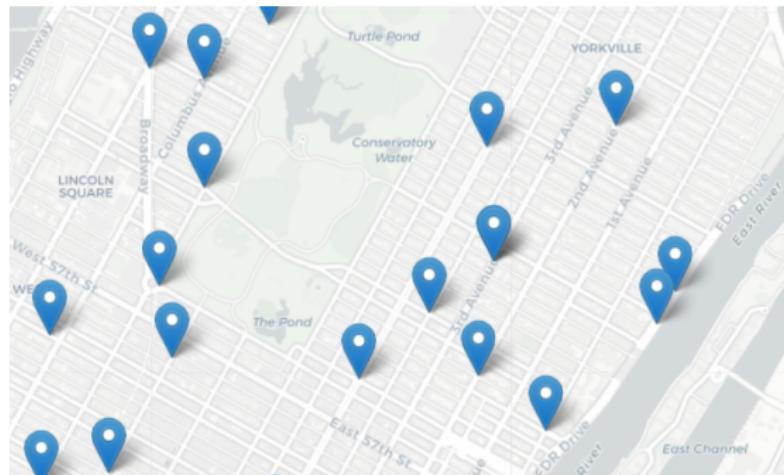


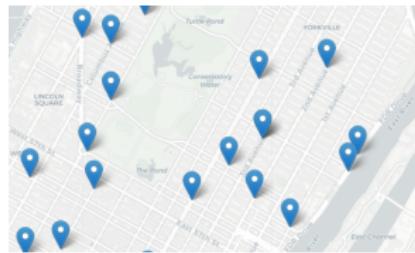
# MfA: Python in the City



Katherine St. John  
City University of New York  
American Museum of Natural History

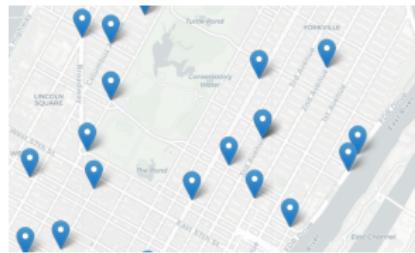
Goal: Sit at a table with someone who you did not sit with the first two sessions.

# Outline



- Recap
- HTML-Scalable Maps: Folium
- Extracting Data
- geoJSON Format & Choropleth Maps
- Break
- Design Challenge: Catchment Areas
- Design Challenge: Clustering Data
- Wrap Up

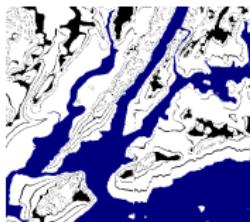
# Outline



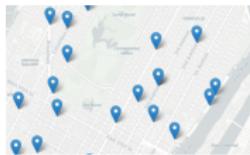
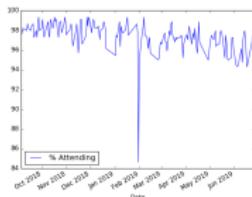
- **Recap**
- HTML-Scalable Maps: Folium
- Extracting Data
- geoJSON Format & Choropleth Maps
- Break
- Design Challenge: Catchment Areas
- Design Challenge: Clustering Data
- Wrap Up

# Recap: Workshop Overview

Three sessions:

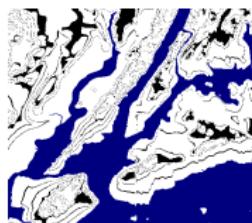


- ① Flood Maps (arrays & images)
- ② School Attendance (structured data, file I/O)
- ③ Mapping Collisions (using objects, mapping coordinates)

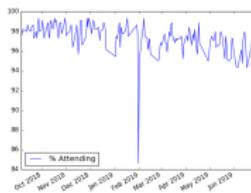


# Recap: Workshop Overview

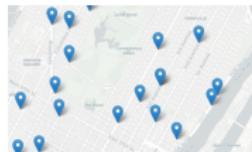
Three sessions:



- ① Flood Maps (arrays & images)
- ② School Attendance (structured data, file I/O)
- ③ Mapping Collisions (using objects, mapping coordinates)

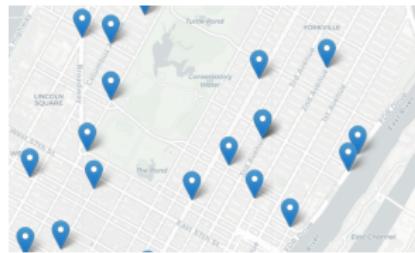


Each session:



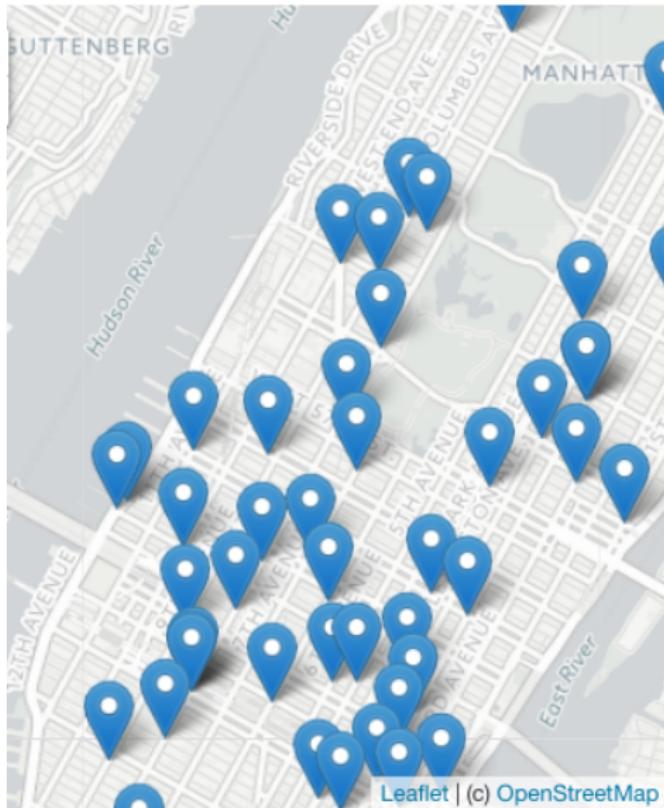
- Design Challenge
  - ▶ Analyze a publicly available dataset
  - ▶ Introduce computing concepts & packages
  - ▶ Write a program to solve the problem
- Variations on the theme
- Design a Challenge

# Outline



- Recap
- **HTML-Scalable Maps: Folium**
- Extracting Data
- geoJSON Format & Choropleth Maps
- Break
- Design Challenge: Catchment Areas
- Design Challenge: Clustering Data
- Wrap Up

# HTML-Scalable Maps: Folium



Leaflet | (c) OpenStreetMap

# Folium

- A module for making HTML maps.

# Folium



# Folium

## Folium



- A module for making HTML maps.
- It's a Python interface to the popular `leaflet.js`.

# Folium

Folium



- A module for making HTML maps.
- It's a Python interface to the popular `leaflet.js`.
- Outputs `.html` files which you can open in a browser.

# Folium

Folium



- A module for making HTML maps.
- It's a Python interface to the popular `leaflet.js`.
- Outputs `.html` files which you can open in a browser.
- An extra step:

# Folium

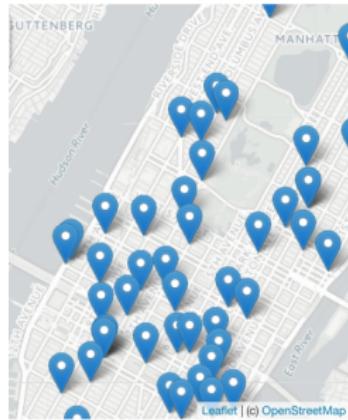
## Folium



- A module for making HTML maps.
- It's a Python interface to the popular `leaflet.js`.
- Outputs `.html` files which you can open in a browser.
- An extra step:

*Write code.* → *Run program.* → *Open .html in browser.*

# Demo



(Map created by Folium.)

# Your turn: Make a map using folium

- To use:

```
import folium
```

## Folium



## Your turn: Make a map using folium

- To use:

```
import folium
```

- Create a map:

```
myMap = folium.Map()
```

## Folium



# Your turn: Make a map using folium

- To use:

```
import folium
```

- Create a map:

```
myMap = folium.Map()
```

- Make markers:

```
newMark = folium.Marker([lat,lon],popup=name)
```

## Folium



# Your turn: Make a map using folium

- To use:

```
import folium
```

- Create a map:

```
myMap = folium.Map()
```

- Make markers:

```
newMark = folium.Marker([lat,lon],popup=name)
```

- Add to the map:

```
newMark.add_to(myMap)
```

## Folium



# Your turn: Make a map using folium

- To use:

```
import folium
```

- Create a map:

```
myMap = folium.Map()
```

- Make markers:

```
newMark = folium.Marker([lat,lon],popup=name)
```

- Add to the map:

```
newMark.add_to(myMap)
```

- Can customize map with starting location, zoom level and background map ("tiles"):

```
myMap = folium.Map(location=[40.75, -74.125],  
zoom_start=10, tiles='Stamen Watercolor')
```

Many options to customize background map ("tiles"):

## Folium



# Your turn: Make a map using folium

- To use:

```
import folium
```

- Create a map:

```
myMap = folium.Map()
```

- Make markers:

```
newMark = folium.Marker([lat,lon],popup=name)
```

- Add to the map:

```
newMark.add_to(myMap)
```

- Can customize map with starting location, zoom level and background map ("tiles"):

```
myMap = folium.Map(location=[40.75, -74.125],  
zoom_start=10, tiles='Stamen Watercolor')
```

Many options to customize background map ("tiles"):

(Some background map options: 'Stamen Terrain',  
'Stamen Watercolor', 'Mapbox Bright',  
'Stamen Toner', 'Cartodb Positron')

- Save to a file:

```
myMap.save(outfile="myMap.html")
```

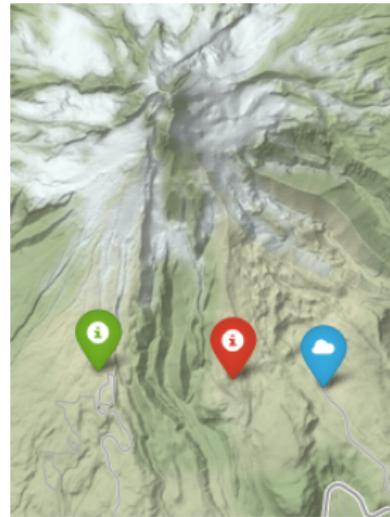
## Folium



# In Pairs of Triples

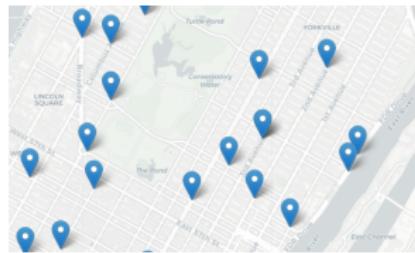
- Predict which each line of code does:

```
m = folium.Map(  
    location=[45.372, -121.6972],  
    zoom_start=12,  
    tiles='Stamen Terrain'  
)  
  
folium.Marker(  
    location=[45.3288, -121.6625],  
    popup='Mt. Hood Meadows',  
    icon=folium.Icon(icon='cloud')  
) .add_to(m)  
  
folium.Marker(  
    location=[45.3311, -121.7113],  
    popup='Timberline Lodge',  
    icon=folium.Icon(color='green')  
) .add_to(m)  
  
folium.Marker(  
    location=[45.3300, -121.6823],  
    popup='Some Other Location',  
    icon=folium.Icon(color='red', icon='info-sign')  
) .add_to(m)
```



(example from Folium documentation)

# Outline



- Recap
- HTML-Scalable Maps: Folium
- **Extracting Data**
- geoJSON Format & Choropleth Maps
- Break
- Design Challenge: Catchment Areas
- Design Challenge: Clustering Data
- Wrap Up

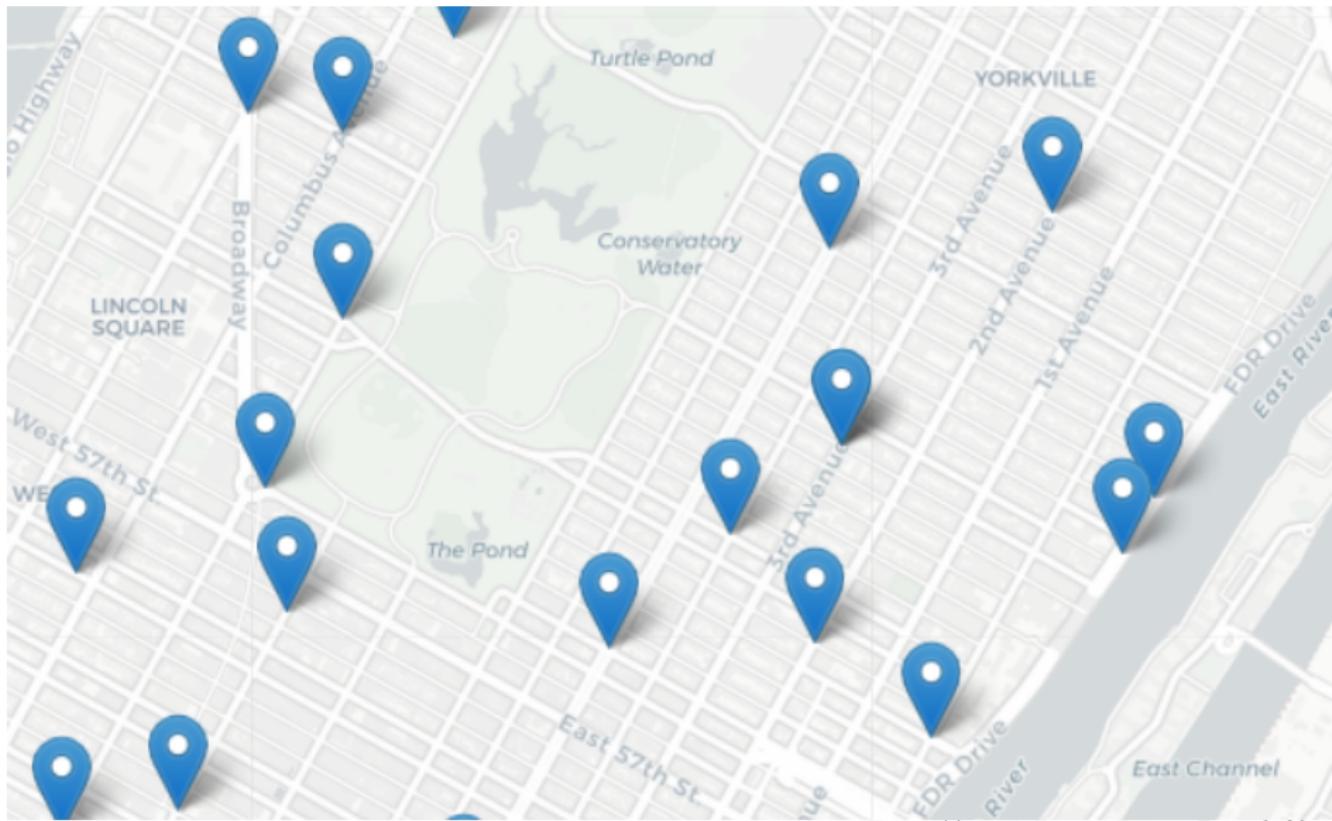
## Recall: Film Permits Example

- Download the data as a CSV file and store on your computer.
  - Python program:

#CSci 127 Teaching Staff  
#March 2019  
#OpenData Film Permits

```
#Import pandas for reading and analyzing CSV data:  
import pandas as pd  
csvFile = "filmPermits.csv" #Name of the CSV file  
tickets = pd.read_csv(csvFile) #Read in the file to a dataframe  
print(tickets) #Print out the dataframe  
print(tickets["ParkingHeld"]) #Print out streets (multiple times)  
print(tickets["ParkingHeld"].value_counts()) #Print out streets & number of times used  
print(tickets["ParkingHeld"].value_counts()[:10]) #Print 10 most popular
```

# Extracting Data for a Map

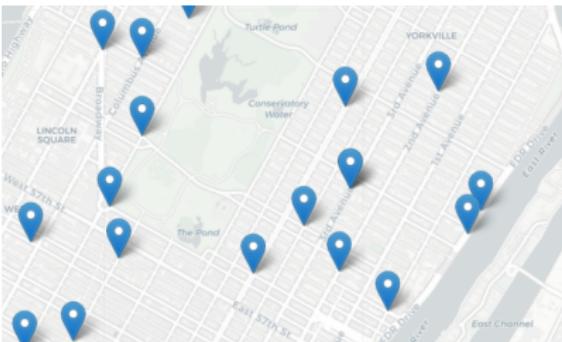


# Extracting Data for a Map

A	B	C	D	E	F	G	H	I	J	K
DATE	TIME	BOROUGH	ZIP CODE	LATITUDE	LONGITUDE	LOCATION	ON STREET	CROSS STREET	OFF STREET	NUMBER OF
10/18/16	8:10	STATEN ISLA	10312	40.5405508	-74.1919197	(40.5405508, -74.1919197)				1
10/18/16	8:10	BROOKLYN	11238	40.8864547	-73.968107	(40.8864547, -73.968107)				1
10/18/16	8:10	BROOKLYN	11234	40.6261739	-73.922234	(40.6261739, -73.922234)				1
10/18/16	8:10	BROOKLYN	11218	40.6261739	-73.922234	(40.6261739, -73.922234)				1
10/18/16	8:10	BROOKLYN	11217	40.6271765	-73.972234	(40.6271765, -73.972234)				1
10/18/16	8:10			40.7415765	-73.82939	(40.7415765, -73.82939)				0
10/18/16	8:10			40.7112758	-73.8771331	(40.7112758, -73.8771331)				0
10/18/16	8:10			40.7415268	-73.78455	(40.7415268, -73.78455)				0
10/18/16	8:10			40.7365373	-73.856099	(40.7365373, -73.856099)				0
10/18/16	8:10			40.7361484	-73.926504	(40.7361484, -73.926504)				0
10/18/16	8:10	QUEENS	11426	40.7351801	-73.718346	(40.7351801, -73.718346)				0
10/18/16	8:10	QUEENS	11417	40.6719314	-73.851635	(40.6719314, -73.851635)				0
10/18/16	8:10	QUEENS	11355	40.7606728	-73.821007	(40.7606728, -73.821007)	144	20	41	0
10/18/16	8:10	QUEENS	11103	40.7623893	-73.911797	(40.7623893, -73.911797)				0
10/18/16	8:10	QUEENS	11101	40.7462598	-73.930409	(40.7462598, -73.930409)				0
10/18/16	8:10	MANHATTAN	10035	40.8050573	-73.939034	(40.8050573, -73.939034)				0
10/18/16	8:10	MANHATTAN	10018	40.7556432	-73.990962	(40.7556432, -73.990962)	607	8	AVI	0
10/18/16	8:10	MANHATTAN	10016	40.7473832	-73.973508	(40.7473832, -73.973508)				0
10/18/16	8:10	MANHATTAN	10011	40.7458255	-74.001812	(40.7458255, -74.001812)				0
10/18/16	8:10	MANHATTAN	10003	40.7317985	-73.882114	(40.7317985, -73.882114)	251	1	AVI	0
10/18/16	8:10	BROOKLYN	11234	40.616612	-73.926504	(40.616612, -73.926504)				0
10/18/16	8:10	BROOKLYN	11220	40.715111	-73.945511	(40.715111, -73.945511)				0
10/18/16	8:10	BROOKLYN	11206	40.707165	-73.984839	(40.707165, -73.984839)				0
10/18/16	8:10	BRONX	10488	40.8716697	-73.897797	(40.8716697, -73.897797)				0

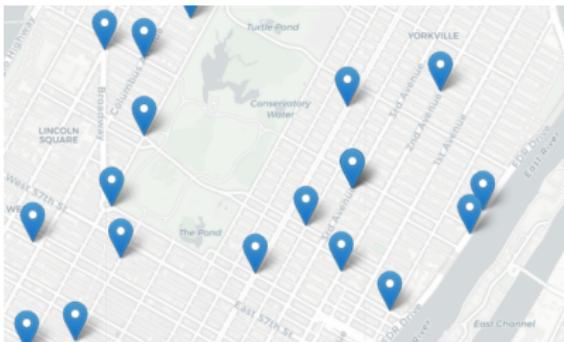
# Extracting Data for a Map

A	B	C	D	E	F	G	H	I	J	K
DATE	TIME	BOROUGH	ZIP CODE	LATITUDE	LONGITUDE	LOCATION	ON STREET A	CROSS STREET B	OFF STREET C	NUMBER OF
10/18/16	8:10	STATEN ISLA	10312	40.5405508	-74.191197	(40.5405508, -74.191197)	NATHBUN AV	NIKKON AVENUE		1
10/18/16	8:10	BROOKLYN	11238	40.8864547	-73.968107	(40.8864547, -73.968107)	107	GREE		1
10/18/16	8:10	BROOKLYN	11234	40.6261739	-73.922234	(40.6261739, -73.922234)	5515	AVE		1
10/18/16	8:10	BROOKLYN	11218	40.6261739	-73.922234	(40.6261739, -73.922234)	107	STREET	EAST 2 STREET	1
10/18/16	8:10	BROOKLYN	11217	40.6271765	-73.972234	(40.6271765, -73.972234)	718	ATLA		1
10/18/16	8:10			40.7415705	-73.82939	(40.7415705, -73.82939)				0
10/18/16	8:10			40.7112758	-73.8771331	(40.7112758, -73.8771331)				0
10/18/16	8:10			40.7415268	-73.78455	(40.7415268, -73.78455)				0
10/18/16	8:10			40.7365373	-73.856099	(40.7365373, -73.856099)				0
10/18/16	8:10			40.7361484	-73.926504	(40.7361484, -73.926504)				0
10/18/16	8:10	QUEENS	11426	40.7351801	-73.718346	(40.7351801, -73.718346)	HILLSIDE AV	249 STREET		0
10/18/16	8:10	QUEENS	11417	40.6719314	-73.851635	(40.6719314, -73.851635)	104	CONI 84	STREET	0
10/18/16	8:10	QUEENS	11355	40.7606728	-73.821007	(40.7606728, -73.821007)	144-20	41		0
10/18/16	8:10	QUEENS	11103	40.7623893	-73.911797	(40.7623893, -73.911797)	45	STREET	30 AVENUE	0
10/18/16	8:10	QUEENS	11101	40.7462598	-73.930409	(40.7462598, -73.930409)	0	34	STREET	43 AVENUE
10/18/16	8:10	MANHATTAN	10035	40.8050573	-73.939034	(40.8050573, -73.939034)	EAST 125 ST	PARK AVENUE		0
10/18/16	8:10	MANHATTAN	10018	40.7556432	-73.990962	(40.7556432, -73.990962)	607	8	AVI	0
10/18/16	8:10	MANHATTAN	10016	40.7473832	-73.973508	(40.7473832, -73.973508)	EAST 34 STR	1 AVENUE		0
10/18/16	8:10	MANHATTAN	10011	40.7458255	-74.001812	(40.7458255, -74.001812)	WEST 22 STR	9 AVENUE		0
10/18/16	8:10	MANHATTAN	10003	40.7317985	-73.882114	(40.7317985, -73.882114)	251	1	AVI	0
10/18/16	8:10	BROOKLYN	11234	40.6166167	-73.926504	(40.6166167, -73.926504)	104	AA	AVENUE D	0
10/18/16	8:10	BROOKLYN	11226	40.7151151	-73.9751151	(40.7151151, -73.9751151)	11206	STR	GREENPOINT AVENUE	0
10/18/16	8:10	BROOKLYN	11206	40.7071655	-73.9398398	(40.7071655, -73.9398398)	0	BUSHWICK	A MONTROSE AVENUE	0
10/18/16	8:10	BRONX	10468	40.8716697	-73.897797	(40.8716697, -73.897797)	WEST 197 ST	RESERVOIR AVENUE		0



# Extracting Data for a Map

A	B	C	D	E	F	G	H	I	J	K	
DATE	TIME	BOROUGH	ZIP CODE	LATITUDE	LONGITUDE	LOCATION	ON STREET A	CROSS STREET B	OFF STREET C	NUMBER OF	
10/18/16	8:10	STATEN ISLA	10312	40.540574	-74.191197	(40.540550, -74.191074)	NATHBUN AV	NIKKON AVENUE		1	
10/18/16	8:10	BROOKLYN	11238	40.886547	-73.968107	(40.886520, -73.968107)				107	GREE
10/18/16	8:10	BROOKLYN	11234	40.626173	-73.922234	(40.626173, -73.922234)				5515	AVE1
10/18/16	8:10	BROOKLYN	11218	40.626173	-73.922234	(40.626173, -73.922234)	BUCHANAN AV	2ND STREET		1	
10/18/16	8:10	BROOKLYN	11217	40.627165	-73.972234	(40.627165, -73.972234)				718	ATLA
10/18/16	8:10			40.7415705	-73.82939	(40.7415705, -73.82939)				1	
10/18/16	8:10			40.7112758	-73.8771331	(40.7112758, -73.8771331)				0	
10/18/16	8:10			40.7415268	-73.78455	(40.7415268, -73.7845496)				0	
10/18/16	8:10			40.7365373	-73.856099	(40.7365373, -73.8560987)				0	
10/18/16	8:10			40.7361484	-73.926504	(40.7361484, -73.9265043)				0	
10/18/16	8:10	QUEENS	11426	40.7351801	-73.718346	(40.7351801, -73.718346)	HILLSIDE AV	249 STREET		0	
10/18/16	8:10	QUEENS	11417	40.6719314	-73.851635	(40.6719314, -73.851635)	NORTH CONI	84 STREET		0	
10/18/16	8:10	QUEENS	11355	40.7606728	-73.821007	(40.7606728, -73.8210074)				144-20	41
10/18/16	8:10	QUEENS	11103	40.7623891	-73.911797	(40.7623891, -73.911797)	45 STREET	30 AVENUE		0	
10/18/16	8:10	QUEENS	11101	40.7462598	-73.930409	(40.7462598, -73.930409)				0	
10/18/16	8:10	MANHATTAN	10035	40.8050573	-73.939034	(40.8050573, -73.939034)	EAST 125 ST	PARK AVENUE		0	
10/18/16	8:10	MANHATTAN	10018	40.7556432	-73.990962	(40.7556432, -73.9909619)				607	8 AVI
10/18/16	8:10	MANHATTAN	10016	40.7473832	-73.973508	(40.7473832, -73.973508)	EAST 34 STR	1 AVENUE		0	
10/18/16	8:10	MANHATTAN	10011	40.7458255	-74.001812	(40.7458255, -74.001812)	WEST 22 STR	9 AVENUE		0	
10/18/16	8:10	MANHATTAN	10003	40.7317985	-73.882114	(40.7317985, -73.8821143)				251	1 AVI
10/18/16	8:10	BROOKLYN	11234	40.616617	-73.926504	(40.616617, -73.926504)				0	
10/18/16	8:10	BROOKLYN	11222	40.715111	-73.945511	(40.715111, -73.945511)	MURKIN STR	GREENPOINT AVENUE		0	
10/18/16	8:10	BROOKLYN	11206	40.707165	-73.938839	(40.707165, -73.938839)	BUSHWICK A	MONTROSE AVENUE		0	
10/18/16	8:10	BRONX	10488	40.8716697	-73.897797	(40.8716697, -73.897797)	WEST 197 ST	RESERVOIR AVENUE		0	

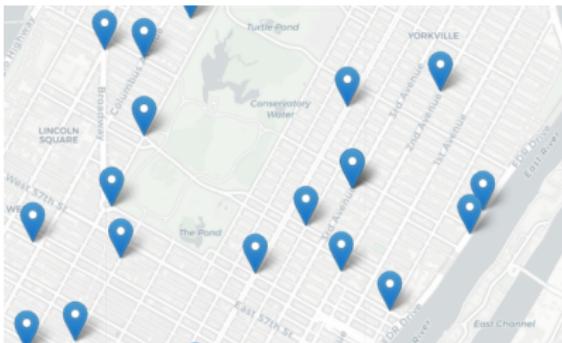


## Steps:

- Download the data as a CSV file and store on your computer.

## Extracting Data for a Map

A	B	C	D	E	F	G	H	I	J	K	LOCATION	ON STREET	CROSS STREET	OFF STREET	NUMBER OF
10/18/16	8:10	STATEN ISL.	10132	40.545058	-74.193137	40.545058	RATHBUN AV	UPPER NARROWS							1
10/18/16	8:10	BROOKLYN	11238	40.668645	-73.986107	40.668645	73.986107								107 GREE
10/18/16	8:10	BROOKLYN	11234	40.626173	-73.922336	40.626173	73.922336								5515 AVEI
10/18/16	8:10	BROOKLYN	11218	40.631756	-73.976072	40.631756	73.976072	UPPER NARROWS	EAST 2 STREET						1
10/18/16	8:10	BROOKLYN	11217	40.631756	-73.976074	40.631756	73.976074	UPPER NARROWS	EAST 2 STREET						1
10/18/16	8:10			40.771406	-73.829353	40.771406	73.829353								718 ATLA
10/18/16	8:10			40.771378	-73.827133	40.771378	73.827133								0
10/18/16	8:10			40.771378	-73.827133	40.771378	73.827133								0
10/18/16	8:10			40.741532	-73.784555	40.741532	73.784555								0
10/18/16	8:10			40.736537	-73.856099	40.736537	73.856097								0
10/18/16	8:10			40.736184	-73.926504	40.736184	73.926504								0
10/18/16	8:10	QUEENS	11426	40.735180	-73.783346	40.735180	73.783346	HILLTOP AV	AV 249 STREET						0
10/18/16	8:10	QUEENS	11417	40.671931	-73.885135	40.671931	73.885135	NORTH CORN 141 ST	AV 249 STREET						0
10/18/16	8:10	QUEENS	11355	40.760672	-73.821007	40.760672	73.821007								144-20 41
10/18/16	8:10	QUEENS	11103	40.762389	-73.911797	40.762389	73.911797	W 234 STREET	34 STREET	34 STREET	34 STREET	34 STREET	34 STREET	34 STREET	30 AVENUE
10/18/16	8:10	QUEENS	11101	40.762458	-73.930404	40.762458	73.930404								43 AVENUE
10/18/16	8:10	MANHATTAN	10935	40.805053	-73.933602	40.805053	73.933602	MANHATTAN	125 EAST 25 PARK AVENUE						0
10/18/16	8:10	MANHATTAN	10935	40.805053	-73.933602	40.805053	73.933602	MANHATTAN	125 EAST 25 PARK AVENUE						607 8 AVI
10/18/16	8:10	MANHATTAN	10916	40.743783	-73.971508	40.743783	73.971508	MANHATTAN	125 EAST 25 PARK AVENUE						0
10/18/16	8:10	MANHATTAN	10911	40.745825	-73.000118	40.745825	73.000118	MANHATTAN	225 WEST 27 5TH AVENUE						0
10/18/16	8:10	MANHATTAN	10003	40.731795	-73.982214	40.731795	73.982214	MANHATTAN	225 WEST 27 5TH AVENUE						251 1 AVI
10/18/16	8:10	BROOKLYN	11234	40.6166112	-73.926628	40.6166112	73.926628	UTICA AVENUE	O VENUE						0
10/18/16	8:10	BROOKLYN	11222	40.7317513	-73.945513	40.7317513	73.945513	UTICA AVENUE	O VENUE						0
10/18/16	8:10	BROOKLYN	11206	40.707655	-73.939886	40.707655	73.939886	W 234 STREET	W 234 STREET	W 234 STREET	W 234 STREET	W 234 STREET	W 234 STREET	W 234 STREET	W 234 STREET
10/18/16	8:10	BRONX	10468	40.8716697	-73.889797	40.8716697	73.889797	W 169 STREET	W 197 1ST RESERVOIR AVENUE						0

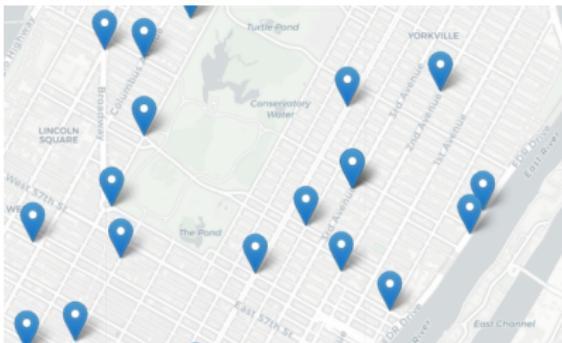


- Steps:

- ▶ Download the data as a CSV file and store on your computer.
  - ▶ Filter for latitude and longitude (not blank).

# Extracting Data for a Map

A	B	C	D	E	F	G	H	I	J	K
DATE	TIME	BOROUGH	ZIP CODE	LATITUDE	LONGITUDE	LOCATION	ON STREET A	CROSS STREET OFF STREET I	NUMBER OF	
10/18/16	8:10	STATEN ISLA	10312	40.540547	-74.191197	(40.540550	NATHUIN AV	NIPPON AVENUE	1	
10/18/16	8:10	BROOKLYN	11238	40.886547	-74.193107	(40.886550	TUTTLE ROND		1	
10/18/16	8:10	BROOKLYN	11234	40.626173	-73.922234	(40.626175	73.922236)		5515	AVEI
10/18/16	8:10	BROOKLYN	11218	40.626173	-73.922234	(40.626175	73.922236)	EAST 2 STREET	1	
10/18/16	8:10	BROOKLYN	11217	40.627165	-73.922234	(40.627167	73.922239)		718	ATLA
10/18/16	8:10			40.7415268	-73.829393	(40.741527	73.829396)		0	
10/18/16	8:10			40.7112158	-73.877133	(40.711216	73.877131)		0	
10/18/16	8:10			40.7415268	-73.784552	(40.741526	73.7845496)		0	
10/18/16	8:10			40.7365373	-73.856099	(40.736537	73.8560987)		0	
10/18/16	8:10			40.7361484	-73.926504	(40.736148	73.9265043)		0	
10/18/16	8:10	QUEENS	11426	40.7351801	-73.718346	(40.735180	HILLSIDE AV	249 STREET	0	
10/18/16	8:10	QUEENS	11417	40.6719314	-73.851635	(40.671931	NORTH CONI	84 STREET	0	
10/18/16	8:10	QUEENS	11355	40.7606728	-73.821007	(40.760672	73.8210074)	144-20	41	
10/18/16	8:10	QUEENS	11103	40.7623893	-73.911797	(40.762389	73.911797)	45 STREET	30 AVENUE	0
10/18/16	8:10	QUEENS	11101	40.7462598	-73.939049	(40.746259	73.939049)	34 STREET	43 AVENUE	0
10/18/16	8:10	MANHATTAN	10035	40.8050573	-73.939034	(40.805057	EAST 125 ST	PARK AVENUE	0	
10/18/16	8:10	MANHATTAN	10018	40.7556432	-73.990962	(40.755643	73.9909619)	607	8 AVI	0
10/18/16	8:10	MANHATTAN	10016	40.7437832	-73.973508	(40.743783	73.973508)	EAST 34 ST	R 1 AVENUE	0
10/18/16	8:10	MANHATTAN	10011	40.7458255	-74.001812	(40.745825	74.001812)	WEST 22 ST	R 9 AVENUE	0
10/18/16	8:10	MANHATTAN	10003	40.7317985	-73.882114	(40.731798	73.8821143)	251	1 AVI	0
10/18/16	8:10	BROOKLYN	11234	40.6166152	-73.926511	(40.616615	73.926511)	AVENUE D	0	
10/18/16	8:10	BROOKLYN	11226	40.7071511	-73.945511	(40.707151	73.945511)	NOVIA STR	GREENPOINT AVENUE	0
10/18/16	8:10	BROOKLYN	11206	40.707165	-73.938839	(40.707165	73.938839)	BUSHWICK	MONTROSE AVENUE	0
10/18/16	8:10	BRONX	10488	40.8716697	-73.897797	(40.871669	73.897797)	WEST 197 ST	RESERVOIR AVENUE	0

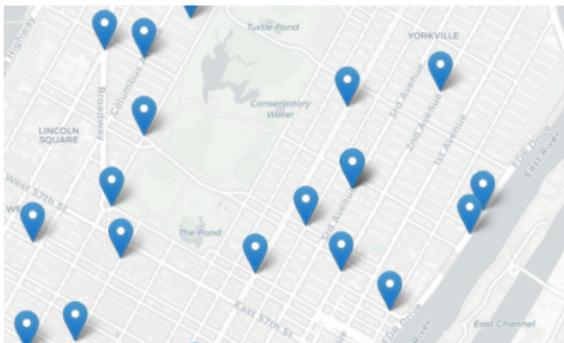


## Steps:

- Download the data as a CSV file and store on your computer.
- Filter for latitude and longitude (not blank).
- Use pandas to read in data.

# Extracting Data for a Map

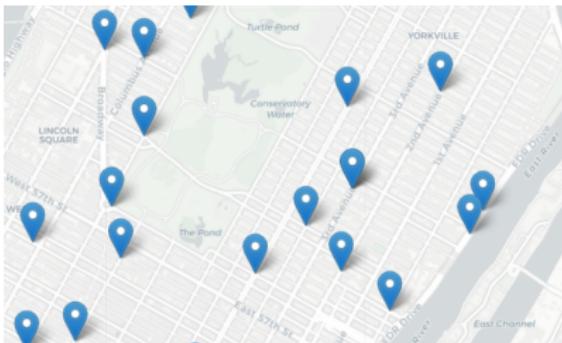
A	B	C	D	E	F	G	H	I	J	K
DATE	TIME	BOROUGH	ZIP CODE	LATITUDE	LONGITUDE	LOCATION	ON STREET A	CROSS STREET OFF STREET I	NUMBER OF	
10/18/16	8:10	STATEN ISLA	10312	40.540550	-74.191197	NATHBUN AV NIPPON AVENUE			1	
10/18/16	8:10	BROOKLYN	11238	40.886547	-73.968107		107	GREE	1	
10/18/16	8:10	BROOKLYN	11234	40.626173	-73.922234	(40.6261739,-73.9222336)	5515	AVEI	1	
10/18/16	8:10	BROOKLYN	11218	40.626173	-73.922234	(40.6261739,-73.9222336)	1	EAST 2 STREET		
10/18/16	8:10	BROOKLYN	11217	40.627165	-73.972234	(40.627165,-73.9722339)	718	ATLA	1	
10/18/16	8:10			40.771278	-73.82933	(40.771278,-73.829339)	0			
10/18/16	8:10			40.771278	-73.877131	(40.771278,-73.8771311)	0			
10/18/16	8:10			40.741526	-73.78455	(40.741526,-73.7845496)	0			
10/18/16	8:10			40.7365373	-73.856099	(40.7365373,-73.8560987)	0			
10/18/16	8:10			40.7361484	-73.926504	(40.7361484,-73.9265043)	0			
10/18/16	8:10	QUEENS	11426	40.7351801	-73.718346	HILLSIDE AVI 249 STREET	0			
10/18/16	8:10	QUEENS	11417	40.6719314	-73.851635	(40.6719314,NORTH CONI BA STREET	0			
10/18/16	8:10	QUEENS	11355	40.7606728	-73.821007	(40.7606728,-73.8210074)	144-20	41		
10/18/16	8:10	QUEENS	11103	40.7623891	-73.911797	(40.7623893, 45 STREET) 30 AVENUE	0			
10/18/16	8:10	QUEENS	11101	40.7462598	-73.930409	(40.7462598, 34 STREET) 43 AVENUE	0			
10/18/16	8:10	MANHATTAN	10035	40.8050573	-73.939034	(40.8050573, EAST 125 ST PARK AVENUE	0			
10/18/16	8:10	MANHATTAN	10018	40.7556432	-73.990962	(40.7556432,-73.9909619)	607	8 AVI	0	
10/18/16	8:10	MANHATTAN	10016	40.7437832	-73.973508	(40.7437832, EAST 34 ST RI AVENUE	0			
10/18/16	8:10	MANHATTAN	10011	40.7458255	-74.001812	(40.7458255, WEST 22 ST) 9 AVENUE	0			
10/18/16	8:10	MANHATTAN	10003	40.7317985	-73.882114	(40.7317985,-73.8821143)	251	1 AVI	0	
10/18/16	8:10	BROOKLYN	11234	40.616651	-73.926504	(40.616651,-73.926504)	0			
10/18/16	8:10	BROOKLYN	11226	40.7075511	-73.945511	(40.7075511,-73.945511)	0			
10/18/16	8:10	BROOKLYN	11206	40.707165	-73.938839	(40.707165,-73.938839)	0			
10/18/16	8:10	BRONX	10488	40.8716897	-73.897797	(40.8716897, WEST 197 ST RESERVOIR AVENUE	0			



- Steps:
  - ▶ Download the data as a CSV file and store on your computer.
  - ▶ Filter for latitude and longitude (not blank).
  - ▶ Use pandas to read in data.
- Many ways to access data from pandas:

# Extracting Data for a Map

A	B	C	D	E	F	G	H	I	J	K
DATE	TIME	BOROUGH	ZIP CODE	LATITUDE	LONGITUDE	LOCATION	ON STREET A	CROSS STREET OFF STREET I	NUMBER OF	
10/18/16	8:10	STATEN ISLA	10312	40.5405508	-74.193197	(40.5405508, -74.193197)	NATHBUN AV	NIPPON AVENUE	107	GREE
10/18/16	8:10	BROOKLYN	11238	40.8865457	-73.968107	(40.8865457, -73.968107)			1	
10/18/16	8:10	BROOKLYN	11234	40.6261739	-73.922234	(40.6261739, -73.922234)			5515	AVEI
10/18/16	8:10	BROOKLYN	11218	40.6261739	-73.922234	(40.6261739, -73.922234)	BUCHANAN AV	EAST 2 STREET	1	
10/18/16	8:10	BROOKLYN	11217	40.6273765	-73.972234	(40.6273765, -73.972234)			718	ATLA
10/18/16	8:10			40.74741705	-73.829393	(40.74741705, -73.829393)			1	
10/18/16	8:10			40.7117578	-73.8771331	(40.7117578, -73.8771331)			0	
10/18/16	8:10			40.7415268	-73.784552	(40.7415268, -73.784552)			0	
10/18/16	8:10			40.7365373	-73.856099	(40.7365373, -73.856099)			0	
10/18/16	8:10			40.7361484	-73.926504	(40.7361484, -73.926504)			0	
10/18/16	8:10	QUEENS	11246	40.7351801	-73.718346	(40.7351801, -73.718346)	HILLSIDE AV	249 STREET	0	
10/18/16	8:10	QUEENS	11417	40.6719314	-73.851635	(40.6719314, -73.851635)	NORTH CONI	84 STREET	0	
10/18/16	8:10	QUEENS	11355	40.7606728	-73.821007	(40.7606728, -73.821007)			144	20 41
10/18/16	8:10	QUEENS	11103	40.7623891	-73.911797	(40.7623891, -73.911797)	345 STREET	30 AVENUE	0	
10/18/16	8:10	QUEENS	11101	40.7462598	-73.930409	(40.7462598, -73.930409)			0	
10/18/16	8:10	MANHATTAN	10035	40.8050573	-73.939034	(40.8050573, -73.939034)	EAST 125 ST	PARK AVENUE	0	
10/18/16	8:10	MANHATTAN	10018	40.7556432	-73.990962	(40.7556432, -73.990962)			607	8 AVI
10/18/16	8:10	MANHATTAN	10016	40.7473832	-73.975108	(40.7473832, -73.975108)	EAST 34 ST	R 1 AVENUE	0	
10/18/16	8:10	MANHATTAN	10011	40.7458255	-74.001812	(40.7458255, -74.001812)	WEST 22 ST	R 9 AVENUE	0	
10/18/16	8:10	MANHATTAN	10003	40.7317985	-73.882114	(40.7317985, -73.882114)			251	1 AVI
10/18/16	8:10	BROOKLYN	11234	40.6166151	-73.926504	(40.6166151, -73.926504)			0	
10/18/16	8:10	BROOKLYN	11226	40.7075511	-73.945511	(40.7075511, -73.945511)	BUCHANAN AV	GREENPOINT AVENUE	0	
10/18/16	8:10	BROOKLYN	11206	40.707165	-73.985898	(40.707165, -73.985898)	BUSHWICK	MONTROSE AVENUE	0	
10/18/16	8:10	BRONX	10488	40.8716897	-73.897797	(40.8716897, -73.897797)	WEST 197 ST	RESERVOIR AVENUE	0	



- Steps:

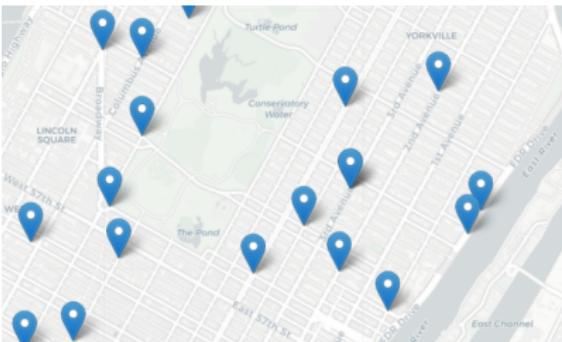
- ▶ Download the data as a CSV file and store on your computer.
- ▶ Filter for latitude and longitude (not blank).
- ▶ Use pandas to read in data.

- Many ways to access data from pandas:

- ▶ Since we are extracting row-by-row to create objects, will iterate through the DataFrame (`df.iterrows()`).

# Extracting Data for a Map

A	B	C	D	E	F	G	H	I	J	K
DATE	TIME	BOROUGH	ZIP CODE	LATITUDE	LONGITUDE	LOCATION	ON STREET A	CROSS STREET OFF STREET I	NUMBER OF	
10/18/16	8:10	STATEN ISLA	10312	40.5405508	-74.191197	ON HATHBUN AV NIPPON AVENUE			1	
10/18/16	8:10	BROOKLYN	11238	40.8866547	-73.968107	40.8866547, -73.9681074	107	GREE	1	
10/18/16	8:10	BROOKLYN	11234	40.6261739	-73.922234	40.6261739, -73.9222336	5515	AVEI	1	
10/18/16	8:10	BROOKLYN	11218	40.7362758	-73.9222758	40.7362758, -73.9222758	1	EAST 2 STREET		
10/18/16	8:10	BROOKLYN	11217	40.6827165	-73.972234	40.6827165, -73.9722339	718	ATLA	1	
10/18/16	8:10			40.74741705	-73.829393	40.74741705, -73.829396			0	
10/18/16	8:10			40.71121758	-73.8771331	40.71121758, -73.8771331			0	
10/18/16	8:10			40.7415268	-73.784552	40.7415268, -73.7845496			0	
10/18/16	8:10			40.7365373	-73.856099	40.7365373, -73.8560987			0	
10/18/16	8:10			40.7361484	-73.926504	40.7361484, -73.9265043			0	
10/18/16	8:10	QUEENS	11426	40.7351801	-73.718346	40.7351801, -73.718346			0	
10/18/16	8:10	QUEENS	11417	40.6719314	-73.851635	40.6719314 NORTH CONI BA STREET			0	
10/18/16	8:10	QUEENS	11355	40.7606728	-73.821007	40.7606728, -73.8210074	144-20	41		
10/18/16	8:10	QUEENS	11103	40.7623891	-73.911797	40.7623891 45 STREET	30 AVENUE		0	
10/18/16	8:10	QUEENS	11101	40.7462598	-73.930409	40.7462598, -73.930409	43 AVENUE		0	
10/18/16	8:10	MANHATTAN	10035	40.8050573	-73.939034	40.8050573 EAST 125 ST PARK AVENUE			0	
10/18/16	8:10	MANHATTAN	10018	40.7556432	-73.990962	40.7556432, -73.9909619	607	8 AVI	0	
10/18/16	8:10	MANHATTAN	10016	40.7473832	-73.973508	40.7473832 EAST 34 ST RI AVENUE			0	
10/18/16	8:10	MANHATTAN	10011	40.7458255	-74.001812	40.7458255 WEST 22 STR 9 AVENUE			0	
10/18/16	8:10	MANHATTAN	10003	40.7317985	-73.882114	40.7317985, -73.8821143	251	1 AVI	0	
10/18/16	8:10	BROOKLYN	11234	40.6166151	-73.926504	40.6166151, -73.926504			0	
10/18/16	8:10	BROOKLYN	11229	40.7075511	-73.945511	40.7075511, -73.945511	1120K	STR GREENPOINT AVENUE	0	
10/18/16	8:10	BROOKLYN	1120K	40.707165	-73.938839	40.707165, -73.938839			0	
10/18/16	8:10	BRONX	10488	40.8716897	-73.897797	40.8716897 WEST 197 ST RESERVOIR AVENUE			0	



- Steps:

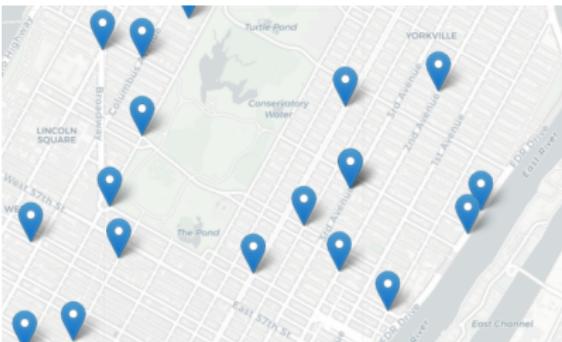
- ▶ Download the data as a CSV file and store on your computer.
- ▶ Filter for latitude and longitude (not blank).
- ▶ Use pandas to read in data.

- Many ways to access data from pandas:

- ▶ Since we are extracting row-by-row to create objects, will iterate through the DataFrame (`df.iterrows()`).
- ▶ Can also, pull comments & iterate over zip:  
`for (lat,lon) in zip(df["LATITUDE"],df["LONGITUDE"])`

# Extracting Data for a Map

A	B	C	D	E	F	G	H	I	J	K
DATE	TIME	BOROUGH	ZIP CODE	LATITUDE	LONGITUDE	LOCATION	ON STREET A	CROSS STREET B	OFF STREET C	NUMBER OF
10/18/16	8:10	STATEN ISLA	10312	40.5405508	-74.191397	(40.5405508, -74.191397)	NATHBUN AV	NIKKON AVENUE		1
10/18/16	8:10	BROOKLYN	11238	40.886547	-73.968107	(40.886547, -73.968107)				107 GREE
10/18/16	8:10	BROOKLYN	11234	40.626173	-73.922234	(40.626173, -73.922234)				5515 AVE1
10/18/16	8:10	BROOKLYN	11218	40.626173	-73.922234	(40.626173, -73.922234)	2ND ST	EAST 2 STREET		1
10/18/16	8:10	BROOKLYN	11217	40.626173	-73.922234	(40.626173, -73.922234)			718 ATL	1
10/18/16	8:10			40.74731765	-73.829393	(40.74731765, -73.829393)				0
10/18/16	8:10			40.7117578	-73.877133	(40.7117578, -73.877133)				0
10/18/16	8:10			40.7415268	-73.784552	(40.7415268, -73.784552)				0
10/18/16	8:10			40.7365573	-73.856099	(40.7365573, -73.856099)				0
10/18/16	8:10			40.7361484	-73.926504	(40.7361484, -73.926504)				0
10/18/16	8:10	QUEENS	11246	40.7351801	-73.718346	(40.7351801, -73.718346)	HILLSIDE AV	I 249 STREET		0
10/18/16	8:10	QUEENS	11417	40.6719314	-73.851635	(40.6719314, -73.851635)	NORTH CONI	84 STREET		0
10/18/16	8:10	QUEENS	11355	40.7606728	-73.821007	(40.7606728, -73.821007)			144-20 41	1
10/18/16	8:10	QUEENS	11103	40.7623891	-73.911797	(40.7623891, -73.911797)	EAST 125 ST	PARK AVENUE		0
10/18/16	8:10	QUEENS	11101	40.7462598	-73.930409	(40.7462598, -73.930409)			43 AVENUE	0
10/18/16	8:10	MANHATTAN	10035	40.8050573	-73.939034	(40.8050573, -73.939034)	10 AVENUE			0
10/18/16	8:10	MANHATTAN	10018	40.7556432	-73.990962	(40.7556432, -73.990962)			607 8 AVI	0
10/18/16	8:10	MANHATTAN	10016	40.7473832	-73.973508	(40.7473832, -73.973508)	EAST 34 ST	R 1 AVENUE		0
10/18/16	8:10	MANHATTAN	10011	40.7458255	-74.001812	(40.7458255, -74.001812)	WEST 22 ST	R 9 AVENUE		0
10/18/16	8:10	MANHATTAN	10003	40.7317985	-73.882114	(40.7317985, -73.882114)			251 1 AVI	0
10/18/16	8:10	BROOKLYN	11234	40.616615	-73.926251	(40.616615, -73.926251)	1ST AVENUE	O		0
10/18/16	8:10	BROOKLYN	11226	40.7075511	-73.945511	(40.7075511, -73.945511)	BUSHWICK STR	GREENPOINT AVENUE		0
10/18/16	8:10	BROOKLYN	11206	40.707165	-73.984839	(40.707165, -73.984839)			BUSHWICK A MONTROSE AVENUE	0
10/18/16	8:10	BRONX	10488	40.8716897	-73.897797	(40.8716897, -73.897797)	WEST 197 ST	RESERVOIR AVENUE		0



- Steps:

- ▶ Download the data as a CSV file and store on your computer.
- ▶ Filter for latitude and longitude (not blank).
- ▶ Use pandas to read in data.

- Many ways to access data from pandas:

- ▶ Since we are extracting row-by-row to create objects, will iterate through the DataFrame (`df.iterrows()`).
- ▶ Can also, pull comments & iterate over zip:  
`for (lat,lon) in zip(df["LATITUDE"],df["LONGITUDE"])`
- ▶ Using `apply()` is possible, but it returns a series or DataFrame, so, processing still required.

# Extracting Data for a Map

- Python program:

# Extracting Data for a Map

- Python program:

```
#Mapping Collisions  
  
#Libraries  
import folium  
import pandas as pd
```

# Extracting Data for a Map

- Python program:

```
#Mapping Collisions  
  
#Libraries  
import folium  
import pandas as pd  
  
#Getting file names:  
inF = input('Enter CSV file name: ')  
outF = input('Enter output file: ')  
coll = pd.read_csv(inF)
```

# Extracting Data for a Map

- Python program:

```
#Mapping Collisions

#Libraries
import folium
import pandas as pd

#Getting file names:
inF = input('Enter CSV file name: ')
outF = input('Enter output file: ')
coll = pd.read_csv(inF)

#Setting up the map:
mapCollisions = folium.Map(location=[40.768731, -73.964915],\
                           tiles="Cartodb Positron",zoom_start=11)
```

# Extracting Data for a Map

- Python program:

```
#Mapping Collisions

#Libraries
import folium
import pandas as pd

#Getting file names:
inF = input('Enter CSV file name: ')
outF = input('Enter output file: ')
coll = pd.read_csv(inF)

#Setting up the map:
mapCollisions = folium.Map(location=[40.768731, -73.964915],\
                           tiles="Cartodb Positron",zoom_start=11)

#Looping through the file:
for index, row in coll.iterrows():
    lat = row["LATITUDE"]
    lon = row["LONGITUDE"]
    popname = row["CRASH TIME"]
    newMarker = folium.CircleMarker([lat, lon], popup=popname,\
                                    radius=5,color='blue')
    newMarker.add_to(mapCollisions)
```

# Extracting Data for a Map

- Python program:

```
#Mapping Collisions

#Libraries
import folium
import pandas as pd

#Getting file names:
inF = input('Enter CSV file name: ')
outF = input('Enter output file: ')
coll = pd.read_csv(inF)

#Setting up the map:
mapCollisions = folium.Map(location=[40.768731, -73.964915],\
                           tiles="Cartodb Positron",zoom_start=11)

#Looping through the file:
for index, row in coll.iterrows():
    lat = row["LATITUDE"]
    lon = row["LONGITUDE"]
    popname = row["CRASH TIME"]
    newMarker = folium.CircleMarker([lat, lon], popup=popname,\
                                    radius=5,color='blue')
    newMarker.add_to(mapCollisions)

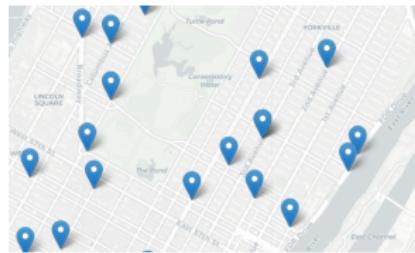
#Saving the HTML file:
mapCollisions.save(outfile=outF)
```

# Extracting Data



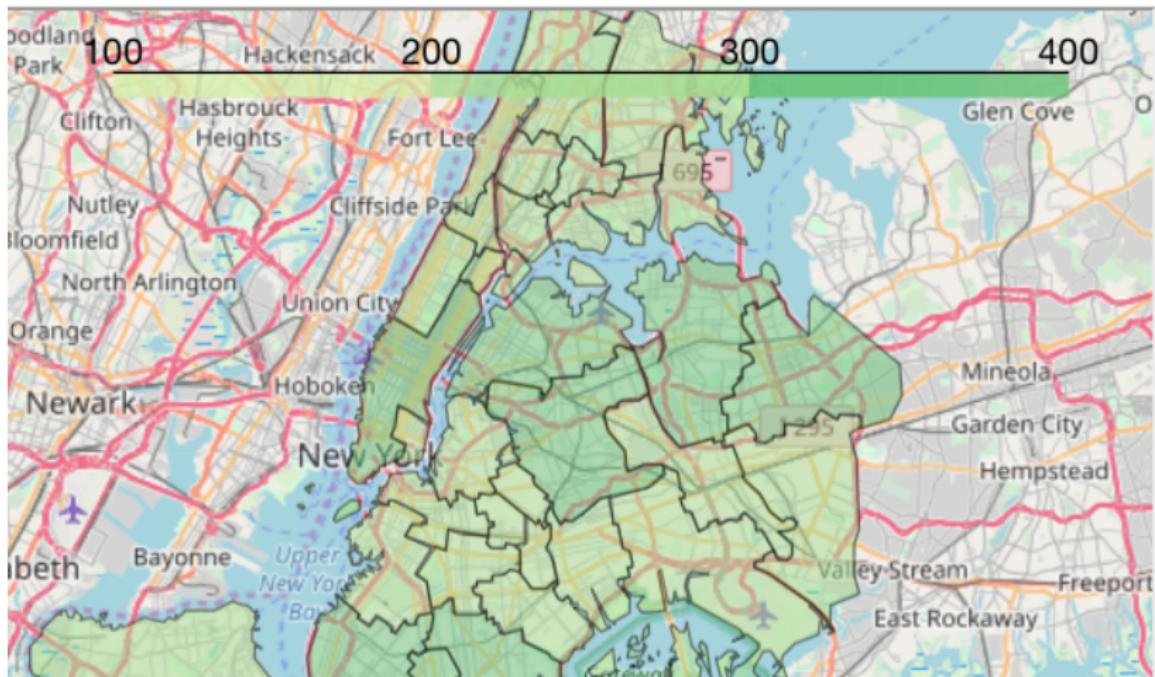
*Make a map with only the accidents during evening rush hour.*

# Outline



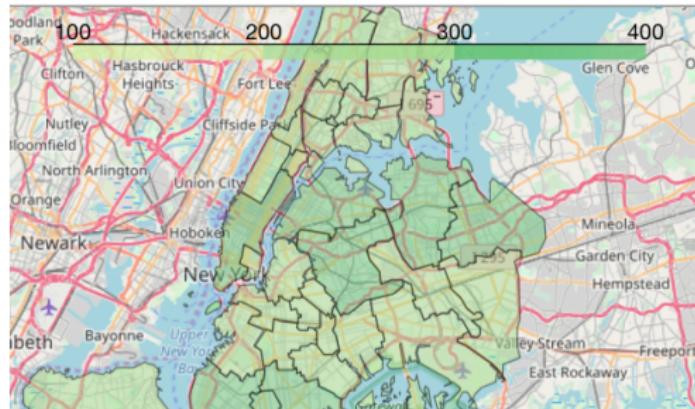
- Recap
- HTML-Scalable Maps: Folium
- Extracting Data
- **geoJSON Format & Choropleth Maps**
- Break
- Design Challenge: Catchment Areas
- Design Challenge: Clustering Data
- Wrap Up

# geoJSON Format & Choropleth Maps



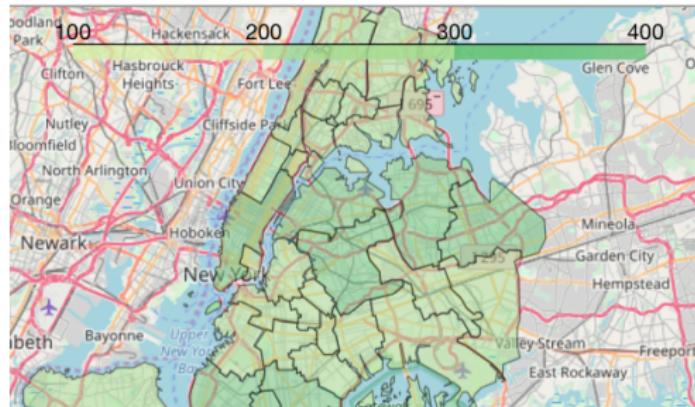
*School districts shaded by math test scores.*

# geoJSON Format & Choropleth Maps



*School districts shaded by math test scores.*

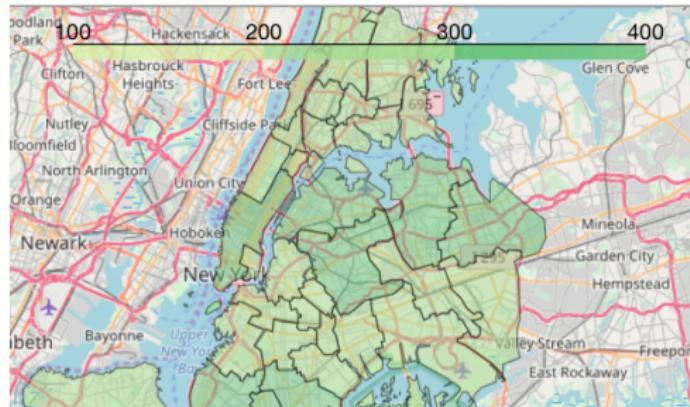
# geoJSON Format & Choropleth Maps



*School districts shaded by math test scores.*

Two data files:

# geoJSON Format & Choropleth Maps

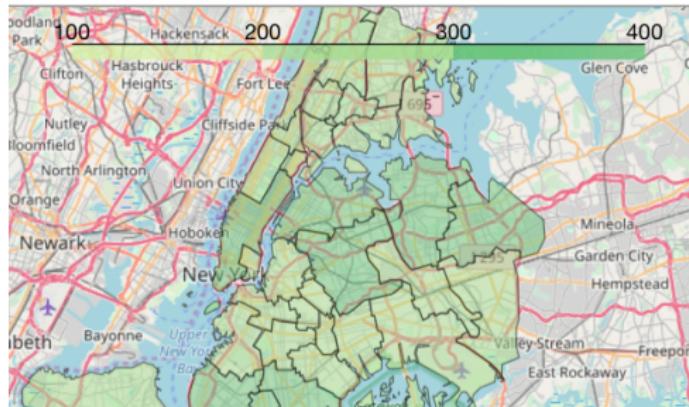


*School districts shaded by math test scores.*

Two data files:

- geoJSON file with polygonal regions (from OpenData NYC Planning)

# geoJSON Format & Choropleth Maps

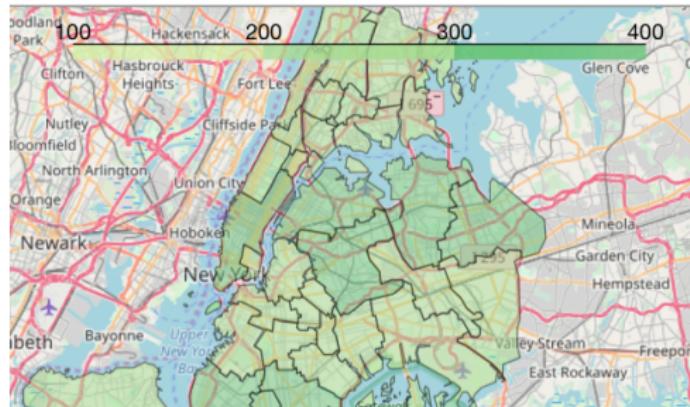


*School districts shaded by math test scores.*

Two data files:

- geoJSON file with polygonal regions (from OpenData NYC Planning)
- CSV file with test scores (NYC Department of Education)

# geoJSON Format & Choropleth Maps



*School districts shaded by math test scores.*

Two data files:

- geoJSON file with polygonal regions (from OpenData NYC Planning)
- CSV file with test scores (NYC Department of Education)

*(Links on webpage.)*

# geoJSON Format & Choropleth Maps

```
#Import folium for maps and pandas for data wrangling
import folium
import pandas as pd

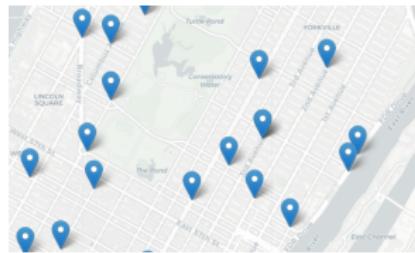
#Read in the test scores
fullData = pd.read_csv('math20132016.csv', skiprows = 6)
#Grab only 2016 data:
scores2016 = fullData[fullData.Year == 2016]
#Focus on 8th grade:
scores8th2016 = scores2016[fullData.Grade == "8"]
print(scores8th2016)

#Create a map:
schoolMap = folium.Map(location=[40.75, -74.125])

#Create a layer, shaded by test scores:
schoolMap.choropleth(geo_path="schoolDistricts.json",
                      fill_color='YlGn', fill_opacity=0.5, line_opacity=0.5,
                      threshold_scale = [100,200,300,400],
                      data = scores8th2016,
                      key_on='feature.properties.SchoolDist',
                      columns = ['district', 'Mean Scale Score']
                     )

#Output the map to an .html file:
schoolMap.save(outfile='testScores.html')
```

# Outline

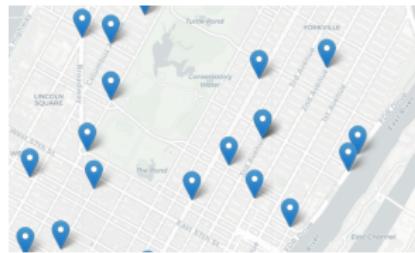


- Recap
- HTML-Scalable Maps: Folium
- Extracting Data
- geoJSON Format & Choropleth Maps
- **Break**
- Design Challenge: Catchment Areas
- Design Challenge: Clustering Data
- Wrap Up

# Break

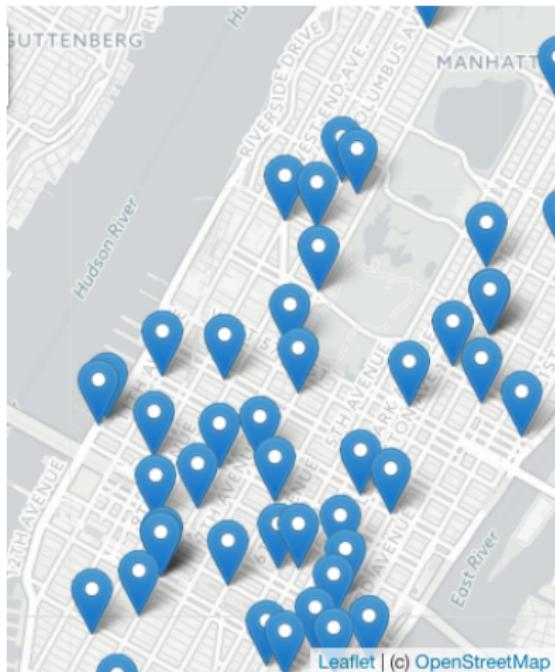


# Outline



- Recap
- HTML-Scalable Maps: Folium
- Extracting Data
- geoJSON Format & Choropleth Maps
- Break
- **Design Challenge: Catchment Areas**
- Design Challenge: Clustering Data
- Wrap Up

# Design Challenge: Catchment Areas



*Worksheet: design an algorithm to find catchment areas for NYC libraries.*

# Design Challenge: Approaching Problems

The diagram illustrates a 7-step approach to solving problems:

- 1 Listen**: Pay very close attention to any info in the problem description. You probably need it all for an optimal algorithm.
- 2 Example**: Most examples are too small or are special cases. Debug your example. Is there any way it's a special case? Is it big enough?
- 3 Brute Force**: Get a brute-force solution as soon as possible. Don't worry about developing an efficient algorithm yet. State a naive algorithm and its runtime, then optimize from there. Don't code yet though!
- 4 Optimize**: Walk through your brute force with BUD optimization or try some of these ideas:
  - Look for any unused info. You usually need all the information in a problem.
  - Solve it manually on an example, then reverse engineer your thought process. How did you solve it?
  - Solve it "incorrectly" and then think about why the algorithm fails. Can you fix those issues?
  - Make a time vs. space tradeoff. Hash tables are especially useful
- 5 Walk Through**: Now that you have an optimal solution, walk through your approach in detail. Make sure you understand each detail before you start coding.
- 6 Implement**: Your goal is to write beautiful code. Modularize your code from the beginning and refactor to clean up anything that isn't beautiful.
- 7 Test**: Test in this order:
  - Conceptual test. Walk through your code like you would for a detailed code review.
  - Unusual or non-standard code.
  - Hot spots, like arithmetic and null nodes.
  - Small test cases. It's much faster than a big first case and just as effective.
  - Special cases and edge cases.
  - And when you find bugs, fix them carefully!

**BUD Optimization** includes: Bottlenecks, Unnecessary Work, and Replicated Work.

**What You Need To Know**

- 1 Data Structures:** Hash Tables, Linked Lists, Stacks, Queues, Trees, Tries, Graphs, Vectors, Heaps.
- 2 Algorithms:** Quick Sort, Merge Sort, Binary Search, Breadth-First Search, Depth-First Search.
- 3 Concepts:** Big-O Time, Big-O Space, Recursion & Memoization, Probability, Bit Manipulation.

**Books by Gayle** include: Cracking the Coding Interview, Cracking the Coding Interview for Career, and Cracking the Coding Interview for Dummies.

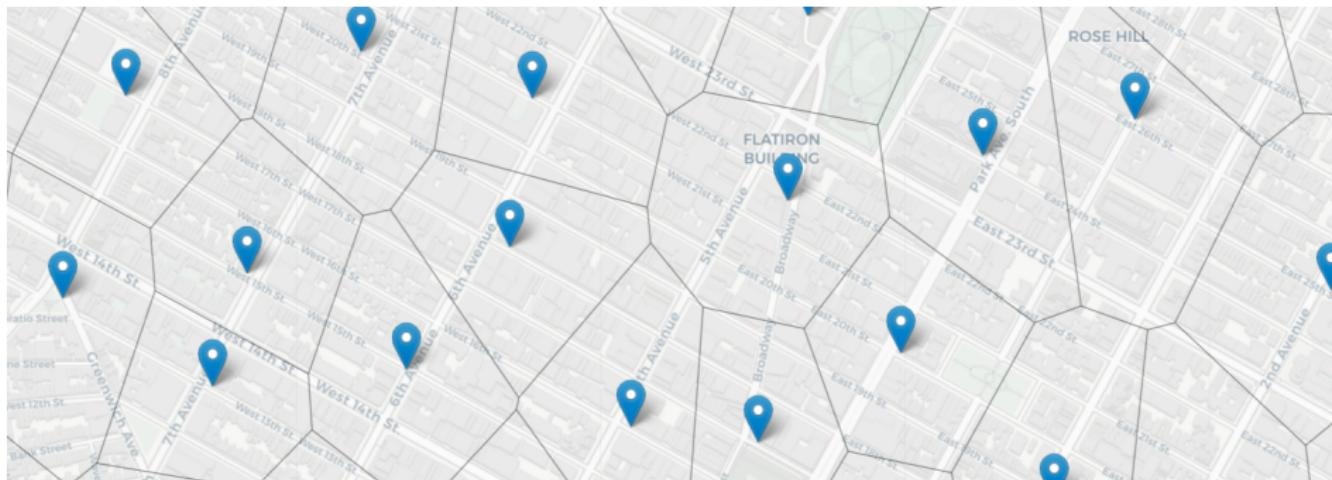
**Do not...**

- Do not ignore information given. Info is there for a reason.
- Do not try to solve problems in your head. Use an example!
- Do not push through code when confused. Stop and think!
- Do not dive into code without interviewer "sign off!"

© CareerCup.com

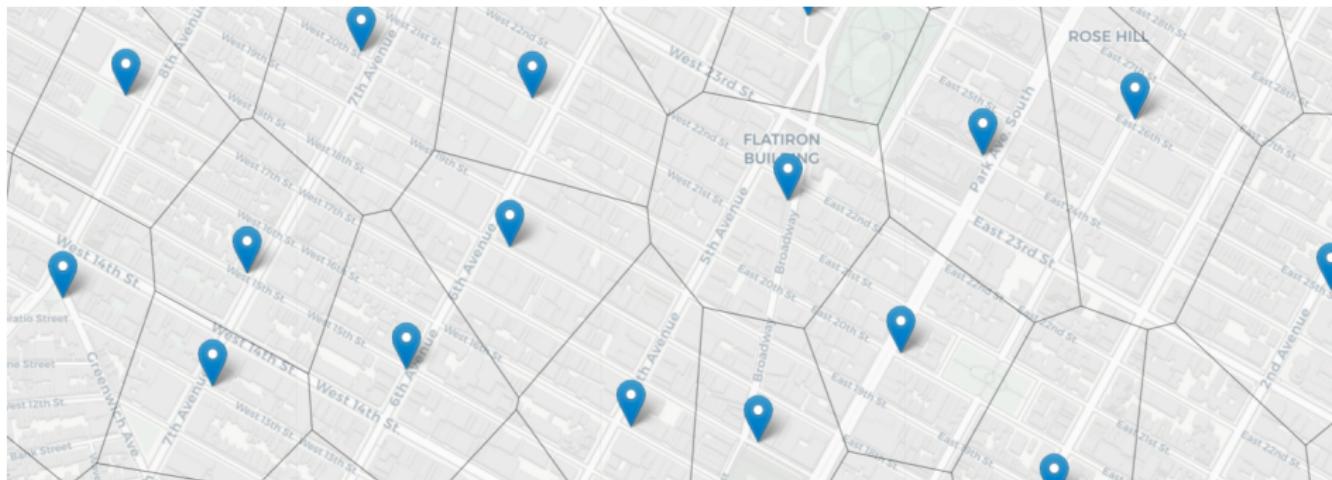
(From: *Cracking the Coding Interview*)

## Design Challenge: Catchment Areas



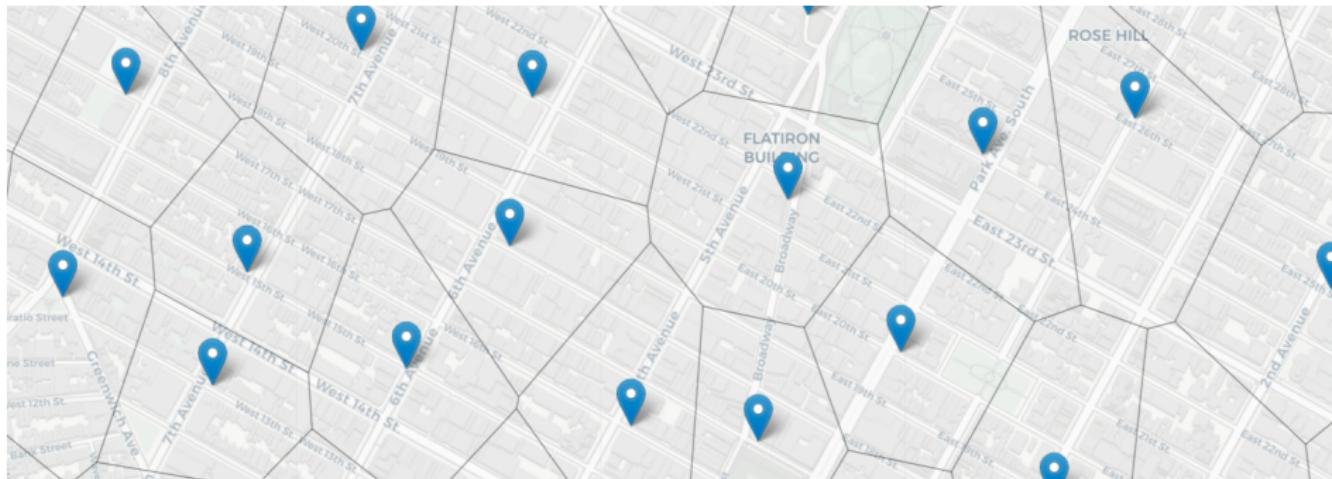
- Called **Voronoi Diagrams**.

# Design Challenge: Catchment Areas



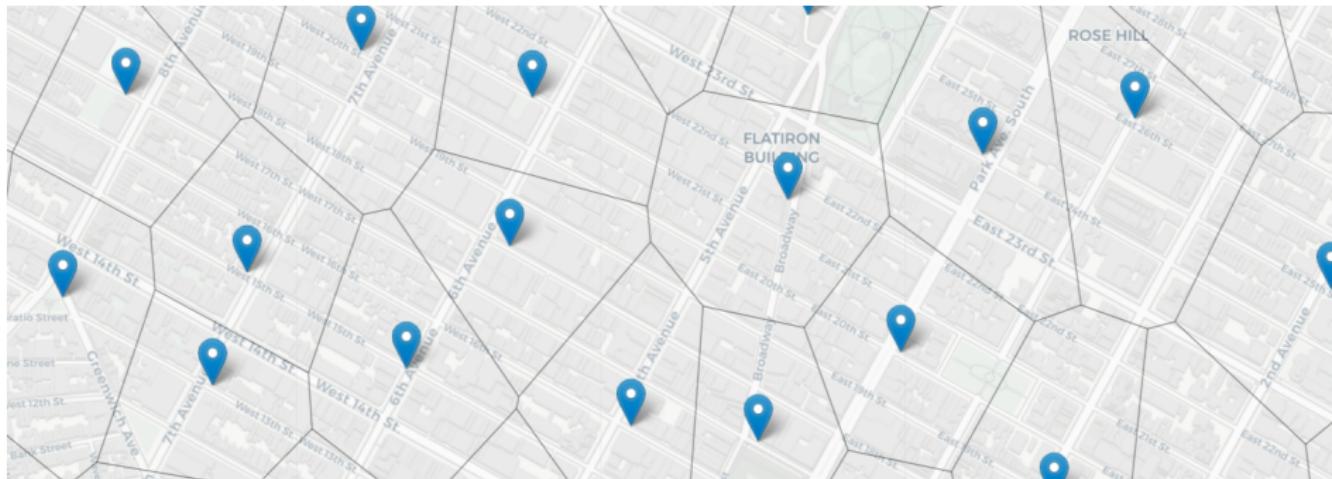
- Called **Voronoi Diagrams**.
- Can be computed in  $O(n^2)$  time: many different approaches.

# Design Challenge: Catchment Areas



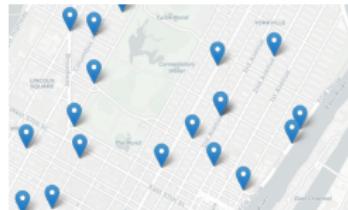
- Called **Voronoi Diagrams**.
- Can be computed in  $O(n^2)$  time: many different approaches.
- Share your approaches.

# Design Challenge: Catchment Areas



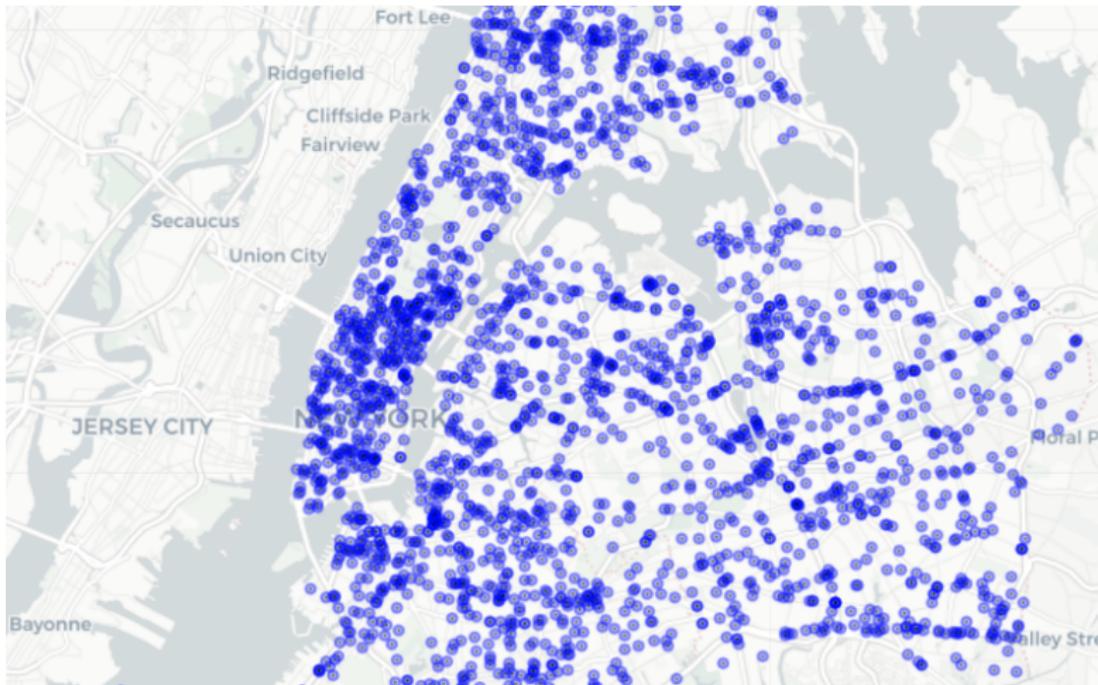
- Called **Voronoi Diagrams**.
- Can be computed in  $O(n^2)$  time: many different approaches.
- Share your approaches. (*Links on webpage of some approaches.*)

# Outline



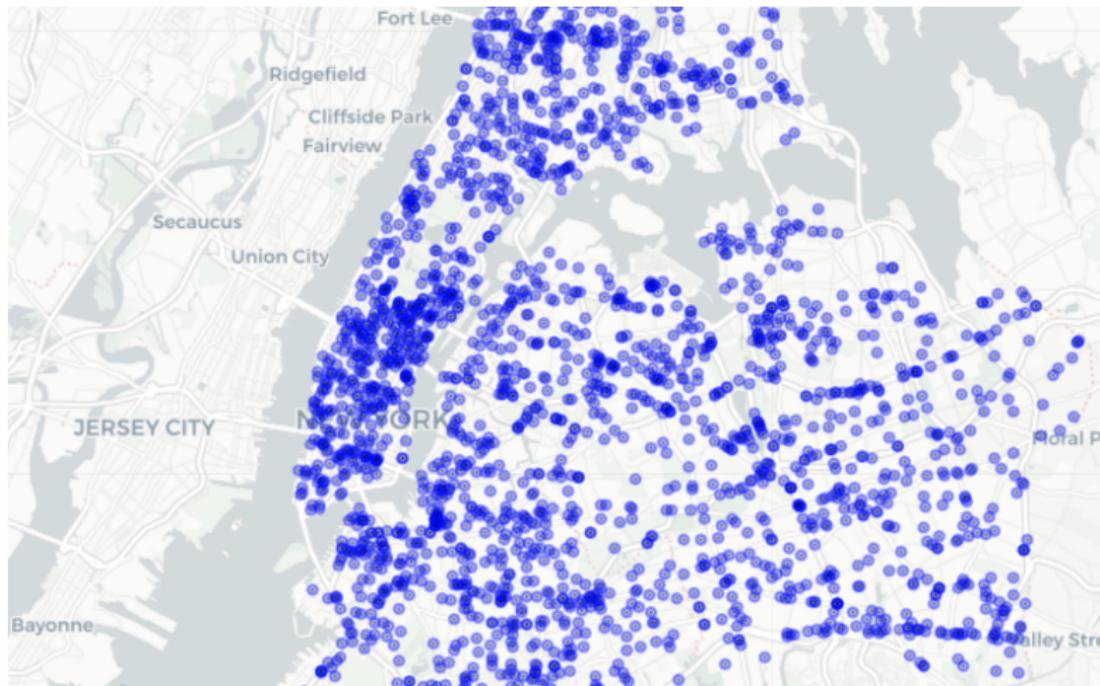
- Recap
- HTML-Scalable Maps: Folium
- Extracting Data
- geoJSON Format & Choropleth Maps
- Break
- Design Challenge: Catchment Areas
- **Design Challenge: Clustering Data**
- Wrap Up

# Design Challenge: Clustering Data



*You have 3 emergency service trucks.*

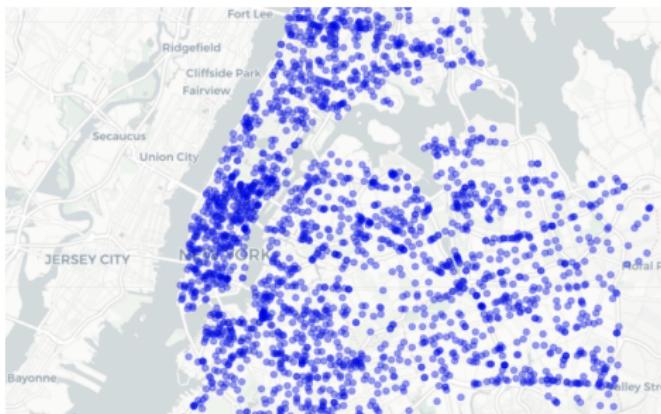
# Design Challenge: Clustering Data



*You have 3 emergency service trucks.*

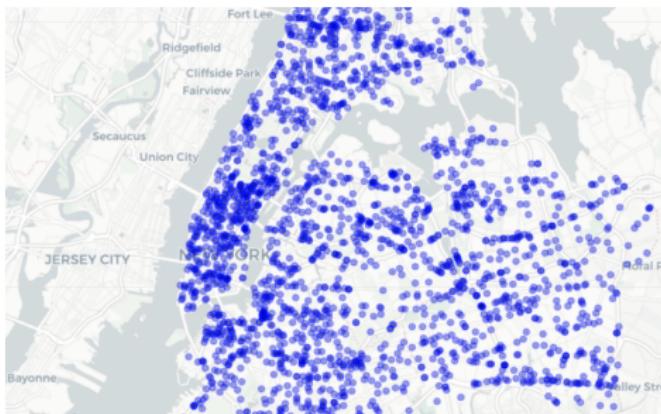
*Where to put them to minimize distances to collisions?*

# Design Challenge: Clustering Data



*You have 3 emergency service trucks.  
Where to put them to minimize distances to collisions?*

# Design Challenge: Clustering Data

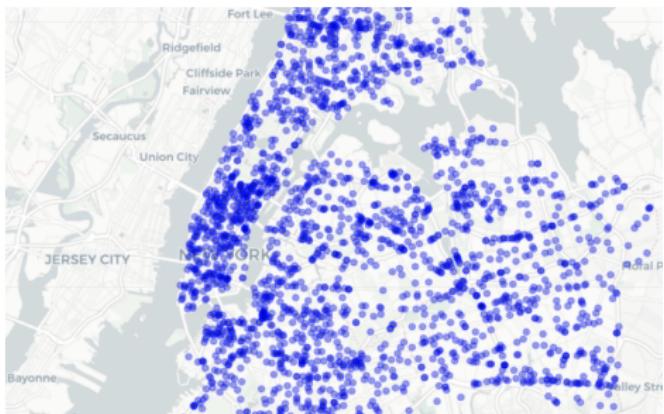


*You have 3 emergency service trucks.*

*Where to put them to minimize distances to collisions?*

- Called ***k-means clustering***.

# Design Challenge: Clustering Data

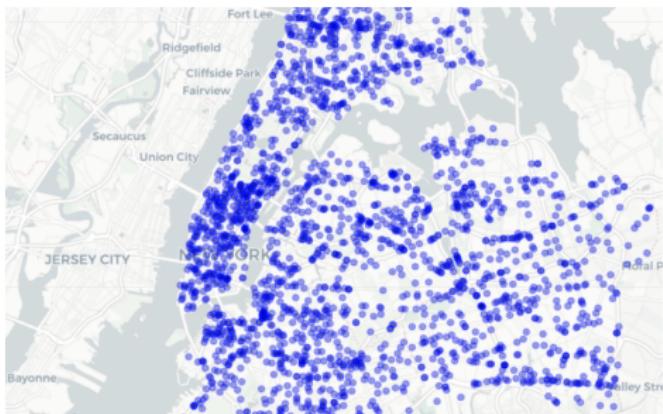


*You have 3 emergency service trucks.*

*Where to put them to minimize distances to collisions?*

- Called  **$k$ -means clustering**.
- Computationally hard to compute.

# Design Challenge: Clustering Data

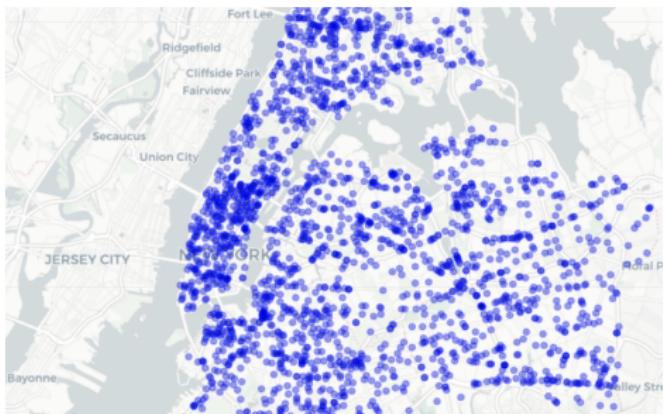


*You have 3 emergency service trucks.*

*Where to put them to minimize distances to collisions?*

- Called  **$k$ -means clustering**.
- Computationally hard to compute.
- Intuition for why: allowed to place the trucks anywhere (not restricted to inputted points) so many, many possible locations.

# Design Challenge: Clustering Data

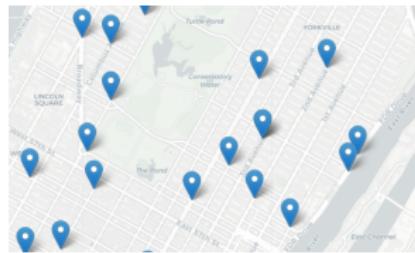


*You have 3 emergency service trucks.*

*Where to put them to minimize distances to collisions?*

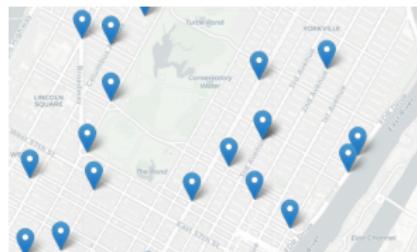
- Called  **$k$ -means clustering**.
- Computationally hard to compute.
- Intuition for why: allowed to place the trucks anywhere (not restricted to inputted points) so many, many possible locations.
- Approximations used instead.

# Outline



- Recap
- HTML-Scalable Maps: Folium
- Extracting Data
- geoJSON Format & Choropleth Maps
- Break
- Design Challenge: Catchment Areas
- Design Challenge: Clustering Data
- **Wrap Up**

# Wrap-Up



- Three sessions:
  - ① Flood Maps (arrays & images)
  - ② School Attendence (structured data, file I/O)
  - ③ Mapping Collisions (using objects, mapping coordinates)
- HTML-Scalable Maps: Folium
- Extracting Data: more on pandas
- geoJSON Format & Choropleth Maps
- Design Challenge: Catchment Areas
- Design Challenge: Clustering Data

# Thank you!



*(Image: NY Times)*