INTERACTIVE MAPMAKING WITH PYTHON

Sangarshanan

DATA + PANDAS <3


My Data backed Insights
Numpy
scikit-learn
Matplotlib
Pandas

In [1]:
```
import pandas
df = pandas.read_csv('data/cities.csv')
df.head()
```

Out[1]:

	name	longitude	latitude
0	Vatican City	12.453387	41.903282
1	San Marino	12.441770	43.936096
2	Vaduz	9.516669	47.133724
3	Luxembourg	6.130003	49.611660
4	Palikir	158.149974	6.916644

DATA WITH A LOCATION COMPONENT

Geometries

- Point (latitude, longitude)
- Polygon [point1, point2]

We can also have linestrings, multipolygons, circles etc

ENTER GEOPANDAS

- Work with a Familiar interface (Dataframes), in this case Geodataframes
- Read/ write GIS data (Fiona) formats like shapefile, geojson, kml etc
- Perform spatial operations like merge/join/overlay etc (Shapely)
- Plot em on a map (Matplotlib)

Also a whole lot of other things like handling projections, recently added vectorized geometrical operations, Indexing with rtree... and more such goodies

Interested in more, https://github.com/jorisvandenbossche/geopandas-tutorial (https://github.com/jorisvandenbossche/geopandas-tutorial) has a comprehensive tutorial by the maintainer

In [2]:
```
df.head()
```

Out[2]:

	name	longitude	latitude
0	Vatican City	12.453387	41.903282
1	San Marino	12.441770	43.936096
2	Vaduz	9.516669	47.133724
3	Luxembourg	6.130003	49.611660
4	Palikir	158.149974	6.916644

In [3]:
```
import geopandas
# Converting latitude, longitude to geometry object
gdf = geopandas.GeoDataFrame(
    df, geometry=geopandas.points_from_xy(df.longitude, df.latitude))
# setting the projection
gdf.crs = 'epsg:4326'
gdf.head()
```

Out[3]:

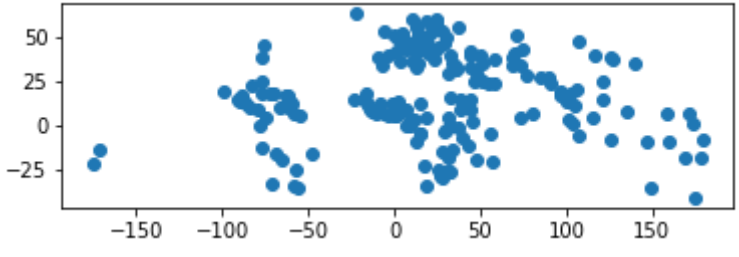
	name	longitude	latitude	geometry
0	Vatican City	12.453387	41.903282	POINT (12.45339 41.90328)
1	San Marino	12.441770	43.936096	POINT (12.44177 43.93610)
2	Vaduz	9.516669	47.133724	POINT (9.51667 47.13372)
3	Luxembourg	6.130003	49.611660	POINT (6.13000 49.61166)
4	Palikir	158.149974	6.916644	POINT (158.14997 6.91664)

HOW DO I PLOT EM ?

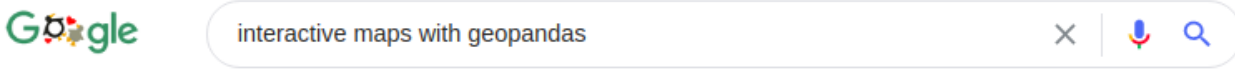
WITH GEOPANDAS, ITS AS SIMPLE AS .PLOT()

In [28]:
```
import matplotlib.pyplot as plt
gdf.plot()
```

Out[28]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f8024596358>
```




Interactivity
Maps



About 25,000 results (0.30 seconds)

towardsdatascience.com › a-complete-guide-to-an-interact...

A Complete Guide to an Interactive Geographical Map using ...

Explore **pandas** and **Geopandas** dataframes. Import **geopandas**. **Geopandas** can read almost any vector ...

Feb 5, 2019 - Uploaded by Shivangi Patel

towardsdatascience.com › walkthrough-mapping-basics...

Walkthrough: Mapping Basics with bokeh and GeoPandas in ...

Nov 13, 2019 - My goal this week was to learn how to make an **interactive map** and then to create a tutorial on how to make that map. Maps can be really ...

automating-gis-processes.github.io › Lesson5-interactiv...

Interactive maps with Bokeh — GeoPython - AutoGIS 1 ...

Our ultimate goal today is to learn few concepts how we can produce nice looking **interactive maps** using **Geopandas** and Bokeh such as:.

automating-gis-processes.github.io › interactive-map-folium

Interactive maps — AutoGIS site documentation

Here, will focus on two Python libraries -mplleaflet and Folium - that are able to convert our data in **(geo)pandas** into **interactive** Leaflet **maps**. Explore also…

FOLIUM PLOTS

In [5]:
```
gdf.head(1)
```

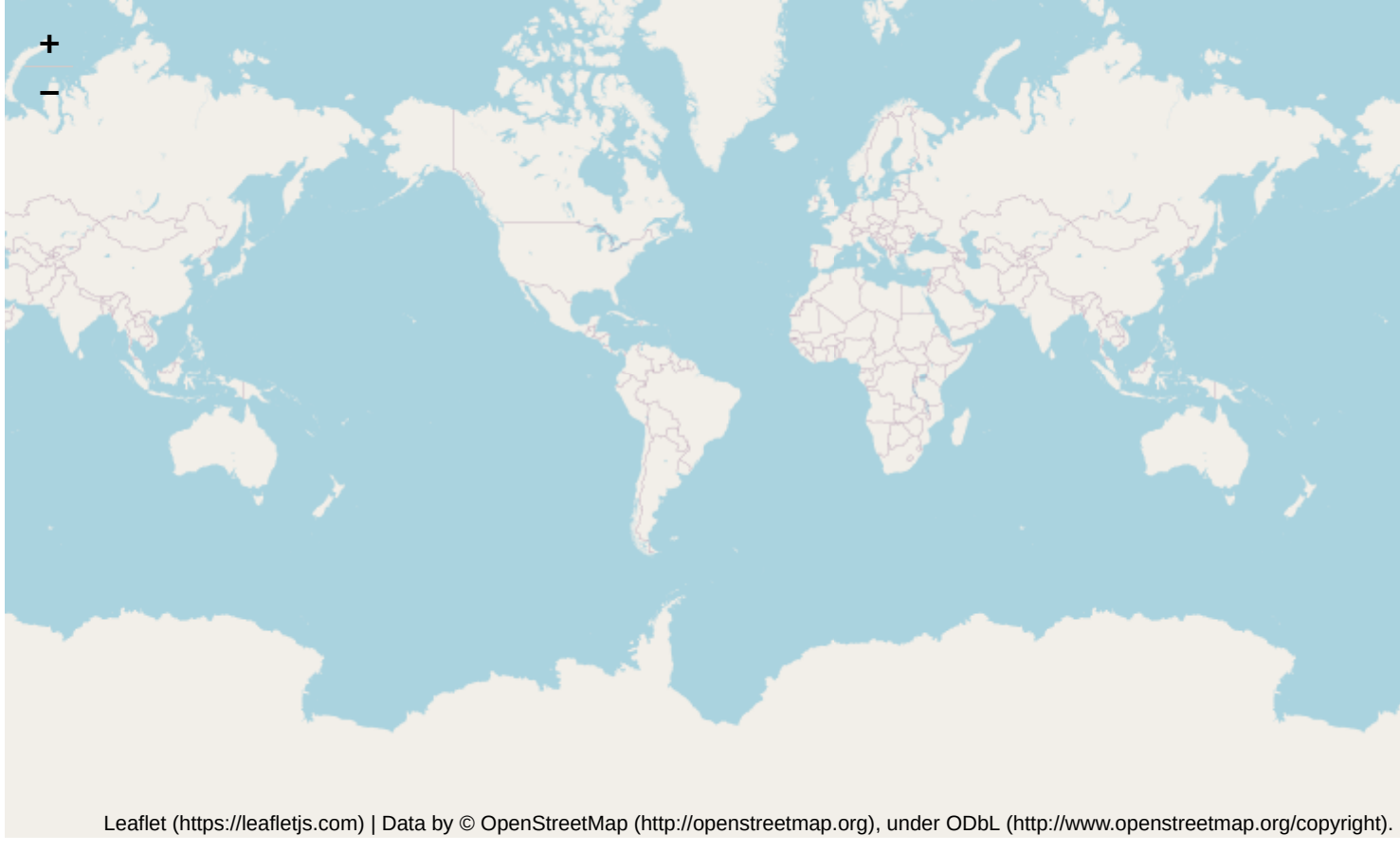
Out[5]:

	name	longitude	latitude	geometry
0	Vatican City	12.453387	41.903282	POINT (12.45339 41.90328)

In [6]:

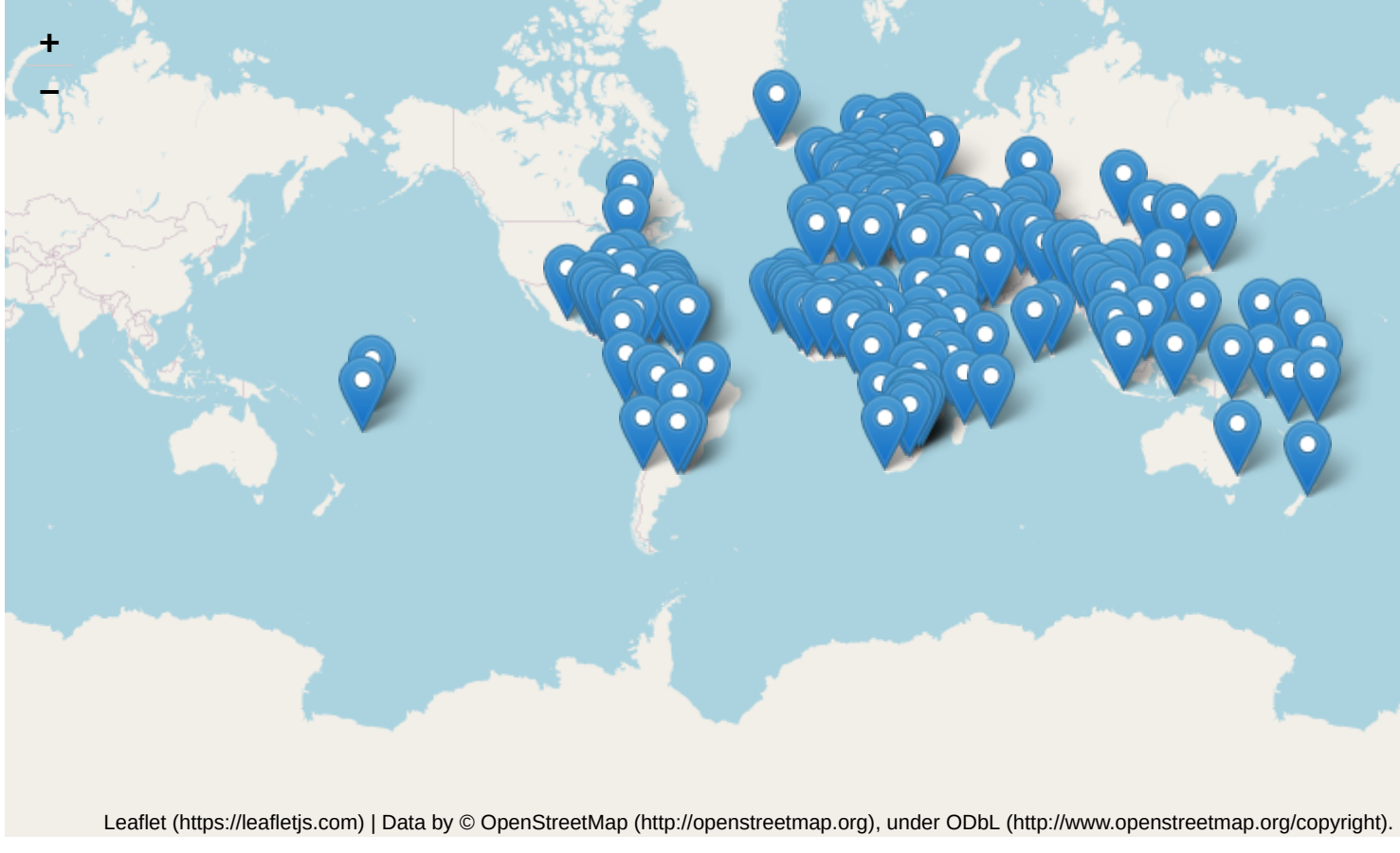
```
# import the library
import folium
# Create a map with a center and zoom level
mapa = folium.Map(location= [-15.783333, -47.866667],
                  zoom_start= 1,
                  tiles= "OpenStreetMap")
mapa
```

Out[6]:



In [7]:
```
# Add the geodataframe as a geojson feature
points = folium.features.GeoJson(gdf,
                                 # tooltip with the name
                                 tooltip=folium.GeoJsonTooltip(fields=['name']))
# Adding the feature to the canvas we created
mapa.add_child(points)
mapa
```

Out[7]:



ONWARD TO POLYGONS

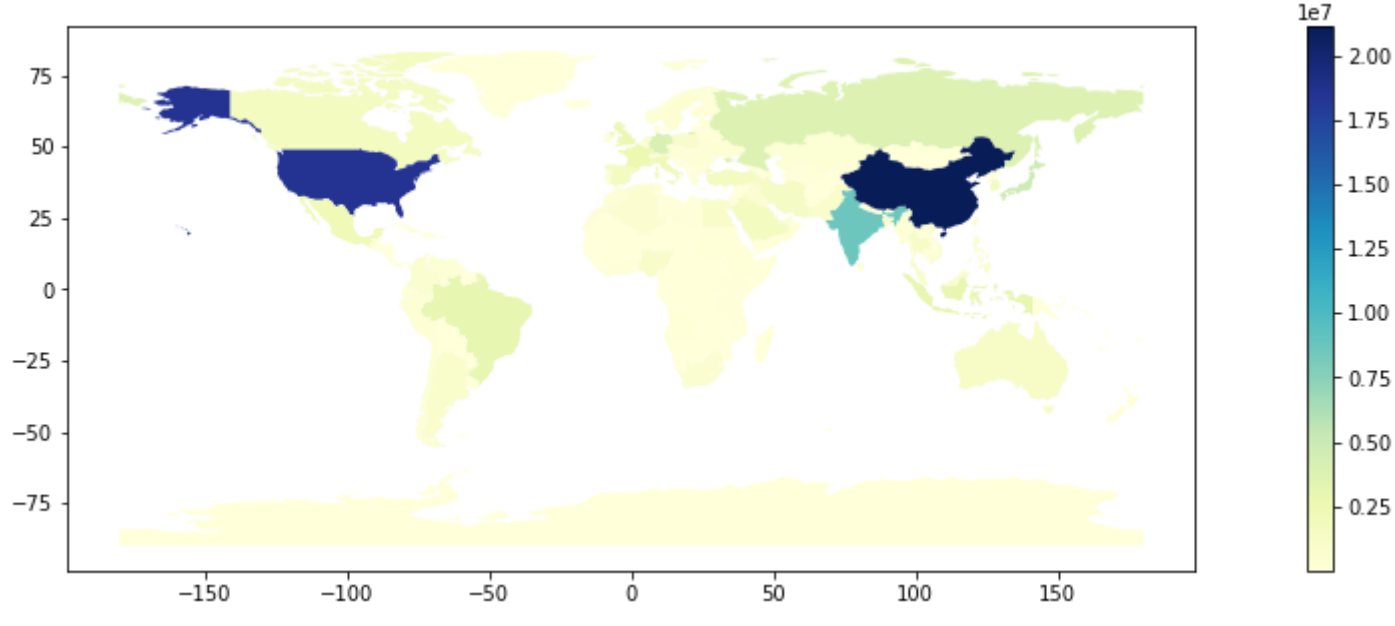
In [8]:
```
world = geopandas.read_file(geopandas.datasets.get_path('naturalearth_lowres'))
world.head(2)
```

Out[8]:

	pop_est	continent	name	iso_a3	gdp_md_est	geometry
0	920938	Oceania	Fiji	FJI	8374.0	MULTIPOLYGON (((180.00000 -16.06713, 180.00000...
1	53950935	Africa	Tanzania	TZA	150600.0	POLYGON ((33.90371 -0.95000, 34.07262 -1.05982...

In [9]:
```
#https://matplotlib.org/3.1.1/gallery/color/colormap_reference.html
world.plot(figsize=(20,5), column= 'gdp_md_est', cmap='YlGnBu', legend=True)
```

Out[9]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f80269f5a20>
```



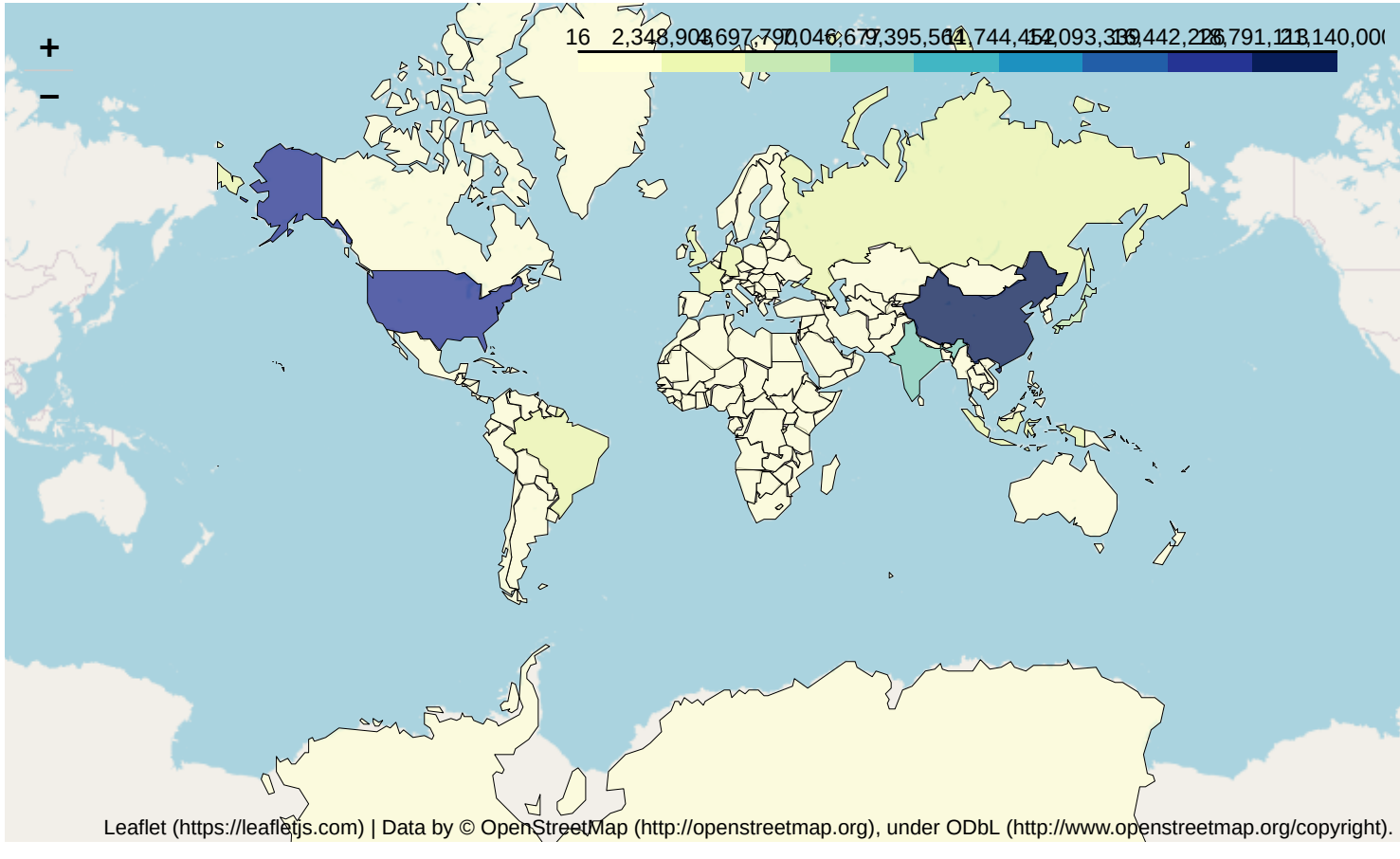
In [10]:
```
import branca.colormap as cm
colormap = cm.linear.YlGnBu_09.to_step(data=world['gdp_md_est'], n=9)
colormap
```

Out[10]:

In [11]:
```
m = folium.Map()
# Setting the style
style_function = lambda x: {
    # How to fill color of the polygon
    'fillColor': colormap(x['properties']['gdp_md_est']),
    # Color of Polygon
    'color': 'black',
    # Weight of the border (Around the Polygon)
    'weight': 0.5,
    # Opacity of filled color
    'fillOpacity': 0.75
}
```

In [12]:
```
# Creating a geojson map with the style
folium.GeoJson(
    world,
    tooltip=folium.GeoJsonTooltip(fields= ["name", "gdp_md_est"]),
    style_function=style_function
).add_to(m)
# Add the legend to the same canvas
colormap.add_to(m)
m
```

Out[12]:

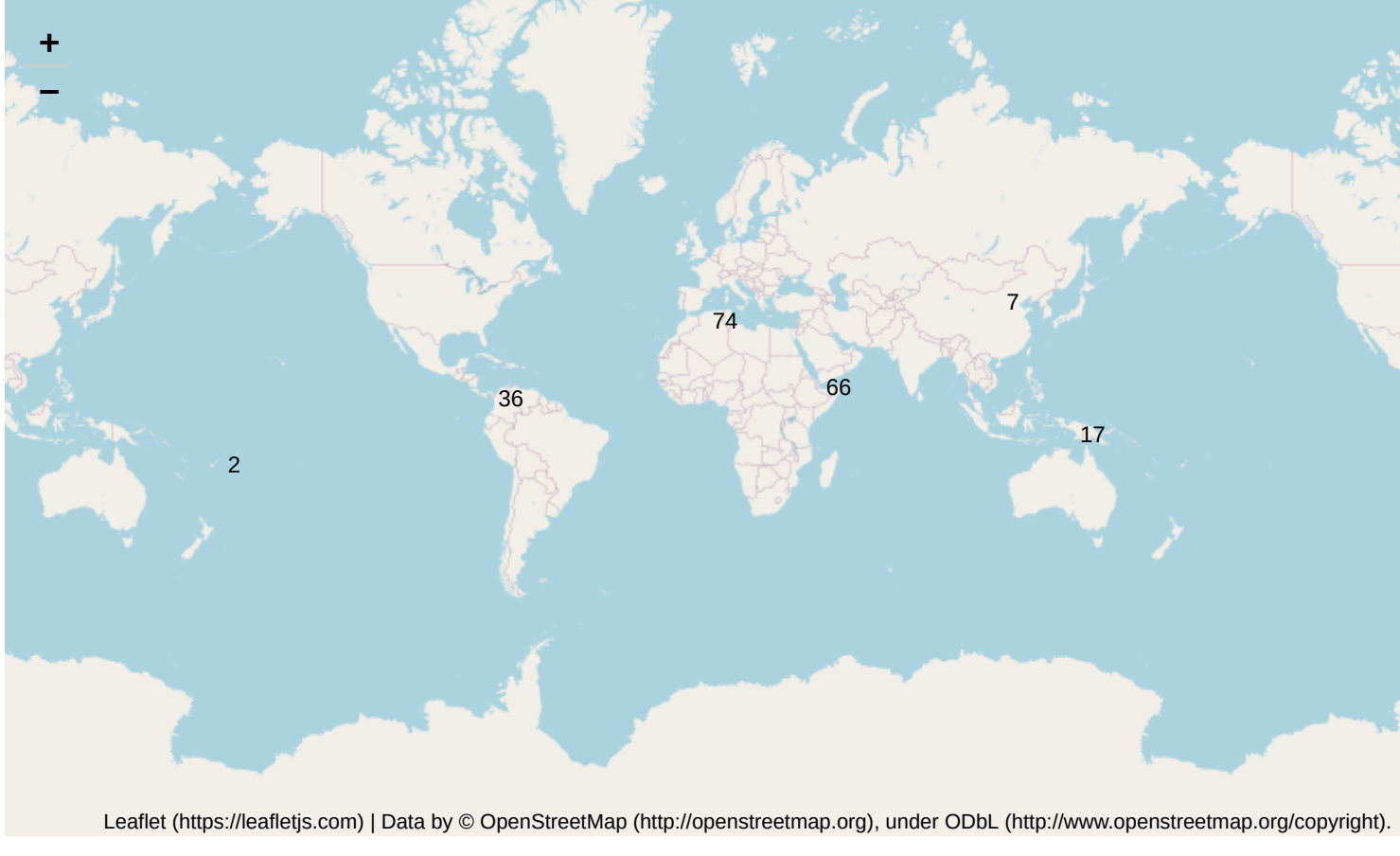


MARKER CLUSTERS

In [13]:

```
from folium.plugins import MarkerCluster
locations = []
# City location geometries to a list of latlongs pairs
for idx, row in gdf.iterrows():
    locations.append([row['geometry'].y, row['geometry'].x])
# Empty canvas
m = folium.Map()
# Markercluster
m.add_child(MarkerCluster(locations=locations))
m
```

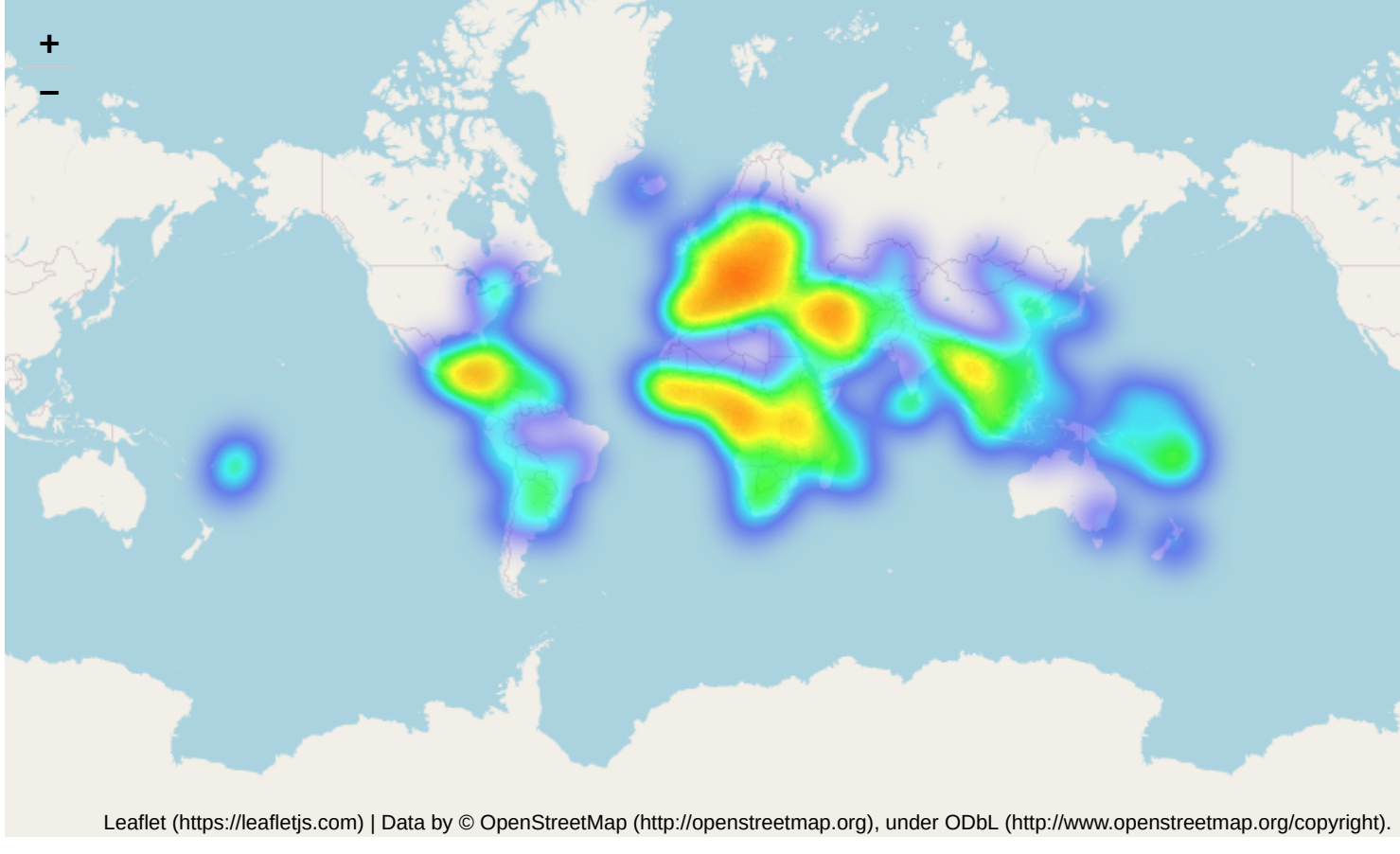
Out[13]:



HEATMAP

In [14]:
```
from folium.plugins import HeatMap
m = folium.Map()
m.add_child(HeatMap(locations, radius=15))
m
```

Out[14]:



Mo data Mo Problems


A million data points
My browser

KEPLER.GL

Kepler.gl is a data-agnostic, high-performance web-based application for visual exploration of large-scale geolocation data sets. Built on top of Mapbox GL and deck.gl, kepler.gl can render millions of points representing thousands of trips and perform spatial aggregations on the fly.

Jupyter Notebook > 5.3

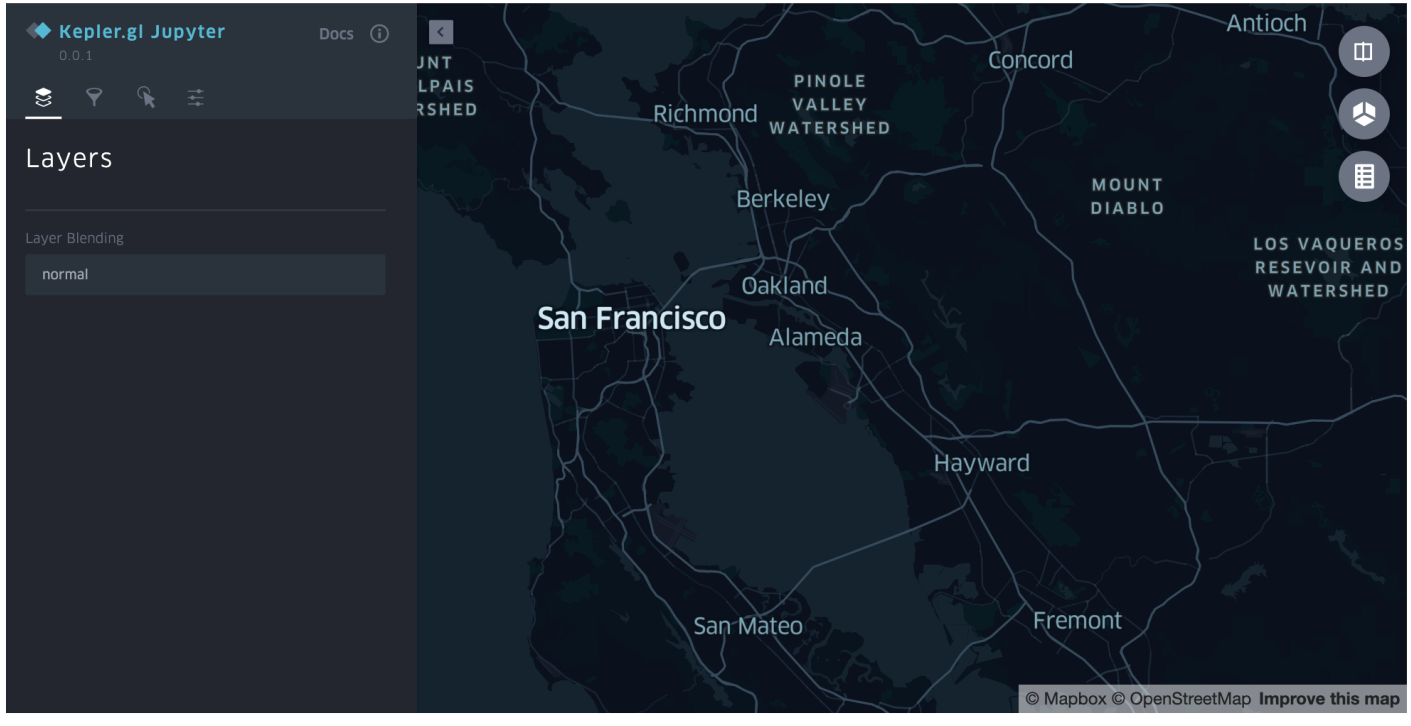
```
pip install keplergl
```

JupyterLab

```
jupyter labextension install @jupyter-widgets/jupyterlab-manager keplergl-jupyter
```

In [10]:
```
import keplergl
w1 = keplergl.KeplerGl(height=500)
w1
```

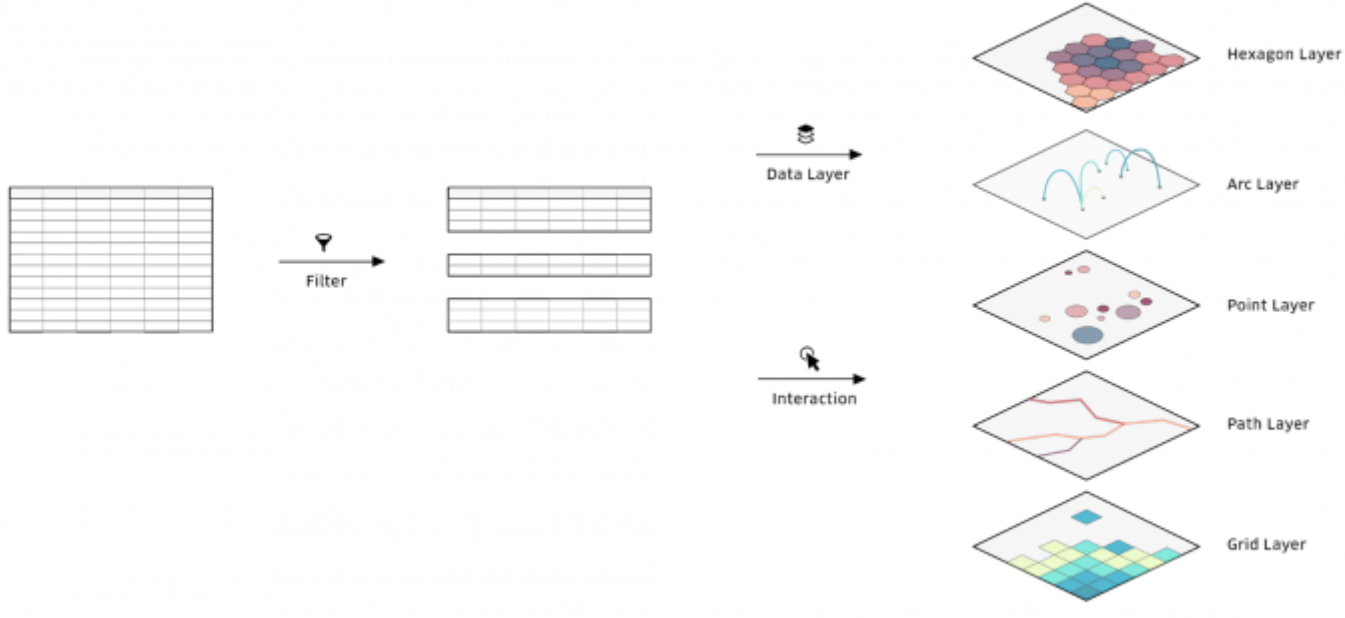
Documentation: https://github.com/keplergl/kepler.gl/blob/master/docs/keplergl-jupyter/user-guide.md



KEPLER USES CONFIG TO CUSTOMIZE ITS MAPS


```
{
  "version": "v1",
  "config": {
    "visState": {
      "filters": [
        {
          "dataId": "earthquakes",
          "id": "vo18yorx",
        }
      ],
      "layers": [
        {
          "id": "hty62yd",
          "type": "point",
          "config": {
            "dataId": "earthquakes",
            "label": "Point",
            "color": [
              23,
              184,
              190
            ],
            "columns": {
              "lat": "Latitude",
              "lng": "Longitude",
              "altitude": null
            },
.....
```

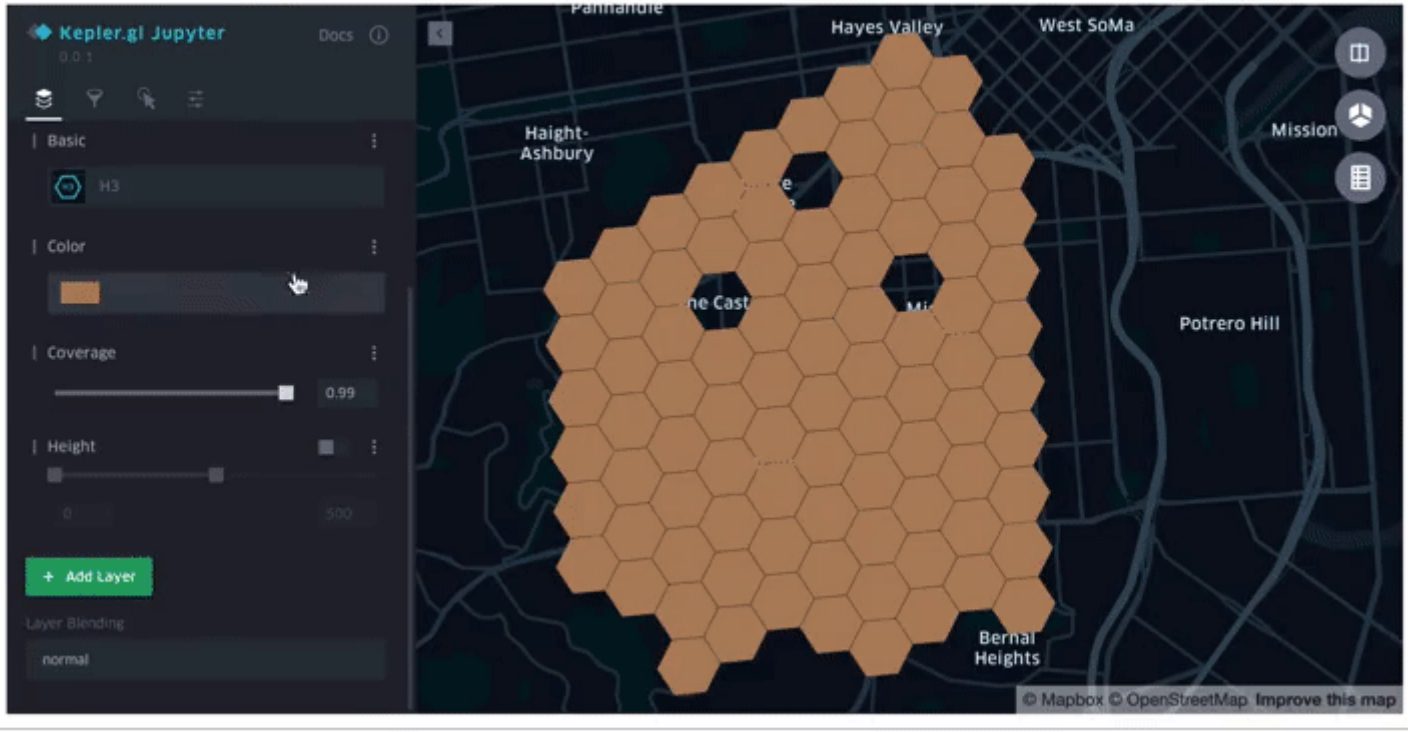
THE UX FLOW IS COMPOSED OF FIVE LAYERS



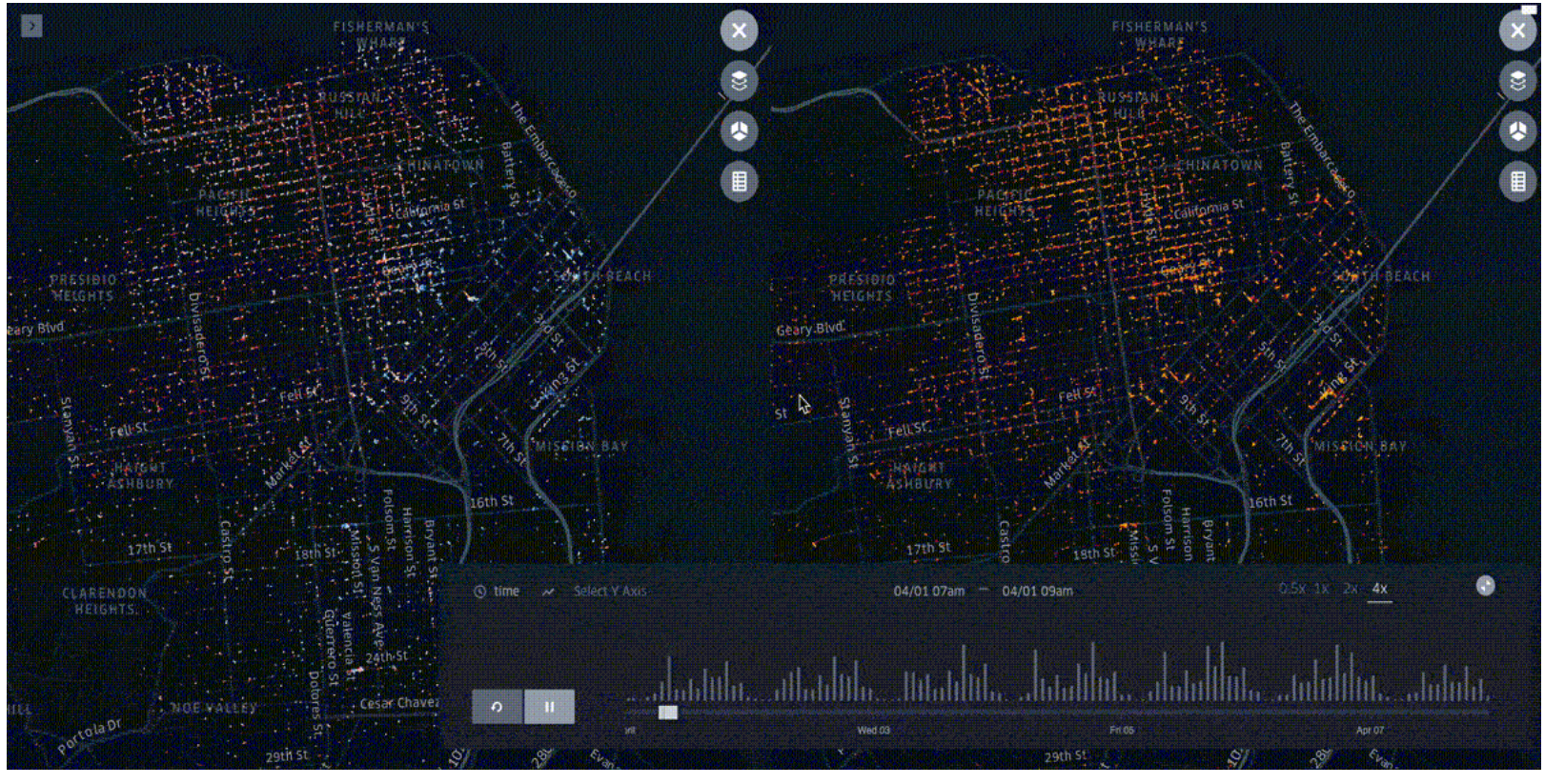
FILTERS, TIMELINES AND OTHER COOL PERKS

```
In [6]: import keplergl_jupyter as keplergl
        wl = keplergl.KeplerGl(height=500)
        wl
```

Documentation: https://github.com/keplergl/kepler.gl/blob/master/docs/keplergl-jupyter/user-guide.md




```
In [7]: wl.add_data(df, 'data_1')
```


FISHERMAN'S WHARF
RUSSIAN HILL
CHINATOWN
PACIFIC HEIGHTS
PRESIDIO HEIGHTS
SOUTH BEACH
MISSION BAY
HAIGHT ASHBURY
CLARENDON HEIGHTS
NOE VALLEY
The Embarcadero
Battery St
California St
Divisadero St
Geary Blvd
Fell St
Stanyan St
Market St
Folsom St
Harrison St
Bryant St
16th St
17th St
18th St
24th St
29th St
Cesar Chavez
Portola Dr
time
Select Y Axis
04/01 07am
04/01 09am
0.5x
1x
2x
4x
Wed 03
Fri 05
Apr 07

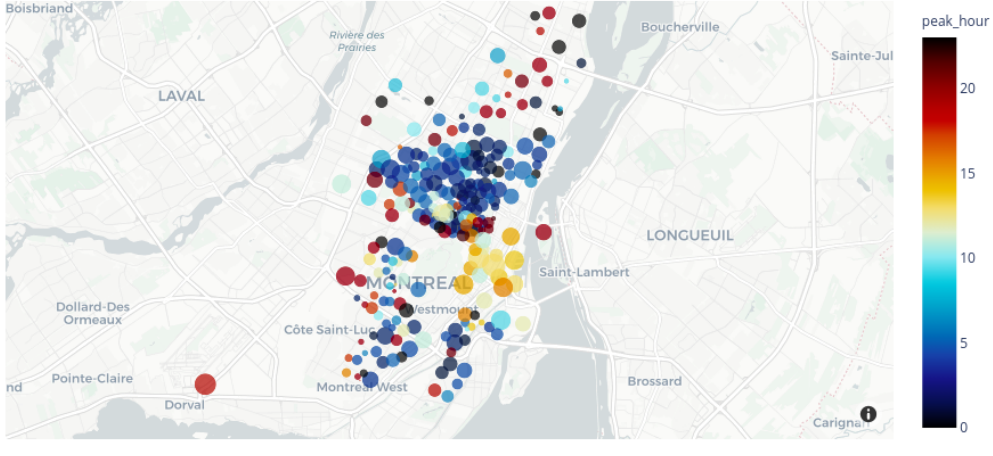

There is another

BOKEH

In [30]:
```
from bokeh.io import output_notebook
from bokeh.plotting import figure, output_file, show
from bokeh.tile_providers import CARTODBPOSITRON, get_provider
output_notebook()
tile_provider = get_provider(CARTODBPOSITRON)
p = figure(x_range=(-2000000, 6000000), y_range=(-1000000, 7000000),
           x_axis_type="mercator", y_axis_type="mercator")
p.add_tile(tile_provider)
show(p)
```

PLOTLY

In [29]:
```
import plotly.express as px
df = px.data.carshare()
fig = px.scatter_mapbox(df, lat="centroid_lat", lon="centroid_lon", color="peak_hour", size="car_hours",
                        color_continuous_scale=px.colors.cyclical.IceFire, size_max=15, zoom=10,
                        mapbox_style="carto-positron")
```



SAY HELLO TO GEOPATRA 👋

https://github.com/sangarshanan/geopatra (https://github.com/sangarshanan/geopatra)

https://geopatra.readthedocs.io/en/latest/ (https://geopatra.readthedocs.io/en/latest/)

In [17]:
```
import geopatra
gdf.folium.plot(zoom=2)
```

Out[17]:

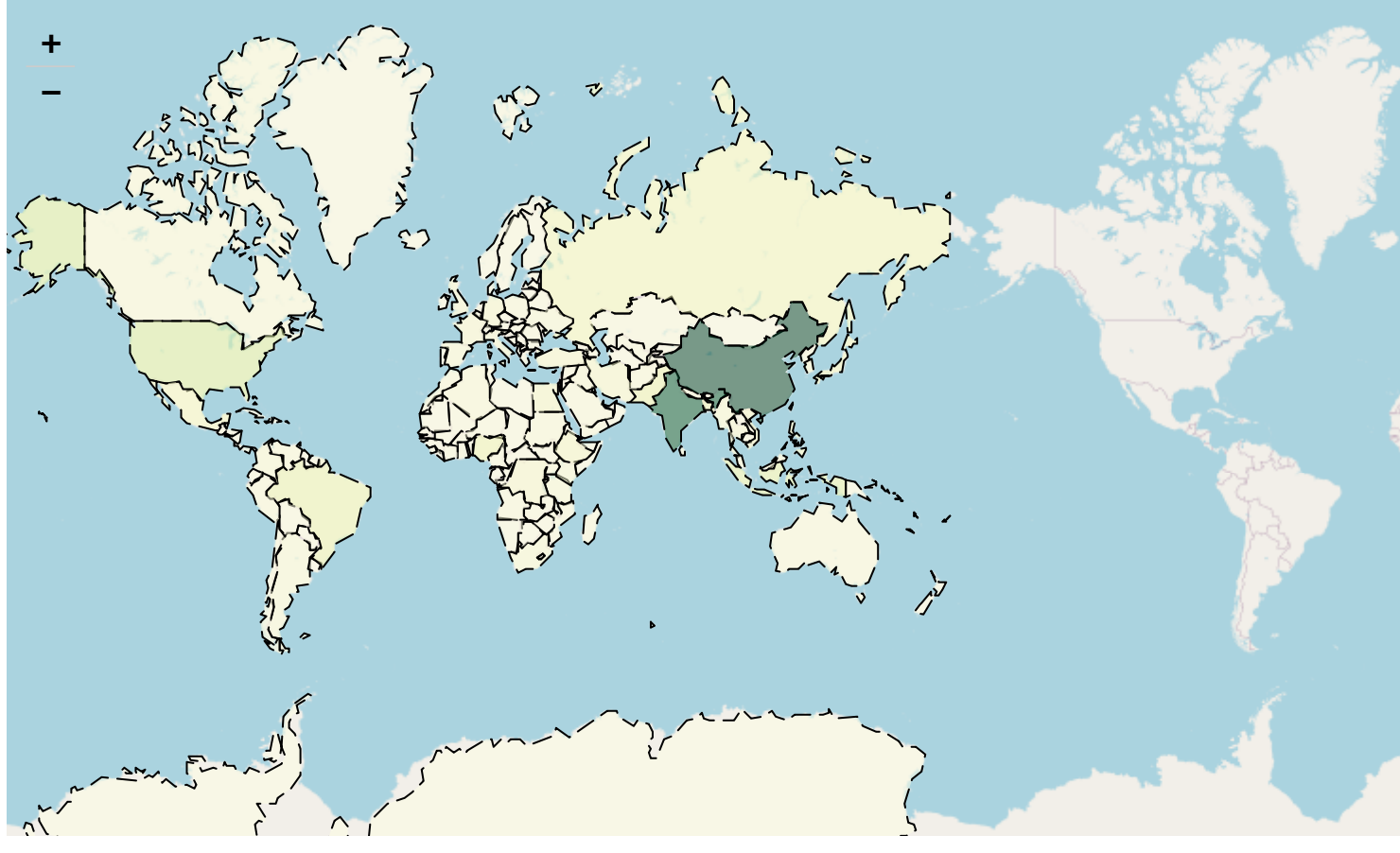
YET ANOTHER ONE, WHY ?

- Different libraries have different APIs
- All of them are awesome and have something new and exciting and offer
- Netflix Syndrome
- I wanna be able to switch between them without having to remember all the interfaces/ spend time googling

CHLOROPETH MAPS

In [18]:
```
world.folium.chloropeth(color_by= 'pop_est',
          color= 'green',
          zoom= 1,
          style = {'color': 'black',
                   'weight': 1,
                   'dashArray': '10, 5',
                   'fillOpacity': 0.5,
                  })
```

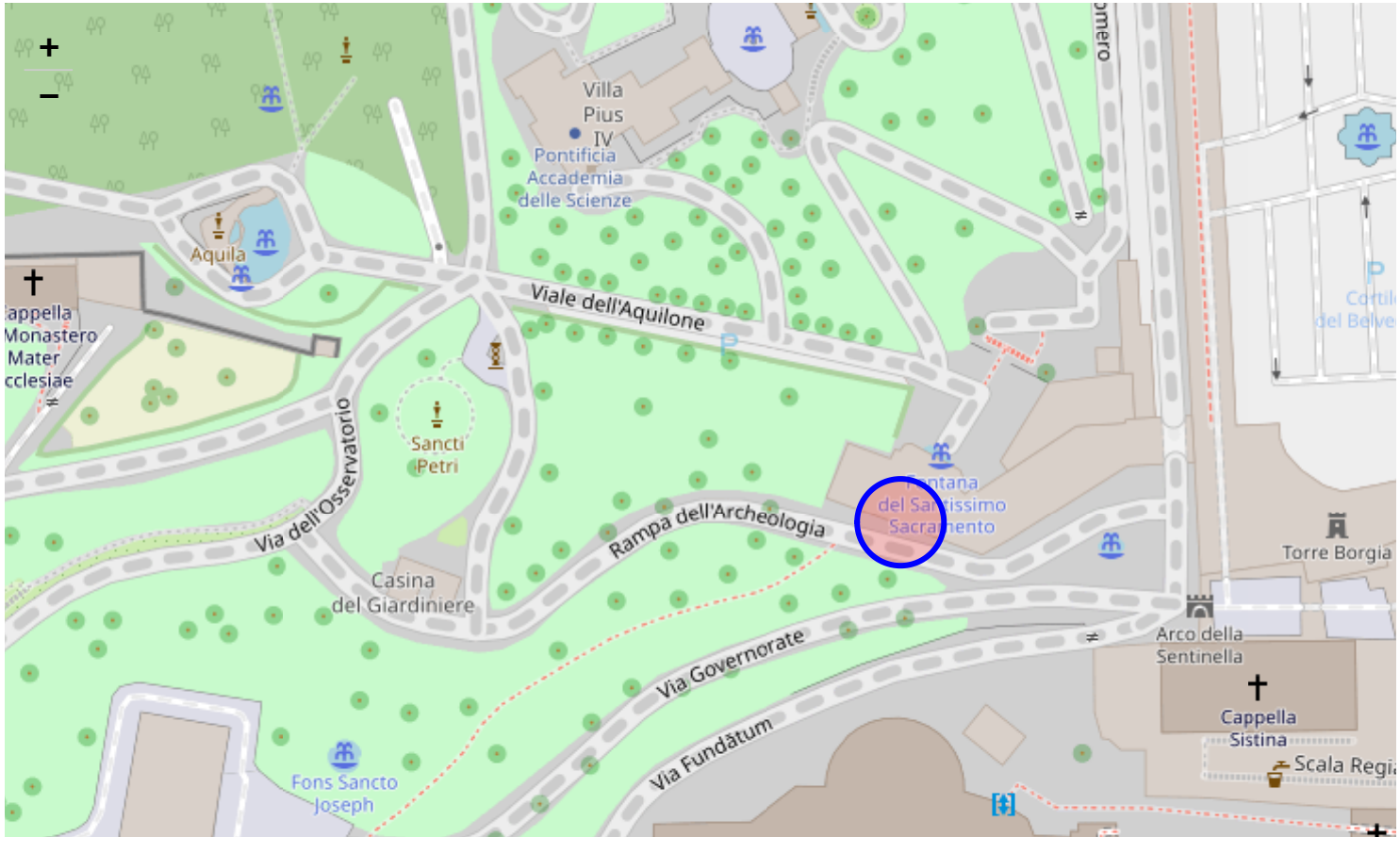
Out[18]:



CIRCLE PLOTS

In [19]:
```
gdf.folium.circle(radius=10, fill=True, fill_color='red', zoom=100, color='blue'
)
```

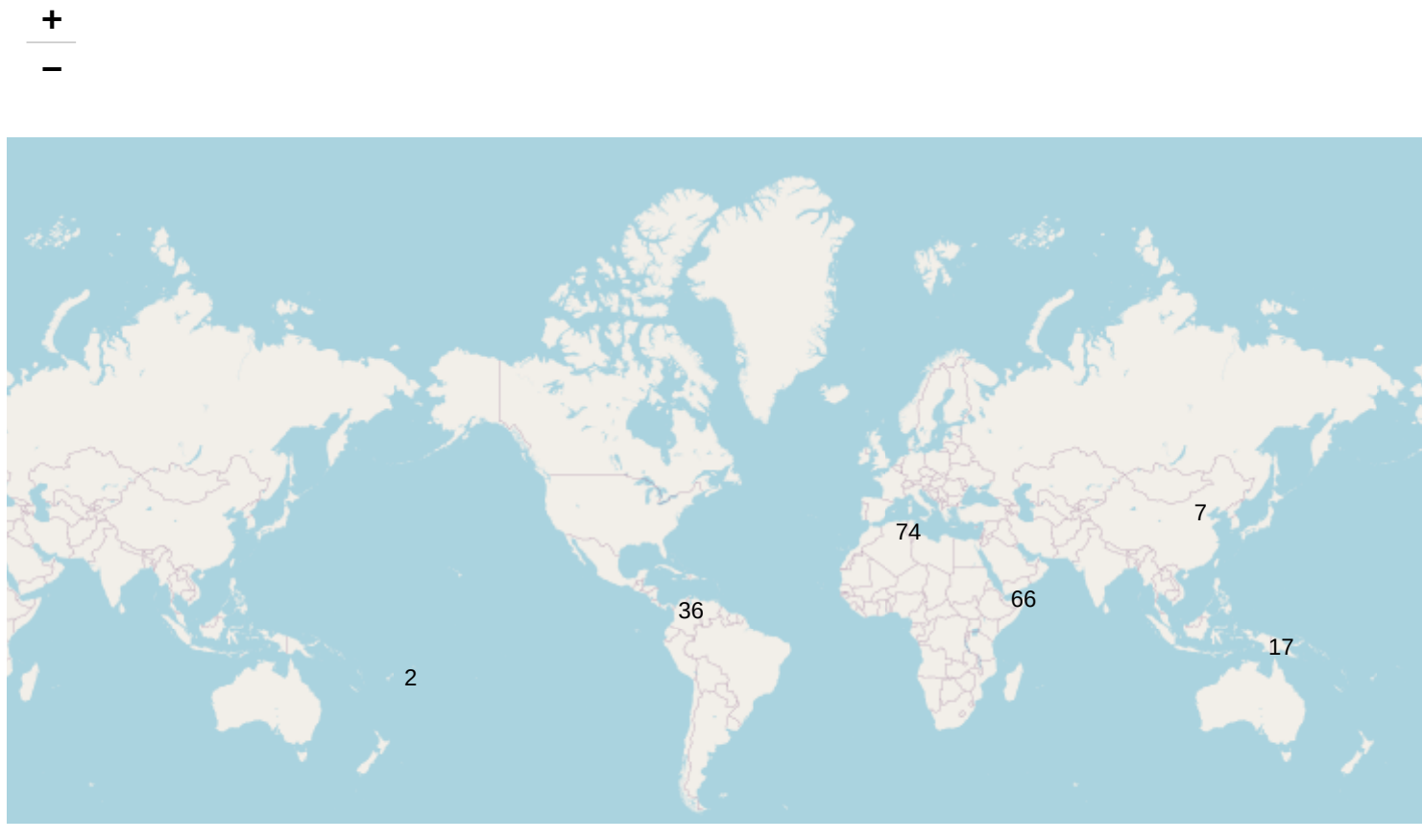
Out[19]:



MARKERCLUSTER

In [26]:
```
gdf.folium.markercluster(zoom=1, tooltip=["name"])
```

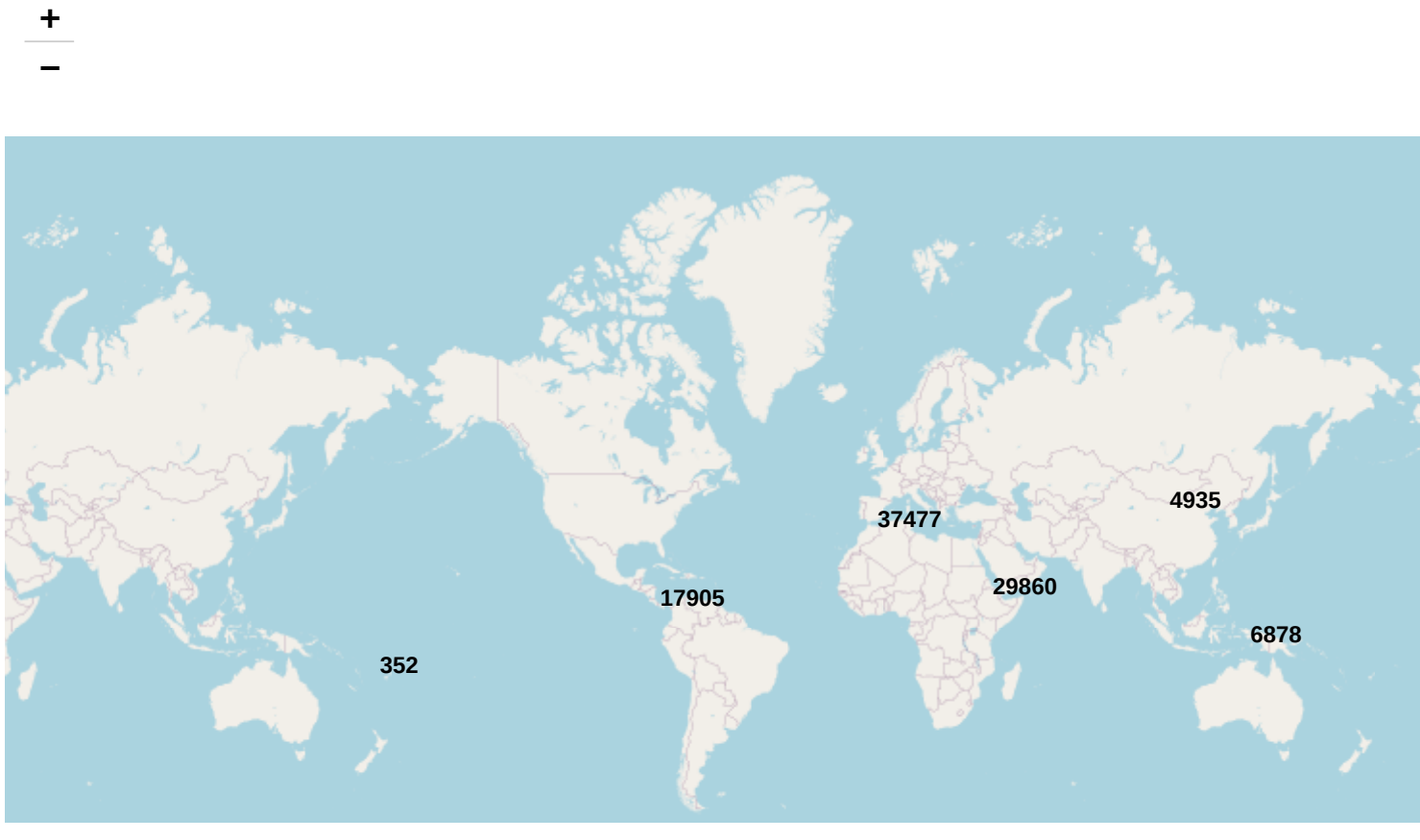
Out[26]:



WEIGHTED MARKERCLUSTER

In [27]:
```
import random
gdf['value'] = [int(random.randint(10, 1000)) for i in range(len(gdf))]
gdf.folium.markercluster(zoom=1, metric='sum', weight='value')
```

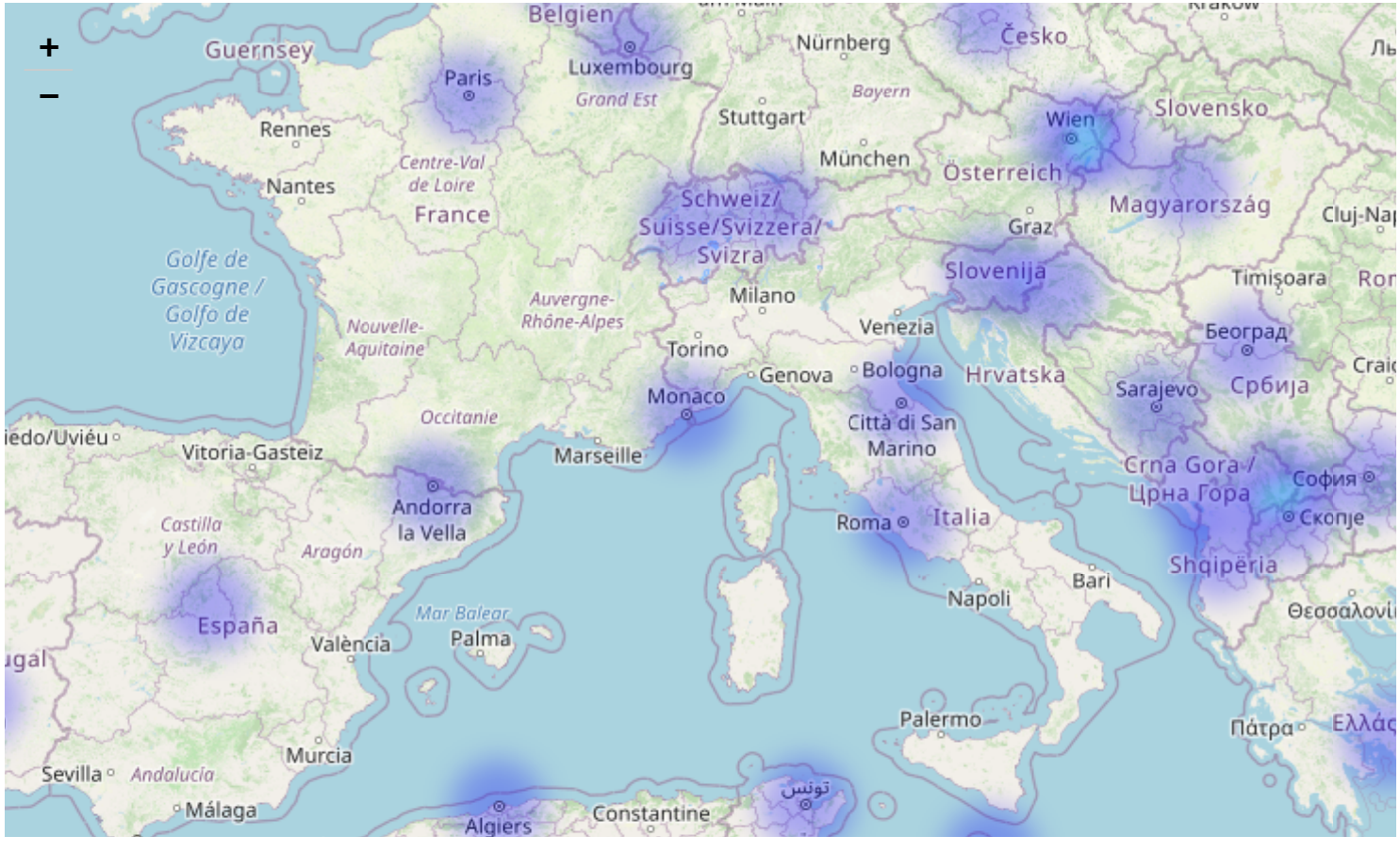
Out[27]:



HEATMAP

In [22]:
```
gdf.folium.heatmap(style={'min_opacity': 0.3}, zoom=5)
```

Out[22]:



KEPLER.GL

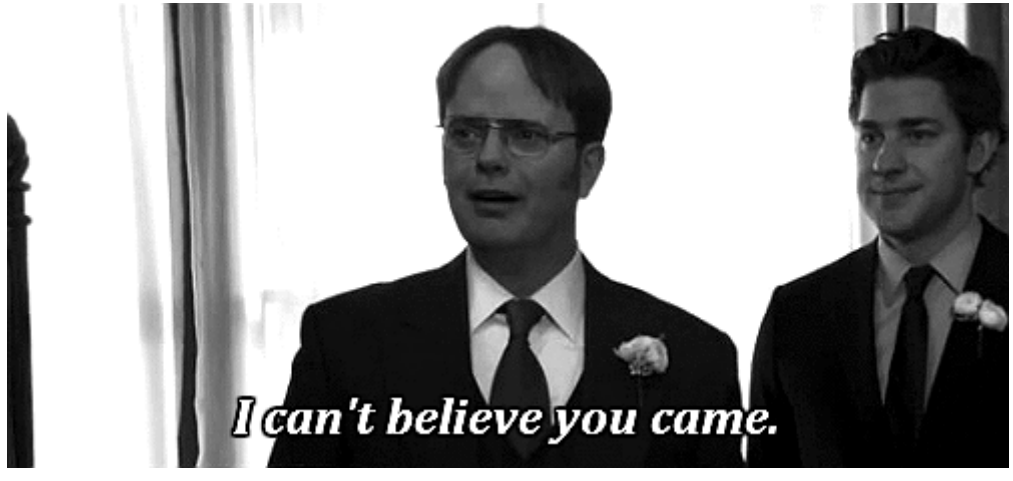
In [24]:

```
from IPython.display import IFrame
kmap1 = gdf.kepler.plot()
kmap1.save_to_html(file_name='./data/kepler_map1.html')
IFrame('./data/kepler_map1.html', width=900, height=500)
```

```
User Guide: https://docs.kepler.gl/docs/keplergl-jupyter
Map saved to ./data/kepler_map1.html!
```

Out[24]:


I AM SPEED
makeameme.org


I can't believe you came.