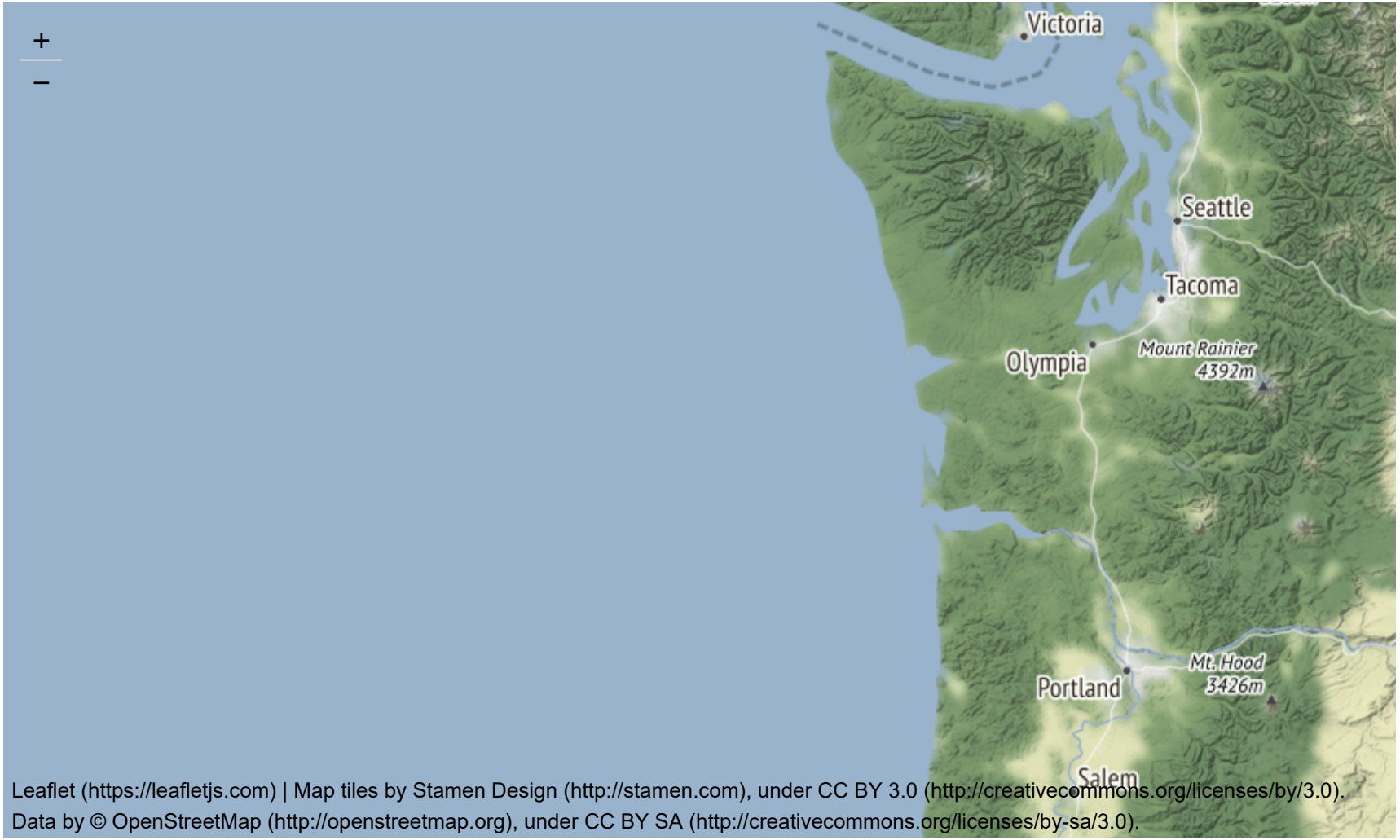
In [ ]:
```
import pandas as pd
import numpy as np
import folium
```

In [ ]:
```
world_map = folium.Map(location = [45.5, -122.675], zoom_start = 7, tiles = 'Stamen Terrain')
world_map
```

Out[ ]:



In [ ]:
```
df_incidents = pd.read_csv('police_incidents.csv')
```

```
print('Dataset downloaded and read into a pandas dataframe!')
```

```
Dataset downloaded and read into a pandas dataframe!
```

In [ ]:

```
df_incidents.head()
```

Out[ ]:

	IncidntNum	Category	Descript	DayOfWeek	Date	Time	PdDistrict	Resolution	Address	X	Y
0	120058272	WEAPON LAWS	POSS OF PROHIBITED WEAPON	Friday	01/29/2016 12:00:00 AM	11:00	SOUTHERN	ARREST, BOOKED	800 Block of BRYANT ST	-122.403405	37.775421
1	120058272	WEAPON LAWS	FIREARM, LOADED, IN VEHICLE, POSSESSION OR USE	Friday	01/29/2016 12:00:00 AM	11:00	SOUTHERN	ARREST, BOOKED	800 Block of BRYANT ST	-122.403405	37.775421
2	141059263	WARRANTS	WARRANT ARREST	Monday	04/25/2016 12:00:00 AM	14:59	BAYVIEW	ARREST, BOOKED	KEITH ST / SHAFTER AV	-122.388856	37.729981
3	160013662	NON-CRIMINAL	LOST PROPERTY	Tuesday	01/05/2016 12:00:00 AM	23:50	TENDERLOIN	NONE	JONES ST / OFARRELL ST	-122.412971	37.785788
4	160002740	NON-CRIMINAL	LOST PROPERTY	Friday	01/01/2016 12:00:00 AM	00:30	MISSION	NONE	16TH ST / MISSION ST	-122.419672	37.765050

In [ ]:

```
df_incidents.shape
```

Out[ ]:

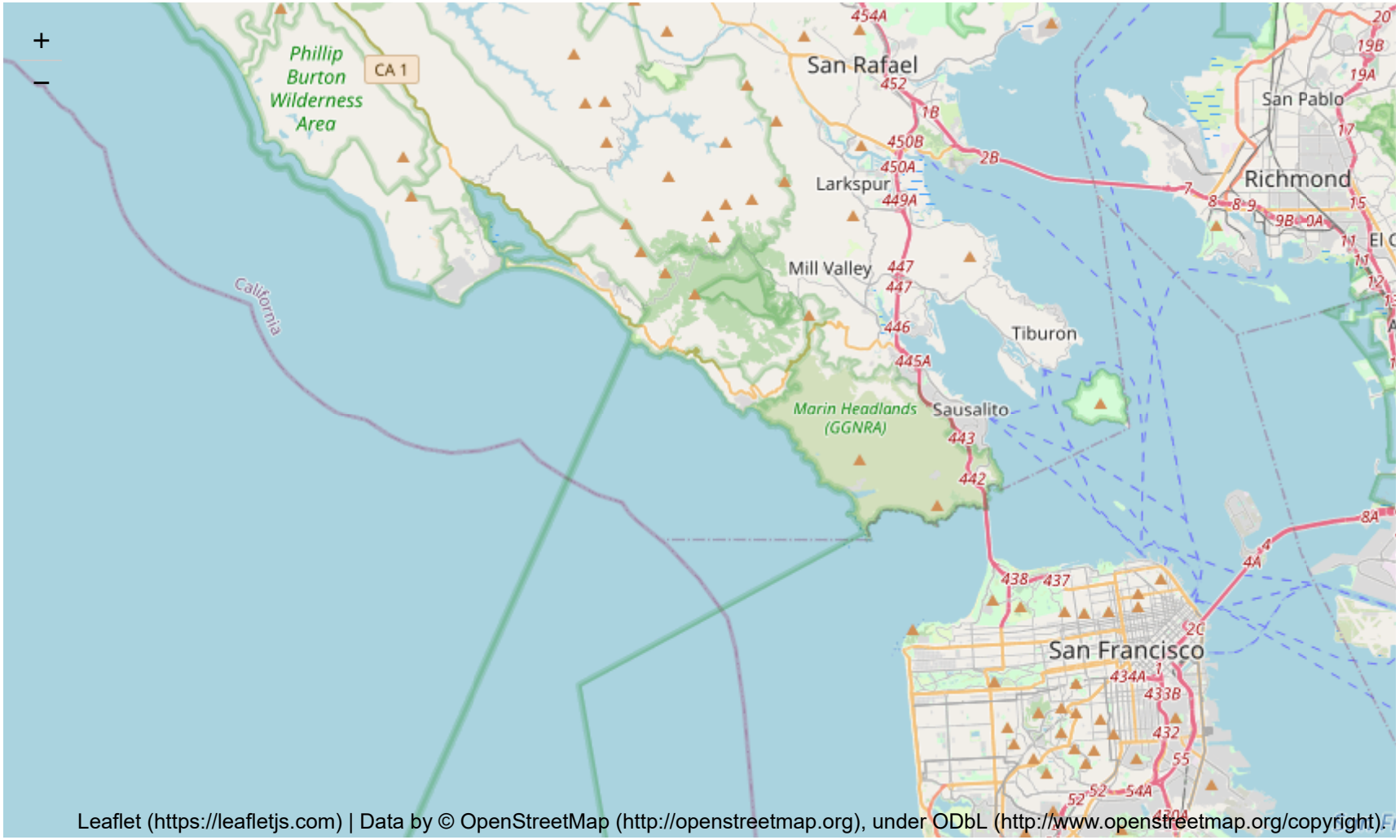
```
(150500, 13)
```

In [ ]:

```
df_incidents_short = df_incidents.iloc[0:100, :]
df_incidents_short.shape
```

Out[ ]:
```
(100, 13)
```

In [ ]:
```
latitude = 37.77
longitude = -122.42
sanfran_map = folium.Map(location = [latitude, longitude], zoom_start = 11)
sanfran_map
```

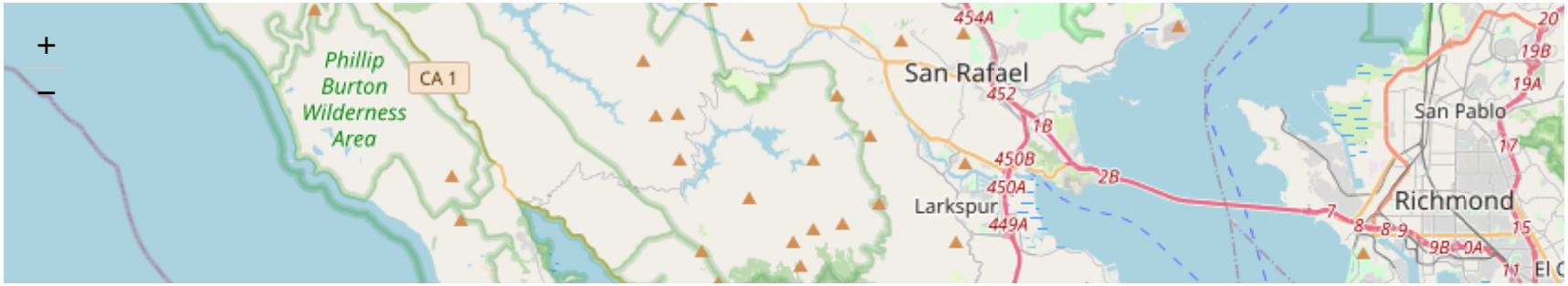
Out[ ]:


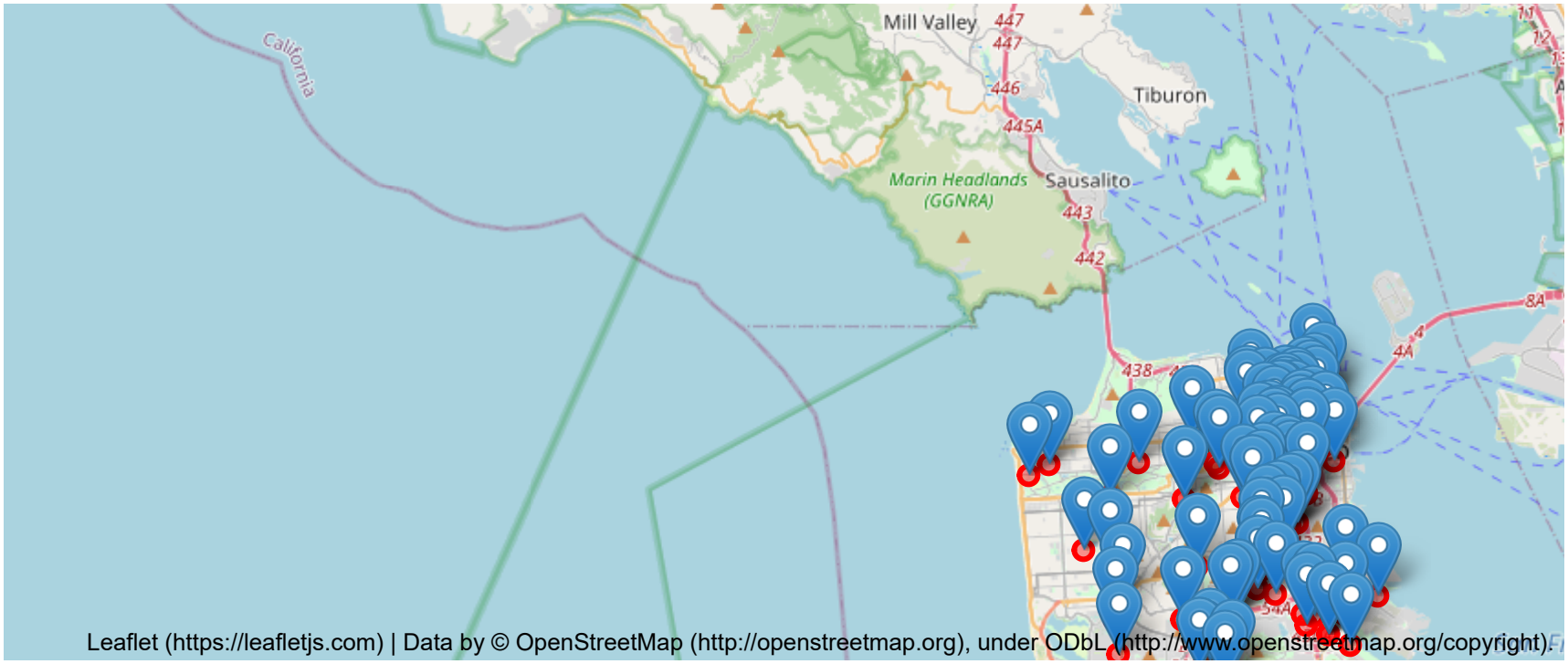
In [ ]:
```
df_incidents.X
```

Out[ ]:
```
0         -122.403405
1         -122.403405
2         -122.388856
3         -122.412971
4         -122.419672
              ...
150495    -122.453982
150496    -122.401857
150497    -122.412269
150498    -122.406659
150499    -122.403405
Name: X, Length: 150500, dtype: float64
```

In [ ]:
```
incidents = folium.map.FeatureGroup()

#Loop through crimes and add to feature gorup
for lat, lng in zip(df_incidents_short.Y, df_incidents_short.X):
    incidents.add_child(
        folium.features.CircleMarker(
        [lat, lng],
        radius = 5,
        color = 'red',
        fill = True,
        fill_color = 'red',
        fill_opacity = 0.4
    ))

latitudes = list(df_incidents_short.Y)
longitudes = list(df_incidents_short.X)
labels = list(df_incidents_short.Category)

for lat, lng, label in zip(latitudes, longitudes, labels):
    folium.Marker([lat, lng], popup = label).add_to(sanfran_map)

sanfran_map.add_child(incidents)
```

Out[ ]:

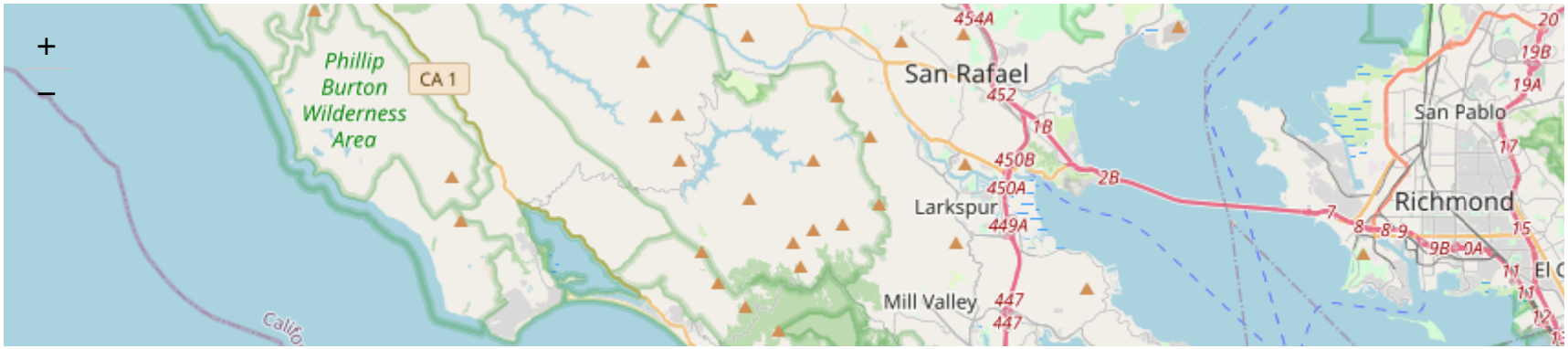


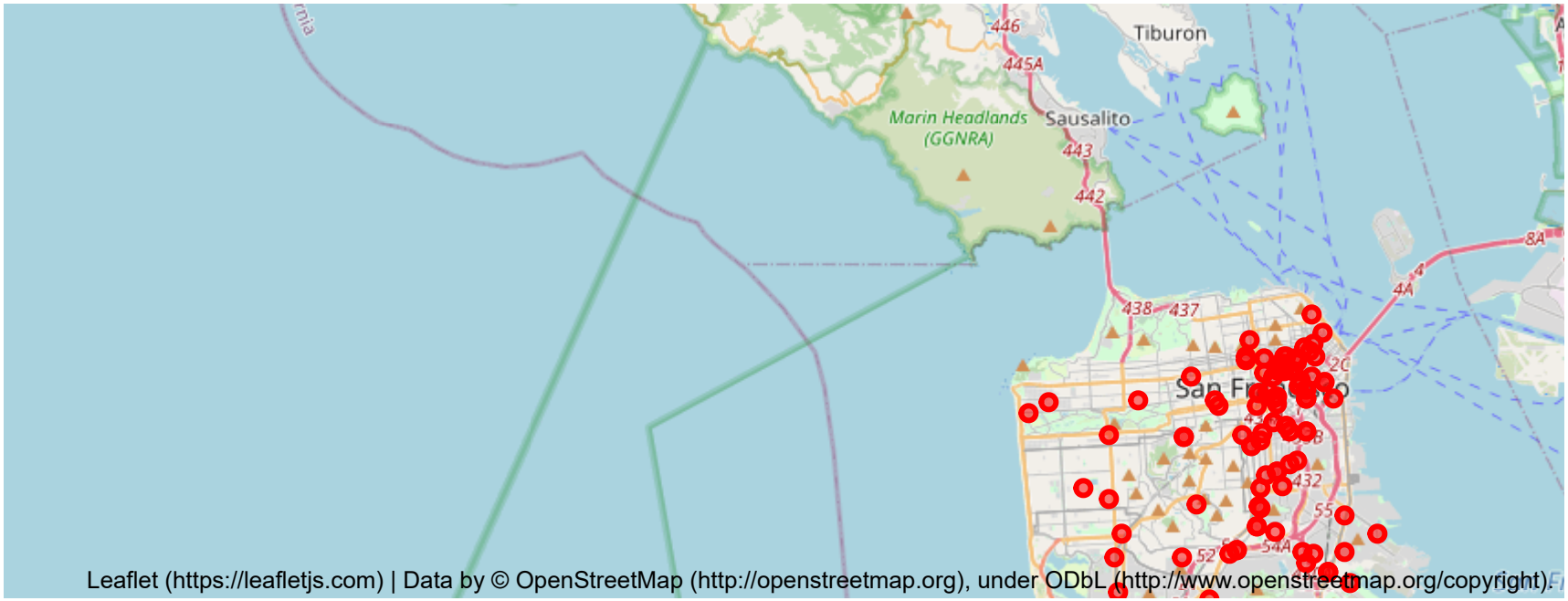


In [ ]:

```
sanfran_map = folium.Map(location = [latitude, longitude], zoom_start = 11)

for lat, lng, label in zip(df_incidents_short.Y, df_incidents_short.X, df_incidents_short.Category):
    folium.features.CircleMarker(
        [lat, lng],
        radius = 4,
        color = 'red', fill = True, popup = label, fill_color = 'red', fill_opacity = '0.5').add_to(sanfran_map)

sanfran_map
```

Out[ ]:





In [ ]:

```
df_incidents_medium = df_incidents.iloc[0:1000, :]
```

In [ ]:

```
from folium import plugins
sanfran_map = folium.Map(location = [latitude, longitude], zoom_start =12)

incidents = plugins.MarkerCluster().add_to(sanfran_map)

for lat, lng, label, in zip(df_incidents_medium.Y, df_incidents_medium.X, df_incidents_medium.Category):
    folium.Marker(location = [lat, lng], icon = None, popup = label).add_to(incidents)

sanfran_map
```

Out[ ]:

In [ ]:
```
df_can = pd.read_excel('Canada.xlsx', sheet_name='Canada by Citizenship', skiprows = range(20), skipfooter=2)
df_can.head()
```

Out[ ]:

	Type	Coverage	OdName	AREA	AreaName	REG	RegName	DEV	DevName	1980	...	2004	2005	2006	2007	2008
0	Immigrants	Foreigners	Afghanistan	935	Asia	5501	Southern Asia	902	Developing regions	16	...	2978	3436	3009	2652	2111
1	Immigrants	Foreigners	Albania	908	Europe	925	Southern Europe	901	Developed regions	1	...	1450	1223	856	702	560
2	Immigrants	Foreigners	Algeria	903	Africa	912	Northern Africa	902	Developing regions	80	...	3616	3626	4807	3623	4005
3	Immigrants	Foreigners	American Samoa	909	Oceania	957	Polynesia	902	Developing regions	0	...	0	0	1	0	0
4	Immigrants	Foreigners	Andorra	908	Europe	925	Southern Europe	901	Developed regions	0	...	0	0	1	1	0

5 rows × 43 columns

In [ ]:
```
df_can.shape
```

Out[ ]:
```
(195, 43)
```

In [ ]:
```
# clean up the dataset to remove unnecessary columns (eg. REG)
df_can.drop(['AREA','REG','DEV','Type','Coverage'], axis=1, inplace=True)

# let's rename the columns so that they make sense
df_can.rename(columns={'OdName':'Country', 'AreaName':'Continent','RegName':'Region'}, inplace=True)

# for sake of consistency, let's also make all column labels of type string
df_can.columns = list(map(str, df_can.columns))

# add total column
df_can['Total'] = df_can.sum(axis=1)

# years that we will be using in this lesson - useful for plotting later on
years = list(map(str, range(1980, 2014)))
print ('data dimensions:', df_can.shape)
```

```
data dimensions: (195, 39)
C:\Users\Gabri\AppData\Local\Temp/ipykernel_3472/2937783283.py:11: FutureWarning: Dropping of nuisance columns in DataFra
me reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select only valid
columns before calling the reduction.
  df_can['Total'] = df_can.sum(axis=1)
```

In [ ]:
```
# download countries geojson file
! wget --quiet https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBMDeveloperSkillsNetwork-DV0101EN-Skil

print('GeoJSON file downloaded!')
```

```
GeoJSON file downloaded!
```

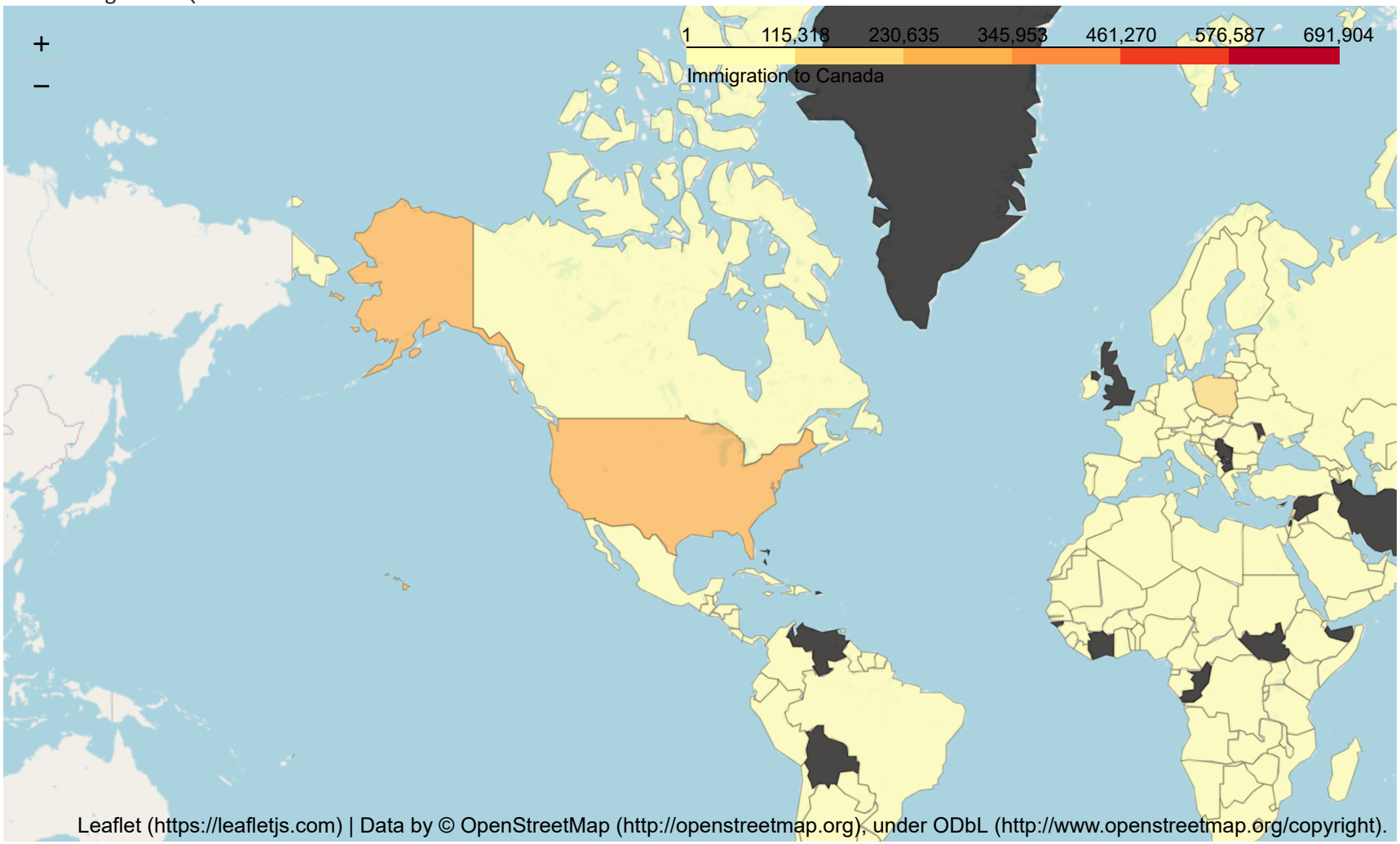
In [ ]:
```
world_geo = r'world_countries.json'
world_map = folium.Map(location = [0,0], zoom_start = 2)
```

In [ ]:
```
world_map.choropleth(geo_data = world_geo, data = df_can, columns = ['Country', 'Total'], key_on = 'feature.properties.na
    fill_color = 'YlOrRd', fill_opacity = 0.7, line_opacity = 0.2, legend_name = 'Immigration to Canada')
```

```
world_map
```

```
C:\Users\Gabri\AppData\Local\Programs\Python\Python39\lib\site-packages\folium\folium.py:409: FutureWarning: The choropleth  method has been deprecated. Instead use the new Choropleth class, which has the same arguments. See the example notebook 'GeoJSON_and_choropleth' for how to do this.
  warnings.warn(
```

Out[ ]:

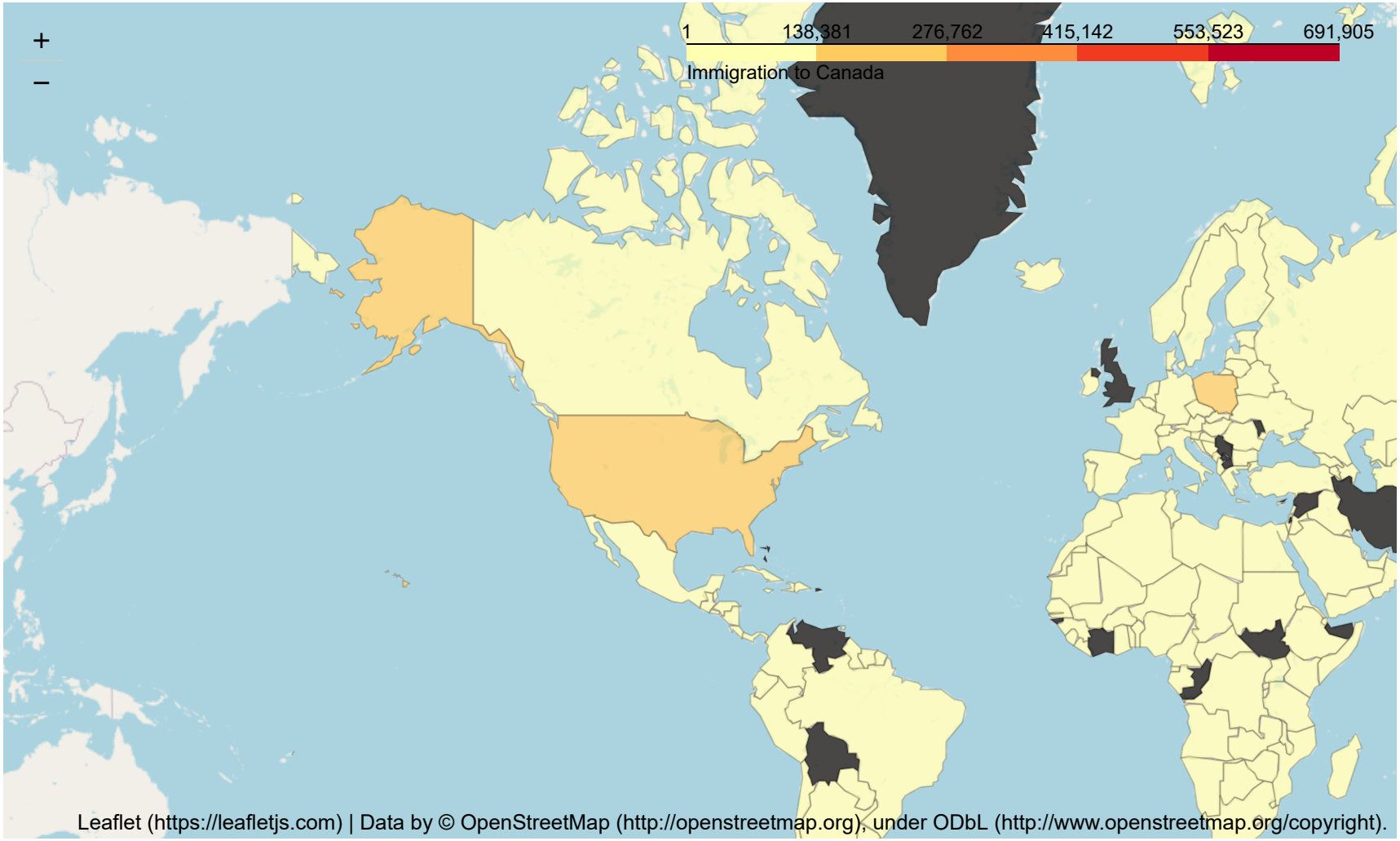


In [ ]:

```
world_geo = r'world_countries.json'
world_map = folium.Map(location = [0,0], zoom_start = 2)

# create a numpy array of length 6 and has linear spacing from the minimum total immigration to the maximum total immigra
```

```
threshold_scale = np.linspace(df_can['Total'].min(),
                              df_can['Total'].max(),
                              6, dtype=int)
threshold_scale = threshold_scale.tolist() # change the numpy array to a list
threshold_scale[-1] = threshold_scale[-1] + 1 # make sure that the last value of the list is greater than the maximum imm

world_map.choropleth(geo_data = world_geo, data = df_can, columns = ['Country', 'Total'], key_on = 'feature.properties.na
    fill_color = 'YlOrRd', fill_opacity = 0.7, line_opacity = 0.2, legend_name = 'Immigration to Canada', threshold_scale

world_map
```

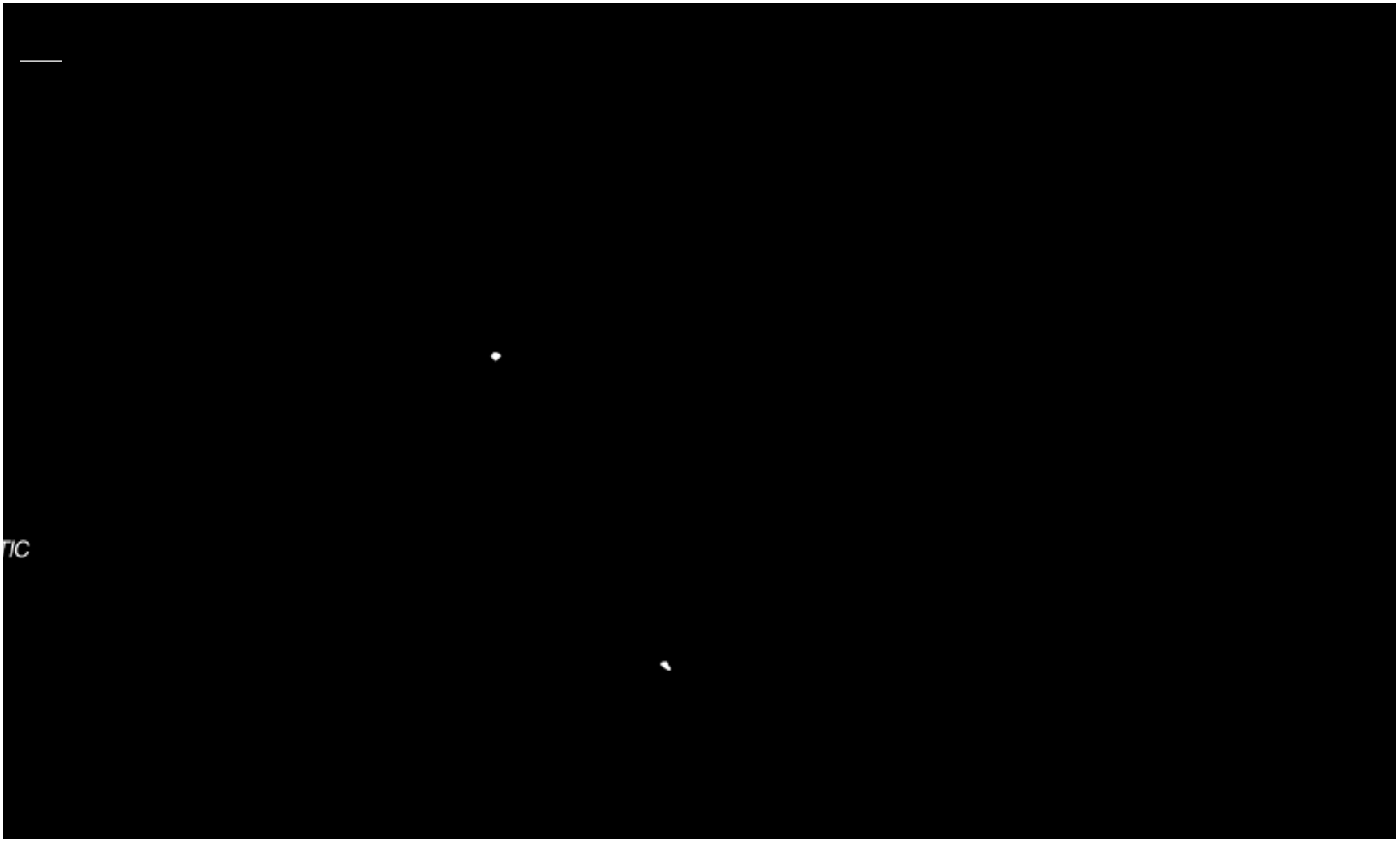
Out[ ]:



In [ ]:

```
folium.Map(location=[-40.4637, -3.7492], zoom_start=6, tiles='Stamen Toner')
```

Out[ ]:



In [ ]: