Explorando os bairros de São Francisco

k-Means Clustering: Um algoritmo de aprendizado de máquina não supervisionado

Eliel Leandro Alves Junior

Outline

1. Introdução
2. O problema
3. Os dados
4. Metodologia
5. Análise de dados
6. Resultado
7. Conclusão

Introdução

A cidade de São Francisco que se encontra no estado da California não é uma grande cidade americana. A vida é em um ritmo desacelerado e tranquila sob o fog constante. Ela é conhecida mundialmente pela Ponte Golden Gate e pela prisão de Alcatraz. São Francisco é situada em uma península de 120 $\mathrm{K}m^2$ e cercada por museus, restaurantes e centros comerciais famosos em lugares históricos

O Problema

Sem dúvida, a Diversidade Alimentar é uma parte importante de uma metrópole etnicamente diversa. A ideia deste projeto é segmentar categoricamente os bairros da cidade de São Francisco em cluster e examinar os *“Dining and Drinking”*. O principal objetivo é examinar os hábitos e gostos alimentares dos bairros. Este projeto, tem a intenção de entender a diversidade gastronômica de cada bairro, para isso coletamos as informações de cada bairro usado a API do Foursquare e utilizar o algoritmo de aprendizado de máquina não supervisionado o K-means clustering. A análise dos dados ajudará a descobrir mais a diversidade gastronômica de cada bairro.

Os Dados

- Utilizamos as seguintes fontes de dados :
    1. Realizamos **Web Scraping** da seguinte página: http://www.healthysf.org/bdi/outcomes/zipmap.htm para encontrar os Cep de cada bairro e depois transforma-los em um *'dataframe'.*
    2. **Foursquare API:** um provedor de dados de localização, usada para recuperar dados sobre locais em diferentes bairros.

Metodologia

Download do conjuntos de dados da cidade de São Francisco

- Por primeiro, baixamos o conjunto de dados utilizando web scraping da URL mencionada na página 4 e transformamos em ‘dataframe’. Como resultado, um é criado um ‘dataframe’ com detalhes do Cep, Bairro, Latitude e Longitude de cada bairro da cidade de São Francisco.

	Zip Code	Neighborhood	Latitude	Longitude
15	94123	Marina	37.80	-122.43
16	94124	Bayview-Hunters Point	37.73	-122.38
17	94127	St. Francis Wood/Miraloma/West Portal	37.74	-122.46
18	94131	Twin Peaks-Glen Park	37.74	-122.44
19	94132	Lake Merced	37.72	-122.48
20	94133	North Beach/Chinatown	37.80	-122.41
21	94134	Visitacion Valley/Sunnydale	37.72	-122.41

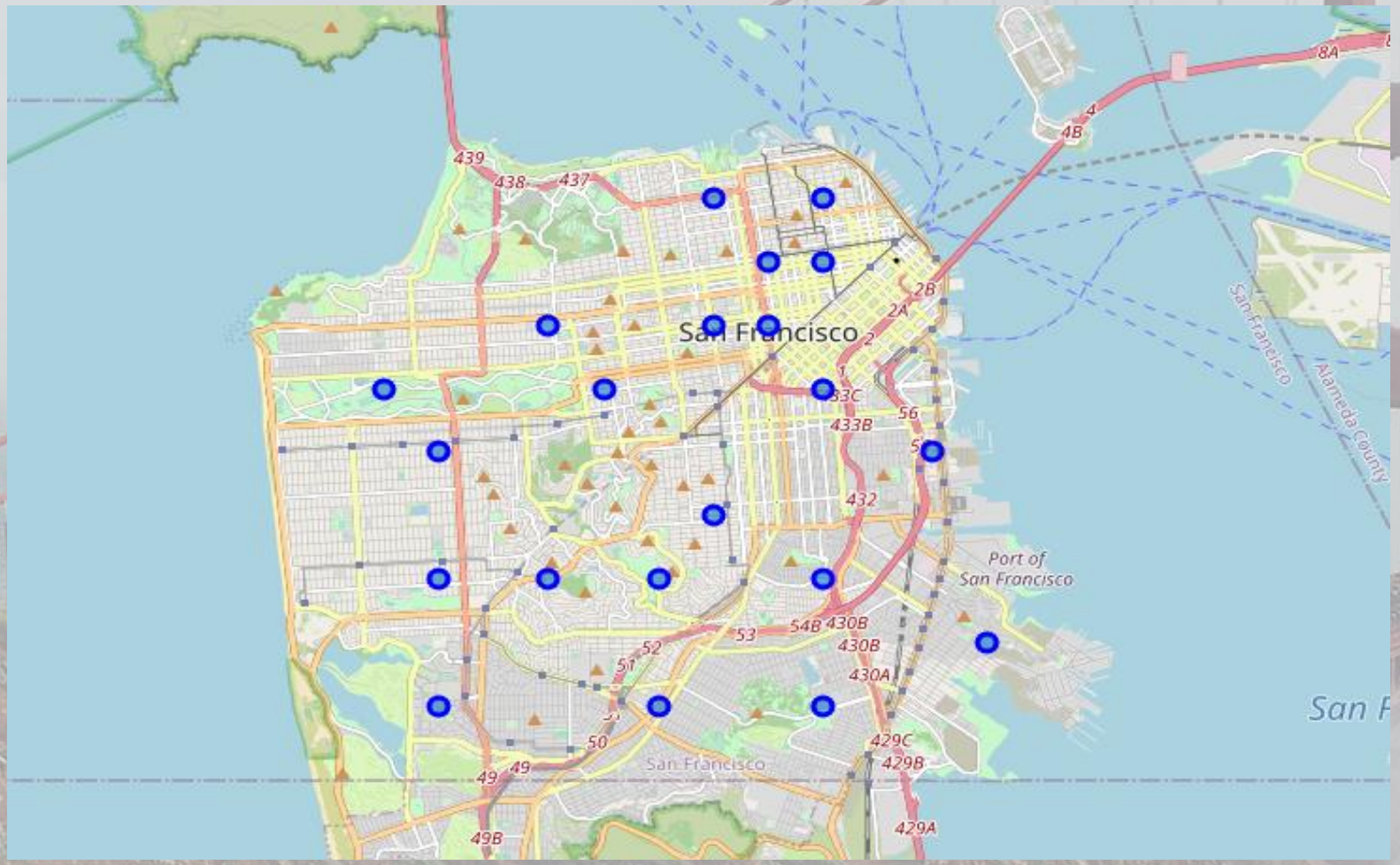
21 rows × 4 columns Open in new tab

Metodologia

- Após a análise, encontramos que o ‘dataframe’ é composto por 21 bairros. Depois, a biblioteca 'geopy' é usada para obter os valores de latitude e longitude da cidade de São Francisco: (37.7790262 ,-122.419906). O ‘dataframe’ é usado para visualizar o mapa da cidade de São Francisco marcando cada bairro da cidade. Assim um mapa é gerado usando a biblioteca *python folium.*

Metodologia

- Mapa da cidade de São Francisco



Metodologia

Dados Coletados na API do FourSquare:

- A API do Foursquare é usada para explorar as vizinhanças de um bairro. Para isso precisamos primeiro fazer um cadastro no site da API e gerar uma chave de autorização e com isso em mãos, devemos selecionar a 'Category ID' da seção 'Dining and Drinking' para exploramos todos os estabelecimentos desta categoria. Utilizamos todas as categorias da seção. Então, uma função é criada para retornar um dicionário com 'Category ID', 'Category Name' com Raio de 500 m e um limite de dados que Api fornecerá.

```
# get data from Foursquare
url = f"https://api.foursquare.com/v3/places/search?ll={lt},
 {lg}&radius=500&categories={category_id}&limit=50"
headers = {
        "Accept": "application/json",
        "Authorization": "[redacted]",
    }
```

Metodologia

Dados Coletados na API do FourSquare:

- Esta função da página 8 seleciona 'Category ID’ que escolhemos ser á 1300 para uma certa latitude e longitude e retorna todos os dados do primeiro bairro de São Franciso e em seguida, é usada novamente de para segundo bairro e assim sucessivamente até o último bairro.
- Paramos fazer o processo acima, é feito um loop percorrendo a latitude e longitude, portanto, percorre todos os bairros da cidade de São Francisco e cria uma URL de solicitação de API com raio = 500, LIMIT = 50. Por limite, é definido que no máximo 50 locais próximos devem ser retornados. Então, a solicitação é feita para a API do Foursquare e apenas as informações relevantes para cada local próximo são extraídas dela. Os dados são então salvos em um arquivo no formato JSON. Por fim, lermos todos os arquivos gerados e colocamos em uma lista e em seguida em um ‘*dataframe’.*

Metodologia

Código utilizado para acessar a API FourSquare

```
for index,rows in df_sf.iterrows():
    neigh=rows['Neighborhood']
    lt=rows['Latitude']
    lg=rows['Longitude']
    category_id="13000"
    print(category_id,lt,lg)


    # get  print category label
    print(
            dining.loc[dining["foursquare_category_label"] == category_id][
                "foursquare_category_label"
            ]
    )
    # get data from Foursquare
    url = f"https://api.foursquare.com/v3/places/search?ll={lt},
     {lg}&radius=500&categories={category_id}&limit=50"
```

```
# get data from Foursquare
url = f"https://api.foursquare.com/v3/places/search?ll={lt},
 {lg}&radius=500&categories={category_id}&limit=50"
headers = {
        "Accept": "application/json",
        "Authorization": "[redacted]",
    }
```

Metodologia

- **Código utilizado**

```
response = requests.get(url, headers=headers)
json_response = response.json()
# save as json file
out_file = f"C:\\Users\eliel\OneDrive\Desktop\Project_2\Json\Foursquare_data_{category_id}_{lt}_{
 .json"
# check if directory exists
if not os.path.exists(os.path.dirname(out_file)):
        os.makedirs(os.path.dirname(out_file))

with open(out_file, "w") as f:
    json.dump(json_response, f)
```

```
list_venues = []
lista=[]
for index,rows in df_sf.iterrows():
    neigh=rows['Neighborhood']
    lt=rows['Latitude']
    lg=rows['Longitude']
    category_id== "1300"
    for name in glob.glob(f"C:\\Users\eliel\OneDrive\Desktop\Project_2\Json\Foursquare_data_{category_id
     }_{lt}_{lg}.json"):
        with open(name, "r") as f:
            data = json.load(f)
        df = pd.json_normalize(data["results"])
        for index,rows in df.iterrows():
            cate = rows['categories']
            venue =rows['name']
            venue_latitude=rows['geocodes.main.latitude']
            venue_longitude=rows['geocodes.main.longitude']
            category = cate[0]['name']
            list_venues.append([neigh,lt,lg,venue,venue_latitude,venue_longitude,category])
```

Metodologia

- Após a leitura dos arquivos JSON juntamos todos os dados em uma *'dataframe'*:

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue
0	Hayes Valley/Tenderloin/North of Market	37.78	-122.42	
1	Hayes Valley/Tenderloin/North of Market	37.78	-122.42	
2	Hayes Valley/Tenderloin/North of Market	37.78	-122.42	
3	Hayes Valley/Tenderloin/North of Market	37.78	-122.42	
4	Hayes Valley/Tenderloin/North of Market	37.78	-122.42	
..	...	...	...	
18	North Beach/Chinatown	37.80	-122.41	

Metodologia

- Portanto, teremos dois '*dataframe' python*, o primeiro contem: os bairros com Cep, Nome do bairro, Latitude e Longitude do bairro da cidade de São Francisco, e o segundo , contem toda a informação do primeiro '*dataframe*' e os dados locais da categoria *Dining and Drinking* pesquisados com um Raio = 500 metros e Limite = 50.

	Zip Code	Neighborhood	Latitude	Longitude
15	94123	Marina	37.80	-122.43
16	94124	Bayview-Hunters Point	37.73	-122.38
17	94127	St. Francis Wood/Miraloma/West Portal	37.74	-122.46
18	94131	Twin Peaks-Glen Park	37.74	-122.44
19	94132	Lake Merced	37.72	-122.48
20	94133	North Beach/Chinatown	37.80	-122.41
21	94134	Visitacion Valley/Sunnydale	37.72	-122.41

21 rows × 4 columns Open in new tab

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue
0	Hayes Valley/Tenderloin/North of Market	37.78	-122.42	
1	Hayes Valley/Tenderloin/North of Market	37.78	-122.42	
2	Hayes Valley/Tenderloin/North of Market	37.78	-122.42	
3	Hayes Valley/Tenderloin/North of Market	37.78	-122.42	
4	Hayes Valley/Tenderloin/North of Market	37.78	-122.42	
..	...	...	...	
18	North Beach/Chinatown	37.80	-122.41	

Análise de Dados

- O 2 ‘dataframe’ mencionado na pagina 13 tem todas as informações necessárias. O tamanho desse ‘dataframe’ é de 823 locais.

```
1 df_sf_new
```

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude
0	Hayes Valley/Tenderloin/North of Market	37.78	-122.42
1	Hayes Valley/Tenderloin/North of Market	37.78	-122.42
2	Hayes Valley/Tenderloin/North of Market	37.78	-122.42
3	Hayes Valley/Tenderloin/North of Market	37.78	-122.42
4	Hayes Valley/Tenderloin/North of Market	37.78	-122.42
...	...	...	...
818	North Beach/Chinatown	37.80	-122.41

823 rows × 7 columns Open in new tab

Análise de Dados

‘Limpeza’ dos Dados

- É fundamental entender que o ponto de interesse do projeto é entender o hábitos alimentar de um bairro usando as informações obtidas dos locais. Então verificamos primeiro se existem valores nulos e tentar substitui-los. Por exemplo, foram encontrados 3 valores nulos na latitude e longitude de 3 estabelecimentos. Então, procuramos encontrar esses dados através do seguinte código:

```
lista=[]
for index, rows in df_sf_new[df_sf_new['Venue Latidude'].isnull()].iterrows():
    address=rows['Venue']
    if address == 'Conchita Taco Truck':
        address = address.split()[0]
        location = geolocator.geocode(address)
        latitude = location.latitude
        longitude = location.longitude
        lista.append([latitude,longitude])
    else:
        location = geolocator.geocode(address)
        latitude = location.latitude
        longitude = location.longitude
        lista.append([latitude,longitude])
```

```
df_sf_new[['Venue Latidude','Venue Longitude']]=df_sf_new[['Venue Latidude','Venue Longitude']].replace
 (np.nan,str(lista))
```

Análise de Dados

- Esta análise de dados e totalmente importante para que não venhamos encontrar nenhum erro em nosso resultados.

- Agora, cada bairro é analisado individualmente para entender a culinária mais comum servida em seus 500 metros de vizinhança.

- O processo é realizado usando a função '*one hot encoding'* da biblioteca *python* 'pandas'. Uma codificação quente converte as variáveis categóricas (que são 'Categoria do local') em um formato que pode ser fornecido aos algoritmos de *Marchine Learning* para encontramos uma melhor na previsão.

Análise de Dados

- Parte do código utilizado para o processo mencionado na página 16.

```
# one hot encoding
df_sf_onehot_new = pd.get_dummies(df_sf_new[['Venue Category']], prefix = "", prefix_sep = "")
# add neighborhood column back to dataframe
df_sf_onehot_new['Neighborhood'] = df_sf_new['Neighborhood']

df_sf_onehot_new = df_sf_onehot_new.reindex(columns=['Neighborhood','African Restaurant', 'American
 Restaurant', 'Arcade','Arts and Entertainment', 'Asian Restaurant', 'Australian Restaurant',
       'BBQ Joint', 'Bagel Shop', 'Bakery', 'Bar', 'Beer Bar', 'Bistro',
       'Breakfast Spot', 'Brewery', 'Bubble Tea Shop', 'Buffet',
       'Burger Joint', 'Burmese Restaurant', 'Cafes, Coffee, and Tea Houses',
       'Café', 'Cajun / Creole Restaurant', 'Cantonese Restaurant',
       'Caribbean Restaurant', 'Chinese Restaurant', 'Cocktail Bar',
       'Coffee Shop', 'Concert Hall', 'Creperie', 'Deli', 'Dessert Shop',
       'Dim Sum Restaurant', 'Diner', 'Dining and Drinking', 'Dive Bar',
       'Donut Shop', 'Falafel Restaurant', 'Fast Food Restaurant','Filipino Restaurant', 'Food Stand',
        'Food Truck', 'French Restaurant',
```

Análise de Dados

- Ao converter as variáveis categóricas, conforme mostrado na página 17. O tamanho do novo '*dataframe*' é examinado e verifica-se que existem cerca de 823 linhas e 99 colunas.
- Agora agrupamos as linhas por bairros e tomando a média da frequência de ocorrência de cada Categoria de Local.
- Parte do '*dataframe*' da etapa acima:

Neighborhood	African Restaurant	American Restaurant	Arcade	Arts
Bayview-Hunters Point	0.0	0.00	0.0	
Castro/Noe Valley	0.0	0.04	0.0	
Chinatown	0.0	0.06	0.0	
Haight-Ashbury	0.0	0.02	0.0	
Hayes Valley/Tenderloin/North of Market	0.0	0.02	0.0	

Análise de Dados

- Como o limite está definido para 50, não haverá muitos locais sendo retornados pela API do Foursquare. Mas os *Dining and Drinking* dos bairro pode ser definido pelos 10 melhores locais em sua vizinhança. Todavia, o '*dataframe*' criado é então alimentado com as 10 principais categorias de locais mais comuns no respectivo bairro.
- Parte do '*dataframe*'gerado pela etapa acima:

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3r
0	Bayview-Hunters Point	Restaurant	Coffee Shop	
1	Castro/Noe Valley	Restaurant	Burger Joint	
2	Chinatown	Restaurant	Cocktail Bar	
3	Haight-Ashbury	Café	Coffee Shop	
4	Hayes Valley/Tenderloin/North of Market	Cocktail Bar	Asian Restaurant	

5 rows × 11 columns Open in new tab

Aprendizado de máquina: Cluster de Bairros – K means

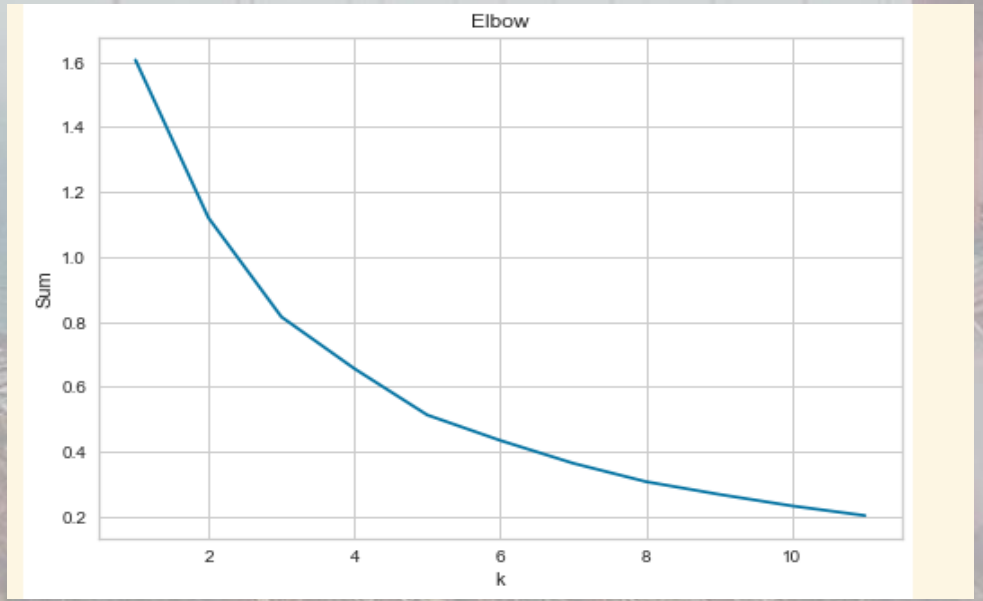
- *k-means'* é um algoritmo de aprendizado de máquina não supervisionado que gera clusters de pontos de dados agregados devido a algumas semelhanças. Então, esse método será usado para contar vizinhanças para cada *"Label"* de cluster para o tamanho do cluster.
- Para então implementarmos este método, é muito importante descobrir qual é o melhor número ótimo de clusters. Existem 2 métodos mais populares para o mesmo, ou seja, 'O Método do Cotovelo' e 'O Método da Silhueta'.
- Aqui iremos utilizar o 'método do cotovelo'.

Aprendizado de máquina: Cluster de Bairros – K means

O Método do Cotovelo:

- O ‘Método do cotovelo’ calcula a soma das distâncias quadradas das amostras ao centro do cluster mais próximo para diferentes valores de 'k'. O número ótimo de clusters é o valor após o qual não há diminuição significativa na soma das distâncias quadradas. O código implementado para esse método

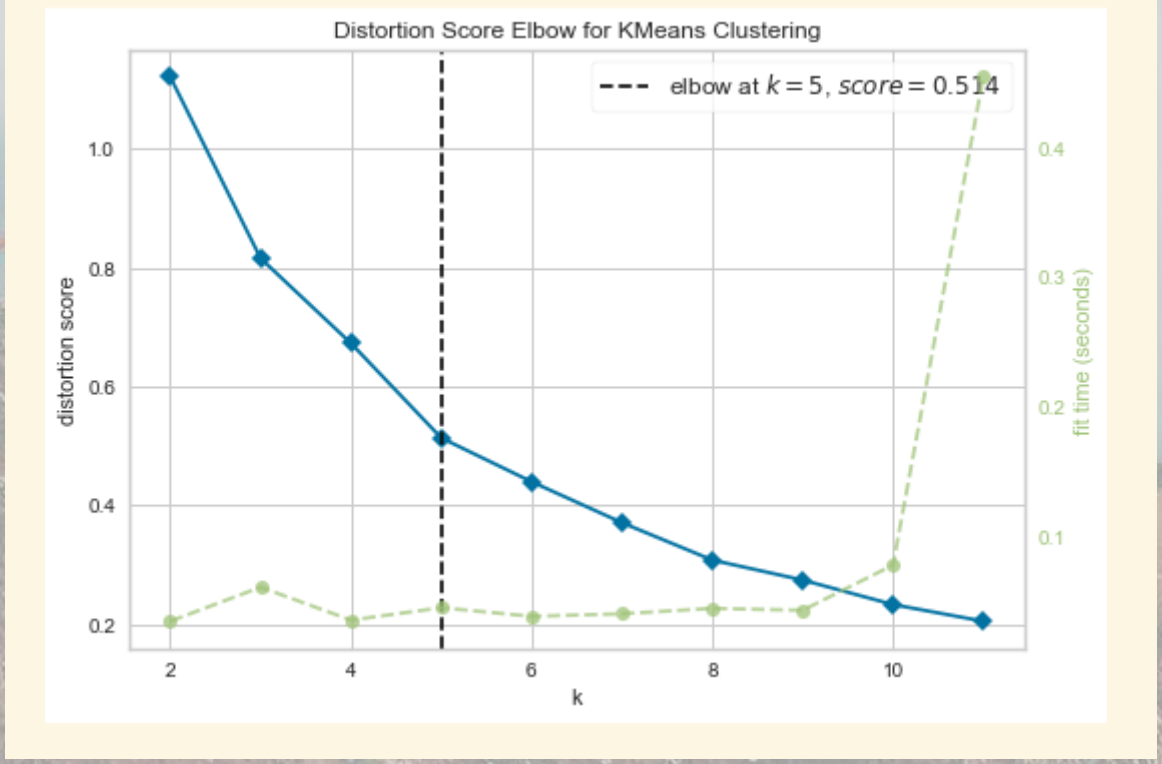
```
sum=[]

k=range(2,12)
for i in k:
    kmeans=KMeans(n_clusters=i).fit(df_sf_grouped_clustering)
    sum.append(kmeans.inertia_)

```



Aprendizado de máquina: Cluster de Bairros – K means

O Método do Cotovelo:

- O gráfico não nos fornece exatamente qual seria o melhor número de cluster a ser utilizado. Então, usaremos *KElbowVisualizer* então temos o gráfico a seguir:



Aprendizado de máquina: Cluster de Bairros – K means

O Método do Cotovelo:

- O gráfico encontra o melhor número k=5, que é um resultado aceitável, pois irá dividir os bairros em 6 grupos e como as vizinhanças em cada cluster é batente semelhante entre si pela características incluídas no conjunto de dados

K means:

- O código a seguir executa o método *k-Means* com o número de clusters = 5 e encontramos as contagens das vizinhanças atribuídas a diferentes clusters:

```
from sklearn.cluster import KMeans
# set number of clusters
kclusters = 6
# run k-means clustering
kmeans = KMeans(n_clusters = kclusters, random_state = 0).fit(df_sf_grouped_clustering)
# check cluster labels generated for each row in the dataframe
kmeans.labels_[0:10]
```

```
array([1, 0, 0, 0, 0, 0, 0, 0, 0, 0])
```

Aprendizado de máquina: Cluster de Bairros – K means

K means:

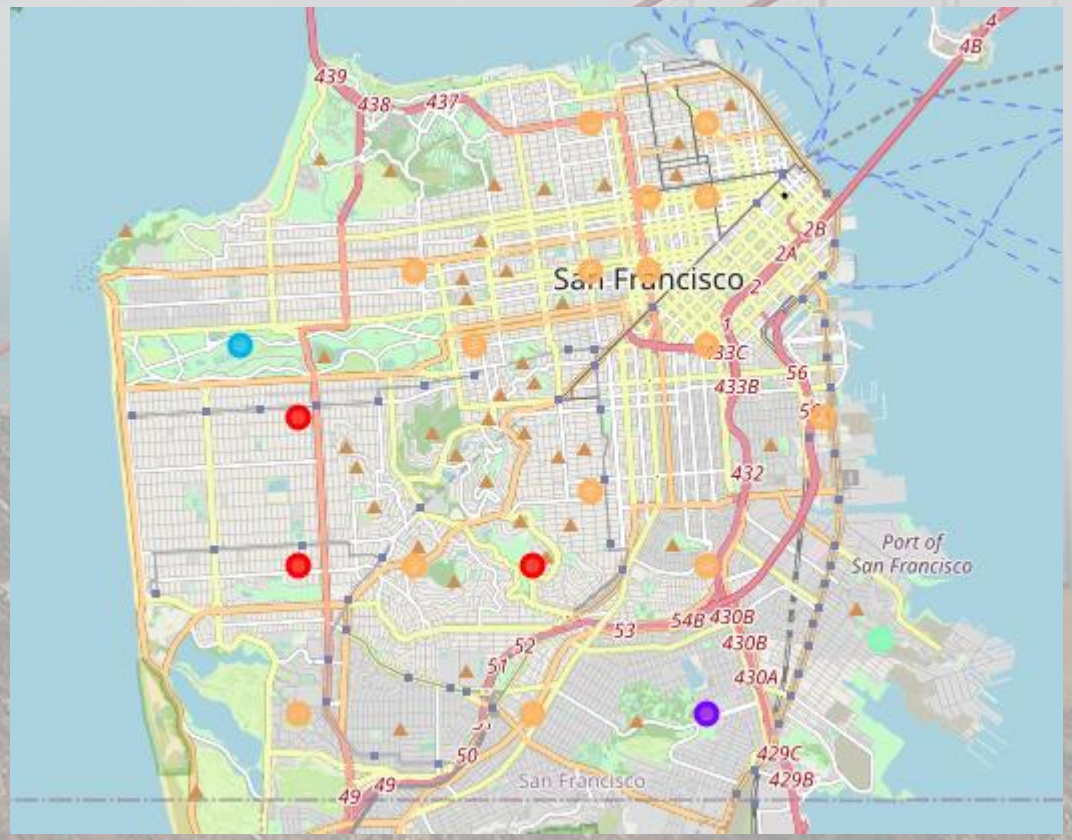
- Os *'Label'* de cluster selecionados são colocados ao *'dataframe'* para encontramos os resultados desejados de segmentação da vizinhança com base nos locais mais comuns em sua vizinhança. Portanto, juntamos em um *'dataframe'.* Parte do *'dataframe'* abaixo:

ode	Neighborhood	Latitude	Longitude	Cluster Labels	1st Most C
94102	Hayes Valley/Tenderloin/North of Market	37.78	-122.42	0	
94103	South of Market	37.77	-122.41	0	
94107	Potrero Hill	37.76	-122.39	0	
94108	Chinatown	37.79	-122.41	0	
94109	Polk/Russian Hill (Nob Hill)	37.79	-122.42	0	

Aprendizado de máquina: Cluster de Bairros – K means

K means:

- A partir disso fizemos um novo mapa de São Francisco adicionando os cluster:



Resultados

- O cluster 5 é o maior cluster com 15 bairros. Consiste principalmente em restaurantes e alguém que prefira variedade de bebidas e alimentos deve considerar a vizinhança nesses grupos para ficar.

	Neighborhood	1st Most Common Venue	2nd Most Common Venue
1	Hayes Valley/Tenderloin/North of Market	Cocktail Bar	Asian Restaurant
2	South of Market	Night Club	Coffee Shop
3	Potrero Hill	Cocktail Bar	Bakery
4	Chinatown	Restaurant	Cocktail Bar
5	Polk/Russian Hill (Nob Hill)	Cocktail Bar	Coffee Shop
6	Inner Mission/Bernal Heights	Restaurant	Bakery
7	Ingelside-Excelsior/Crocker-Amazon	Latin American Restaurant	Asian Restaurant

15 rows × 11 columns Open in new tab

```
Restaurant                    5
Cocktail Bar                  4
Night Club                    1
Latin American Restaurant     1
Fast Food Restaurant          1
Café                          1
Vietnamese Restaurant         1
Thai Restaurant               1
Name: 1st Most Common Venue, dtype: int64
---------
```

Conclusão

- Apos aplicar um *algoritmo de Clustering* , k-Means, a um conjunto de dados multidimensional. Os bairros da cidade de São Francisco foram brevemente agrupados em 5 clusters e, após análise, foi possível voltar a renomear eles com base nas categorias de locais dentro e ao redor desse bairro. Então como o principal objetivo era examinar os *Dining and Drinking na cidade de São Francisco*, e podemos destacar a predominância de restaurantes, cocktail bar e coffee shop na cidade de São Francisco.