Spatial data

Manipulation and plotting

Andreas Bjerre-Nielsen

Overview

- Introduction
- Spatial fundementals: shapes and coordinates

- Working with spatial data

    - Geopandas for data stucturing
    - Plotting: static and interactive

- Spatial procedures:

    - input-output
    - set-operations and joins
    - interpolation
- Other: assignments and exam project

Motivation

Why spatial data?

- Data supply is exploding
    - Free public big data
        - Infrastructure and buildings
        - Weather, housing market, traffic, job openings
        - Other APIs
    - Private big data
        - Smartphone: GPS sensors/wifi
- Spatial proximity > social relation?

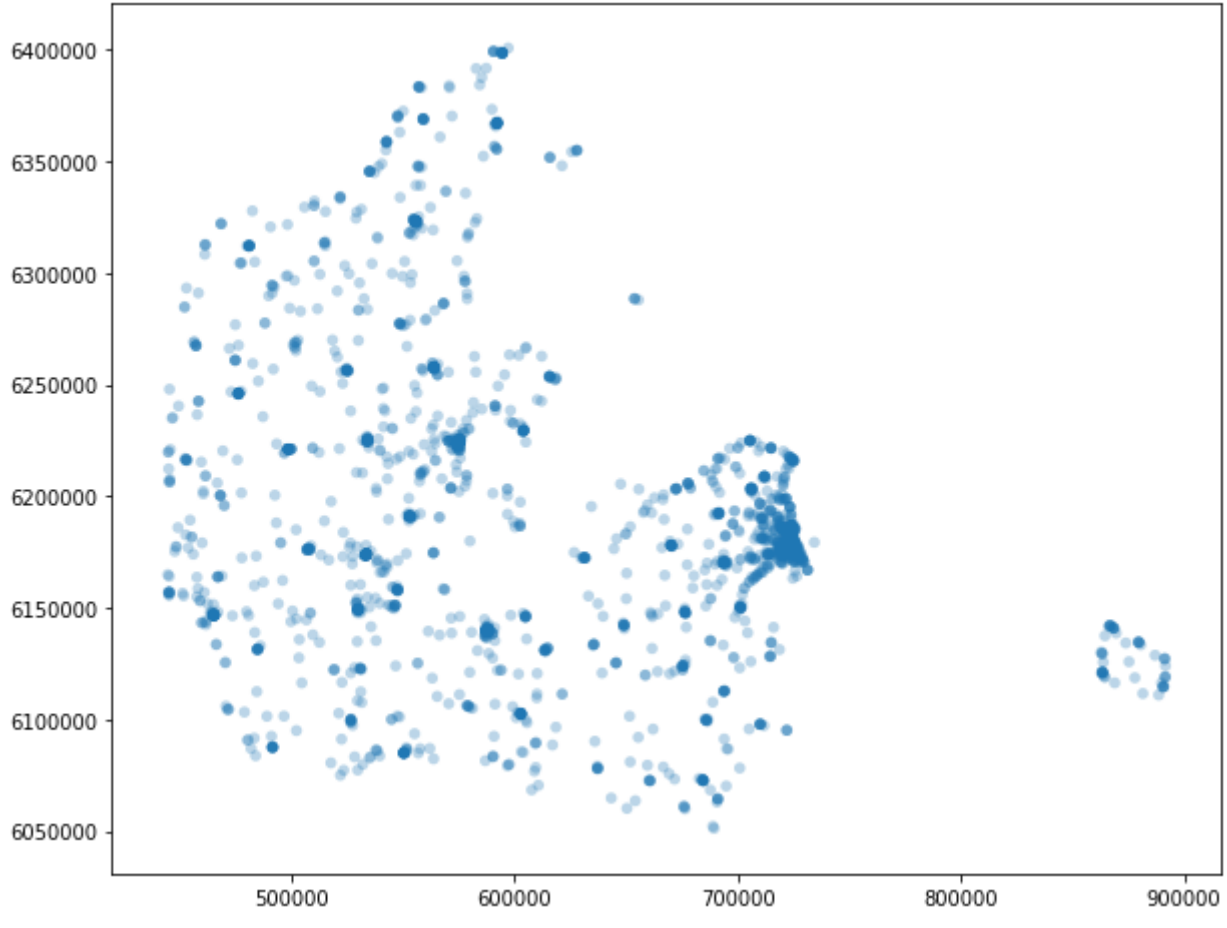
Why spatial data? (2)

Let's try some free public data:

- Where are the restaurants in Denmark? (i.e. where do people live)

In [2]:
```
f_restaurant_buffer
```

Out[2]:



Why spatial data? (3)

- Feature engineering
    - Approximate local measures (interpolation)
    - Intersect behavior and context:
        - Did you visit the supermarket yesterday?
        - Are people who spend more time in greenspace less streesed, less likely to use their phone?
- Identification strategy
    - Spatial RD: use separation administrative boundaries as causal effect
    - Heterogeneous policy responses:

Core concepts

What is spatial data?

- What are spatial objects? Do you know any?

- Spatial objects include classic geometric objects (lines, circles, squares etc.)
- Divided into four different generic shapes:
    - Point
    - LineString
    - Polygon (also approximates circles)
    - Multipolygon

What is spatial data? (2)

Example of a LineString

In [3]:
```
from shapely.geometry import LineString

line_coords = np.array([[0,3],[1,3],[2,4]])

LineString(line_coords)
```

Out[3]:

What is spatial data? (3)

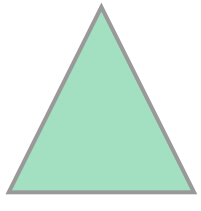
Example of a polygon

In [4]:
```
from shapely.geometry import Polygon

triangle_coords = np.array([[0,0],[2,0],[1,2]])

Polygon(triangle_coords)
```

Out[4]:



What is spatial data? (4)

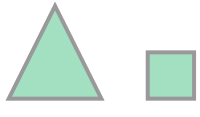
Examples of multi-polygons

In [5]:
```
from shapely.geometry import MultiPolygon

square = np.array([[3,0],[3,1],[4,1],[4,0]])

MultiPolygon([Polygon(triangle_coords),
              Polygon(square)])
```

Out[5]:



How do we measure spatial data?

We need to define the space we work in. What could this be?

- Observations on earth use different Coordinate References System (CRS).
- Some CRS are local and some are global.

How do we measure spatial data? (2)

The standard CRS is:

- The Global Positioning System (GPS)
    - Technically called WGS84
    - Has EPSG code 4326
        - (EPSG: *European Petroleum Survey Group*, now extinct)
- Used on Google Maps, Apple Maps etc.

How do we measure spatial data? (3)

Working with Danish spatial data you often encounter:

- The Danish mapping reference
    - Technically called ETRS89 UTM Zone 32 North
    - Has EPSG code 25832
    - This is used in Danish admin data.
    - Coordinates measured in meters: can use Euclidian distance

Applied spatial data

How do we work with spatial data?

GeoPandas:

- Collection of spatial objects
    - Powerful data structuring
    - Static plots
- Combines Pandas with
    - Shapely (Python shape objects)
    - Fiona (low level, ultra fast for complex operations)

Folium

- Interactive plots: map overlay for OpenStreetMaps and zoom

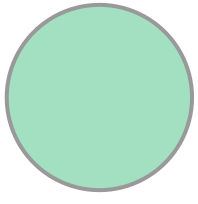
Spatial Data Structures

Spatial shapes

We have already seen how to employ `shapely` to construct spatial shapes.

In [6]:
```
points = np.linspace(0, 2*np.pi, 101)
unit_circle_points = [(np.cos(d),np.sin(d)) for d in points]
Polygon(unit_circle_points)
```

Out[6]:



What if we have more than one spatial object?

We could store spatial objects in lists but that is not smart. Why?

GeoSeries

A smart container for spatial data is the GeoSeries:

- 1d array with labels (like Pandas series)
- useful spatial tools

In [7]:
```
world_map = gpd.read_file(gpd.datasets.get_path('naturalearth_lowres'))
world_map.geometry.head(3)
```

Out[7]:
```
0    POLYGON ((61.21081709172574 35.65007233330923,...
1    (POLYGON ((16.32652835456705 -5.87747039146621...
2    POLYGON ((20.59024743010491 41.85540416113361,...
Name: geometry, dtype: object
```

GeoDataFrame

There is also the GeoDataFrame which has a dedicated column for geometry:

In [8]:
```
world_map.crs
```

Out[8]:
```
{'init': 'epsg:4326'}
```

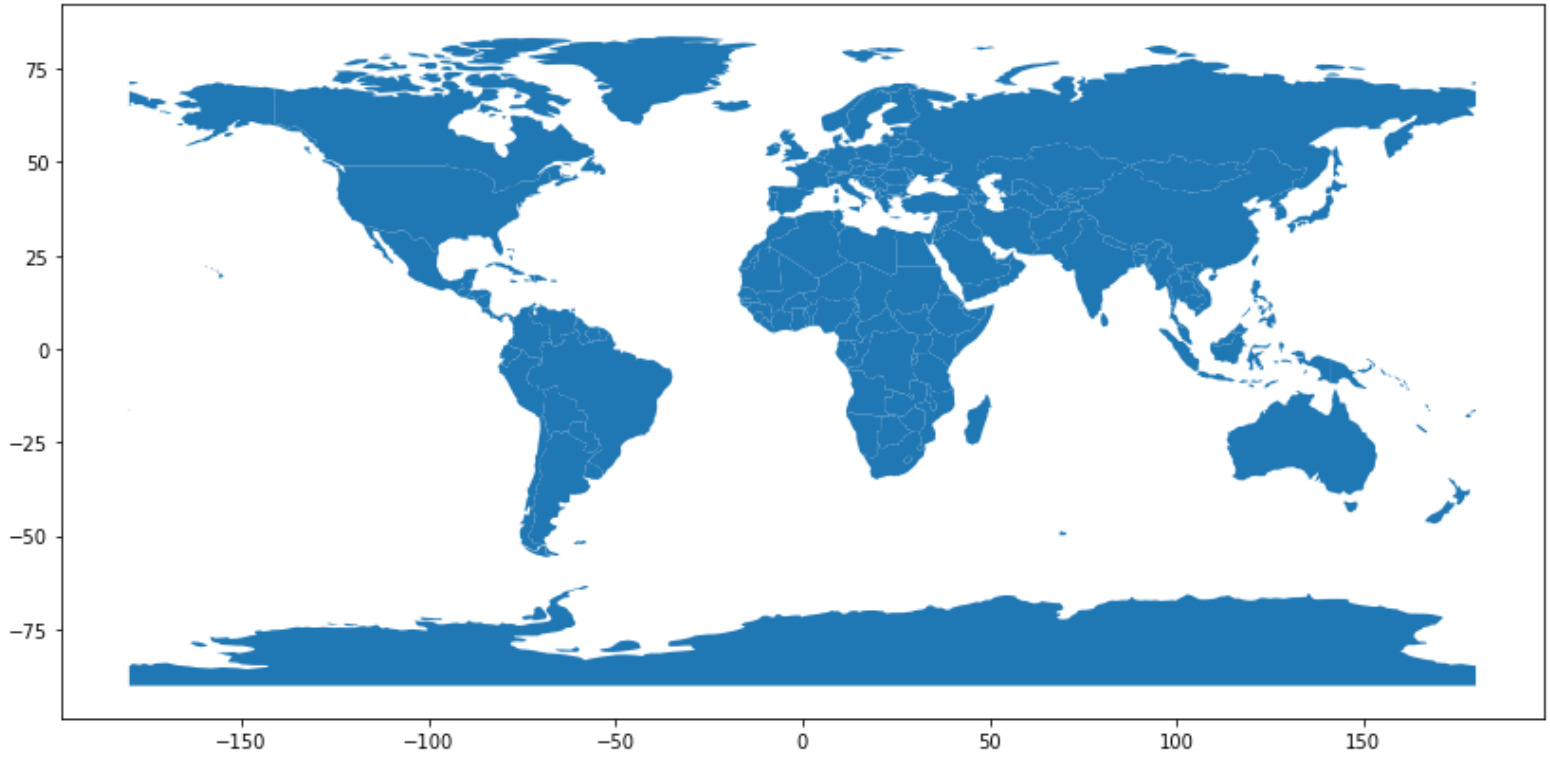
Map making

Static plots

Let's try to plot the world's countries. Easy as hell with GeoPandas:

In [9]:
```
world_map.plot(figsize=(14,7))
```

Out[9]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x19ccb377470>
```



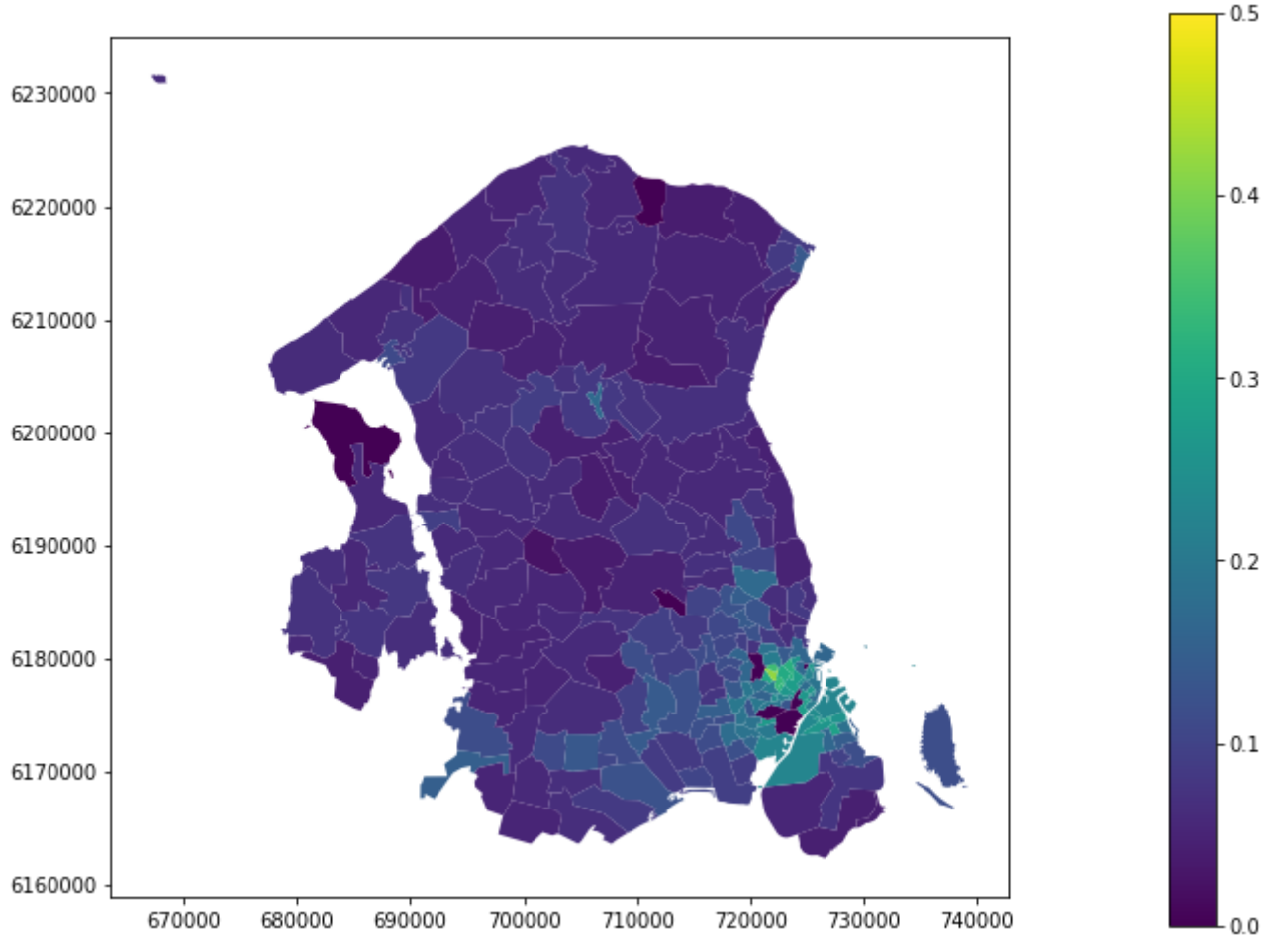
Static plots (2)

Now let's try to plot the share of young around in the Capital Region of Denmark (excluding Bornholm).

Data is from Statistics Denmark at Parish level.

In [10]:
```
f_cph_young_share
```

Out[10]:

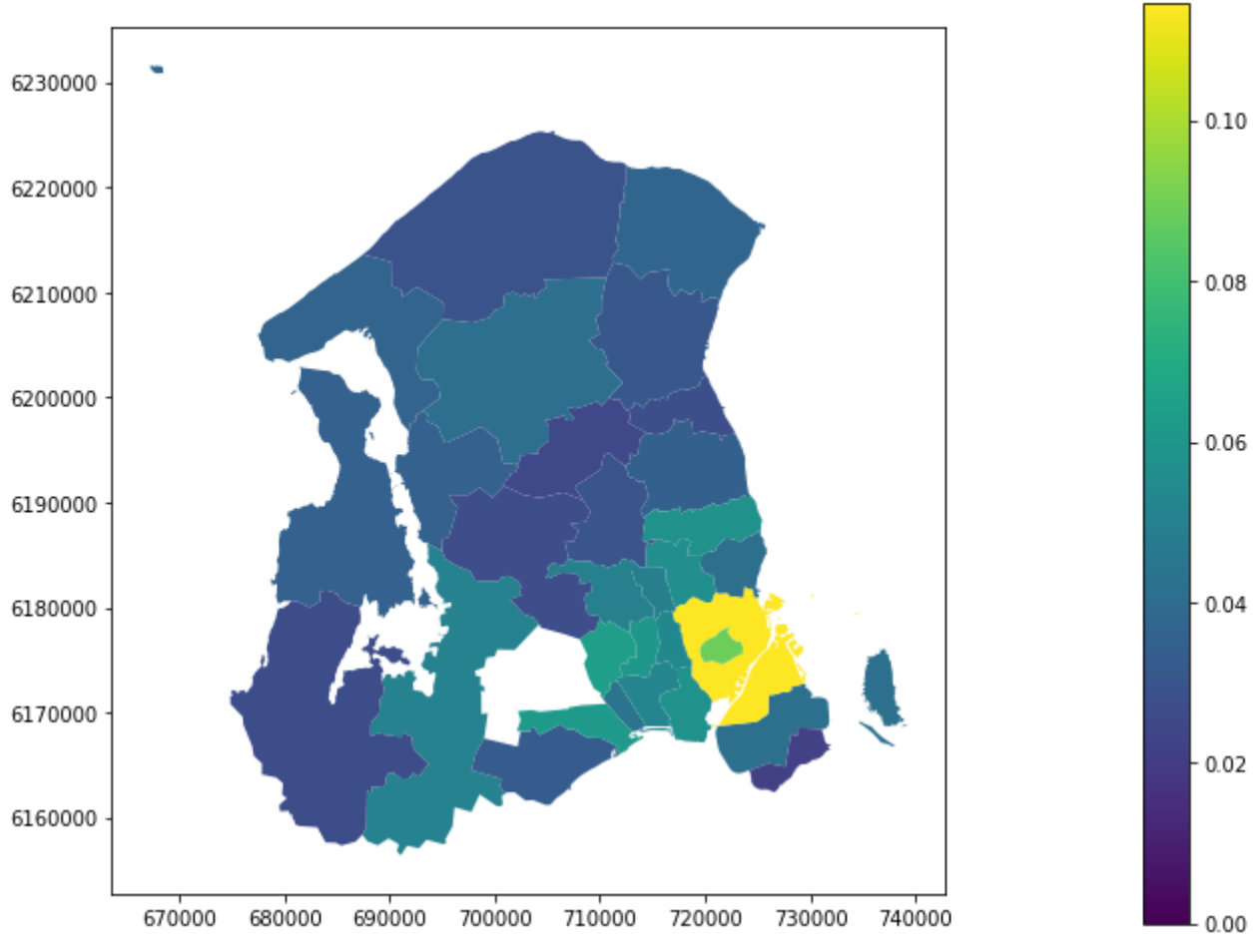


Static plots (3)

Using municipal level statistics we see much smaller variation.

In [11]:
```
f_cph_young_share_mun
```

Out[11]:



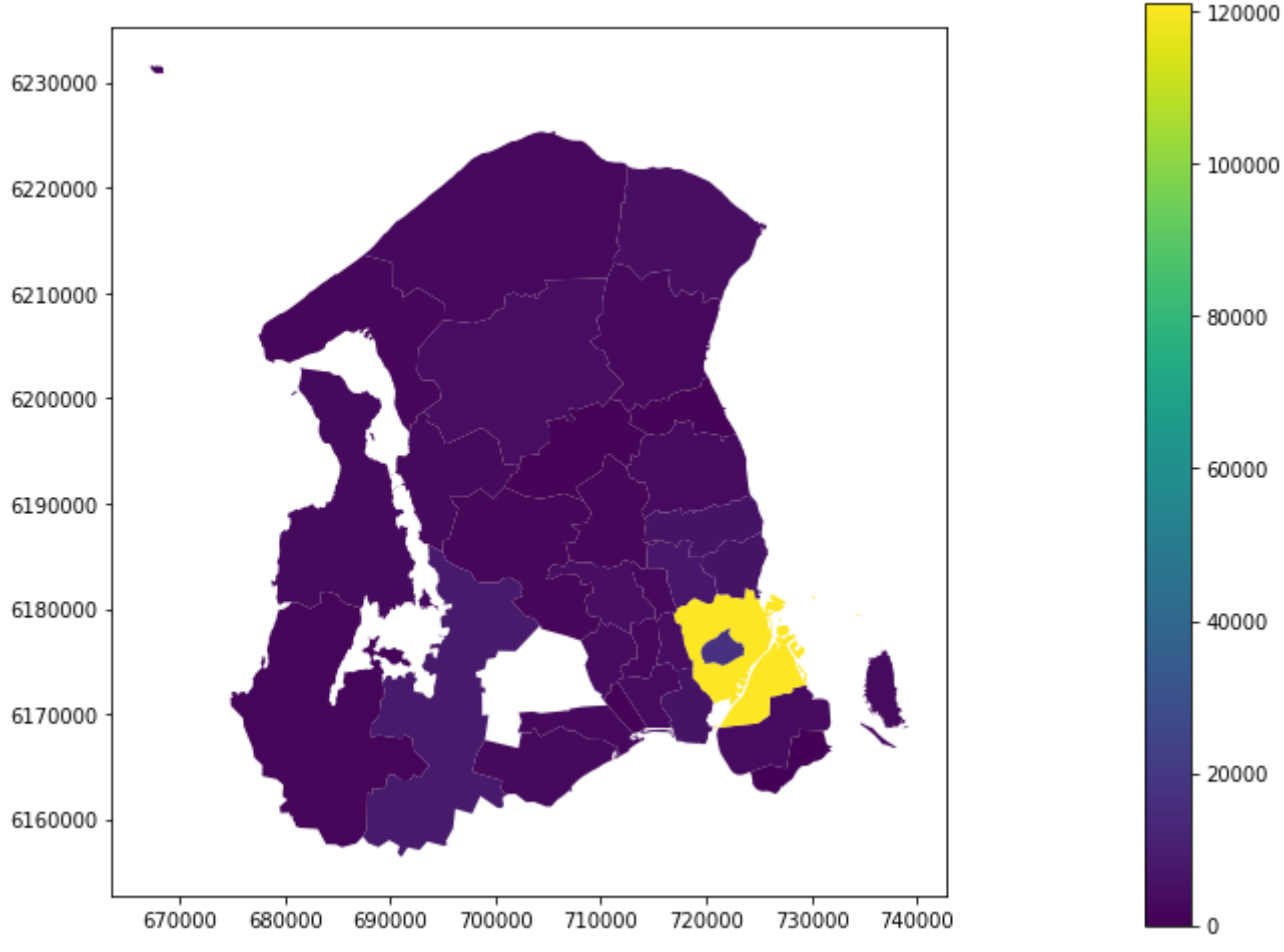
Static plots (4)

What should be very careful about when making plots?

Never plot sum or count !!! We mix in population density.

In [12]:
```
f_cph_young_count_mun
```

Out[12]:

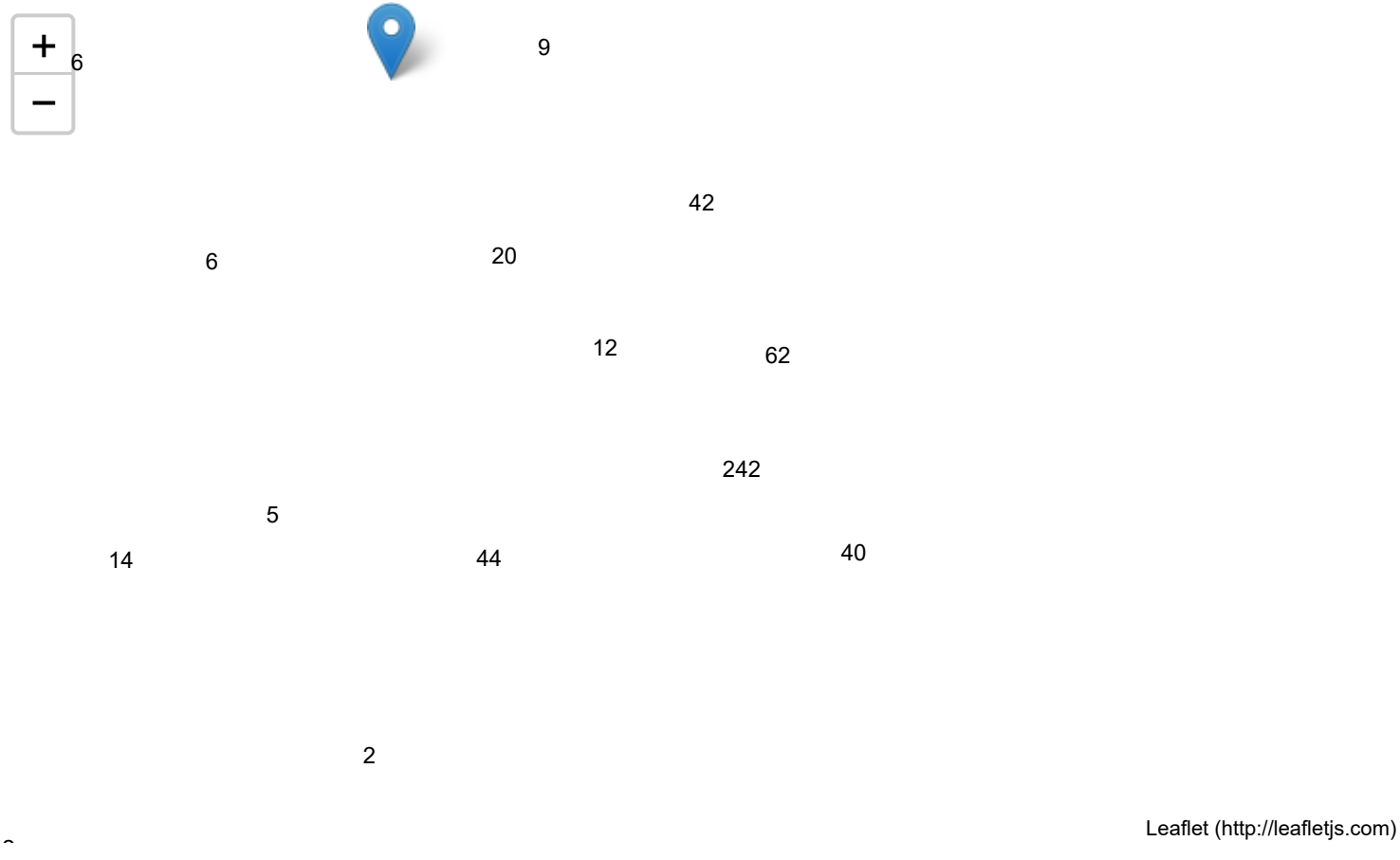


Folium: Plots with interaction

Make a map of many points for Danish supermarkets.

In [13]:
```
m_dk_supermarket
```

Out[13]:



Loading and storing spatial data

There are two standard formats:

- GeoJSON (.geojson):
    - This is getting more common, especially in more modern applications.
- Shapefiles (.shp)
    - The de-facto standard format. This is a collection of files where the central one is the `.shp` file.

Example - load house price data:

In [14]:
```
import geopandas as gpd
gdf = gpd.read_file('house_prices.geojson')
```

Spatial operations

When having two or more spatial objects it is possible to make interactions between them.

Basic operations

- Shortest distance between shapes (e.g. Euclidian distance between two points)
    - What about in spherical coordinates? (using Haversine formula is one solution)

Spatial operations (2)

Advanced operations

- Set like operations ~ make new shapes:
    - Intersection (A and B)
    - Union (A or B)
    - Difference (A not B)
- We can ask whether two shapes
    - Intersect or not
    - Touch or not

Spatial operations (3)

Basic 'set-like' operations in `shapely`.

In [15]:
```
rect_coords = [[0, 0], [0, 2], [1, 2], [1, 0]]

A = Polygon(triangle_coords)
B = Polygon(rect_coords)



A - B # union
# A & B # intersection
# A - B # set difference
```

Out[15]:

Spatial operations (4)

Example of spatial join through intersection with `GeoPandas' sjoin`.

In [16]:
```
kommuner_cph.plot()
```

Out[16]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x19ccb836208>
```

In [17]:
```
supermarket_cph = gpd.sjoin(gdf_supermarket, kommuner_cph)
supermarket_cph.head(3)
```

Out[17]:

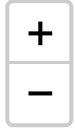
	id	lat	lon	tags	type	geometry	name	index_right	komkode	komnavn
8	272860527	55.655972	12.485416	{'name': 'Netto', 'shop': 'supermarket'}	node	POINT (719255.6405412797 6173300.325984053)	Netto	167	0101	København
9	272860703	55.643196	12.501363	{'name': 'Netto', 'shop': 'supermarket'}	node	POINT (720330.1542702764 6171929.991743754)	Netto	167	0101	København
15	282680168	55.647999	12.526752	{'name': 'SuperBrugsen', 'opening_hours': 'Mo-...	node	POINT (721899.9097495852 6172545.122327959)	SuperBrugsen	167	0101	København

Spatial operations (5)

Check output of spatial join

In [18]:
```
centroid = supermarket_cph.centroid.to_crs(epsg=4326)
for lat,lon in zip(centroid.y, centroid.x):
    marker_cluster_cph.add_child(folium.Marker(location=[lat, lon]))
marker_cluster_cph.add_to(m_dk_supermarket_cph)
m_dk_supermarket_cph
```

Out[18]:



31

206

28

Leaflet (http://leafletjs.com)

Spatial interpolation

Why spatial interpolation?

- Often we are interested in
    - Look up local weather
    - Data for neighborhoods not just points
    - Having measure available everywhere

- Spatial data is often sparse and pointwise
    - What about areas in between?
    - Which points to use?

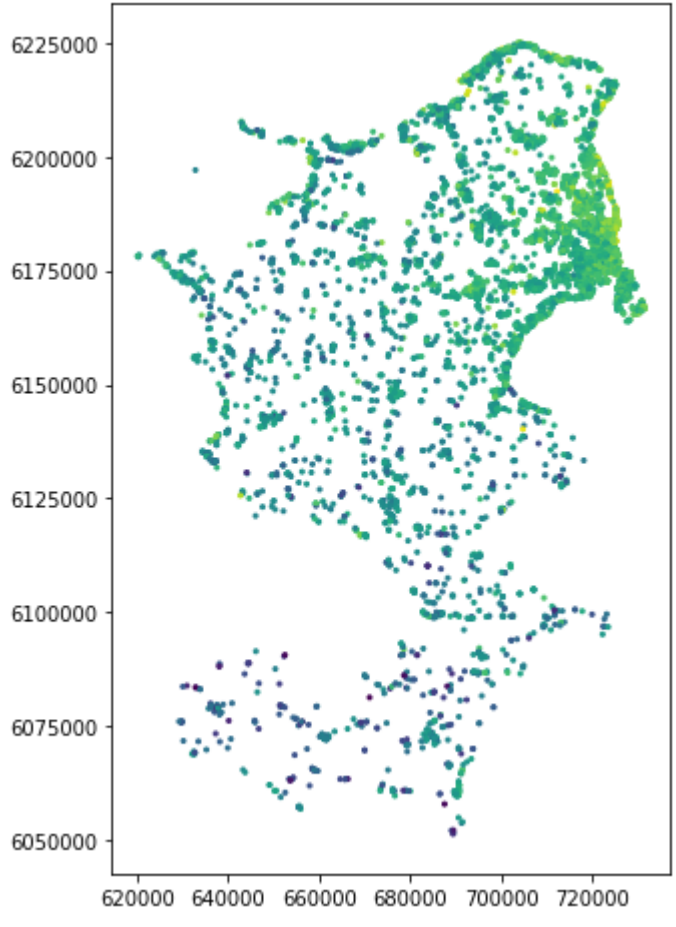
Why spatial interpolation? (2)

Let's plot 2012 house sale prices.

- We notice some rural regions have little coverage

In [19]:
```
f_price_pointcloud
```

Out[19]:



What is spatial interpolation?

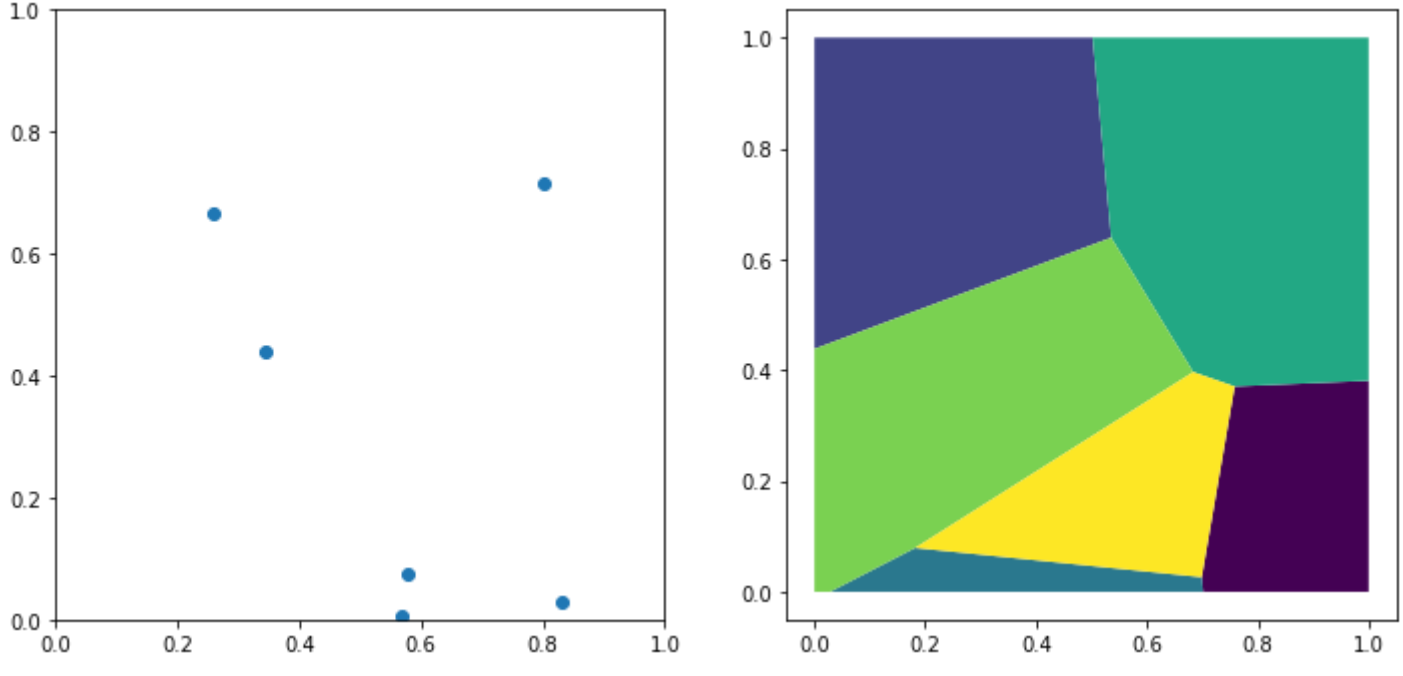
What is the simplest possible spatial interpolation pattern?

- The nearest neighbor?
- The nearest neighbor plot is known a the Voronoi plot

What is spatial interpolation? (2)

Let's see what a Voronoi plot looks like

In [26]:
```
make_vor_plot()
```



What is spatial interpolation? (3)

The Voronoi plot for Danish house sale prices in 2012:

What is spatial interpolation? (4)

The nearest neighbor method (exercise)

- Find the k nearest points
    - Regression: Take mean of k-nearest
    - Classification: Take mode of k-nearest
- The measurement space can be physical e.g. surface distance i 2d
- A machine learning model
    - non-parametric approach
    - requires normalization (e.g. L2)

Summary

We have learned about

- Spatial shapes and coordinate systems
- Perform operations on single shapes and collections of shapes
- Make interactive maps
- Interpolate spatial data