DATA FOR RESTAURANT CUISINE/LOCATION SELECTOR

The sources data for the project are:

- FourSquareAPI for data on restaurants
- Geopy for translating address to coordinates

First, I used the Geopy to fetch the coordinates of the city (San Jose, CA)

```
#city coordinates
address = "San Jose,CA"

locator = Nominatim(user_agent = "foursquare_api")
coordinates = locator.geocode(address)
latitude = coordinates.latitude
longitude = coordinates.longitude
```

```
latitude , longitude
```

```
(37.3361905, -121.8905833)
```

Then using the category code for each cuisine, I fetched the data for restaurants of each cuisine type.

```
#first let us make a list of the top cuisines in san jose
cuisines = ['American','Italian','French','Indian',
            'Chinese','Japanese','Thai','Mexican',
           'Spanish']
cat_id = ['4bf58dd8d48988d14e941735','4bf58dd8d48988d110941735','4bf58dd8d48988d10c941735','4bf58dd8d48988d10f941735',
          '4bf58dd8d48988d145941735','4bf58dd8d48988d111941735','4bf58dd8d48988d149941735','4bf58dd8d48988d1c1941735',
         '4bf58dd8d48988d150941735']
```

```
df = pd.DataFrame({})
for index,code in enumerate(cat_id):
    url = "https://api.foursquare.com/v2/venues/search?client_id={}&client_secret={}&ll={},{}&v={}&categoryId={}&radius=
    results = requests.get(url).json()
    venues = results['response']['venues']
    data = json_normalize(venues)
    data["cuisine"] = cuisines[index]
    df = pd.concat([df,data],axis=0)
    print(index,cuisines[index])
```

```
0 American
1 Italian
2 French
3 Indian
4 Chinese
5 Japanese
6 Thai
7 Mexican
8 Spanish
```

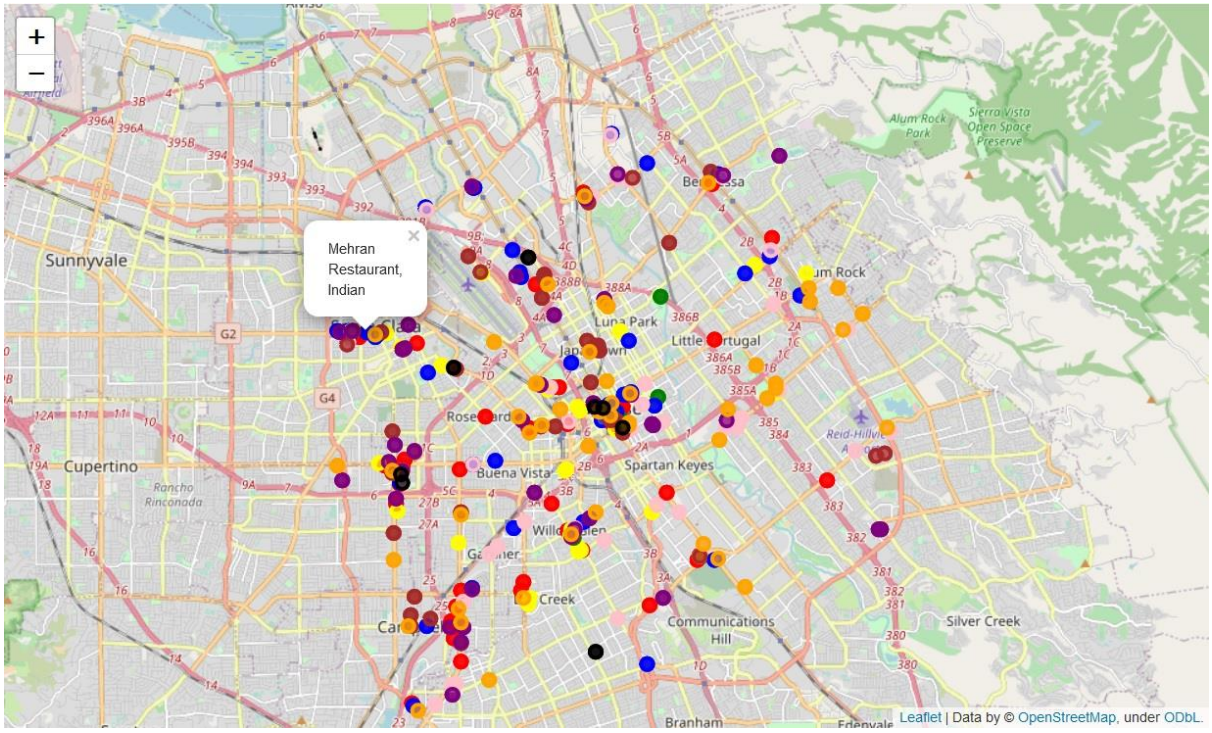
Then I convert the json file to a DataFrame and filter out the required data from the data fetched.

```
data = df[["name","location.lat","location.lng","cuisine"]]
data = data.reset_index(drop = True)
data
```

	name	location.lat	location.lng	cuisine
0	The Farmers Union	37.335140	-121.893030	American
1	Liquid Restaurant & Lounge	37.336702	-121.887864	American
2	San Jose Beer Union	37.369526	-121.929421	American
3	Water Tower Kitchen	37.286541	-121.944007	American
4	McCormick & Schmick's Seafood & Steak	37.332280	-121.888766	American
...	...	...	...	...
326	Oveja Negra	37.320950	-121.948080	Spanish
327	cherry garden lane	37.281319	-121.893692	Spanish
328	corner of first street and union	37.369453	-121.912521	Spanish
329	District	37.336166	-121.893901	Spanish
330	Chalateco - The Alameda	37.344835	-121.933628	Spanish

331 rows × 4 columns

I then plot a map of the restaurants categorised based on the cuisines

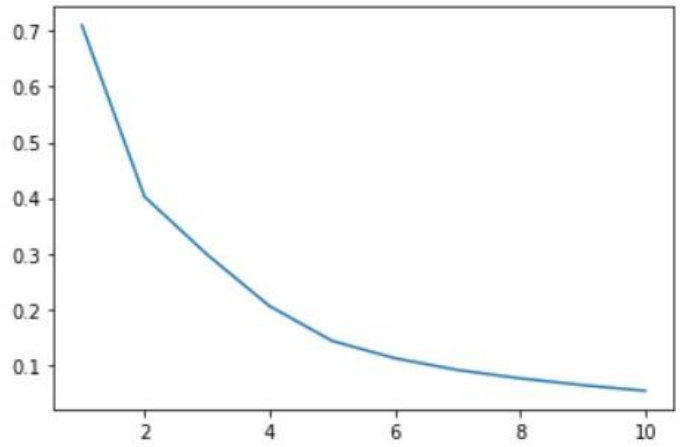


Choose the right number of clusters using the elbow method

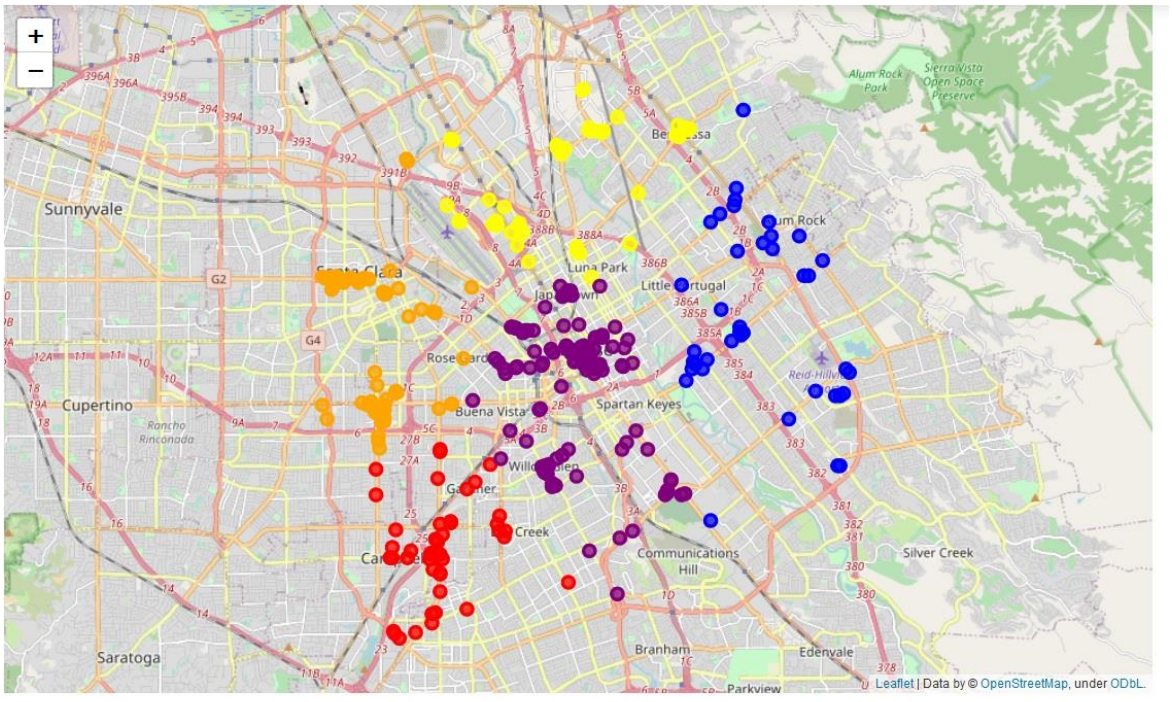
```
k = [1,2,3,4,5,6,7,8,9,10]
inertia = []
for i in k:
    kmeans = KMeans(n_clusters = i)
    kmeans.fit(df[["location.lat","location.lng"]])
    inertia.append(kmeans.inertia_)
```

```
plt.plot(k,inertia)
```

```
[<matplotlib.lines.Line2D at 0x1cfb7580160>]
```



Choosing 5 clusters, plot a map of all the clusters




```
# red - cluster 0
# blue - cluster 1
# orange - cluster 2
# yellow - cluster 3
# purple - cluster 4
```

Information about cuisines per cluster

```
Cluster  0
American    10
Mexican      8
Japanese     7
Chinese      6
Thai         5
Indian       4
Italian      3
French       1
Spanish      1
Name: cuisine, dtype: int64
_____________________________________
Cluster  1
Mexican     13
Chinese     11
Thai         4
Italian      3
American     3
Indian       3
Japanese     2
Name: cuisine, dtype: int64
_____________________________________
Cluster  2
Indian      14
American    11
Thai        11
Japanese     8
Italian      7
Chinese      6
Mexican      4
Spanish      3
French       1
Name: cuisine, dtype: int64
_____________________________________
Cluster  3
Chinese     13
Japanese    11
Indian       9
Thai         7
Mexican      5
Italian      3
American     3
French       1
Spanish      1
Name: cuisine, dtype: int64
_____________________________________
Cluster  4
American    23
Japanese    21
Mexican     20
Italian     18
Indian      16
Chinese     13
Thai        12
French       3
Spanish      3
Name: cuisine, dtype: int64
```

We can select the least common cuisine and the cluster with lowest ratio of cuisine(x) restaurants : total restaurants.