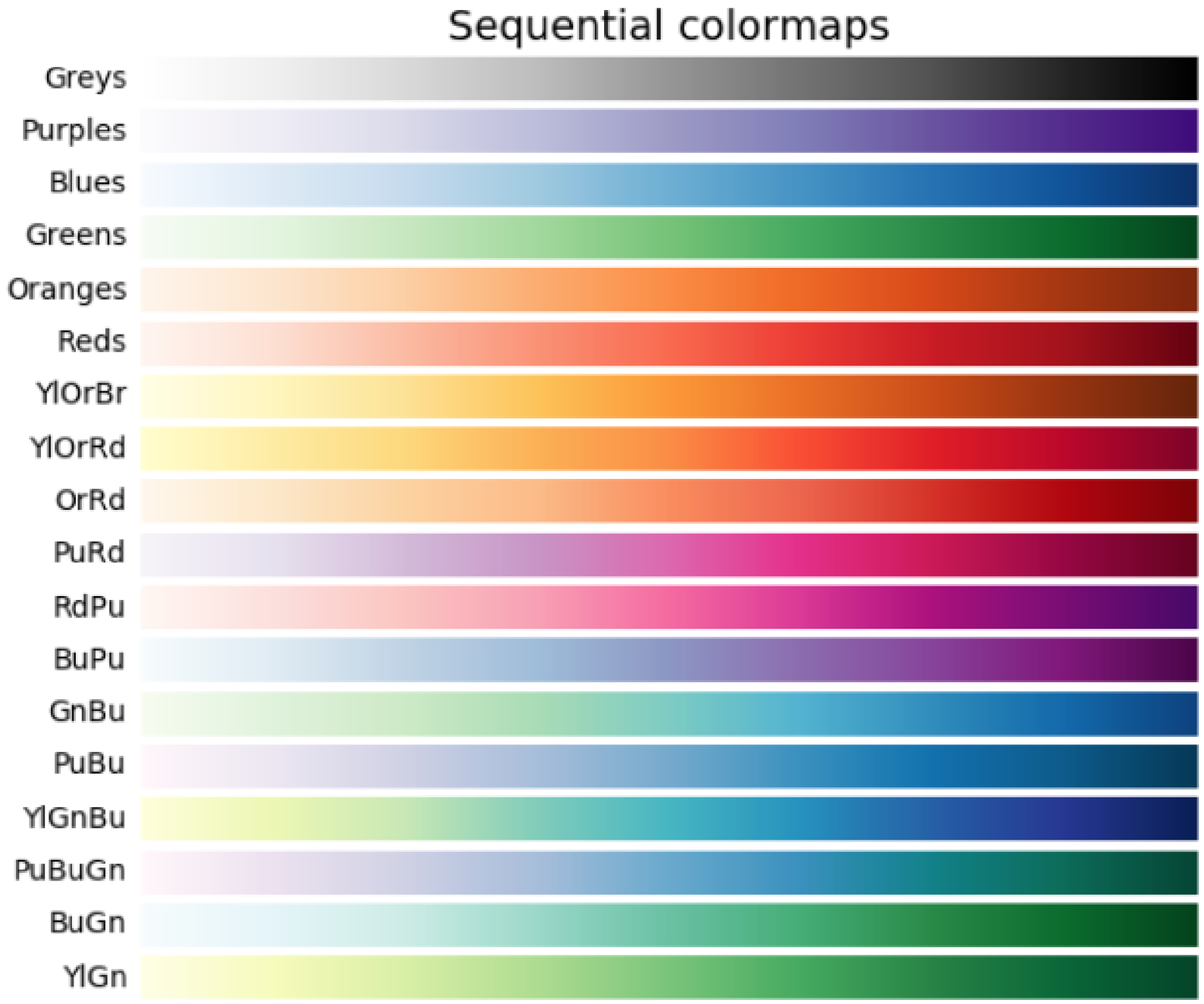
VISUALIZING GEOSPATIAL DATA IN PYTHON

Let's Practice!

VISUALIZING GEOSPATIAL DATA IN PYTHON

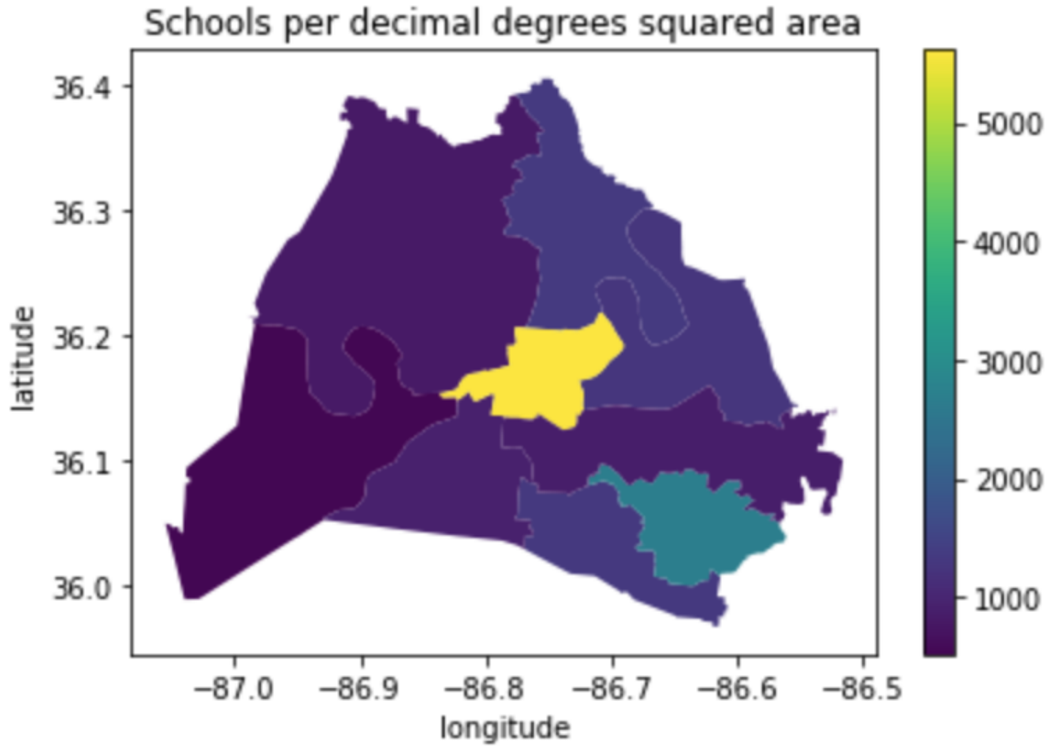
Choropleths with geopandas

Mary van Valkenburg
Data Science Program Manager, Nashville Software School


Sequential colormaps
Greys
Purples
Blues
Greens
Oranges
Reds
YlOrBr
YlOrRd
OrRd
PuRd
RdPu
BuPu
GnBu
PuBu
YlGnBu
PuBuGn
BuGn
YlGn

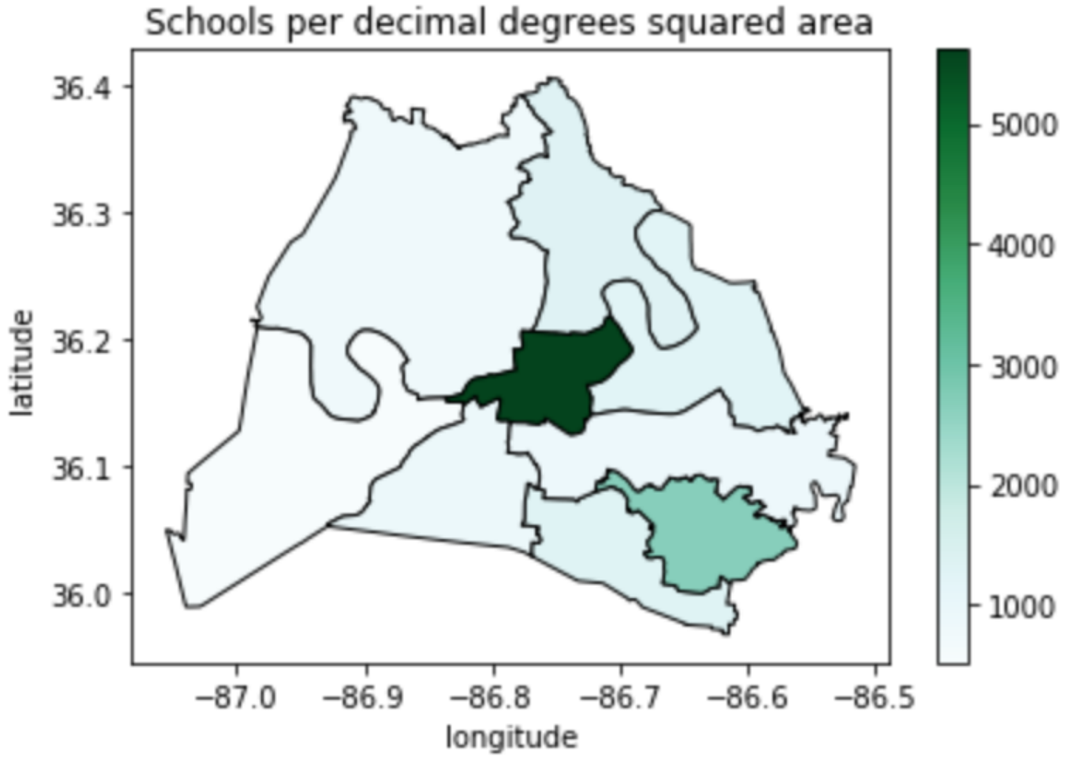
Choropleth with GeoDataFrame.plot()

```
districts_with_counts.plot(column = 'school_density', legend = True)
plt.title('Schools per decimal degrees squared area')
plt.xlabel('longitude')
plt.ylabel('latitude');
```



Choropleth with GeoDataFrame.plot()

```
districts_with_counts.plot(column = 'school_density', cmap = 'BuGn',
                           edgecolor = 'black', legend = True)
plt.title('Schools per decimal degrees squared area')
plt.xlabel('longitude')
plt.ylabel('latitude');
```



Area in Kilometers Squared

```
# starting CRS
print(school_districts.crs)
```

```
{'init': 'epsg:4326'}
```

```
# convert to EPSG 3857
school_districts = school_districts.to_crs(epsg = 3857)
print(school_districts.crs)
```

```
{'init': 'epsg:3857', 'no_defs': True}
```

Area in Kilometers Squared

```
# define a variable for m^2 to km^2
sqm_to_sqkm = 10**6

school_districts['area'] = school_districts.geometry.area / sqm_to_sqkm
school_districts.head(2)
```

```
district    geometry                                area
1          (POLYGON ((-965.055 4353528.766...    563.134380
3          (POLYGON ((-965.823 4356392.677...    218.369949
```

Latitude and longitude in decimal degrees

```
# change crs back to 4326
school_districts = school_districts.to_crs(epsg = 4326)
print(school_districts.crs)
```

```
{'init': 'epsg:4326', 'no_defs': True}
```

```
print(school_districts.head(2))
```

```
district      geometry                          area
1             (POLYGON ((-86.771 36.383...   563.134380
3             (POLYGON ((-86.753 36.404...   218.369949
```

```
# spatial join to get districts that contain schools
schools_in_districts = gpd.sjoin(school_districts,
                                 schools_geo, op = 'contains')
```

Counting schools in each district

```
# aggregate to get counts
school_counts = schools_in_districts.groupby(['district']).size()
```

```
# convert school_counts to a df
school_counts_df = school_counts.to_frame()
school_counts_df.reset_index(level=0, inplace=True)
school_counts_df.columns = ['district', 'school_count']
```

```
# merge
districts_with_counts = pd.merge(school_districts,
                                 school_counts_df, on = 'district')
```

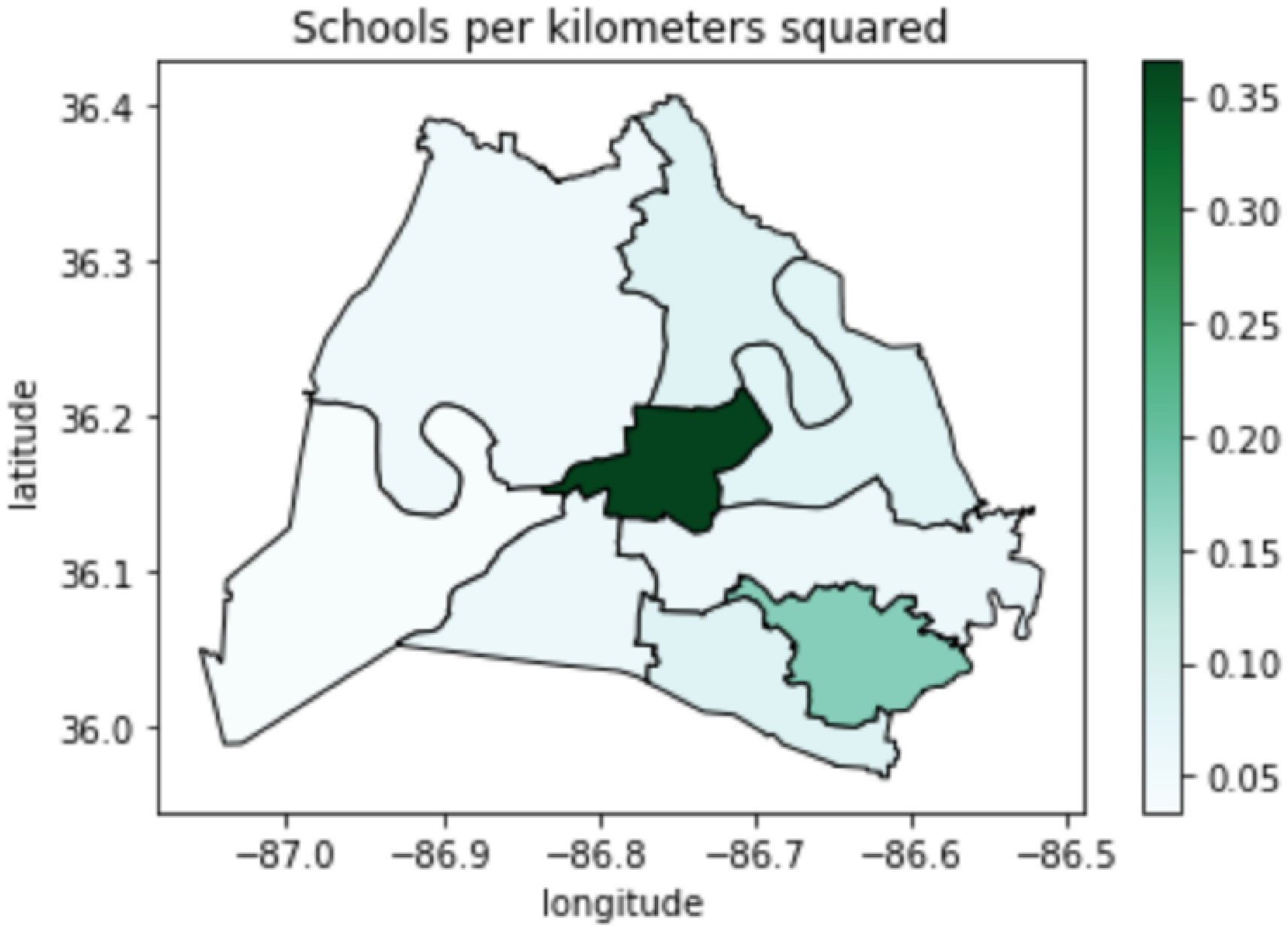
```
districts_with_counts.head(2)
```

```
district  geometry                        area        school_count
1         (POLYGON ((-86.771 36.383..     563.134380    30
3         (POLYGON ((-86.753 36.404...    218.369949    19
```

Calculating school density

```
# create school_density
districts_with_counts['school_density'] = districts_with_counts.apply(
    lambda row: row.school_count/row.area, axis = 1)
```

```
# plot it
districts_with_counts.plot(column = 'school_density', cmap = 'BuGn',
                           edgecolor = 'black', legend = True)
plt.title('Schools per kilometers squared')
plt.xlabel('longitude')
plt.ylabel('latitude');
```


Schools per kilometers squared
36.4
36.3
36.2
36.1
36.0
latitude
−87.0
−86.9
−86.8
−86.7
−86.6
−86.5
longitude
0.35
0.30
0.25
0.20
0.15
0.10
0.05

VISUALIZING GEOSPATIAL DATA IN PYTHON

Let's practice!

VISUALIZING GEOSPATIAL DATA IN PYTHON

Choropleths with folium

Mary van Valkenburg
Data Science Program Manager, Nashville Software School

folium.Map choropleth

```
# Construct a map object for Nashville
nashville = [36.1636,-86.7823]
m = folium.Map(location=nashville, zoom_start=10)

# Create a choropleth
m.choropleth(...)
```

Arguments of the folium choropleth

- `geo_data` - the source data for the polygons (geojson file or a GeoDataFrame)
- `name` - the name of the geometry column (or geojson property) for the polygons
- `data`- the source DataFrame or Series for the normalized data
- `columns`- a list of columns: one that corresponds to the polygons and one that has the value to plot

Additional arguments of the folium choropleth

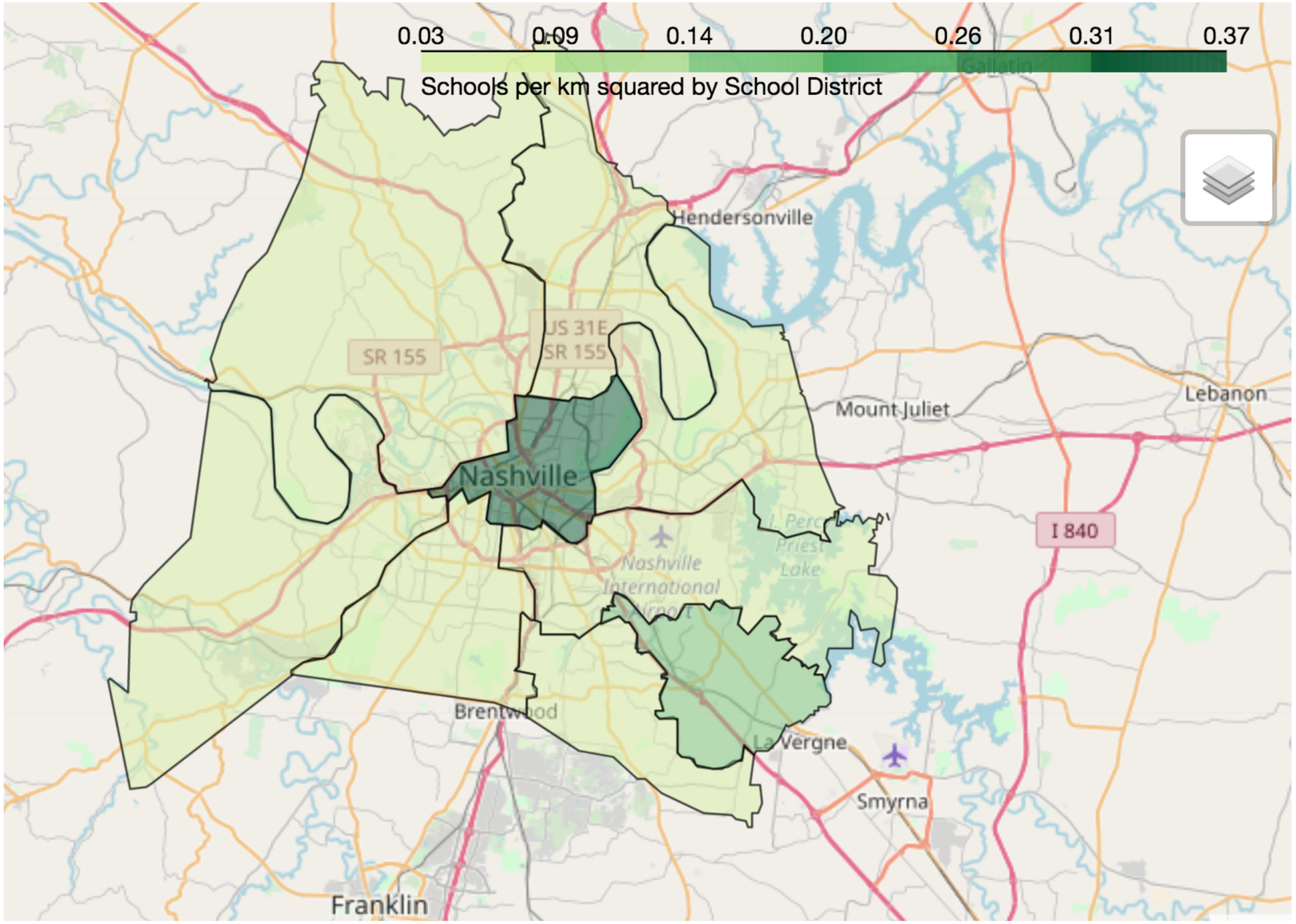
- `key_on` - a GeoJSON variable to bind the data to (always starts with `feature`)
- `fill_color` - polygon fill color (defaults to blue)
- `fill_opacity` - range between 0 (transparent) and 1 (completely opaque)
- `line_color` - color of polygon border lines (defaults to black)
- `line_opacity` - range between 0 (transparent) and 1 (completely opaque)
- `legend_name` - creates a title for the legend

Folium choropleth of school density

```
# Center point and map for Nashville
nashville = [36.1636,-86.7823]
m = folium.Map(location=nashville, zoom_start=10)
```

```
# Define a choropleth layer for the map
m.choropleth(
    geo_data=districts_with_counts,
    name='geometry',
    data=districts_with_counts,
    columns=['district', 'school_density'],
    key_on='feature.properties.district',
    fill_color='YlGn',
    fill_opacity=0.75,
    line_opacity=0.5,
    legend_name='Schools per km squared by School District'
   )
```

```
# Add layer control and display
folium.LayerControl().add_to(m)
display(m)
```


0.03
0.09
0.14
0.20
0.26
0.31
0.37
Schools per km squared by School District
Hendersonville
SR 155
US 31E
SR 155
Mount Juliet
Lebanon
Nashville
I 840
Nashville International
Priest Lake
Brentwood
La Vergne
Smyrna
Franklin

VISUALIZING GEOSPATIAL DATA IN PYTHON

Let's Practice!

VISUALIZING GEOSPATIAL DATA IN PYTHON

Congratulations!

Mary van Valkenburg
Data Science Program Manager, Nashville Software School

Skills list

- how to work with shapefiles and GeoJSON
- how to work with geometries
- how to use geopandas, shapely, and folium to extract meaning from geospatial data
- how to create beautiful and informative geospatial visualizations


WARNING!
NASHVILLE
#PeaceLoveGoodDeeds
PeaceLoveGoodDeeds.com
CASH FOR CLUES TO CRIME
EASTSIDEMURALS

VISUALIZING GEOSPATIAL DATA IN PYTHON

Goodbye