

# What is a choropleth?

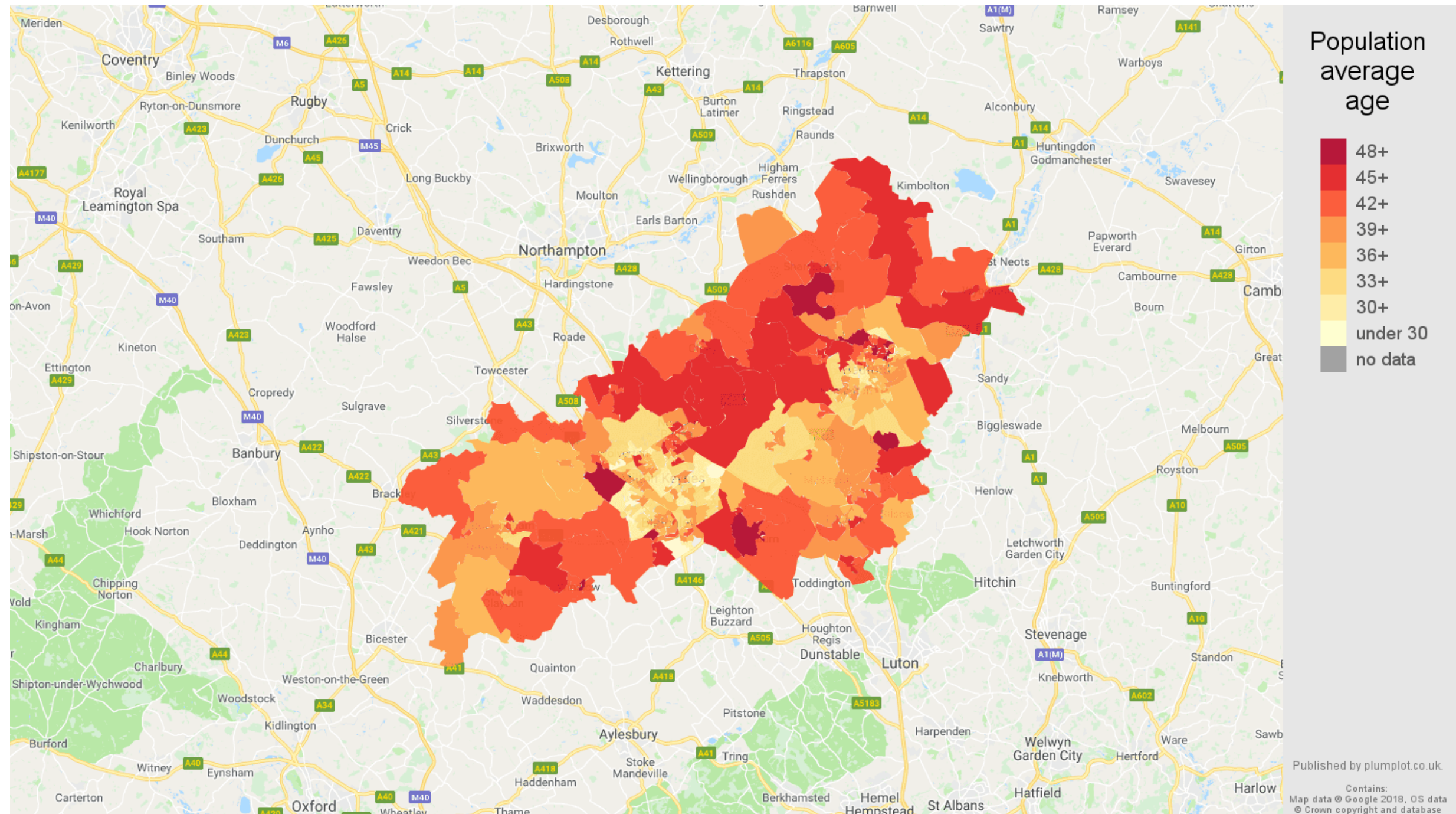
VISUALIZING GEOSPATIAL DATA IN PYTHON

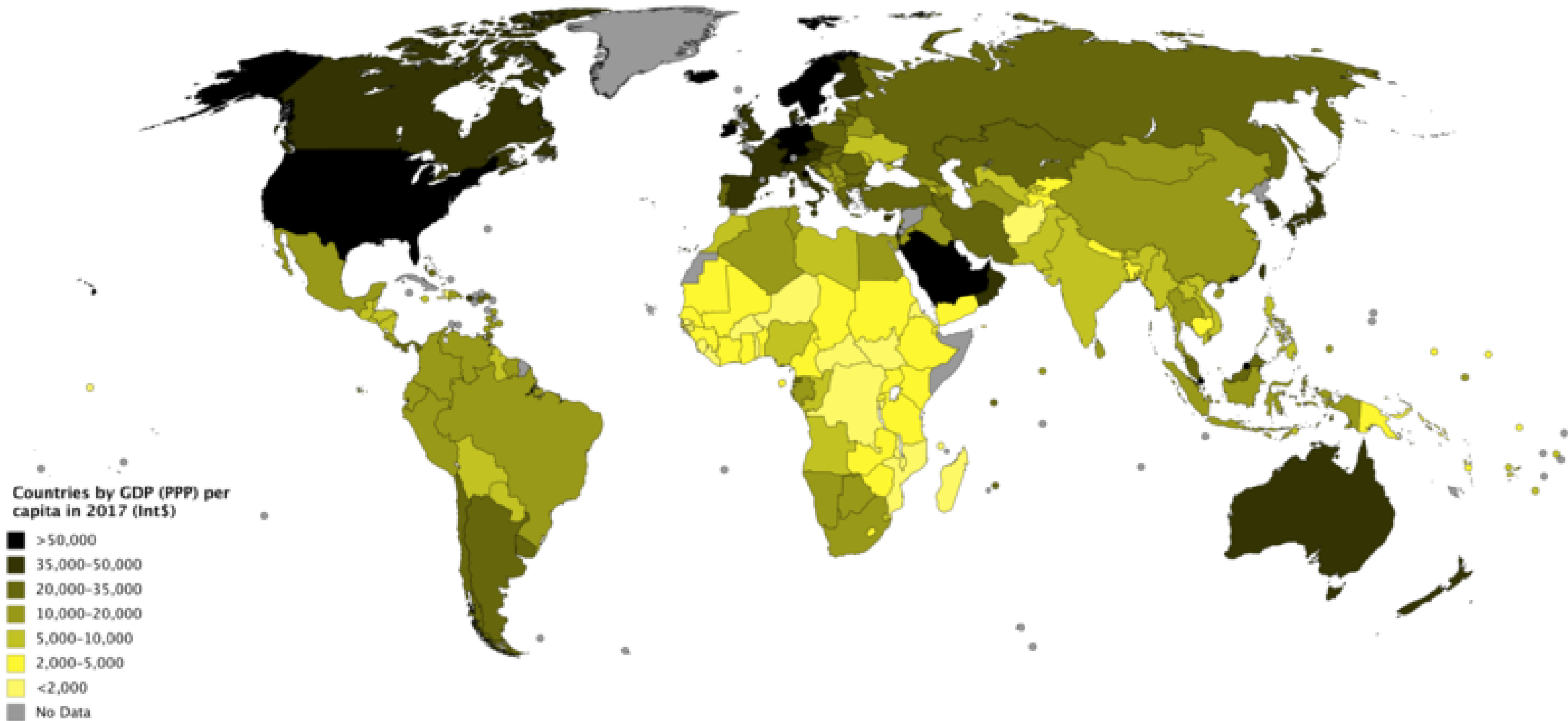


**Mary van Valkenburg**

Data Science Program Manager,  
Nashville Software School

# Definition of a choropleth





# 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 = school_counts_df.reset_index()
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 areas

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

# Let's Practice!

VISUALIZING GEOSPATIAL DATA IN PYTHON



# Choropleths with geopandas

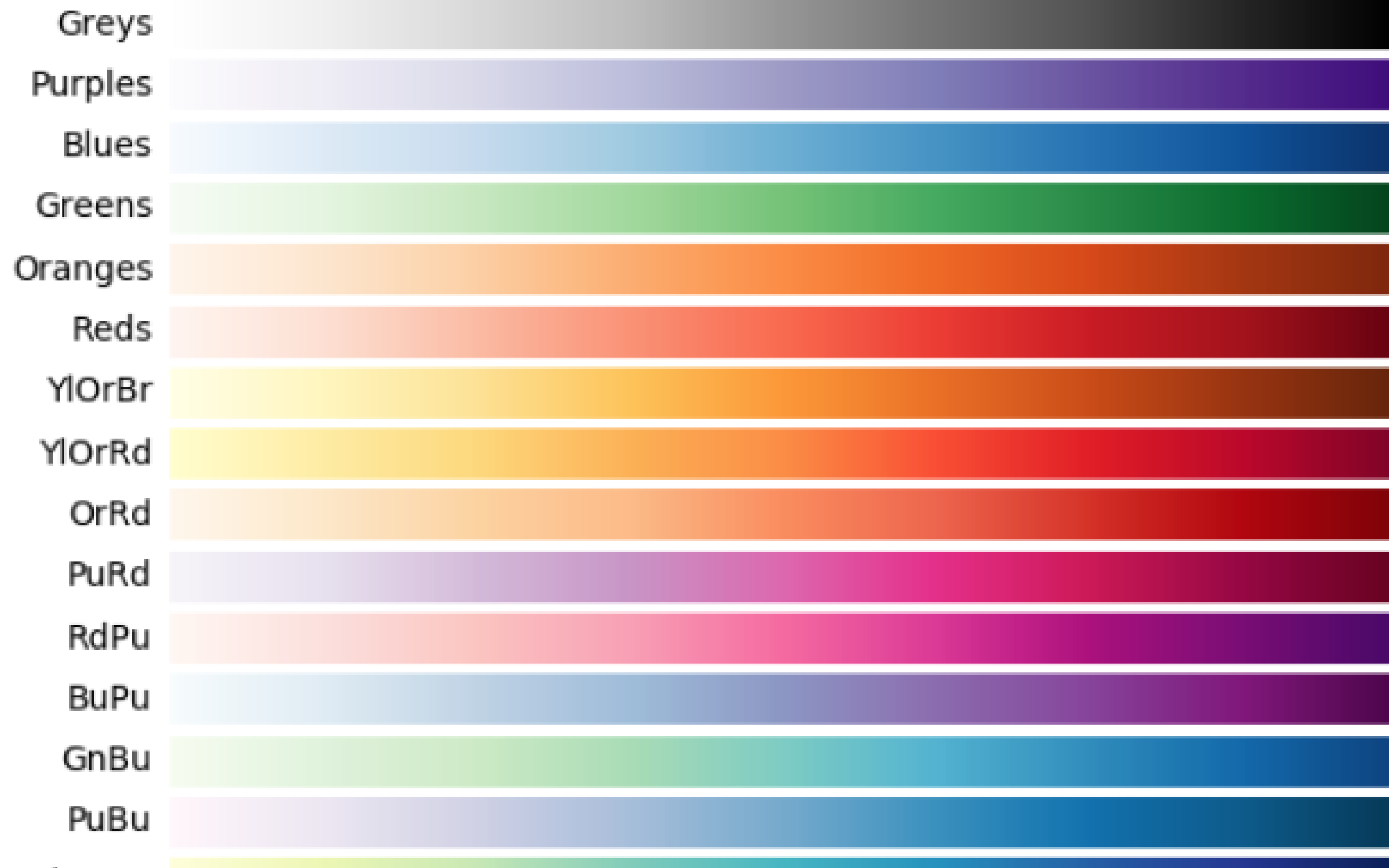
VISUALIZING GEOSPATIAL DATA IN PYTHON



**Mary van Valkenburg**

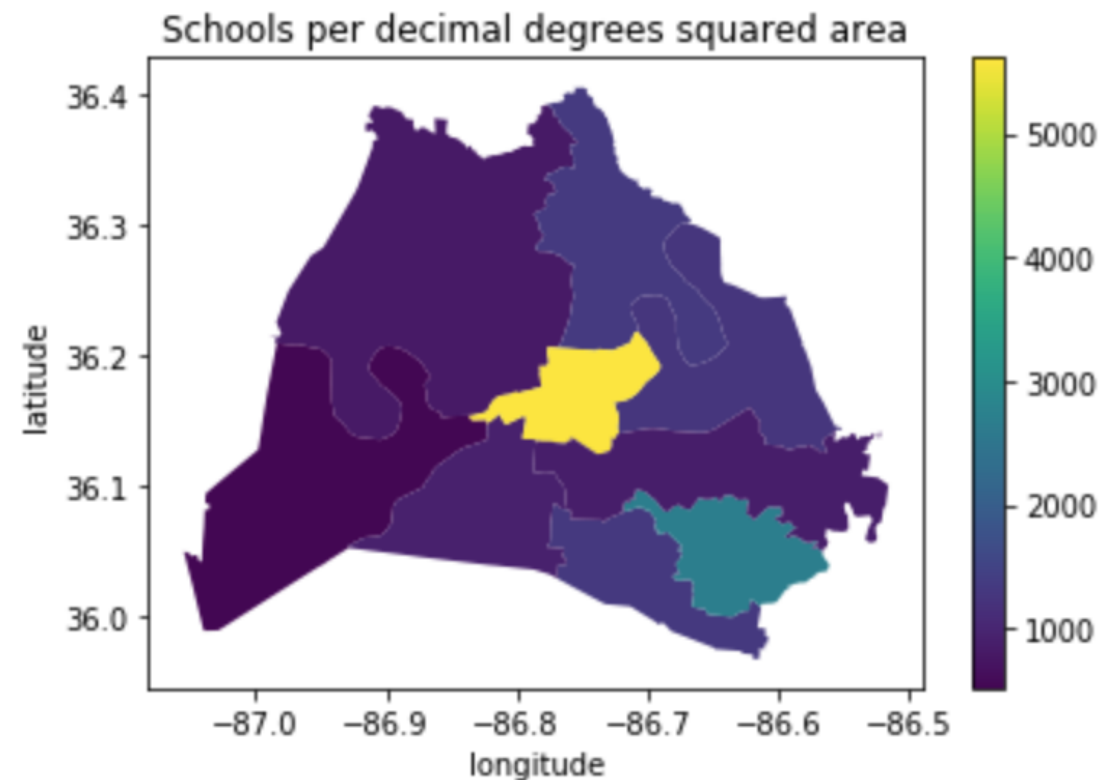
Data Science Program Manager,  
Nashville Software School

## Sequential colormaps



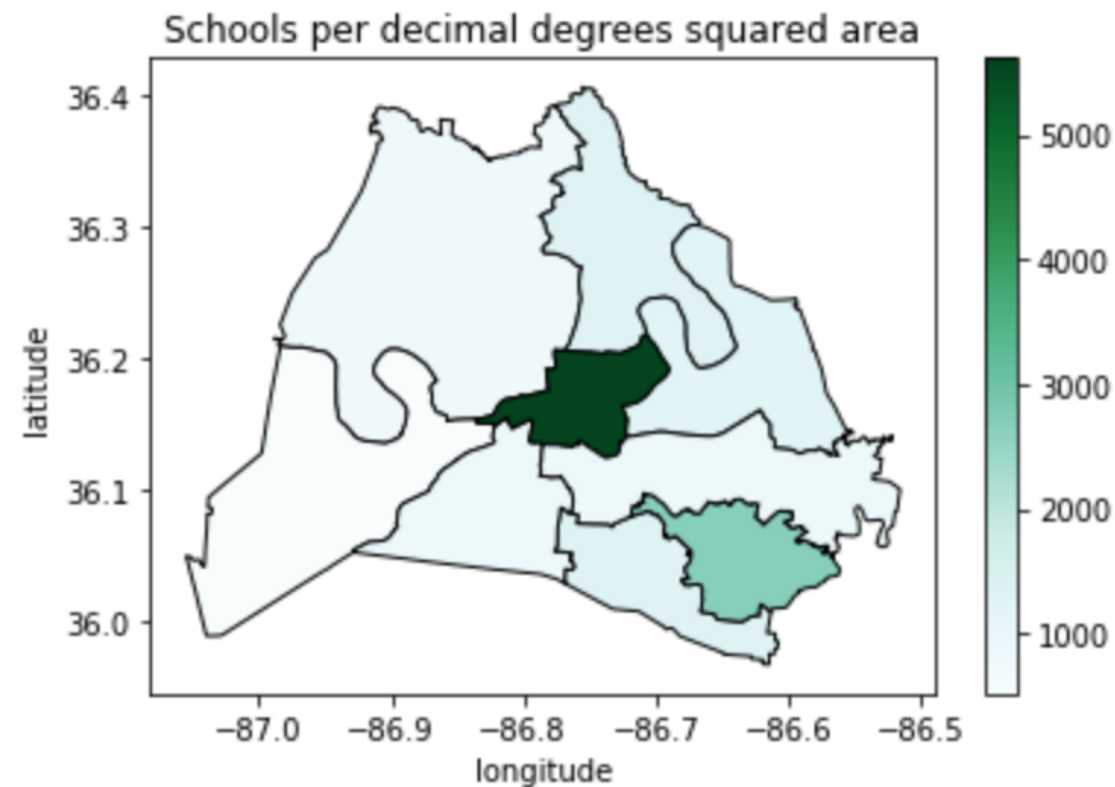
# Choropleth with GeoDataFrame.plot()

```
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)
```

```
epsg:4326
```

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

```
epsg:3857
```

# Area in Kilometers Squared

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

school_districts['area'] = school_districts.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

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

```
epsg:4326
```

```
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, predicate = 'contains')
```

```
# 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 = school_counts_df.reset_index(level=0)
school_counts_df.columns = ['district', 'school_count']
```

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

```
districts_with_counts.head(1)
```

```
district  geometry  area  school_count
1         (POLYGON ((-86.771 36.383..  563.134380    30
```

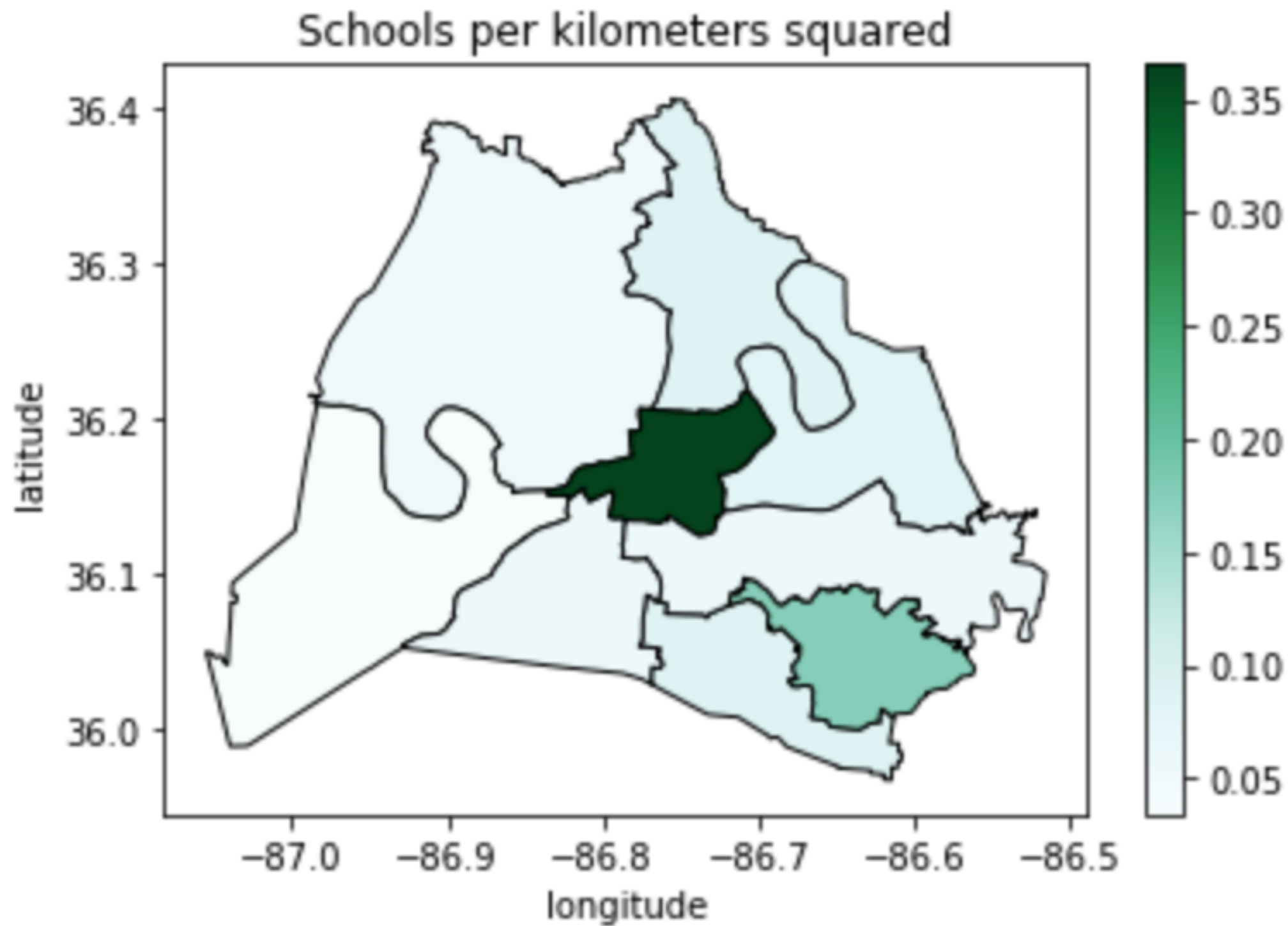


# Calculating school density

```
# 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')
plt.show();
```



# Let's practice!

VISUALIZING GEOSPATIAL DATA IN PYTHON

# Choropleths with folium

VISUALIZING GEOSPATIAL DATA IN PYTHON



**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 using the folium Choropleth class
folium.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

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

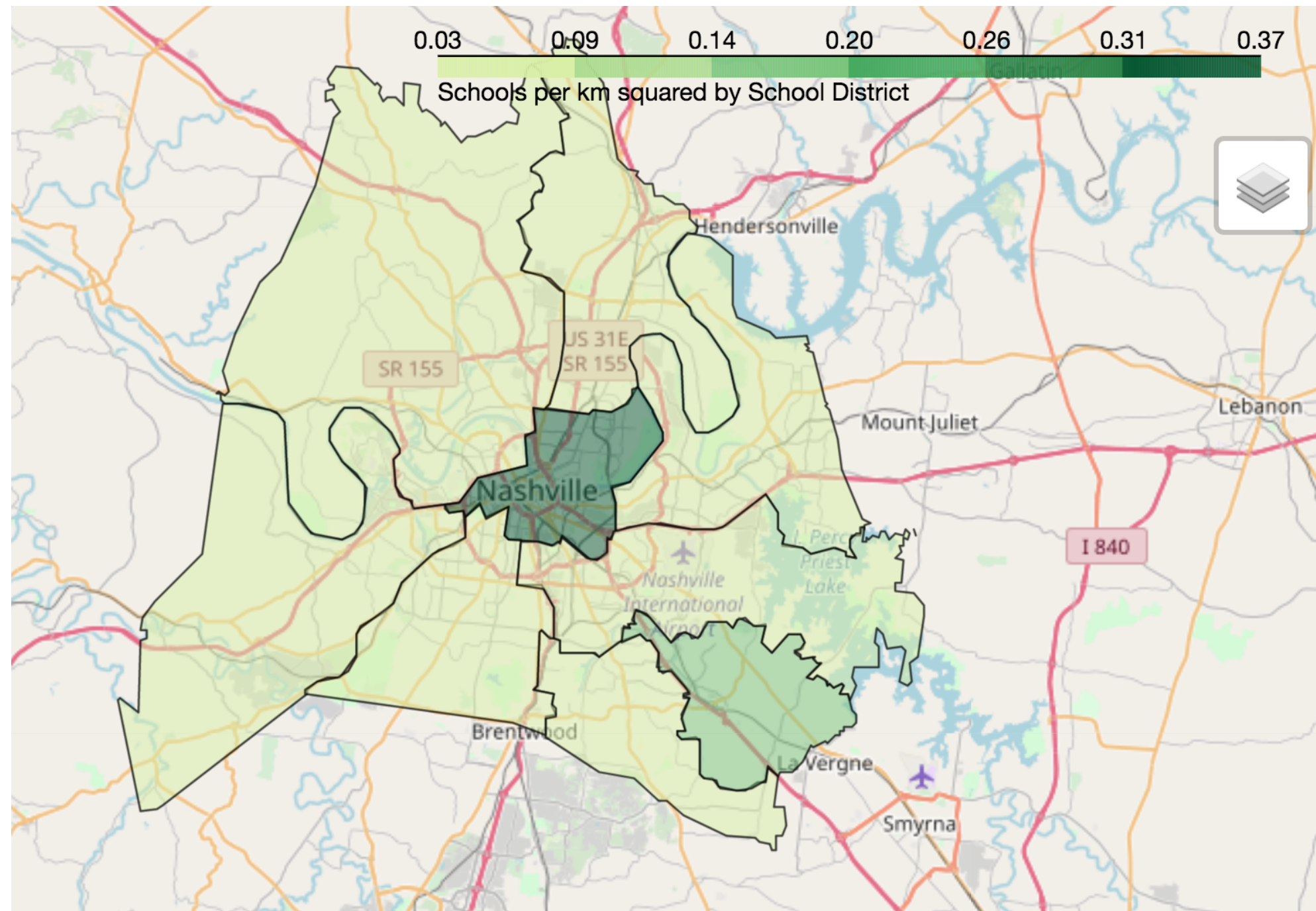
```
# Define a choropleth layer for the map
folium.Choropleth(
    geo_data=districts_with_counts,
    name='geometry',
    data=districts_with_counts,
    columns=['district', 'school_density'],
    key_on='feature.properties.district',
    fill_color='YlGn',
    fill_opacity=0.75,
    line_opacity=0.5,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

folium.LayerControl().add_to_map()
```

```
display(m)
```



# Folium choropleth of school density



# Let's Practice!

VISUALIZING GEOSPATIAL DATA IN PYTHON

# Congratulations!

VISUALIZING GEOSPATIAL DATA IN PYTHON



**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







# Goodbye

VISUALIZING GEOSPATIAL DATA IN PYTHON