How to Visualize Data on top of a Map in Python using the Geoviews library

Christos Zeglis Nov 9, 2019 · 5 min read ★

So let's start with the problem we are about to tackle. Say you have some data that represent a specific figure (e.g. population) which differs from place to place (e.g. different cities) and you want to make a plot to visualize that data. How do you proceed with that?

One way to do that (and the most common one) is to create a bar plot. The y-axis represents the figure(e.g. population) and the x-axis represents the place (e.g. cities). I bet the vast majority of plots on that kind of data is of that type. As a result, there is a countless number of such examples on the web and therefore there is no need for me to add another one on the stack.

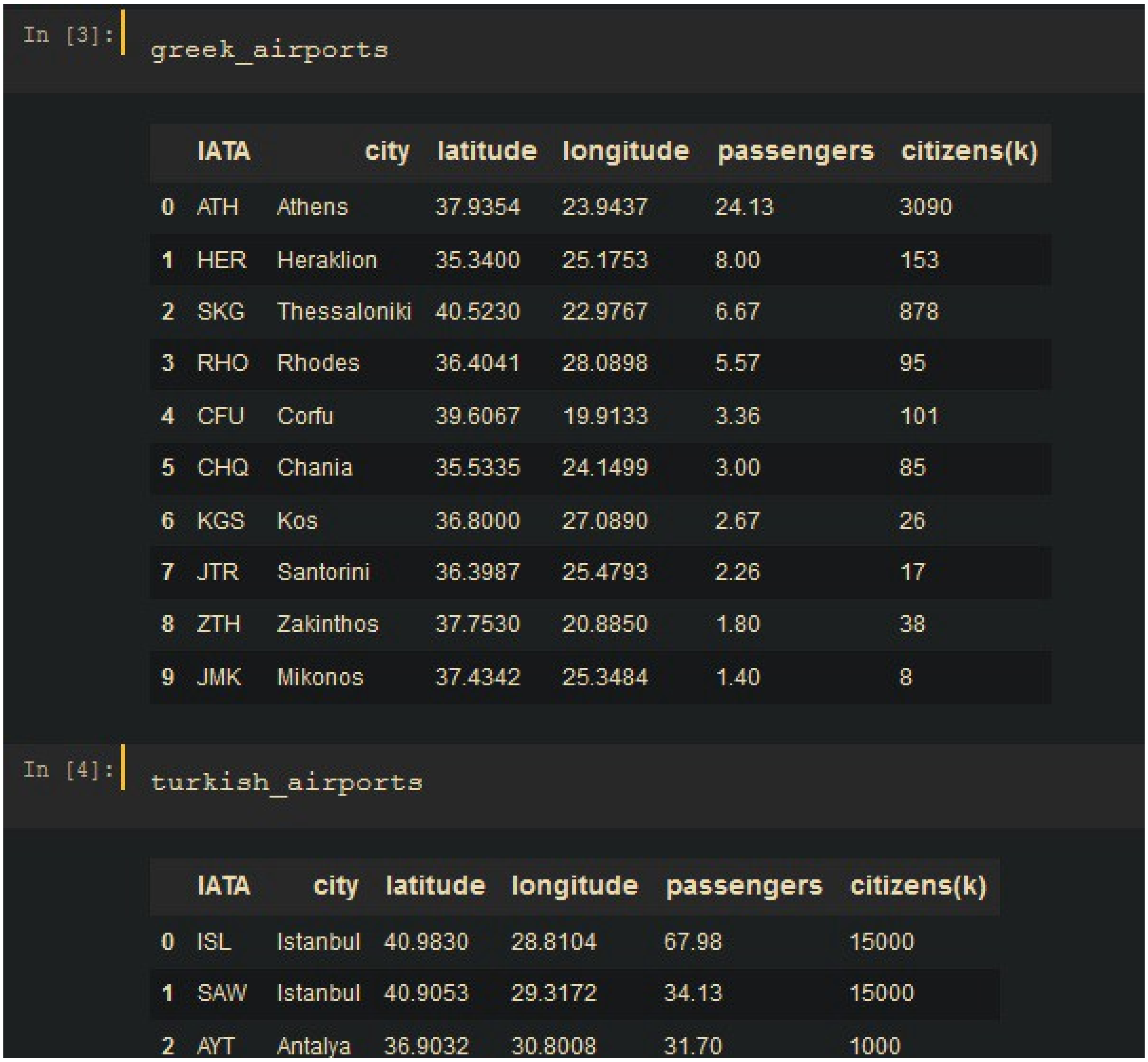
Fortunately, there is a better way to visualize that kind of data. Remember, plots have to be intuitive for the viewers to get a better grasp of what's in front of them. So, in this case, a more intuitive way to visualize that data would be to plot them on a map. What's more intuitive than an interactive map where you can zoom in, out, and over the place or figure you look for?

for comparison reasons.

First, we need to import the libraries and the methods we are about to use.

```
import pandas as pd
import numpy as np
import geoviews as gv
import geoviews.tile_sources as gvts
from geoviews import dim, opts
gv.extension('bokeh')
```

Our two dataframes, `greek_aiports` and `turkish_airports` , consist of the top 10 greek airports and the top 5 turkish airports in passengers volume respectively.

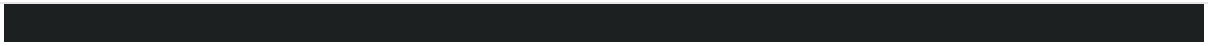


```
In [3]: greek_airports
```

	IATA	city	latitude	longitude	passengers	citizens(k)
0	ATH	Athens	37.9354	23.9437	24.13	3090
1	HER	Heraklion	35.3400	25.1753	8.00	153
2	SKG	Thessaloniki	40.5230	22.9767	6.67	878
3	RHO	Rhodes	36.4041	28.0898	5.57	95
4	CFU	Corfu	39.6067	19.9133	3.36	101
5	CHQ	Chania	35.5335	24.1499	3.00	85
6	KGS	Kos	36.8000	27.0890	2.67	26
7	JTR	Santorini	36.3987	25.4793	2.26	17
8	ZTH	Zakinthos	37.7530	20.8850	1.80	38
9	JMK	Mikonos	37.4342	25.3484	1.40	8

```
In [4]: turkish_airports
```

	IATA	city	latitude	longitude	passengers	citizens(k)
0	ISL	Istanbul	40.9830	28.8104	67.98	15000
1	SAW	Istanbul	40.9053	29.3172	34.13	15000
2	AYT	Antalya	36.9032	30.8008	31.70	1000



Stats for 2018

To these dataframes we will add an extra column `country` .

```
greek_airports['country']= 'GR'
turkish_airports['country']= 'TR'
```

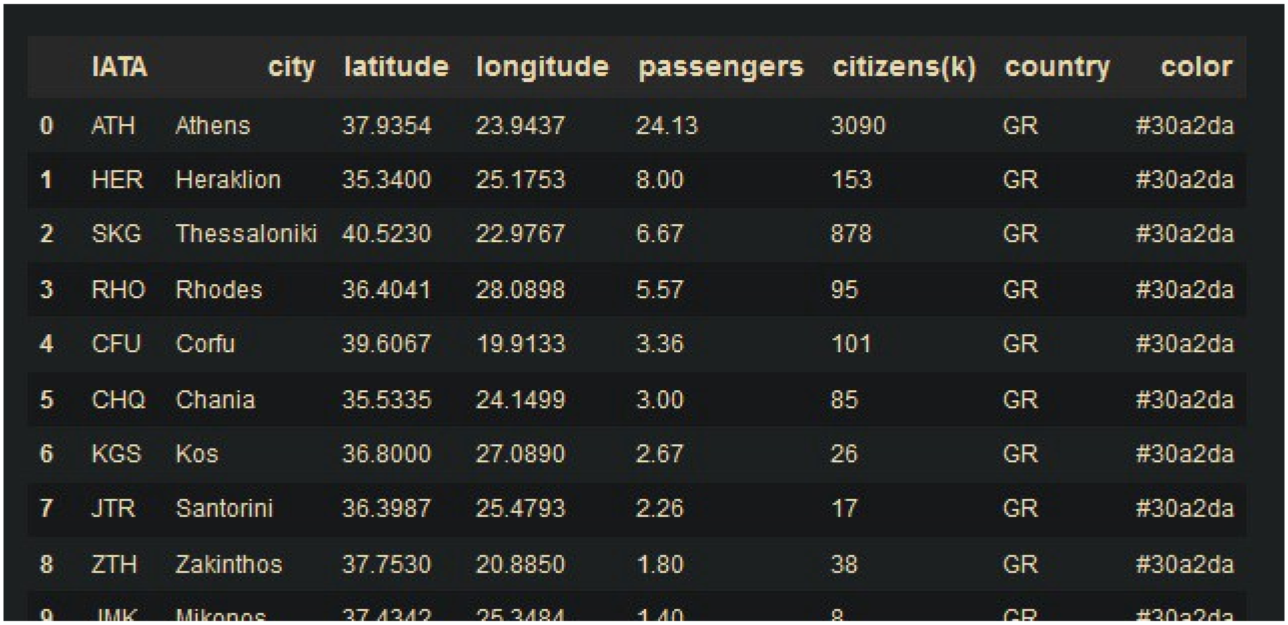
We will also add the column `color` . You will see later on why we did that.

```
greek_airports['color']= '#30a2da'
turkish_airports['color']= '#fc4f30'
```

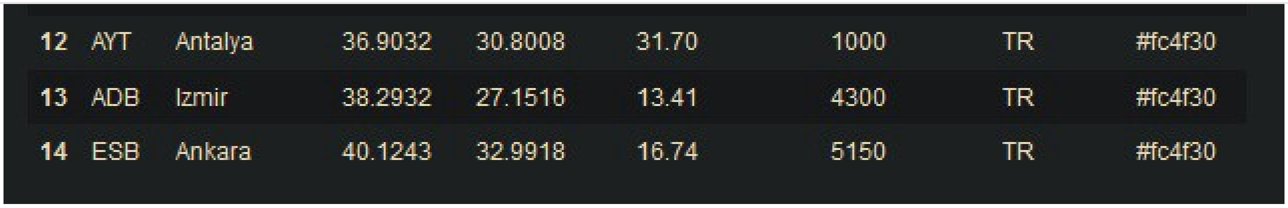
Now if we merge these two dataframes into `airports`

```
airports = pd.merge(greek_airports, turkish_airports, how='outer')
```

The `airports` dataframe will look like this.



	IATA	city	latitude	longitude	passengers	citizens(k)	country	color
0	ATH	Athens	37.9354	23.9437	24.13	3090	GR	#30a2da
1	HER	Heraklion	35.3400	25.1753	8.00	153	GR	#30a2da
2	SKG	Thessaloniki	40.5230	22.9767	6.67	878	GR	#30a2da
3	RHO	Rhodes	36.4041	28.0898	5.57	95	GR	#30a2da
4	CFU	Corfu	39.6067	19.9133	3.36	101	GR	#30a2da
5	CHQ	Chania	35.5335	24.1499	3.00	85	GR	#30a2da
6	KGS	Kos	36.8000	27.0890	2.67	26	GR	#30a2da
7	JTR	Santorini	36.3987	25.4793	2.26	17	GR	#30a2da
8	ZTH	Zakinthos	37.7530	20.8850	1.80	38	GR	#30a2da
9	JMK	Mikonos	37.4342	25.3484	1.40	8	GR	#30a2da



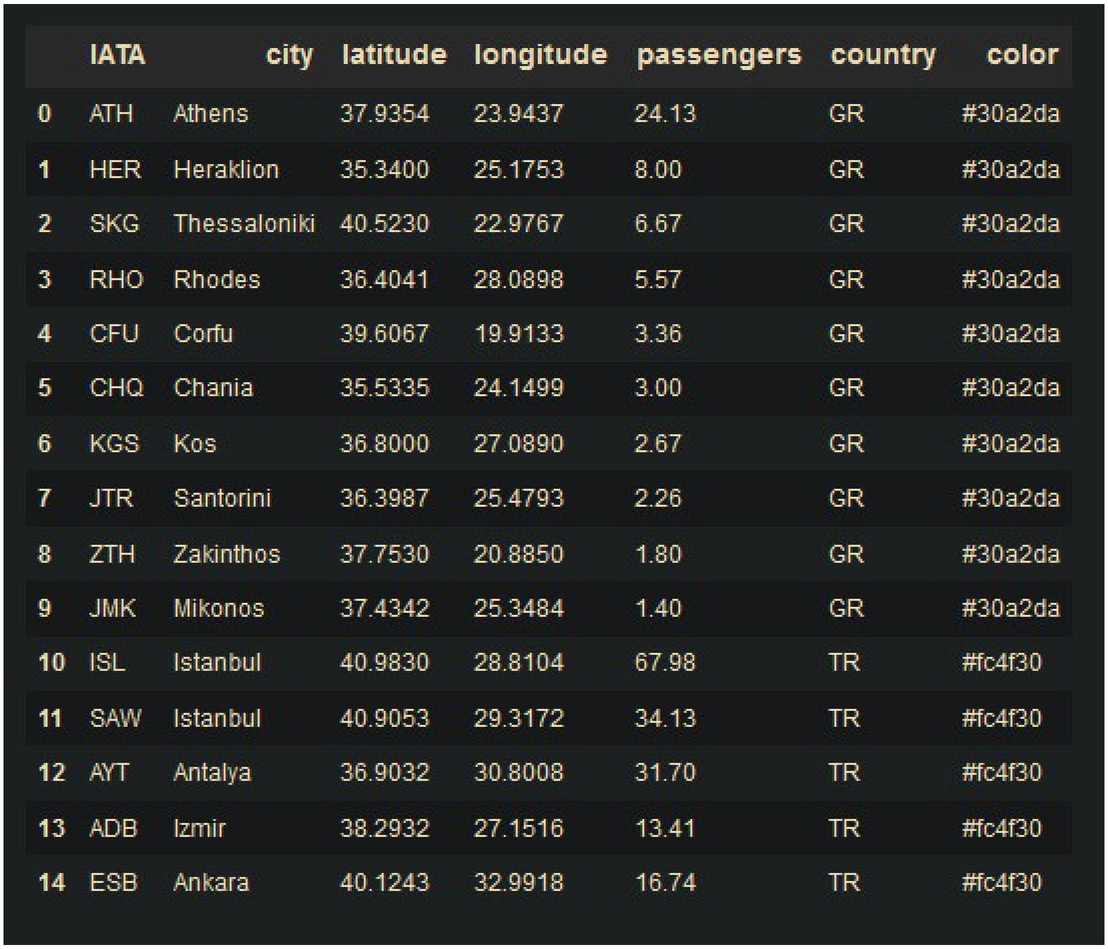
12	AYT	Antalya	36.9032	30.8008	31.70	1000	TR	#fc4f30
13	ADB	Izmir	38.2932	27.1516	13.41	4300	TR	#fc4f30
14	ESB	Ankara	40.1243	32.9918	16.74	5150	TR	#fc4f30

airports dataframe

I don’t need the `citizens(k)` column for that example so I will drop it.

```
airports.drop('citizens(k)', axis=1, inplace=True)
```

So the final `airports` dataframe will look like this.

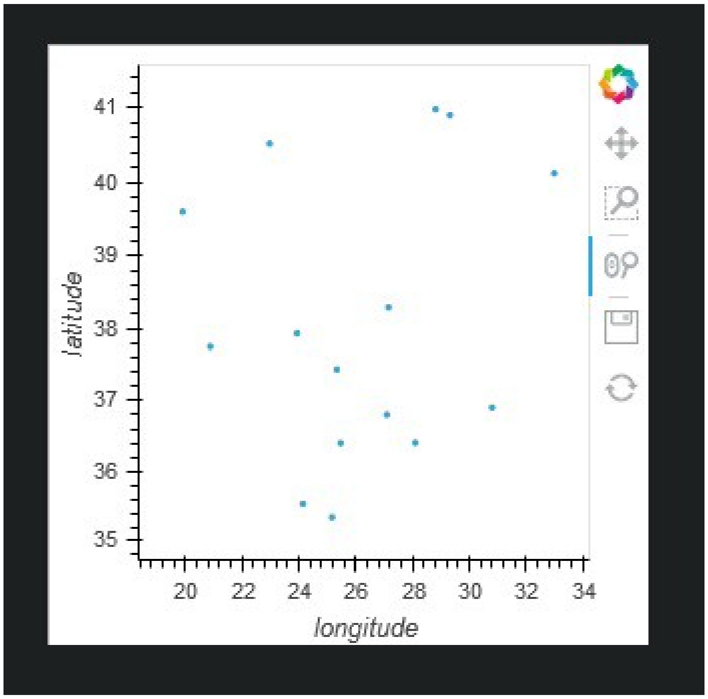


	IATA	city	latitude	longitude	passengers	country	color
0	ATH	Athens	37.9354	23.9437	24.13	GR	#30a2da
1	HER	Heraklion	35.3400	25.1753	8.00	GR	#30a2da
2	SKG	Thessaloniki	40.5230	22.9767	6.67	GR	#30a2da
3	RHO	Rhodes	36.4041	28.0898	5.57	GR	#30a2da
4	CFU	Corfu	39.6067	19.9133	3.36	GR	#30a2da
5	CHQ	Chania	35.5335	24.1499	3.00	GR	#30a2da
6	KGS	Kos	36.8000	27.0890	2.67	GR	#30a2da
7	JTR	Santorini	36.3987	25.4793	2.26	GR	#30a2da
8	ZTH	Zakinthos	37.7530	20.8850	1.80	GR	#30a2da
9	JMK	Mikonos	37.4342	25.3484	1.40	GR	#30a2da
10	ISL	Istanbul	40.9830	28.8104	67.98	TR	#fc4f30
11	SAW	Istanbul	40.9053	29.3172	34.13	TR	#fc4f30
12	AYT	Antalya	36.9032	30.8008	31.70	TR	#fc4f30
13	ADB	Izmir	38.2932	27.1516	13.41	TR	#fc4f30
14	ESB	Ankara	40.1243	32.9918	16.74	TR	#fc4f30

airports dataframe

Now let's start using the geoviews module. In specific, let's use the `geoviews.Points` function to create a plot with our points.

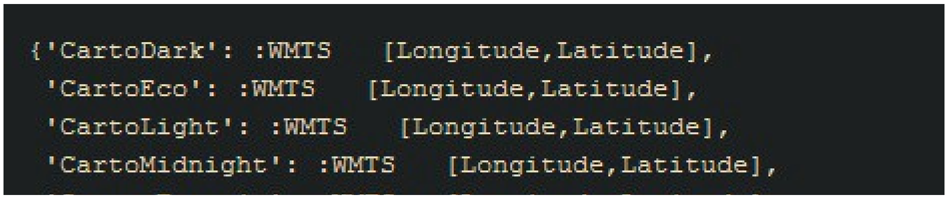
```
airports_gv_points = gv.Points(airports, ['longitude', 'latitude'],
                               ['IATA', 'city', 'passengers',
                               'country', 'color'])
```



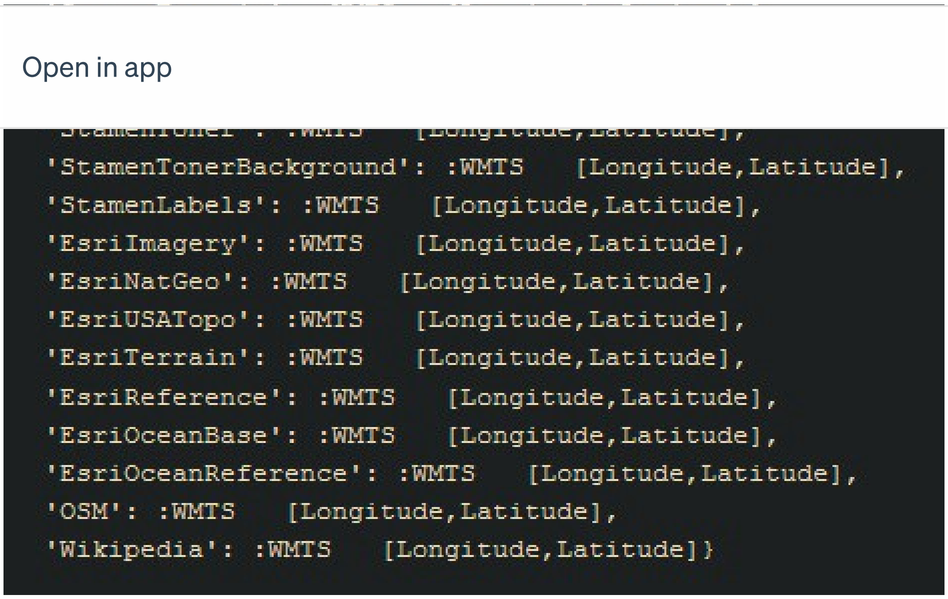
airports_gv_points

In order to plot these points on a map we need a… map. The geoviews module offers a lot of tilemaps we can use. We can see what's available if we type in

```
gvts.tile_sources
```



```
{'CartoDark': :WMTS   [Longitude,Latitude],
 'CartoEco': :WMTS   [Longitude,Latitude],
 'CartoLight': :WMTS   [Longitude,Latitude],
 'CartoMidnight': :WMTS   [Longitude,Latitude],
```

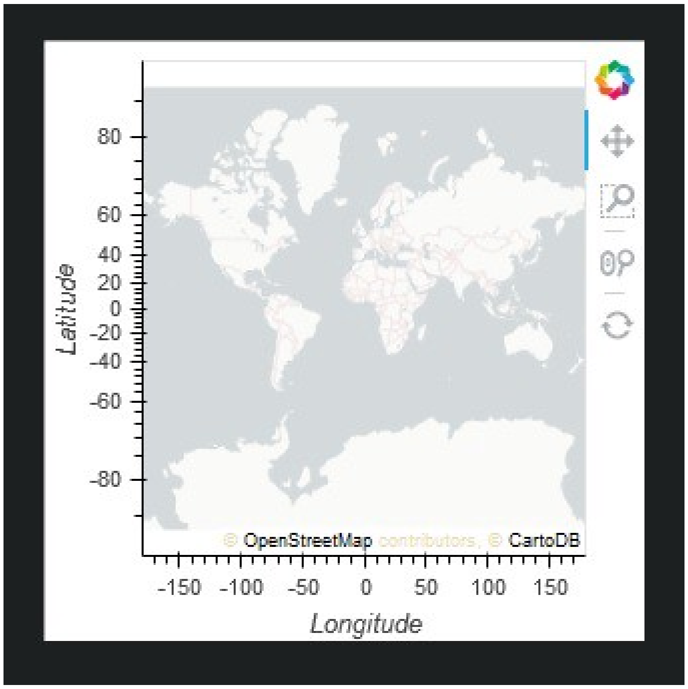


```
 'StamenToner': :WMTS   [Longitude,Latitude],
 'StamenTonerBackground': :WMTS   [Longitude,Latitude],
 'StamenLabels': :WMTS   [Longitude,Latitude],
 'EsriImagery': :WMTS   [Longitude,Latitude],
 'EsriNatGeo': :WMTS   [Longitude,Latitude],
 'EsriUSATopo': :WMTS   [Longitude,Latitude],
 'EsriTerrain': :WMTS   [Longitude,Latitude],
 'EsriReference': :WMTS   [Longitude,Latitude],
 'EsriOceanBase': :WMTS   [Longitude,Latitude],
 'EsriOceanReference': :WMTS   [Longitude,Latitude],
 'OSM': :WMTS   [Longitude,Latitude],
 'Wikipedia': :WMTS   [Longitude,Latitude]}
```

Available tilemaps

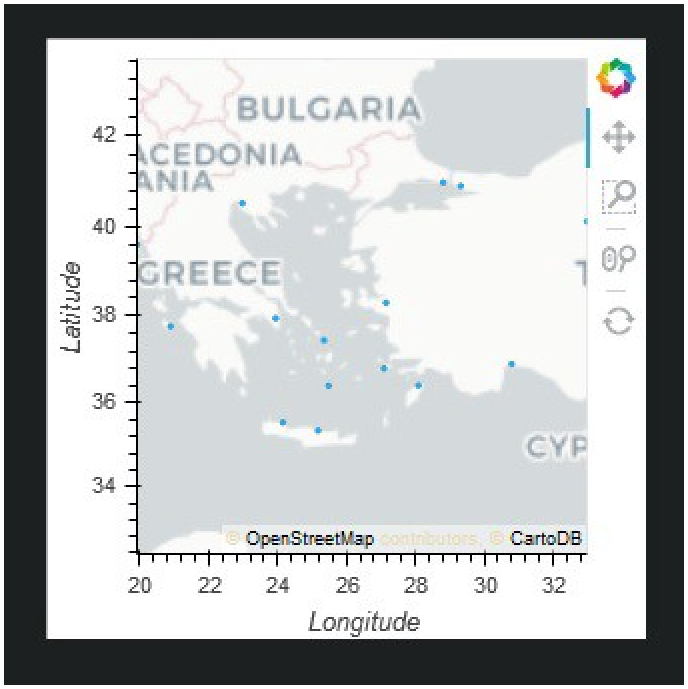
For that example let's use the CartoLight tilemap. Let's see what we get for

```
gvts.CartoLight
```



CartoLight

Now, we can plot the points on top of CartoLight with



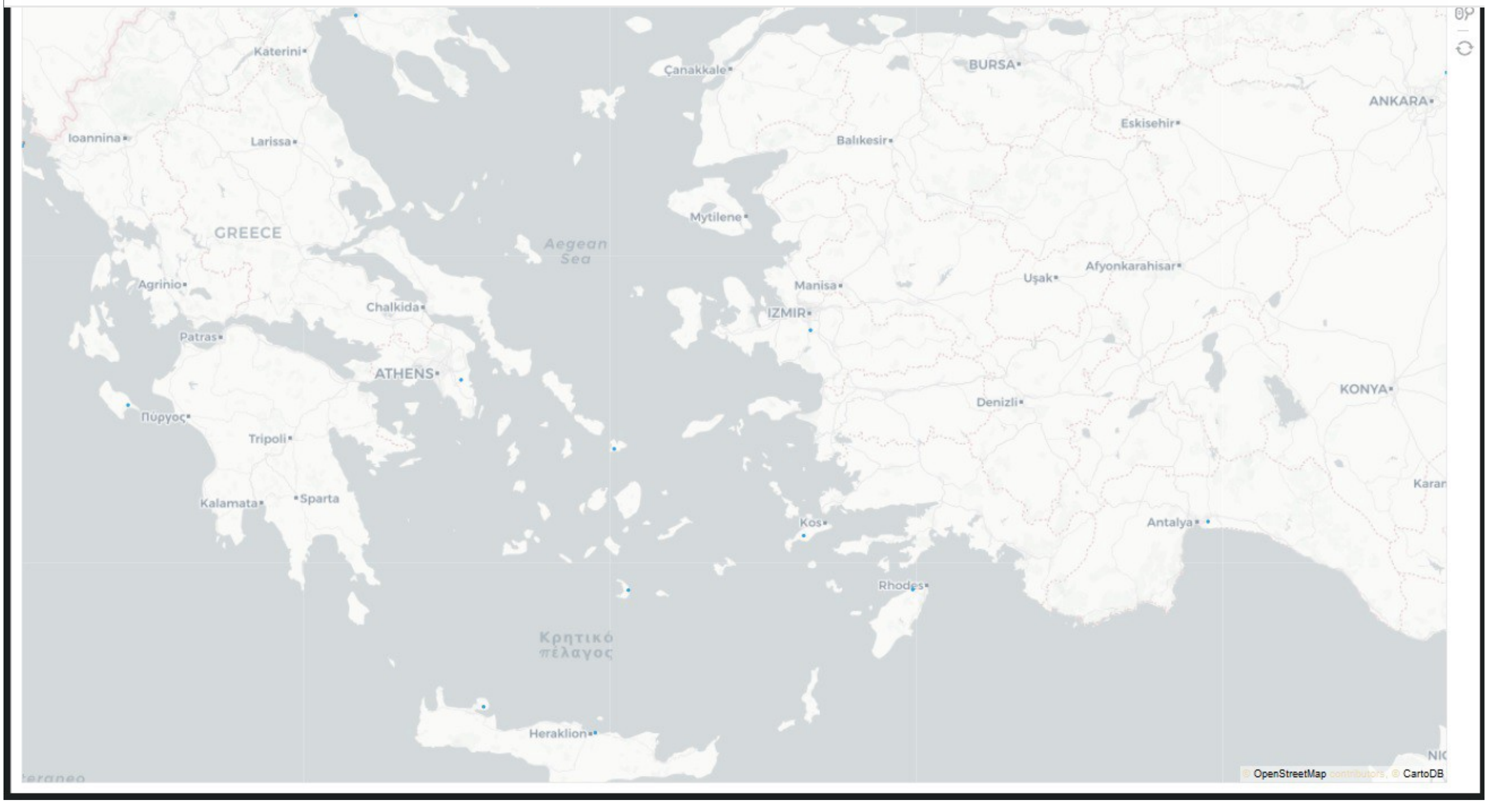
gv.Points on CartoLight

There are a few issues here:

- The plot dimensions are really small.
- I don't like axis labels for latitude and longitude.
- I don't like grids on maps.

I can fix these by adding a few options on the `gvts.CartoLight` like

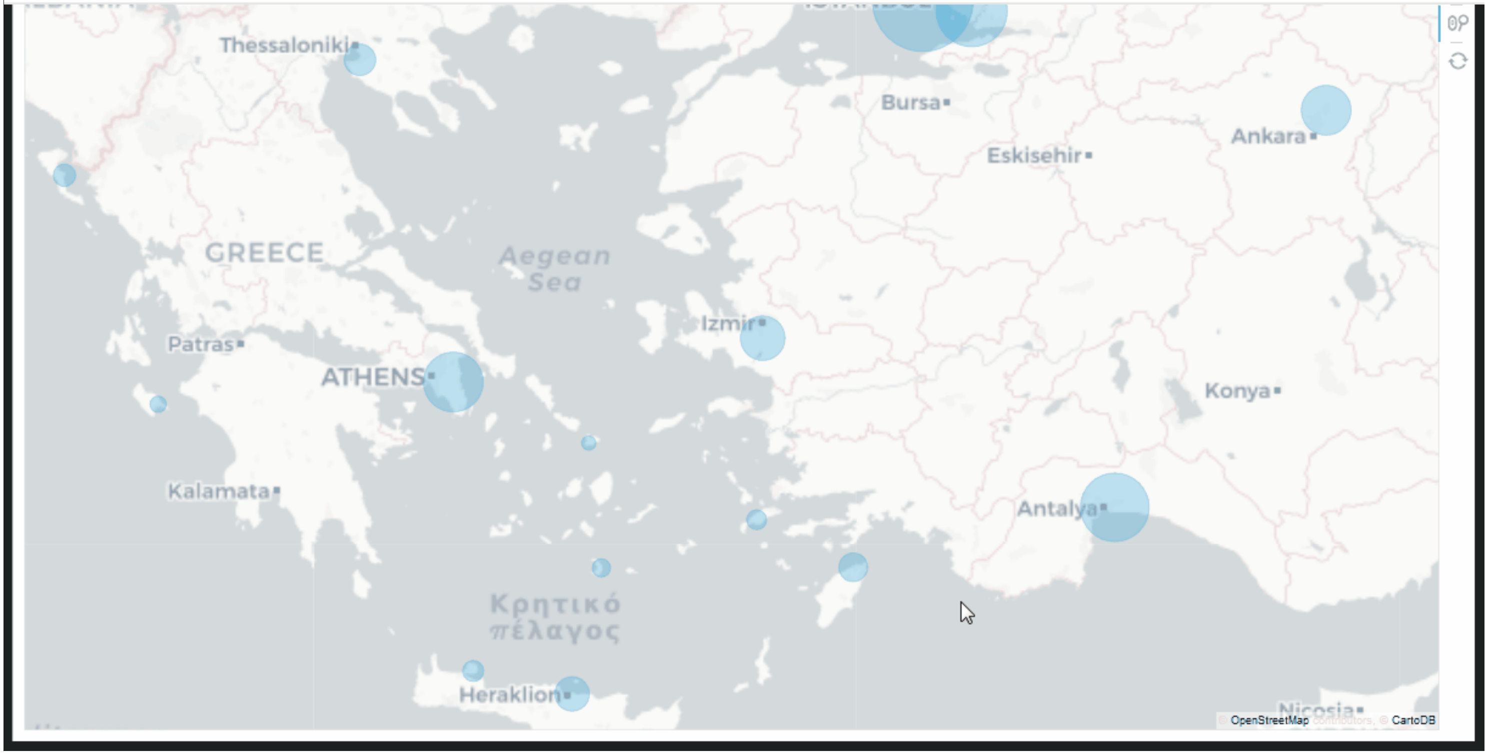
```
gvts.CartoLight.options(width=1300, height=800, xaxis=None,
yaxis=None, show_grid=False)  * airports_gv_points
```



geoviews.Points on CartoLight

Now, because the points are still just dots, and therefore they cannot illustrate the volume of passengers, I will use the `opts` method we imported before. The parameter we need is `size` . Moreover, in order to get the numbers somewhat closer, I will use the `np.sqrt` function to get their square roots.

```
airports_plot = (gvts.CartoLight * airports_gv_points).opts(
    opts.Points(width=1200, height=700, alpha=0.3,
                xaxis=None, yaxis=None,
                size=np.sqrt(dim('passengers'))*10))
```

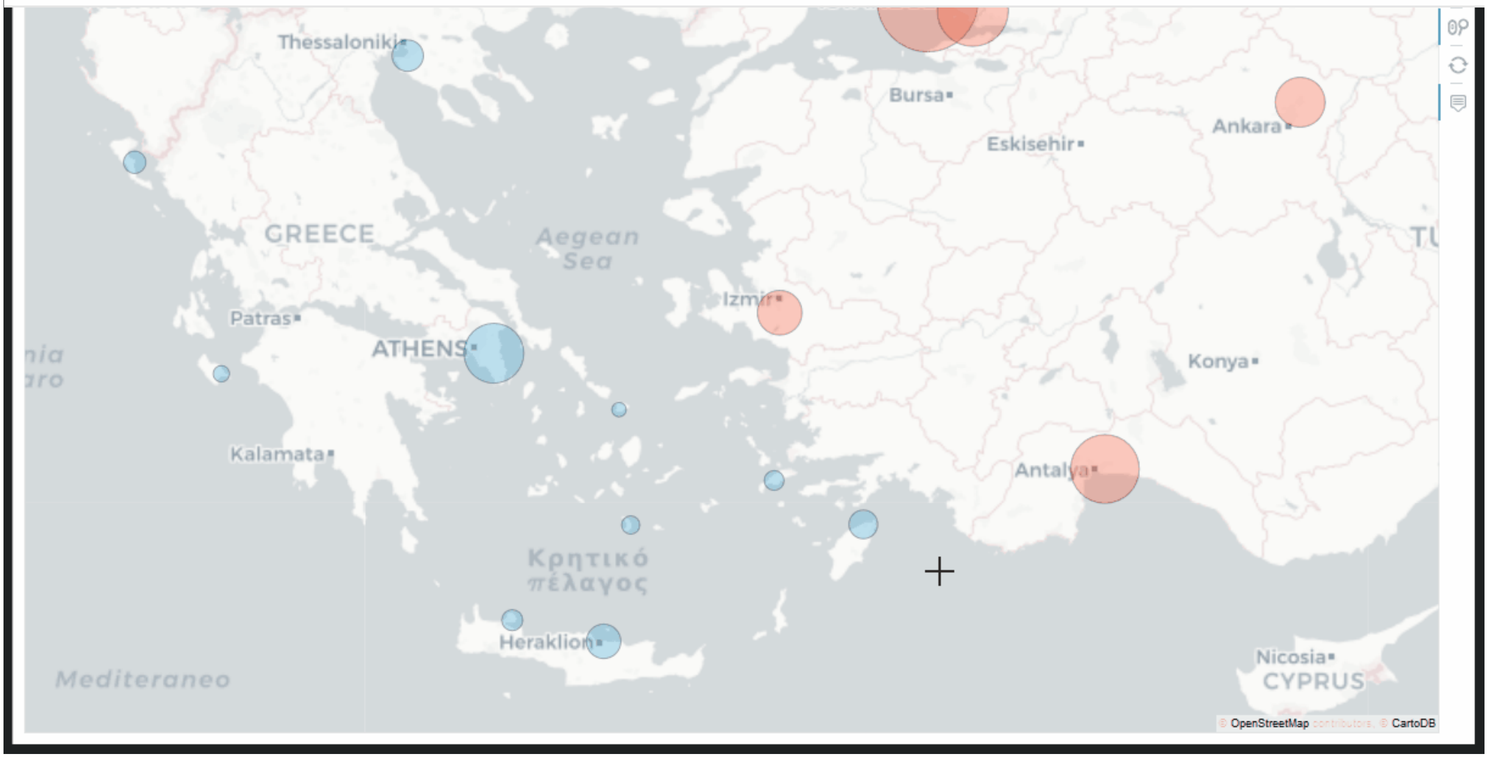


airports_plot

As you can see, we now have our interactive map that illustrates how busy the airports in Greece and in Turkey are. Obviously, Athens has the busiest airport in Greece whereas the busiest airport in Turkey is located in Istanbul.

To make the plot better, we can use different colors for each country while we can also add a hover tool to get some information about the airports once we move the cursor above them. To add color we add the parameter `color=dim('color')` which uses the color we specified on the `color` column, and for the hover tool we add the parameter `tools=['hover']` .

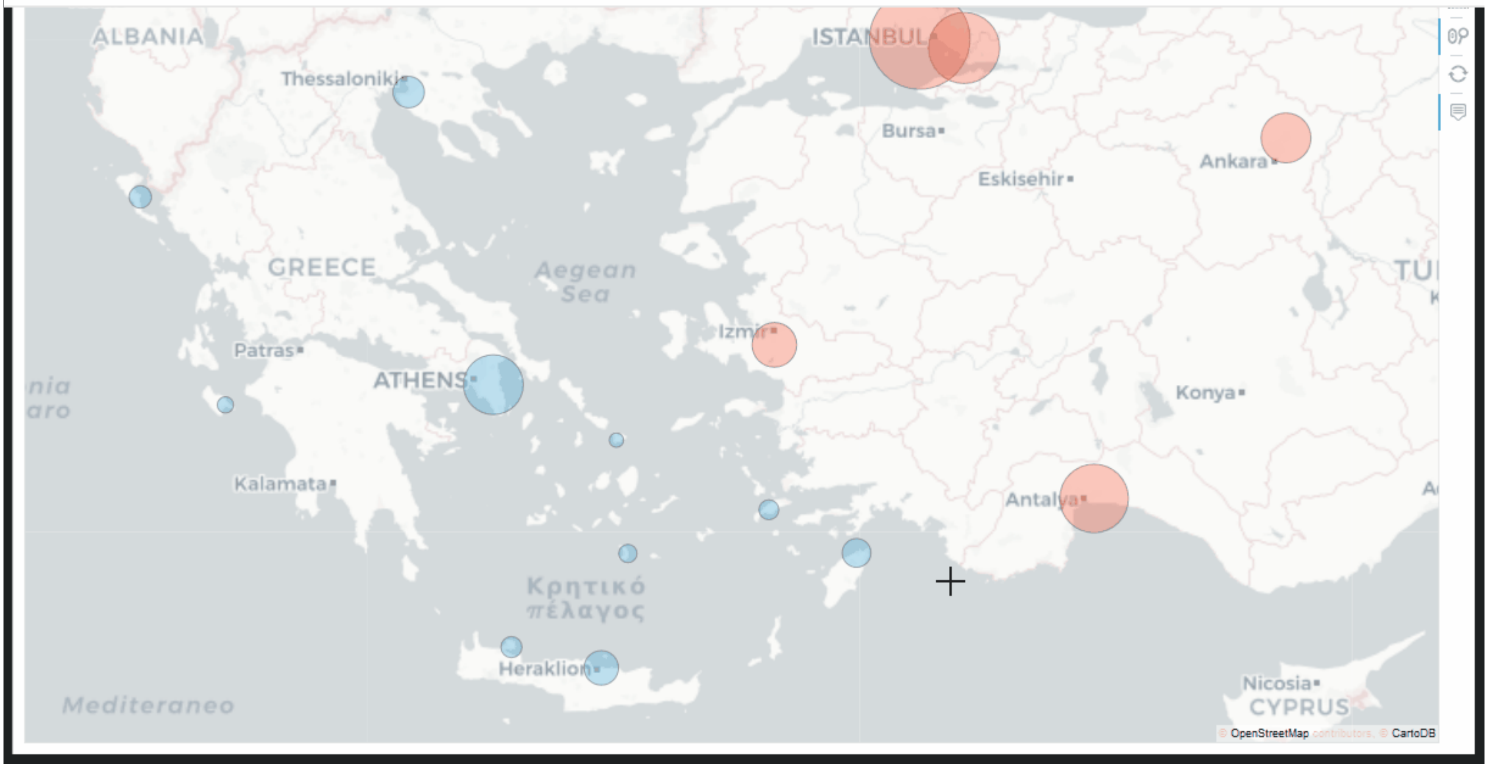
```
airports_plot = (gvts.CartoLight * airports_gv_points).opts(
    opts.Points(width=1200, height=700, alpha=0.3,
                color=dim('color'), hover_line_color='black',
                line_color='black', xaxis=None, yaxis=None,
                tools=['hover'],size=np.sqrt(dim('passengers'))*10))
```



airports_plot

We can even create our own hover tool to get 100% control of what the hover tool shows. For example, I don't want it to show the color code used for each country. Lastly, I want the color of each point to darken a little bit when I pass the cursor over them so I add the parameter `hover_fill_alpha=0.5` .

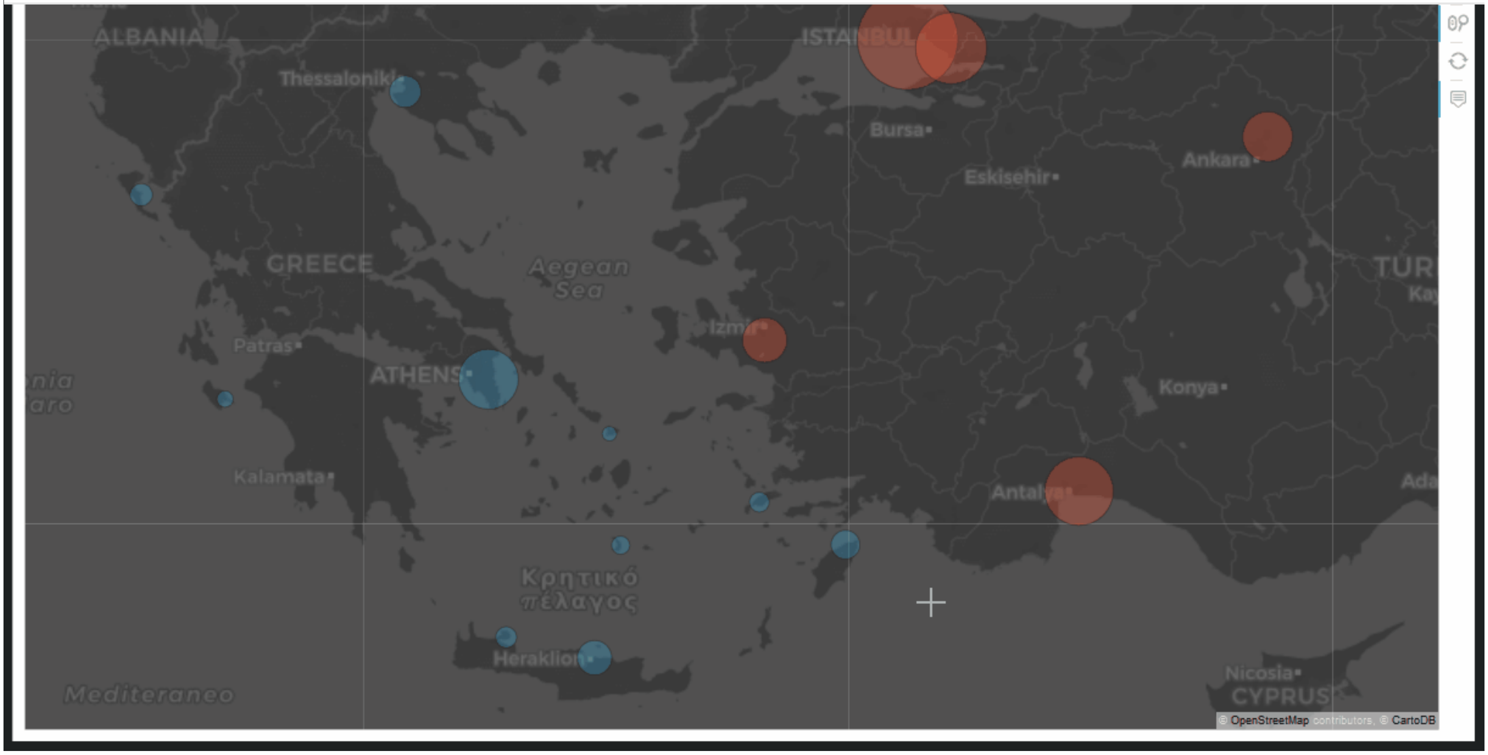
```
from bokeh.models import HoverTool
tooltips = [('IATA', '@IATA'),
            ('Passengers', '@passengers{0.00 a}m'),
            ('City', '@city'),
            ('Country', '@country'),
            ('Longitude', '$x'),
            ('Latitude', '$y'),
            ]
hover = HoverTool(tooltips=tooltips)

airports_plot = (gvts.CartoLight * airports_gv_points).opts(
    opts.Points(width=1200, height=700, alpha=0.3,
                color=dim('color'), hover_line_color='black',
                line_color='black', xaxis=None, yaxis=None,
                tools=[hover],size=np.sqrt(dim('passengers'))*10,
                hover_fill_color=None, hover_fill_alpha=0.5))
```



airports_plot

If you are a fan of dark color themes, as I am, you can always use the `gvts.CartoDark` tilemap.

```
airports_plot = (gvts.CartoDark.options(alpha=0.8) *
airports_gv_points).opts(
    opts.Points(width=1200, height=700, alpha=0.3,
                color=dim('color'), hover_line_color='black',
                line_color='black', xaxis=None, yaxis=None,
                tools=[hover],size=np.sqrt(dim('passengers'))*10,
                hover_fill_color=None, hover_fill_alpha=0.5))
```



airports_plot

That’s it for today. Hope you found it useful. Till next time!

You can always find me on LinkedIn by clicking here.