# Exploring San Francisco Bay Area
# -Date Science in Real Life

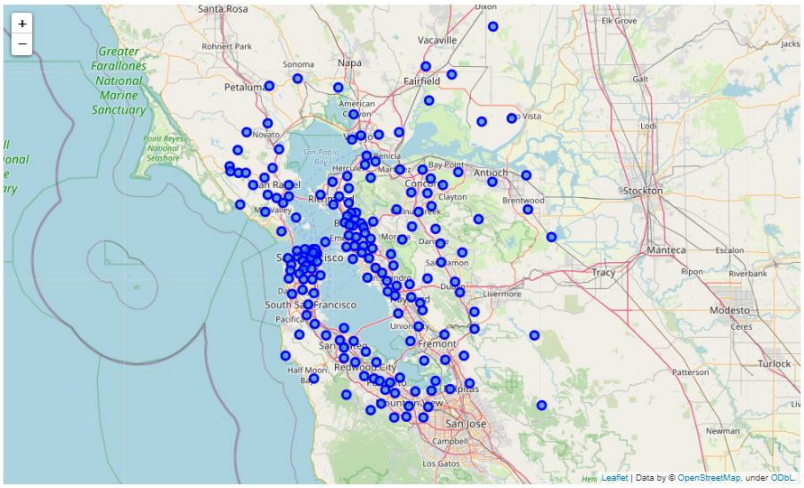
## 1. Introduction

This project is the final capstone project for the course Applied Data Science Capstone which is the last course of the IBM Data Science Professional Certificate program. In this project, I will use the data science skills and tools learned from the series of courses to use location data to explore a geographical location. The goal is to prove that I am capable of defining an idea or problem, looking for data in the web, using the Foursquare location data to explore different areas, effectively analyzing the data, and presenting the results.

### 1.1 Description of the problem and a discussion of the background

I live in a suburban city in the grand Houston area. My neighborhood is very diverse, including a large amount resident originally from China. However, when I am tired of cooking and want to dine in a Chinese restaurant, I have very limited choices in the nearby neighborhood. I have to drive 15-20 minutes (without traffic) away to get one. Based on the information from my neighbors and friends, there is actually a demand to the traditional Chinese food. Since I am equipped with skills and tools from this series courses, an idea to my mind is to explore the relation between the number of Chinese restaurants and the population distribution in an area. This idea can be used to explore the similar relation between other venues and the population distribution.

Bay Area zip codes destitution.



At the beginning I used Foursquare to search the venues information in my city but found out the search didn't work well since some restaurants I know well are not searched out. I choose to explore the San Francisco Bay Area because it is a metropolitan region with 7 million inhabitants and more than 100 cities/town in the 7000 square miles of land. The high-density population supports a sufficient number of venues to use in this project. In addition, I don't have a chance to visit the Bay Area yet and it is in my visit list in near future, so I'd like to get some impression of it in advance.

### 1.2 Target audience

Some data scientists might be interested in this project since it includes some real-life data and exploratory data analysis methods, and the data is able to tell a story out of it. Business personnel who want to invest a venue such as a Chinese restaurant, might be interested in this project too since its analysis results provide a comprehensive understanding on the relation between the number of a venue in an area and the population distribution. Some analysis is kind of easy to understand, such as that the number of Chinese restaurants is positively correlated to the Chinese population. Some others might be surprise.

## 2. Description of the data and how it will be used to solve the problem

Four sets of data are used in this project.

**2.1 Bay Area ZIP Codes from [https://catalog.data.gov/dataset/bay-area-zip-codes](https://catalog.data.gov/dataset/bay-area-zip-codes).** The data was downloaded as a csv file. It includes zip codes, city names, geometry locations, area information. This data is used to limit the zip codes within the Bay Area.

```
path='D:/bayarea_zipcodes.csv'
bay_geom=pd.read_csv(path)
bay_geom.head()
```

| | PO_NAME | the_geom | ZIP | STATE | Area__ | Length__ |
|---|---|---|---|---|---|---|
| 0 | NAPA | MULTIPOLYGON (((-122.10329200180091 38.5132829... | 94558 | CA | 1.231326e+10 | 995176.225313 |
| 1 | FAIRFIELD | MULTIPOLYGON (((-121.947475002335 38.301511000... | 94533 | CA | 9.917861e+08 | 200772.556587 |
| 2 | DIXON | MULTIPOLYGON (((-121.65335500334429 38.3133870... | 95620 | CA | 7.236950e+09 | 441860.201400 |
| 3 | SONOMA | MULTIPOLYGON (((-122.406843003057 38.155681999... | 95476 | CA | 3.001414e+09 | 311318.546326 |
| 4 | NAPA | MULTIPOLYGON (((-122.29368500225117 38.1552379... | 94559 | CA | 1.194302e+09 | 359104.646602 |

Here only the zip codes, city names, and area of the data are used in this project. With renaming of the columns, a data frame named bay_geom is obtained as:

```
bay_geom=bay_geom[['ZIP','PO_NAME','Area__']]
bay_geom=bay_geom.rename(columns={'ZIP':'ZipCode','PO_NAME':'City','Area__':'Area'})
bay_geom.head()
```

| | ZipCode | City | Area |
|---|---|---|---|
| 0 | 94558 | NAPA | 1.231326e+10 |
| 1 | 94533 | FAIRFIELD | 9.917861e+08 |
| 2 | 95620 | DIXON | 7.236950e+09 |
| 3 | 95476 | SONOMA | 3.001414e+09 |
| 4 | 94559 | NAPA | 1.194302e+09 |

### 2.2 Scraping the population distribution of California from [http://zipatlas.com/us/ca/zip-code-comparison/population-density.htm](http://zipatlas.com/us/ca/zip-code-comparison/population-density.htm).

This data includes zip codes, geographic location (latitude, longitude), population of all zip codes in California.

```
urls=['http://zipatlas.com/us/ca/zip-code-comparison/population-density.htm',
     'http://zipatlas.com/us/ca/zip-code-comparison/population-density.2.htm',
     'http://zipatlas.com/us/ca/zip-code-comparison/population-density.3.htm',
     'http://zipatlas.com/us/ca/zip-code-comparison/population-density.4.htm',
     'http://zipatlas.com/us/ca/zip-code-comparison/population-density.5.htm',
     'http://zipatlas.com/us/ca/zip-code-comparison/population-density.6.htm',
     'http://zipatlas.com/us/ca/zip-code-comparison/population-density.7.htm',
     'http://zipatlas.com/us/ca/zip-code-comparison/population-density.8.htm',
     'http://zipatlas.com/us/ca/zip-code-comparison/population-density.9.htm',
     'http://zipatlas.com/us/ca/zip-code-comparison/population-density.10.htm',
     'http://zipatlas.com/us/ca/zip-code-comparison/population-density.11.htm',
     'http://zipatlas.com/us/ca/zip-code-comparison/population-density.12.htm',
     'http://zipatlas.com/us/ca/zip-code-comparison/population-density.13.htm',
     'http://zipatlas.com/us/ca/zip-code-comparison/population-density.14.htm',
     'http://zipatlas.com/us/ca/zip-code-comparison/population-density.15.htm',
     'http://zipatlas.com/us/ca/zip-code-comparison/population-density.16.htm',
     'http://zipatlas.com/us/ca/zip-code-comparison/population-density.17.htm'
    ]

cadata=pd.DataFrame()
for url in urls:
    cadata=cadata.append(pd.read_html(url,header=0)[10].drop(['#'], axis=1), ignore_index=True)

cadata=cadata.rename(columns={'Zip Code':'ZipCode'})
cadata.head()
```

| | ZipCode | Location | City | Population | People / Sq. Mile | National Rank |
|---|---|---|---|---|---|---|
| 0 | 94108 | 37.791998, -122.408653 | San Francisco, California | 13716 | 53134.47 | #48 |
| 1 | 90057 | 34.061918, -118.277939 | Los Angeles, California | 43986 | 49226.28 | #54 |
| 2 | 94109 | 37.794487, -122.422270 | San Francisco, California | 56322 | 46521.46 | #64 |
| 3 | 94102 | 37.779500, -122.419233 | San Francisco, California | 28991 | 44719.24 | #71 |
| 4 | 94133 | 37.802071, -122.411004 | San Francisco, California | 26827 | 40117.97 | #77 |

This set of original data need to be cleaned and refined, and then merged with the data set 2.1 to get a dataframe named bay_data, which include 'ZipCode', 'City', 'Latitude', 'Longitude', 'Area', 'Population', 'People / Sq. Mile' data in the Bay Area.

```
bay_data=pd.merge(bay_geom.drop('City', axis=1), cadata, on='ZipCode')
bay_data['Latitude']=bay_data['Location'].apply(lambda x:float(x.split(', ')[0]))
bay_data['Longitude']=bay_data['Location'].apply(lambda x:float(x.split(', ')[1]))
bay_data=bay_data[['ZipCode','City','Latitude','Longitude','Area','Population','People / Sq. Mile']]
bay_data.head()
```

| | ZipCode | City | Latitude | Longitude | Area | Population | People / Sq. Mile |
|---|---|---|---|---|---|---|---|
| 0 | 94558 | Napa, California | 38.489789 | -122.270110 | 1.231326e+10 | 63932 | 155.68 |
| 1 | 94533 | Fairfield, California | 38.287136 | -122.027110 | 9.917861e+08 | 77666 | 2290.55 |
| 2 | 95620 | Dixon, California | 38.392821 | -121.799917 | 7.236950e+09 | 18510 | 91.81 |
| 3 | 95476 | Sonoma, California | 38.254850 | -122.461799 | 3.001414e+09 | 34310 | 309.44 |
| 4 | 94559 | Napa, California | 38.232389 | -122.324944 | 1.194302e+09 | 26891 | 800.91 |

### 2.3 Download California population by race, including detailed Asian race from State of California Department of Finance, https://www.dof.ca.gov/Reports/Demographic_Reports/Census_2010/#SF1

```
path2='D:/2010Census_DemoProfile3a_ZCTA.xls'
population=pd.read_excel(path2,sheet_name='2010_3a',header=8)
population.columns=['ZipCode','Total Population','Total Population of One Race','White','Black or Afric
population
```

| | ZipCode | Total Population | Total Population of One Race | White | Black or African American | American Indian and Alaska Native | Total Asian | Asian Indian | Chinese | Filipino | Japa |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | California | 37253956.0 | 35438572.0 | 21453934.0 | 2299072.0 | 362801.0 | 4861007.0 | 528176.0 | 1253102.0 | 1195580.0 | 272 |
| 1 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | |
| 2 | ZCTA5 89010 (California part only) | 31.0 | 31.0 | 18.0 | 0.0 | 8.0 | 0.0 | 0.0 | 0.0 | 0.0 | |
| 3 | ZCTA5 89019 (California part only) | 69.0 | 64.0 | 55.0 | 6.0 | 0.0 | 2.0 | 0.0 | 0.0 | 0.0 | |
| 4 | ZCTA5 89060 | | | | | | | | | | |

This set of data need to be cleaned, refined, and merged with data set 2.2 to get a dataframe named population, which include detailed population distribution in each zip code in the Bay Area, as well as the location data.

```
population=population.drop(['Total Population of One Race','Total Asian','Total NHOPI','Total Population of Two or More Races'],
population=population.drop([0,1,2,3,4,5,6,1770],axis=0)
population=population.reset_index(drop=True)
population['ZipCode']=population['ZipCode'].apply(lambda x:int(x[6:]))
population=pd.merge(bay_data.drop(['Population'], axis=1), population, on='ZipCode')
population.head()
```

| | ZipCode | City | Latitude | Longitude | Area | People / Sq. Mile | Total Population | White | Black or African American | American Indian and Alaska Native | Asian Indian | Chinese | Filipino | Japanese | Korean |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 94558 | Napa, California | 38.489789 | -122.270110 | 1.231326e+10 | 155.68 | 66830.0 | 51624.0 | 654.0 | 491.0 | 133.0 | 286.0 | 478.0 | 308.0 | 152.0 |
| 1 | 94533 | Fairfield, California | 38.287136 | -122.027110 | 9.917861e+08 | 2290.55 | 69277.0 | 29074.0 | 12261.0 | 660.0 | 802.0 | 540.0 | 5408.0 | 419.0 | 233.0 |
| 2 | 95620 | Dixon, California | 38.392821 | -121.799917 | 7.236950e+09 | 91.81 | 20553.0 | 14581.0 | 578.0 | 204.0 | 84.0 | 110.0 | 314.0 | 75.0 | 28.0 |
| 3 | 95476 | Sonoma, California | 38.254850 | -122.461799 | 3.001414e+09 | 309.44 | 35394.0 | 28449.0 | 178.0 | 253.0 | 56.0 | 138.0 | 178.0 | 105.0 | 47.0 |
| 4 | 94559 | Napa, California | 38.232389 | -122.324944 | 1.194302e+09 | 800.91 | 27184.0 | 20256.0 | 189.0 | 265.0 | 78.0 | 97.0 | 171.0 | 75.0 | 38.0 |

### 2.4 Foursquare location date.

Foursquare is a location technology platform offering business solutions and consumer products through a deep understanding of location. It gives the venues information in Bay Area including venue category, geographic location, etc.

Create a function to use Foursquare API to explore the neighborhoods.

```
def getNearbyVenues(postalcodes,names, latitudes, longitudes, radius=2000):

    venues_list=[]
    for postalcode, name, lat, lng in zip(postalcodes,names, latitudes, longitudes):
        print(name)

        # create the API request URL
        url = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secret={}&v={}&ll={},{}&radius={}&limit={}'.forr
            CLIENT_ID,
            CLIENT_SECRET,
            VERSION,
            lat,
            lng,
            radius,
            LIMIT)

        # make the GET request
        results = requests.get(url).json()["response"]['groups'][0]['items']

        # return only relevant information for each nearby venue
        venues_list.append([(
            postalcode,
            name,
            lat,
            lng,
            v['venue']['name'],
            v['venue']['location']['lat'],
            v['venue']['location']['lng'],
            v['venue']['categories'][0]['name']) for v in results])

    nearby_venues = pd.DataFrame([item for venue_list in venues_list for item in venue_list])
    nearby_venues.columns = ['ZipCode',
                  'City',
                  'City Latitude',
                  'City Longitude',
                  'Venue',
                  'Venue Latitude',
                  'Venue Longitude',
                  'Venue Category']

    return(nearby_venues)
```

Then write the code to run the above function on each postal code/borough and create a new dataframe called bay_venues. There are total 10072 venues included.

```
zipcodes=bay_data['ZipCode']
names=bay_data['City']
latitudes=bay_data['Latitude']
longitudes=bay_data['Longitude']

bay_venues=getNearbyVenues(zipcodes, names, latitudes, longitudes, 2000)
#bay_venues.head()
```

```
San Jose, California
Alviso, California
Redwood City, California
Sunnyvale, California
Palo Alto, California
Menlo Park, California
Palo Alto, California
Milpitas, California
Mount Hamilton, California
Redwood City, California
Mountain View, California
Palo Alto, California
Stanford, California
Palo Alto, California
Portola Valley, California
Mountain View, California
Los Altos, California
Sunnyvale, California
Los Altos, California
Sunnyvale, California
```

```
bay_venues.head()
```

| | ZipCode | City | City Latitude | City Longitude | Venue | Venue Latitude | Venue Longitude | Venue Category |
|---|---|---|---|---|---|---|---|---|
| 0 | 94558 | Napa, California | 38.489789 | -122.27011 | Turtle Rock | 38.494408 | -122.251862 | Bar |
| 1 | 94558 | Napa, California | 38.489789 | -122.27011 | Brown Estate | 38.505196 | -122.276495 | Winery |
| 2 | 94558 | Napa, California | 38.489789 | -122.27011 | Turtle Rock Bar & Cafe | 38.496575 | -122.250497 | Bar |
| 3 | 94533 | Fairfield, California | 38.287136 | -122.02711 | In-Shape Health Clubs | 38.284730 | -122.025750 | Gym / Fitness Center |
| 4 | 94533 | Fairfield, California | 38.287136 | -122.02711 | Raley's | 38.289158 | -122.033253 | Grocery Store |

```
len(bay_venues['ZipCode'])
```

```
10072
```