Capstone Project Restaurant Location Research Report

Mark Mu

(contact information hidden for privacy reasons)

2019.3.25

Table of Contents

Capstone Project Restaurant Location Research Report....1
Table of Contents....2
Abstract....3
Introduction....4
Methodology....5
Results....11
Discussion....16
Conclusion....17
References....18

Abstract

The purpose of this project is to make recommendations to a Chinese restaurant chain owner regarding the location of their next branch in San Francisco, CA.

This project utilizes the Foursquare API, as well as the sklearn package in Python, in order to provide more insight into the geographical distributions of existing Chinese restaurants, so that we can provide ideas as to where to open the next Chinese restaurant.

In the end, I find that Chinese restaurants in San Francisco tend to appear in clusters, meaning that they tend to open next to each other, thus creating more competition for new comers. In addition, I have also found that Chinese restaurants prefer to open in less expensive residential areas and near China town. Such observations does provide some insights into the decision-making process of my customer.

Introduction

Restaurants are always a welcomed business for a big city. For the city, they brings consistent tax income, make the neighborhood more attractive and of course, satisfy people's needs for good food. A restaurant, if run properly, can even raise the reputation of the neighborhood it's in. For a business owner, a restaurant might also be a good idea since its business model is relative simple and it can provide reliable cash flows. Before opening a restaurant, however, one important decision to make is the its location. Should one open their restaurant in popular yet competitive districts? Or should one open where there is less competition but also less potential customers? Or should one target only the wealthy neighborhood?

In this project, I aim to answer these questions using data collected from the Foursquare API, as well as some magic data science methods. Hopefully the following findings can provide some insight that can help the decision making process of the owner.

In this project, I will be focusing on San Francisco, CA and help a owner of a quality Chinese food restaurant chain determine where to open their next restaurant in San Francisco city.

Methodology

In this project, I use the Foursquare data to find the geographical location information of competing restaurants.

In order to retrieve data from the Foursquare database, the Foursquare API is utilized. The url of the Foursquare API is typically as follows.

```
url = 'https://api.foursquare.com/v2/venues/explore'
```

To send a request through the Foursquare API, I use the get() function in the Python *requests* package. This function has two parameters: *url* and *params*. The *url* parameter contains the url of the targeted API, while the *params* parameter contains all required fields by the API. In my case, the *url* is the Foursquare API address, while the *params* includes the following fields

```
params = dict(
  client_id=CLIENT_ID,
  client_secret=CLIENT_SECRET,
  v='20180323',
  ll='37.79413,-122.404838',
  query=search_query,
  radius=500,
  limit=60
)
```

Where

- *client_id* and *client_secret* are the credentials required by the Foursquare API
- *v* is the version of the API that I desire
- *ll* is the latitude and longitude of the center of the area that I wish to make a search query
- *query* is the keyword that I wish to search for. In this case, my search query is “Chinese”, since I’m researching on Chinese food restaurant.
- *radius* is the size of the area that I want to search
- *limit* controls the number of results to be returned by the API

The results returned by the get() function would be json file and needs to be decoded before I can properly read it and pass it into a Pandas dataframe. In order to do this, the *loads()* function from the *json* package is utilized. This step works as follows.

```
resp = requests.get(url=url, params=params)
results = json.loads(resp.text)
```

The results, being a string variable, looks like the following

In [43]:
```
results
```

Out[43]:
```
{'meta': {'code': 200, 'requestId': '5c9549f81ed219272970332f'},
 'response': {'groups': [{'items': [{'reasons': {'count': 0,
       'items': [{'reasonName': 'globalInteractionReason',
         'summary': 'This spot is popular',
         'type': 'general'}]},
      'referralId': 'e-0-56f2092e498eac8f5f0799c8-0',
      'venue': {'beenHere': {'count': 0,
        'lastCheckinExpiredAt': 0,
        'marked': False,
        'unconfirmedCount': 0},
       'categories': [{'icon': {'prefix': 'https://ss3.4sqi.net/img/categories_v2/food/asian_',
          'suffix': '.png'},
         'id': '4bf58dd8d48988d145941735',
         'name': 'Chinese Restaurant',
         'pluralName': 'Chinese Restaurants',
         'primary': True,
         'shortName': 'Chinese'}],
       'contact': {},
       'hereNow': {'count': 0, 'groups': [], 'summary': 'Nobody here'},
```

But of course, most of the information contained here are not useful to my research. Therefore, it is necessary to clean up the data first. For this project, I only care about the information of each restaurant that comes up in the search result. I collect such information from the string variable and pass it through the *json_normalize()* function to transform it into a dataframe. The final result of this step looks like the following.

In [10]:
```
# assign relevant part of JSON to venues
venues = results['response']['groups'][0]['items']

# tranform venues into a dataframe
dataframe = json_normalize(venues)
dataframe
```

Out[10]:

	reasons.count	reasons.items	referralId	venue.beenHere.count
0	0	[{'type': 'general', 'reasonName': 'globalInte...	e-0-56f2092e498eac8f5f0799c8-0	0
1	0	[{'type': 'general', 'reasonName': 'globalInte...	e-0-4a47ee99f964a52053aa1fe3-1	0
2	0	[{'type': 'general', 'reasonName': 'globalInte...	e-0-49e4d4fdf964a52058631fe3-2	0
3	0	[{'type': 'general', 'reasonName': 'globalInte...	e-0-4abfdac7f964a5209f9220e3-3	0

Where each row represents a certain restaurant and each column is an attribute of that restaurant. The dataframe has a total of 40 columns as follows

```
Data columns (total 40 columns):
reasons.count                         28 non-null int64
reasons.items                         28 non-null object
referralId                            28 non-null object
venue.beenHere.count                  28 non-null int64
venue.beenHere.lastCheckinExpiredAt   28 non-null int64
venue.beenHere.marked                 28 non-null bool
venue.beenHere.unconfirmedCount       28 non-null int64
venue.categories                      28 non-null object
venue.delivery.id                     12 non-null object
venue.delivery.provider.icon.name     12 non-null object
venue.delivery.provider.icon.prefix   12 non-null object
venue.delivery.provider.icon.sizes    12 non-null object
venue.delivery.provider.name          12 non-null object
venue.delivery.url                    12 non-null object
venue.hereNow.count                   28 non-null int64
venue.hereNow.groups                  28 non-null object
venue.hereNow.summary                 28 non-null object
venue.id                              28 non-null object
venue.location.address                28 non-null object
venue.location.cc                     28 non-null object
venue.location.city                   28 non-null object
venue.location.country                28 non-null object
venue.location.crossStreet            18 non-null object
venue.location.distance               28 non-null int64
venue.location.formattedAddress       28 non-null object
venue.location.labeledLatLngs         28 non-null object
venue.location.lat                    28 non-null float64
venue.location.lng                    28 non-null float64
venue.location.neighborhood           3 non-null object
venue.location.postalCode             28 non-null object
venue.location.state                  28 non-null object
venue.name                            28 non-null object
venue.photos.count                    28 non-null int64
venue.photos.groups                   28 non-null object
venue.stats.checkinsCount             28 non-null int64
venue.stats.tipCount                  28 non-null int64
venue.stats.usersCount                28 non-null int64
venue.stats.visitsCount               28 non-null int64
venue.venuePage.id                    1 non-null object
venue.verified                        28 non-null bool
```

Among this 40 attributes, only location attributes and restaurant-specific attributes are of interest to me. So I ran a check on such attributes to see what information they convey. Here is the result.

venue.photos.count	venue.stats.checkinsCount	venue.stats.tipCount	venue.stats.usersCount	venue.stats.visitsCount
28.0	28.0	28.0	28.0	28.0
0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0

As seen here, most of the restaurant-specific attributes, including tipCount, usersCount, visitsCount all have the value of 0 for all restaurants. Therefore, such

attributes does not convey any meaningful information and should be excluded from the research afterwards. The reason that caused this might have to do with Foursquare, since such information is missing from their database and cannot be retrieved one way or another. Now I subset the dataframe to only include restaurant names and location data. I also rename the columns of the dataframe so that it is more easily readable.

In [21]:
```
nearby_res=dataframe[['venue.name','venue.location.lat','venue.location.lng']]
nearby_res.columns = ['Restaurant', 'Latitude','Longitude']
nearby_res.head()
```

Out[21]:

	Restaurant	Latitude	Longitude
0	Ton Kiang	37.780181	-122.481924
1	Hong Kong Lounge 穗香酒家	37.780558	-122.476644
2	Jiangnan Cuisine	37.775890	-122.495616
3	Hot Pot Buffet	37.780617	-122.478636
4	Taste of Formosa	37.782037	-122.485469

The data cleaning step is thus finished here. Next I can move on to the clustering analysis.

For the clustering analysis, I use the K-means method, since the data consists mostly of geographical location information and the dimension is low. The use of K-means method makes sense here.

```
# set number of clusters
kclusters = 5

nearby_res_clustering = nearby_res.drop('Restaurant', 1)

# run k-means clustering
kmeans = KMeans(n_clusters=kclusters, random_state=0).fit(nearby_res_clustering)

# check cluster labels generated for each row in the dataframe
kmeans.labels_[0:10]

nearby_res['Cluster Label']=kmeans.labels_
nearby_res.head()
```

The number of clusters may change depending on which area of the city I'm researching. The clustering results looks like the following.

	Restaurant	Latitude	Longitude	Cluster Label
0	Mister Jiu's	37.793790	-122.406615	0
1	Hunan Home's Restaurant	37.796193	-122.405581	1
2	Hong Kong Clay Pot Restaurant	37.795844	-122.406594	1
3	City View Restaurant	37.794191	-122.403918	3
4	R&G Lounge 嶺南小館	37.794072	-122.404724	3

Where if two restaurants have the same cluster label, then they belong to the same cluster.

Having got the clustering results, I then utilize the *folium* package in order to further present the data and find meaningful solution to the questions raised in the introduction section. The related codes looks something like the following.

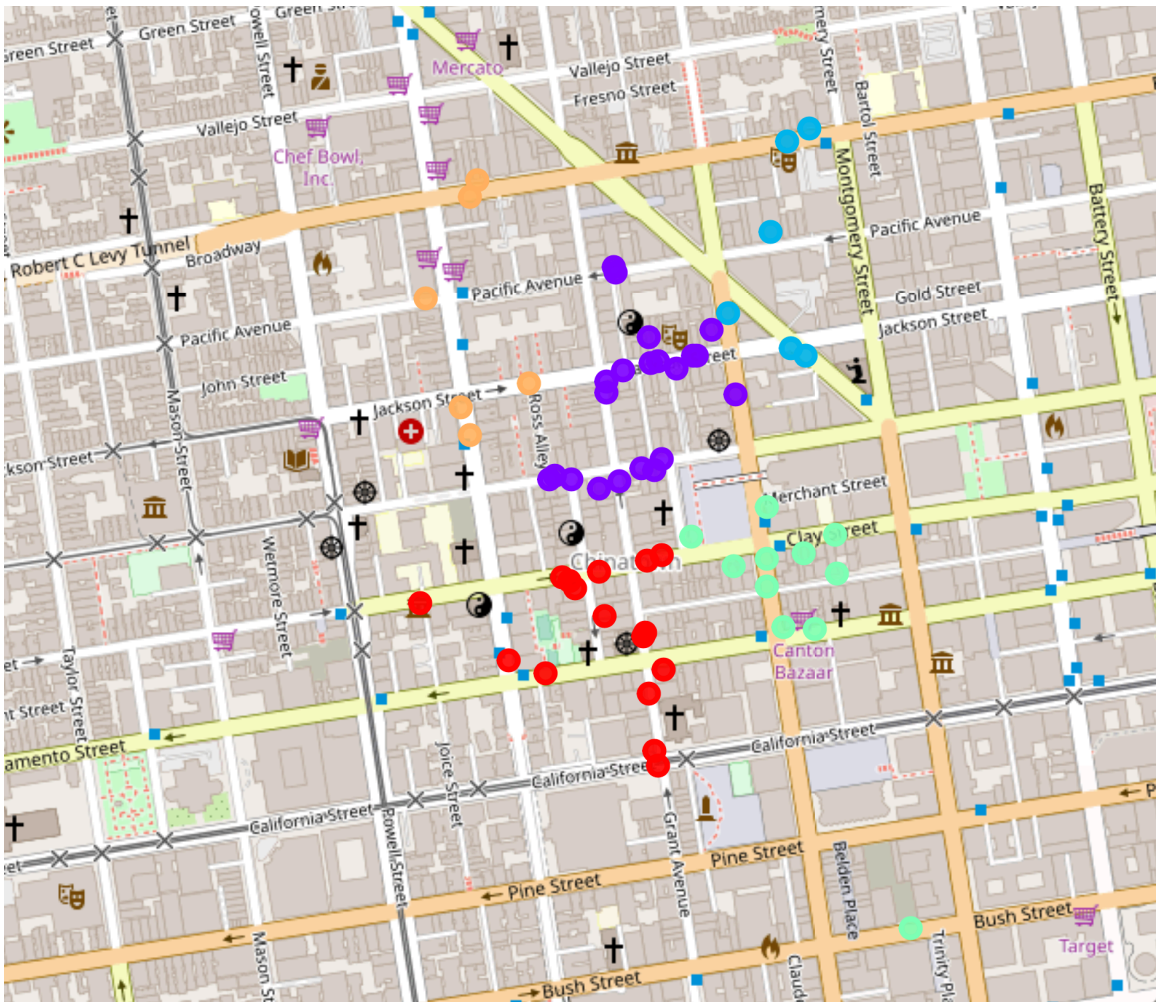
```
# create map
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=16)

# set color scheme for the clusters
x = np.arange(kclusters)
ys = [i + x + (i*x)**2 for i in range(kclusters)]
colors_array = cm.rainbow(np.linspace(0, 1, len(ys)))
rainbow = [colors.rgb2hex(i) for i in colors_array]

# add markers to the map
markers_colors = []
for lat, lon, poi, cluster in zip(nearby_res['Latitude'], nearby_res['Longitude'], nearby_res['Restaurant'], nearby_res['Cluster Label']):
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_html=True)
    folium.CircleMarker(
        [lat, lon],
        radius=5,
        popup=label,
        color=rainbow[cluster-1],
        fill=True,
        fill_color=rainbow[cluster-1],
        fill_opacity=0.9).add_to(map_clusters)

map_clusters
```

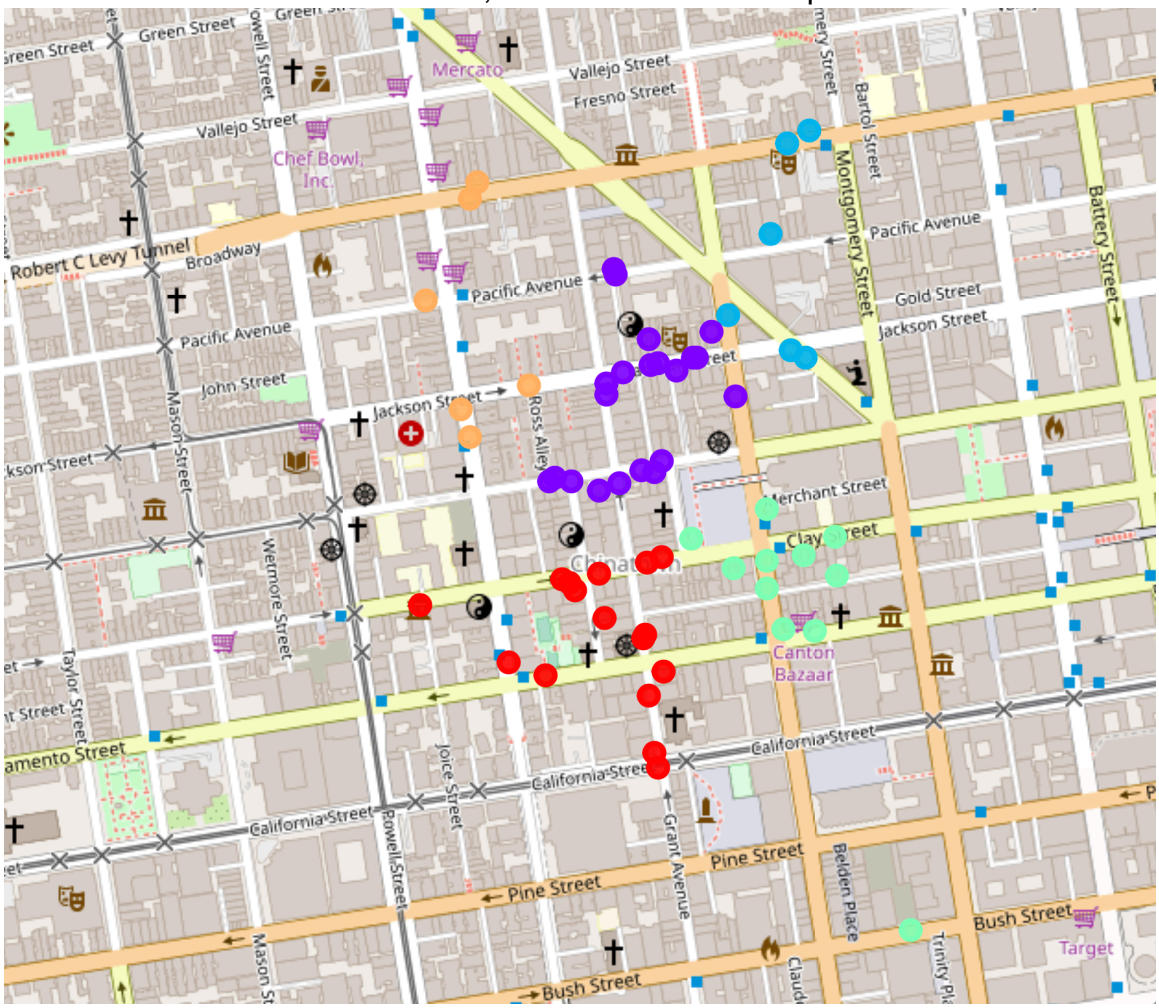
The generated map, with restaurant clusters on top of it, look like follows.


Green Street
Vallejo Street
Mercato
Fresno Street
Chef Bowl, Inc.
Robert C Levy Tunnel
Broadway
Pacific Avenue
Montgomery Street
Battery Street
Gold Street
Jackson Street
John Street
Mason Street
Ross Alley
Merchant Street
Clay Street
Wermore Street
Canton Bazaar
California Street
Joice Street
Powell Street
Grant Avenue
Pine Street
Belden Place
Bush Street
Target

Results

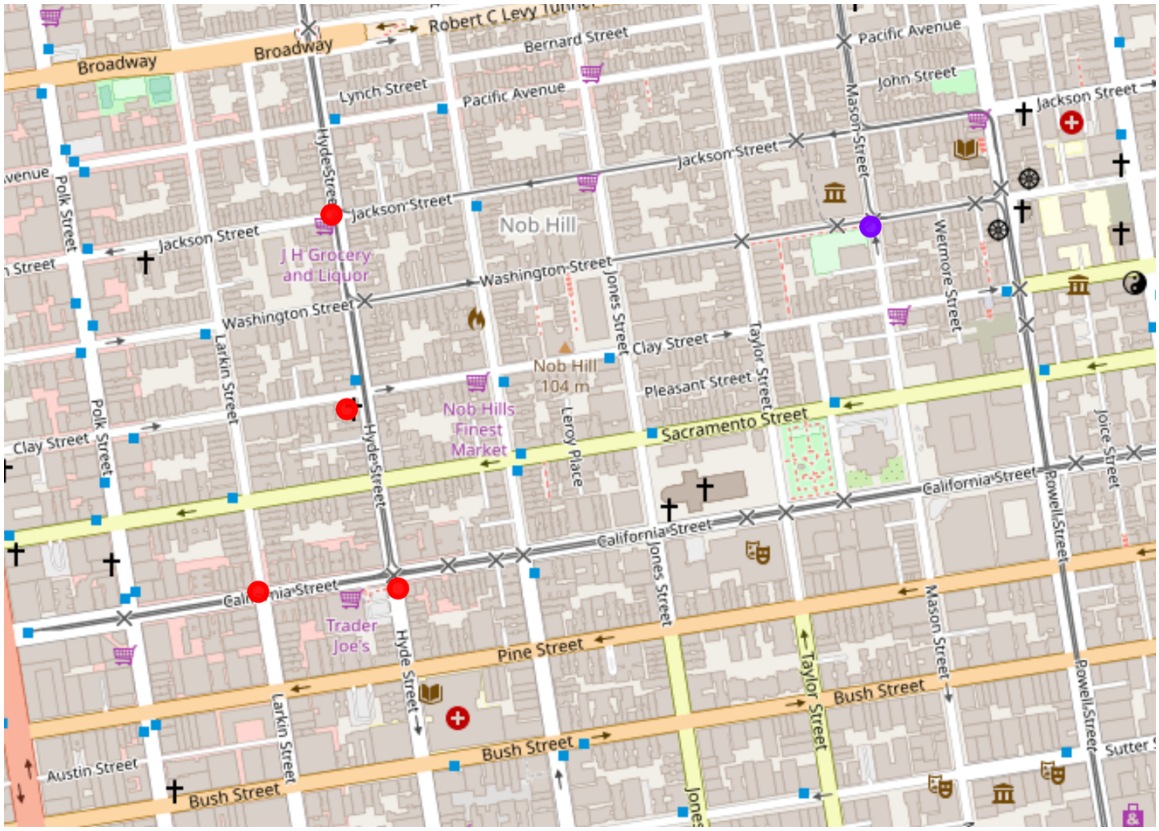
After repeating the process discussed in the Methodology section on 4 major different areas in San Francisco city, I come up with 4 different cluster maps for Chinese restaurants.

For the area near China Town, here is the cluster map.



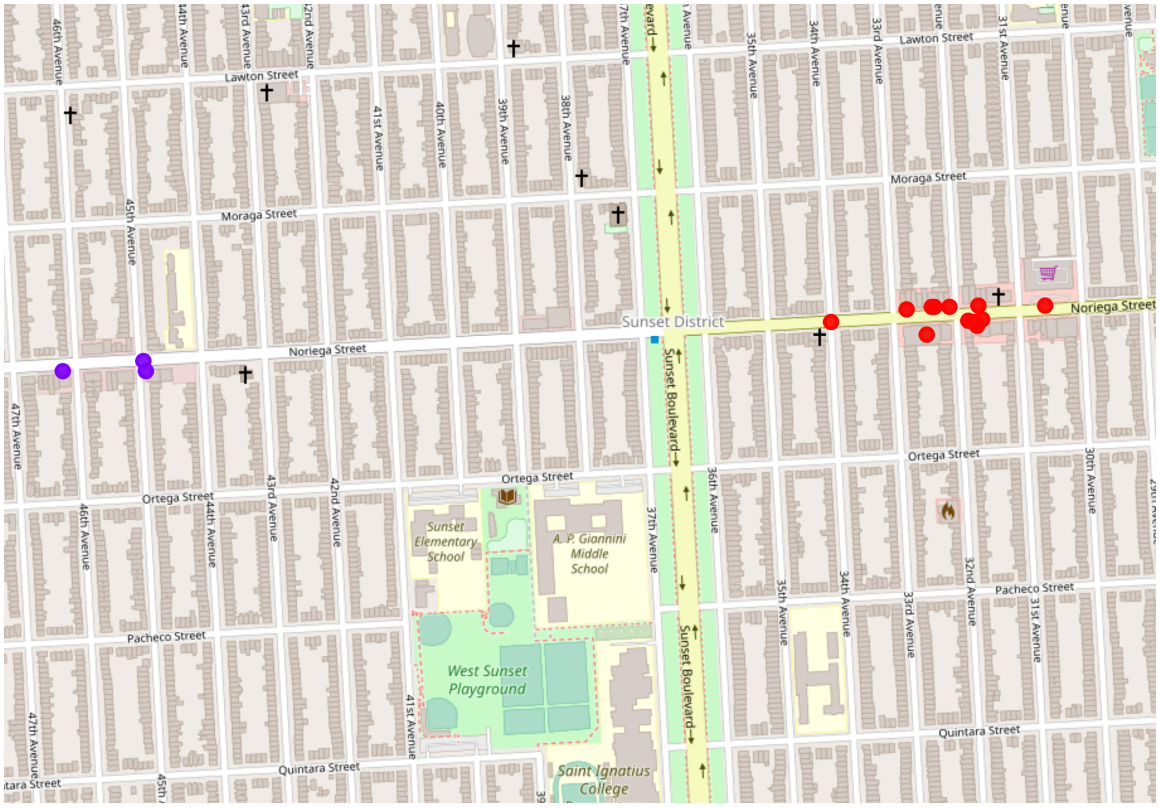
It is quite clear that, the Chinese restaurants forms 5 clusters, occupying the South, South-East, North-East, North-West and Middle part of China Town-Financial District area. This makes sense, since the North, West and East side of this area are occupied by Churches, stores and office buildings. If one wants to open a Chinese restaurant here, it is sensible to choose among those 5 clusters, since they already attracts huge amount of customers each day. However, do so may put one in fierce competition, as there are too many choices for customers.

For the Nob Hill area, here is the cluster map



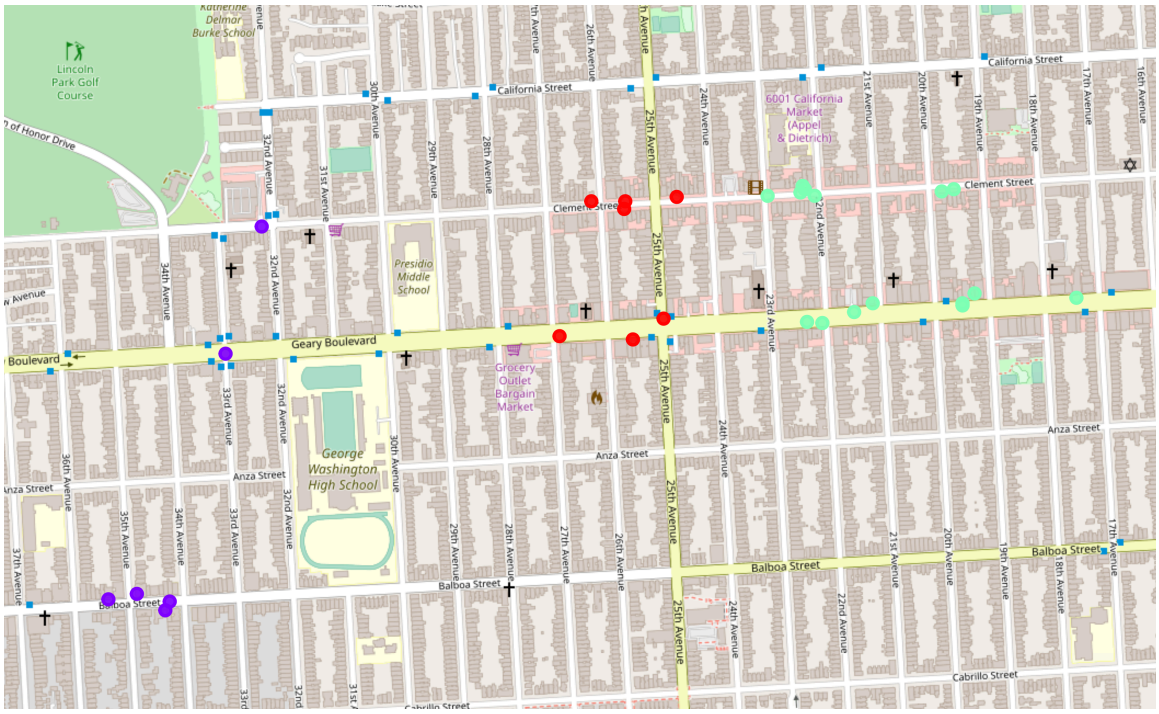
Obviously, there are not a lot of Chinese restaurants in this area. There are a few, but they hardly form more than 2 clusters. The reason for this is because Nob Hill is one of the most expensive and high-end residential areas in San Francisco city. Less restaurants can afford the land expenses in such district. However, a high-end restaurant chain, like the one my customer has, may find this area attractive, since there are far fewer competitions and more wealthy potential customers.

For the Sunset area, here is the cluster map



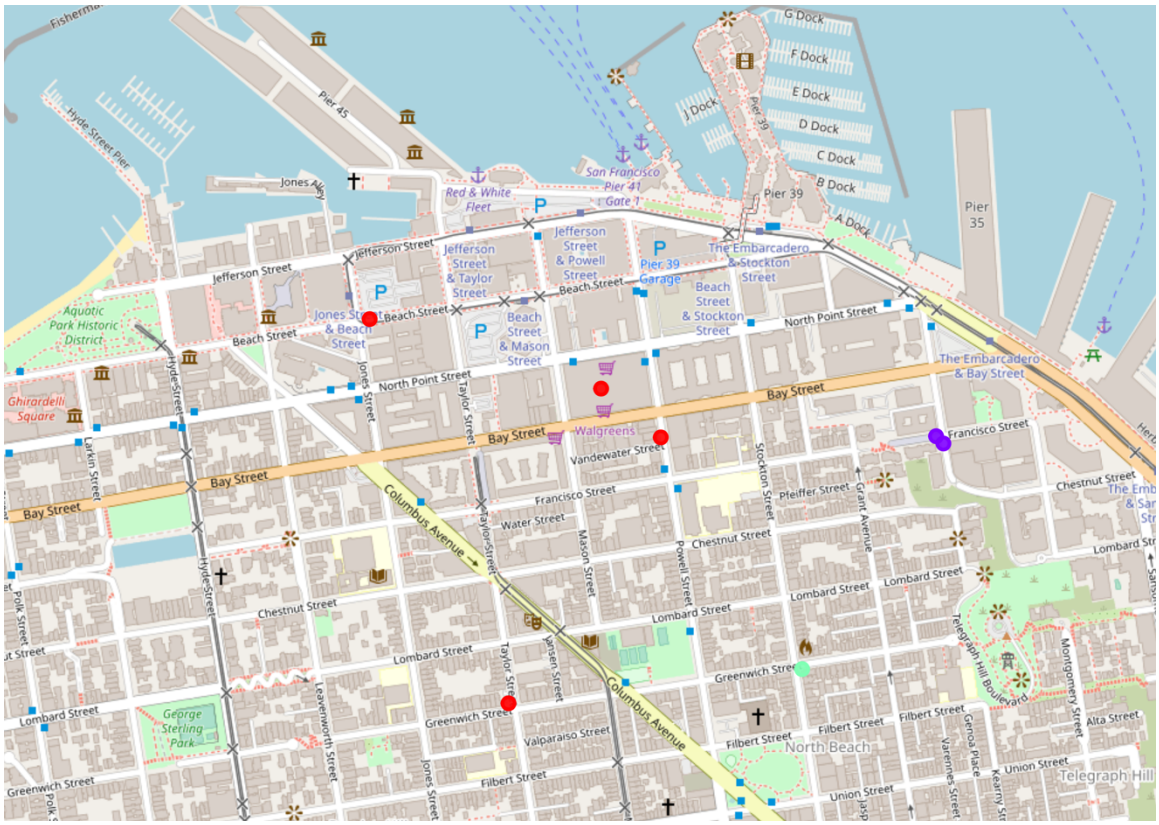
The Sunset area is another district where many middle-class population lives. Here we can find much more Chinese restaurants than in Nob Hill, since the land expenses are much lower. The Chinese restaurants here also appear in clusters and they are all on the same street. The reason is probably because Chinese restaurants tend to open where big market places are, and the two big market place in Sunset district happen to be built on the same street, hence the distribution of Chinese restaurants. This area may appeal to many new restaurants owners, including my customer, because it is less competitive comparing to China Town, while the number of potential customers is also affluent.

Let's look at the Richmond district.



Richmond is another residential area in San Francisco city where middle class population live. Here the distribution of Chinese restaurants is more diverse and there is no obvious pattern for the distribution. One may argue that most of them appear on the three most popular street in this area and that could be true. An owner may be interested in this area because there are already some Chinese restaurants settling here but no so many that it becomes overcrowded. In addition, the land expenses in the Richmond area is certainly less expensive than in areas like Nob Hill.

Last but not least, let's look at the Fisherman Wharf area.



Fisherman's Wharf is one of the top tourist attraction in San Francisco city. Chinese restaurants is so sparse in this area that calling this a cluster is not appropriate. This indicates that tourists who come here look for local treats are not very interested in Chinese food. The land expenses may also be a valid reason.

Discussion

All in all, after all these researches, I have found that the location distribution of Chinese restaurants does have certain patterns. Here are some distinct ones.

- Chinese restaurants prefer to settle near where China Town is (Obviously!).
- Chinese restaurants prefer appear where there are more office buildings, in another word, downtown areas.
- Chinese restaurants tend to appear in clusters near big market places. This is probably to reduce marketing costs and enjoy big visitor flows in popular places.
- Chinese restaurants tend to avoid high-end residential areas, probably due to land expenses.
- Chinese restaurants tend to avoid local tourist attraction spots, since tourists are not that into Chinese food when they are look for local delicacies.

Based upon such findings, it is safe to suggest that when look for a spot to open a brand new Chinese restaurant, it is a good idea to first categorize one's goal. As in my case, since my customer is an owner of a high-end Chinese restaurant chain, I would suggest either Sunset, or Nob Hill. Opening a brand-new Chinese restaurant near China Town or Richmond area would be a bad idea because these areas are already crowned and there are few potential wealthy customers who are interested in high-end Chinese food. Fisherman's Wharf would also be a bad idea since the area is crowed with tourists who are not interested in Chinese food anyways. Hence, Nob Hill and Sunset seems to be better choices.

Conclusion

Choosing a location for your new restaurant might be a laborious task. However, with the magic of data science and a set of useful tools, such tasks might become fun and simpler. The methods discussed before doesn't only apply to Chinese restaurants. One can try different kind of restaurants and get an idea of different location distributions for different cuisines. One important thing to have in mind before starting this analysis, is to first decide what potential customers one wants to target—wealthy residentials, or office workers, or tourists? Having some idea to these questions would make it far easier to conduct the research and get the desired answer. Finally, bon appétit!

References

1. *Data Science Orientation Use Cases for Data Science Reading*
2. *Getting Started with Data Science' Publisher: IBM Press; 1 edition (Dec 13 2015) Print. Author: Murtaza Haider*