Geospatial Data Analysis Using Boston Crime Data

Import Libraries

In [1]:
```python
import pandas as pd
import folium #https://python-visualization.github.io/folium/quickstart.html
              #https://python-visualization.github.io/folium/
import math
from folium.plugins import MarkerCluster,HeatMap,CirclePattern
import seaborn as sns
import plotly.graph_objects as go
import plotly.express as px
import datetime
from plotly.subplots import make_subplots
```

Import Data

In [2]:
```python
df=pd.read_csv('https://raw.githubusercontent.com/VinitaSilaparasetty/coursera-spatial-data-analysis/master/boston-crime%202.csv')
df
```

Out[2]:

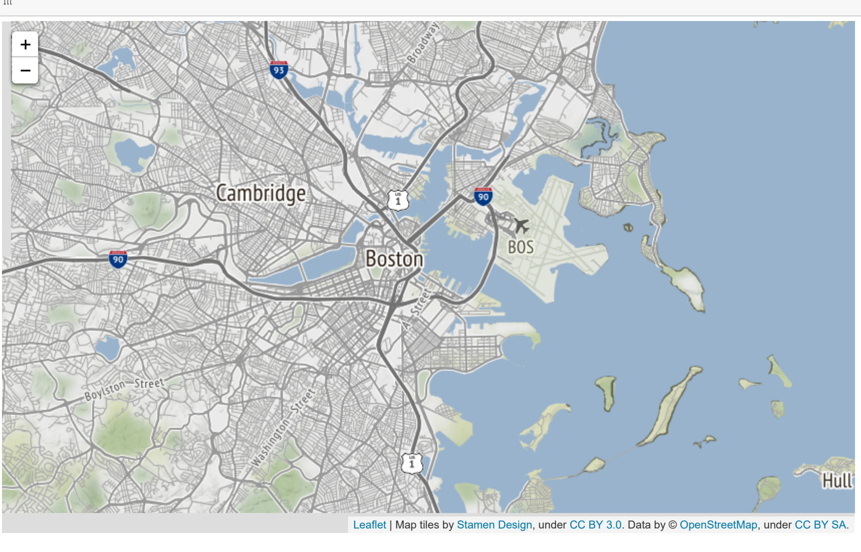
	INCIDENT_NUMBER	OFFENSE_CODE	OFFENSE_CODE_GROUP	OFFENSE_DESCRIPTION	DISTRICT	REPORTING_AREA	SHOOTING
0	I192082859	724	Auto Theft	AUTO THEFT	E18	519	NaN
1	I192082751	724	Auto Theft	AUTO THEFT	E18	493	NaN
2	I192082680	727	Auto Theft	AUTO THEFT - LEASED/RENTED VEHICLE	D14	794	NaN
3	I192082577	724	Auto Theft	AUTO THEFT	D4	130	NaN
4	I192079582	727	Auto Theft	AUTO THEFT - LEASED/RENTED VEHICLE	A15	47	NaN
...	...	...	...	...	...	...	...
467	I192078066	3831	Motor Vehicle Accident Response	M/V - LEAVING SCENE - PROPERTY DAMAGE	B2	259	NaN
468	I192078065	616	Larceny	LARCENY THEFT OF BICYCLE	D14	759	NaN
469	I192078064	3201	Property Lost	PROPERTY - LOST	D4	140	NaN
470	I192078063	3207	Property Found	PROPERTY - FOUND	E5	691	NaN
471	I192078062	3006	Medical Assistance	SICK/INJURED/MEDICAL - PERSON	D4	135	NaN

472 rows × 19 columns

Generate Base Map

In [3]:
```python
boston=(42.358443, -71.05977)
m=folium.Map(location=boston, tiles='Stamen Terrain', zoom_start=12)
m
```

Out[3]:

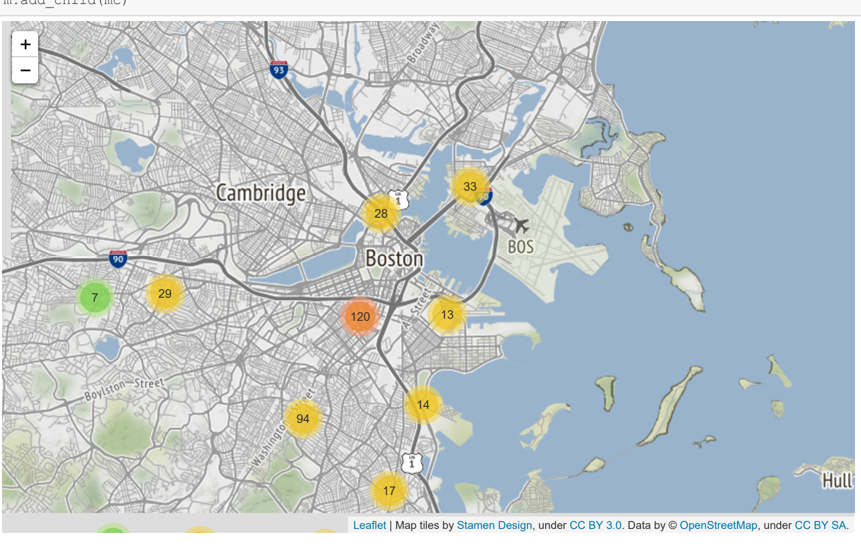


Mark Crime Scenes

In [4]:
```python
# marker clustering utility helps you to manage multiple markers at different zoom levels. When a user views the map at a high zoom level, the individual markers show on the map. When the user zooms out, the markers gather together into clusters, to make viewing the map easier.
mc=MarkerCluster()
```

In [5]:
```python
for idx,row in df.iterrows():
    if not math.isnan(row['Long']) and not math.isnan(row['Lat']):
        mc.add_child(folium.Marker([row['Lat'], row['Long']]))
m.add_child(mc)
```

Out[5]:



View Districts with Highest Crime Rates

In [6]:
```python
crime=df.groupby(['DISTRICT', 'STREET', 'REPORTING_AREA', 'Lat', 'Long']).sum().reset_index()
crime.head(10)
```

Out[6]:

	DISTRICT	STREET	REPORTING_AREA	Lat	Long	OFFENSE_CODE	SHOOTING	YEAR	MONTH	HOUR
0	A1	ARLINGTON ST	128	42.348670	-71.069389	3201	0.0	2019	9	10
1	A1	ATLANTIC AVE	110	42.354009	-71.052853	3831	0.0	2019	9	3
2	A1	ATLANTIC AVE	114	42.354383	-71.052133	2647	0.0	2019	9	15
3	A1	CAUSEWAY ST	75	42.364331	-71.063193	1825	0.0	4038	18	8
4	A1	CAUSEWAY ST	75	42.364975	-71.062044	2405	0.0	2019	9	21
5	A1	CHARLES ST	64	42.361928	-71.070820	10183	0.0	12114	54	128
6	A1	CHESTNUT ST	70	42.357640	-71.067032	3410	0.0	2019	9	7
7	A1	COURT ST	92	42.358923	-71.058354	3623	0.0	4038	18	24
8	A1	FANEUIL HALL MARKETPLACE	93	42.359980	-71.054741	1830	0.0	2019	9	10
9	A1	FANEUIL HALL SQ	93	42.360205	-71.056208	613	0.0	2019	9	19

In [7]:
```python
crime.update(crime['DISTRICT'].map('District:{}'.format))
```

In [8]:
```python
crime
```

Out[8]:

	DISTRICT	STREET	REPORTING_AREA	Lat	Long	OFFENSE_CODE	SHOOTING	YEAR	MONTH	HOUR
0	District:A1	ARLINGTON ST	128	42.348670	-71.069389	3201	0.0	2019	9	10
1	District:A1	ATLANTIC AVE	110	42.354009	-71.052853	3831	0.0	2019	9	3
2	District:A1	ATLANTIC AVE	114	42.354383	-71.052133	2647	0.0	2019	9	15
3	District:A1	CAUSEWAY ST	75	42.364331	-71.063193	1825	0.0	4038	18	8
4	District:A1	CAUSEWAY ST	75	42.364975	-71.062044	2405	0.0	2019	9	21
...	...	...	...	...	...	...	...	...	...	...
386	District:E5	SOUTH ST	676	42.288025	-71.143697	3802	0.0	2019	9	12
387	District:E5	TOBIN RD	714	42.259006	-71.160679	3301	0.0	2019	9	21
388	District:E5	TYNDALE ST	672	42.287186	-71.136805	3503	0.0	2019	9	17
389	District:E5	WALTER ST	663	42.293797	-71.130566	3006	0.0	2019	9	19
390	District:E5	WELD ST	678	42.293364	-71.144940	3301	0.0	2019	9	16

391 rows × 10 columns

In [9]:
```python
crime.update(crime['REPORTING_AREA'].map('Reports:{}'.format))
```

In [10]:
```python
crime
```

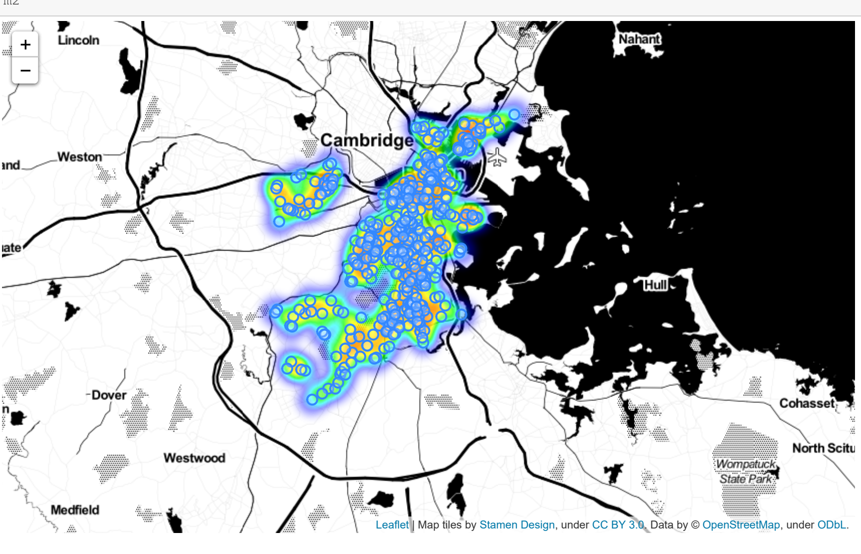
Out[10]:

	DISTRICT	STREET	REPORTING_AREA	Lat	Long	OFFENSE_CODE	SHOOTING	YEAR	MONTH	HOUR
0	District:A1	ARLINGTON ST	Reports:128	42.348670	-71.069389	3201	0.0	2019	9	10
1	District:A1	ATLANTIC AVE	Reports:110	42.354009	-71.052853	3831	0.0	2019	9	3
2	District:A1	ATLANTIC AVE	Reports:114	42.354383	-71.052133	2647	0.0	2019	9	15
3	District:A1	CAUSEWAY ST	Reports:75	42.364331	-71.063193	1825	0.0	4038	18	8
4	District:A1	CAUSEWAY ST	Reports:75	42.364975	-71.062044	2405	0.0	2019	9	21
...	...	...	...	...	...	...	...	...	...	...
386	District:E5	SOUTH ST	Reports:676	42.288025	-71.143697	3802	0.0	2019	9	12
387	District:E5	TOBIN RD	Reports:714	42.259006	-71.160679	3301	0.0	2019	9	21
388	District:E5	TYNDALE ST	Reports:672	42.287186	-71.136805	3503	0.0	2019	9	17
389	District:E5	WALTER ST	Reports:663	42.293797	-71.130566	3006	0.0	2019	9	19
390	District:E5	WELD ST	Reports:678	42.293364	-71.144940	3301	0.0	2019	9	16

391 rows × 10 columns

In [11]:
```python
m2=folium.Map(location=boston, tiles="Stamen Toner", zoom_start=12)
HeatMap(data=crime[['Lat', 'Long']], radius=15).add_to(m2)
#https://python-visualization.github.io/folium/modules.html search circlemaker
#circlemaker:A circle of a fixed size with radius specified in pixels.
def plotDot(point):
    folium.vector_layers.CircleMarker(location=[point.Lat,point.Long],
                        radius=5,
                        weight=2,
                        popup=[point.DISTRICT,point.REPORTING_AREA],
                        fill_color='#ffffff').add_to(m2)
crime.apply(plotDot,axis=1)
#Computes the bounds of the object itself (not including it's children) in the form [[lat_min, lon_min], [lat_max, lon_max]]
m2.fit_bounds(m2.get_bounds())
m2
```

Out[11]:



Medical Assistance Analysis

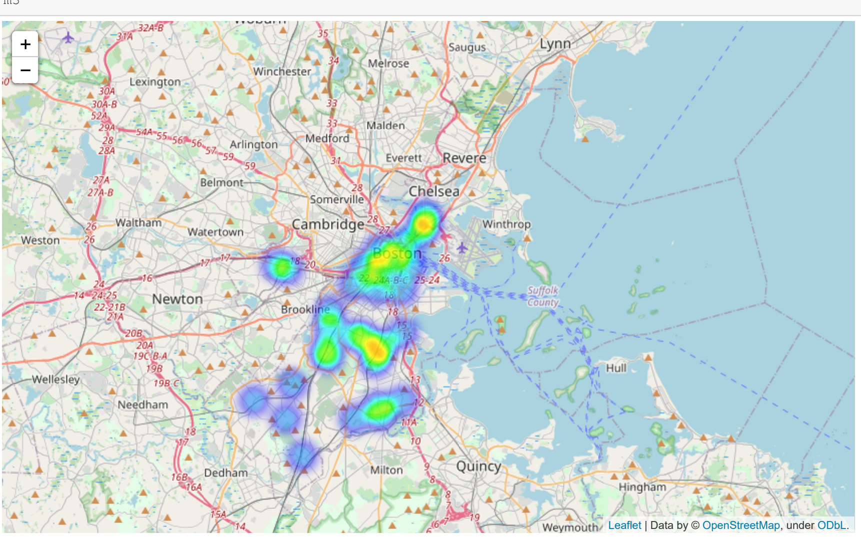
In [12]:
```python
med=df.loc[df.OFFENSE_CODE_GROUP=='Medical Assistance'][['Lat', 'Long']]
med
```

Out[12]:

	Lat	Long
23	42.355942	-71.062583
53	42.307342	-71.105441
58	42.309692	-71.072859
73	42.341288	-71.054679
75	42.320778	-71.105538
83	42.319757	-71.054289
95	42.276131	-71.087654
103	42.319052	-71.084114
108	42.285398	-71.156161
117	42.343734	-71.074192
152	42.323877	-71.104093
169	42.276511	-71.134176
172	42.318576	-71.087672
195	42.313634	-71.067780
211	42.278004	-71.069505
260	42.374762	-71.037312
270	42.351824	-71.133134
272	42.353254	-71.048724
279	42.364754	-71.054695
281	42.361928	-71.070820
291	42.330826	-71.104370
292	42.351435	-71.069693
293	42.310749	-71.107662
295	42.370261	-71.042900
297	42.376203	-71.039799
299	42.257675	-71.123671
300	42.304613	-71.072966
304	42.354136	-71.141165
333	42.293797	-71.130566
363	42.285235	-71.055097
368	42.382891	-71.039309
373	42.358923	-71.058354
405	42.282779	-71.071355
420	42.317362	-71.077246
432	42.343765	-71.088669
456	42.339542	-71.069409
459	42.357690	-71.059388
471	42.353357	-71.080177

In [13]:
```python
m3=folium.Map(location=boston,tiles='openstreetmap',zoom_start=11)
HeatMap(data=med,radius=16).add_to(m3)
m3
```

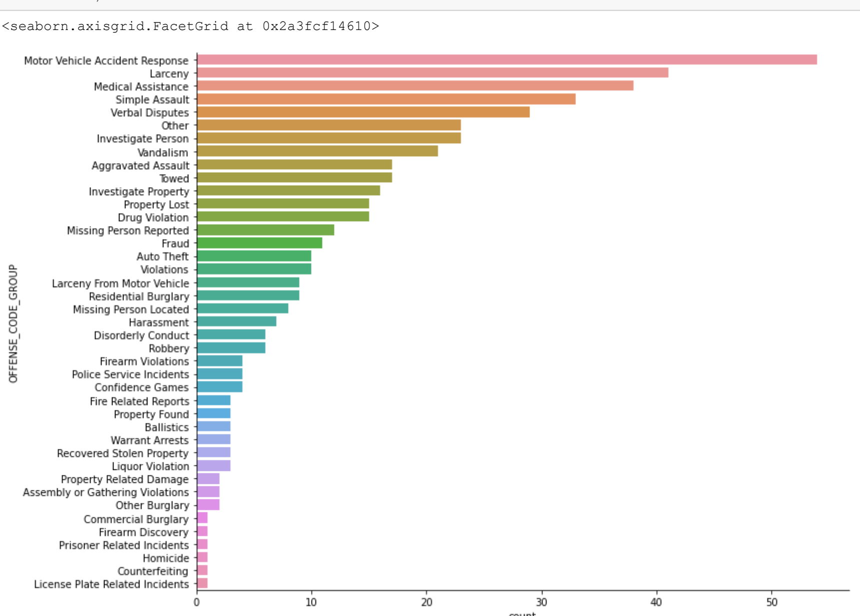
Out[13]:



Crimes Commited

In [14]:
```python
#catplot show the relationship between a numerical and one or more categorical variables using one of several visual representations
sns.catplot(y='OFFENSE_CODE_GROUP',
            kind = 'count',
            height=8,
            aspect=1.5,
            order=df.OFFENSE_CODE_GROUP.value_counts().index,
            data=df
           )
```

Out[14]: <seaborn.axisgrid.FacetGrid at 0x2a3fcf14610>



Motor Vehicle Accident Response

In [15]:
```python
mv=df.loc[df.OFFENSE_CODE_GROUP=='Motor Vehicle Accident Response'][['Lat', 'Long']]
```

In [16]:
```python
mv
```

Out[16]:

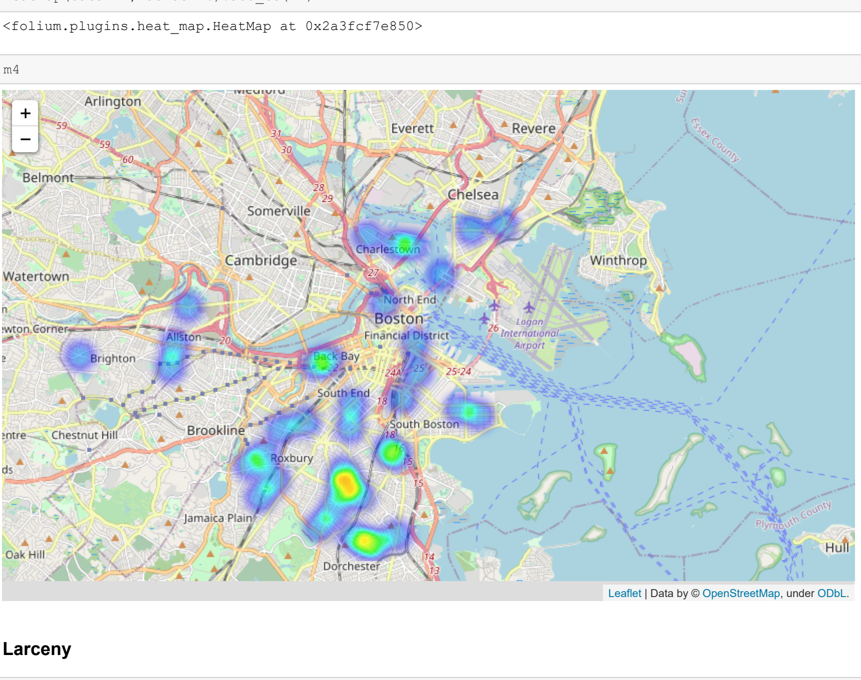
	Lat	Long
9	42.332419	-71.075013
16	42.287579	-71.075273
20	42.286065	-71.070010
24	42.382589	-71.033420
50	42.339761	-71.057092
54	42.264837	-71.099668
55	42.270801	-71.105812
56	42.325610	-71.104500
74	42.363122	-71.130563
86	42.302698	-71.066015
105	42.304904	-71.074140
106	42.350570	-71.167812
121	42.348406	-71.086883
122	42.335718	-71.030256
124	42.334025	-71.090390
130	42.265197	-71.126864
151	42.364065	-71.063337
168	42.371295	-71.043424
170	42.305142	-71.058981
177	42.280679	-71.158175
179	42.381208	-71.069143
191	42.346475	-71.051506
194	42.311628	-71.080943
196	42.283628	-71.093496
201	42.288025	-71.143697
209	42.302673	-71.070195
246	42.264305	-71.100613
251	42.317094	-71.078030
252	42.325276	-71.059145
254	42.354009	-71.052853
256	42.307242	-71.085517
257	42.286332	-71.086423
264	42.274130	-71.116746
274	42.303441	-71.066746
280	42.314950	-71.105995
289	42.332096	-71.098606
298	42.287701	-71.106713
310	42.318399	-71.075305
332	42.347513	-71.136893
348	42.335700	-71.031285
350	42.323094	-71.109661
389	42.383916	-71.022662
393	42.326966	-71.061986
394	42.378971	-71.059343
414	42.349686	-71.082179
424	42.275316	-71.110819
426	42.337292	-71.037466
433	42.378689	-71.053465
434	42.303576	-71.068808
446	42.353426	-71.135084
448	42.321532	-71.077385
453	42.319157	-71.100676
457	42.338413	-71.074278
467	42.314784	-71.071610

In [17]:
```python
mv.fillna(0, inplace=True)
mv.Lat.fillna(0, inplace=True)
mv.Long.fillna(0, inplace=True)
m4=folium.Map(location=boston, tiles='openstreetmap',zoom_start=12)
HeatMap(data=mv,radius=16).add_to(m4)
```

Out[17]: <folium.plugins.heat_map.HeatMap at 0x2a3fcf7e850>

In [18]:
```python
m4
```

Out[18]:



Larceny

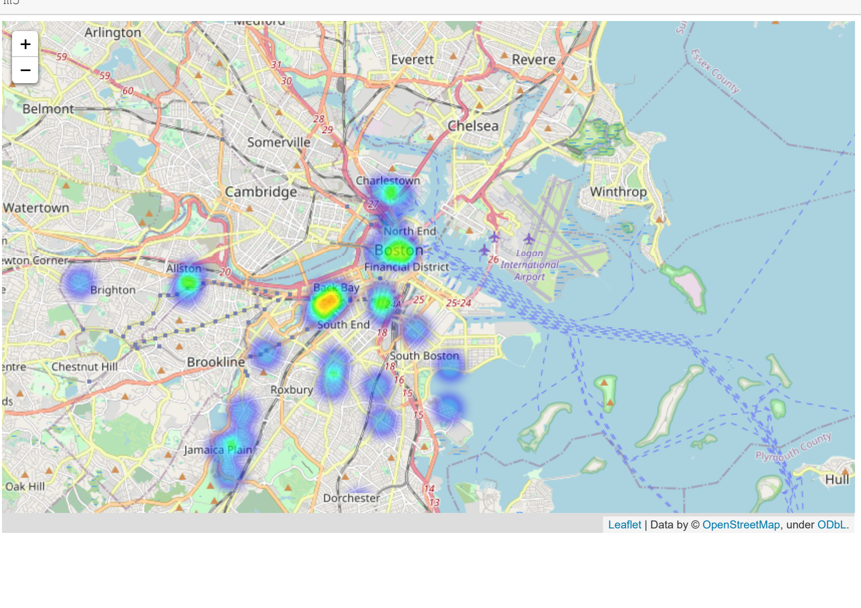
In [19]:
```python
lar=df.loc[df.OFFENSE_CODE_GROUP=='Larceny'][['Lat', 'Long']]
```

In [20]:
```python
lar.Lat.fillna(0, inplace=True)
lar.Long.fillna(0, inplace=True)
m5=folium.Map(location=boston, tiles='openstreetmap',zoom_start=12)
HeatMap(data=lar,radius=16).add_to(m5)
```

Out[20]: <folium.plugins.heat_map.HeatMap at 0x2a3fd69d7c0>

In [21]:
```python
m5
```

Out[21]:



type this in cmd to extract an html file of these visualization and delete markdown cell :

jupyter nbconvert file.ipynb --no-input --no-prompt to

In [ ]: