Applied Data Science Capstone

THE BATTLE OF NEIBORHOODS

Introduction

With its scenic views, mild climate, and friendly people, Vancouver is known around the world as both a popular tourist attraction and one of the best places to live. It is also one of the most ethnically and linguistically diverse cities in Canada.

Vancouver is made up of a few smaller neighbourhoods and communities. Neighbourhood boundaries provide a way to break up the city's large geographical area for delivering services and resources and identify the distinct culture and character of different areas of our diverse population.

Goals

In this project, we will study, analyze, cluster, and compare the neighborhoods of Vancouver. We will investigate on what kinds of businesses are most common in the city, which outdoors and recreation activities and what kinds of restaurants are most common between the neighborhoods.

Data Acquisition and Preparation

The process of acquiring, cleaning, and preparing the dataset used in this project for the next stages will be specified. To be able to do this project, two types of data are needed:

- **Neighborhood Data:** datasets with the list names of the neighborhoods of Vancouver and their latitude and longitude coordinates. The neighborhoods names were obtained in the website of Vancouver (https://vancouver.ca/news-calendar/areas-of-the-city.aspx) and the latitude and longitude data were obtained using a recursive function that would return the geocode of the address passed into it.

- **Venues Data:** data that describes the top 100 venues (restaurants, cafes, parks, museums, etc) in each neighborhood of Vancouver. The data should list the venues of each neighborhood with their categories. This data will be retrieved from Foursquare which is one of the world largest sources of location and venues data. Foursquare API will be utilized to get and download the data.

Neighborhood Data – Vancouver

```
[2]: # Vancouver neighborhoods
van_neighborhoods = ['Arbutus Ridge', 'Cedar Cottage', 'Champlain Heights', 'Chinatown', 'Coal Harbour', 'Collingwood',
                     'Commercial Drive', 'Creekside', 'Downtown', 'Downtown Eastside', 'Dunbar-Southlands', 'Fairview',
                     'False Creek North', 'False Creek South', 'Gastown', 'Grandview-Woodland', 'Granville Island',
                     'Hastings-Sunrise', 'Hastings Crossing', 'Hastings East', 'Kensington-Cedar Cottage', 'Kerrisdale',
                     'Killarney', 'Kitsilano', 'Knight', 'Langara', 'Little Mountain', 'Main', 'Marpole', 'Mole Hill',
                     'Mount Pleasant', 'Musqueam', 'Oakridge', 'Quilchena', 'Renfrew-Collingwood', 'Riley Park',
                     'Shaughnessy', 'South Cambie', 'South Granville', 'South Hill', 'South Vancouver', 'Southlands',
                     'Southwest Marine', 'Sunrise', 'Sunset', 'Victoria-Fraserview', 'West Broadway', 'West End', 'West Point Grey', 'Yaletown']
```

```
[3]: df_van = pd.DataFrame(van_neighborhoods)
df_van.columns = ['Neighborhood']
df_van.head()
```

[3]:

	Neighborhood
0	Arbutus Ridge
1	Cedar Cottage
2	Champlain Heights
3	Chinatown
4	Coal Harbour

```
[4]: #create a function to handle TimeOuts from Geocoder
from geopy.exc import GeocoderTimedOut
locator = Nominatim(user_agent = "bostonagent")

def do_geocode(address):
    try:
        return locator.geocode(address)
    except GeocoderTimedOut:
        return do_geocode(address)
```

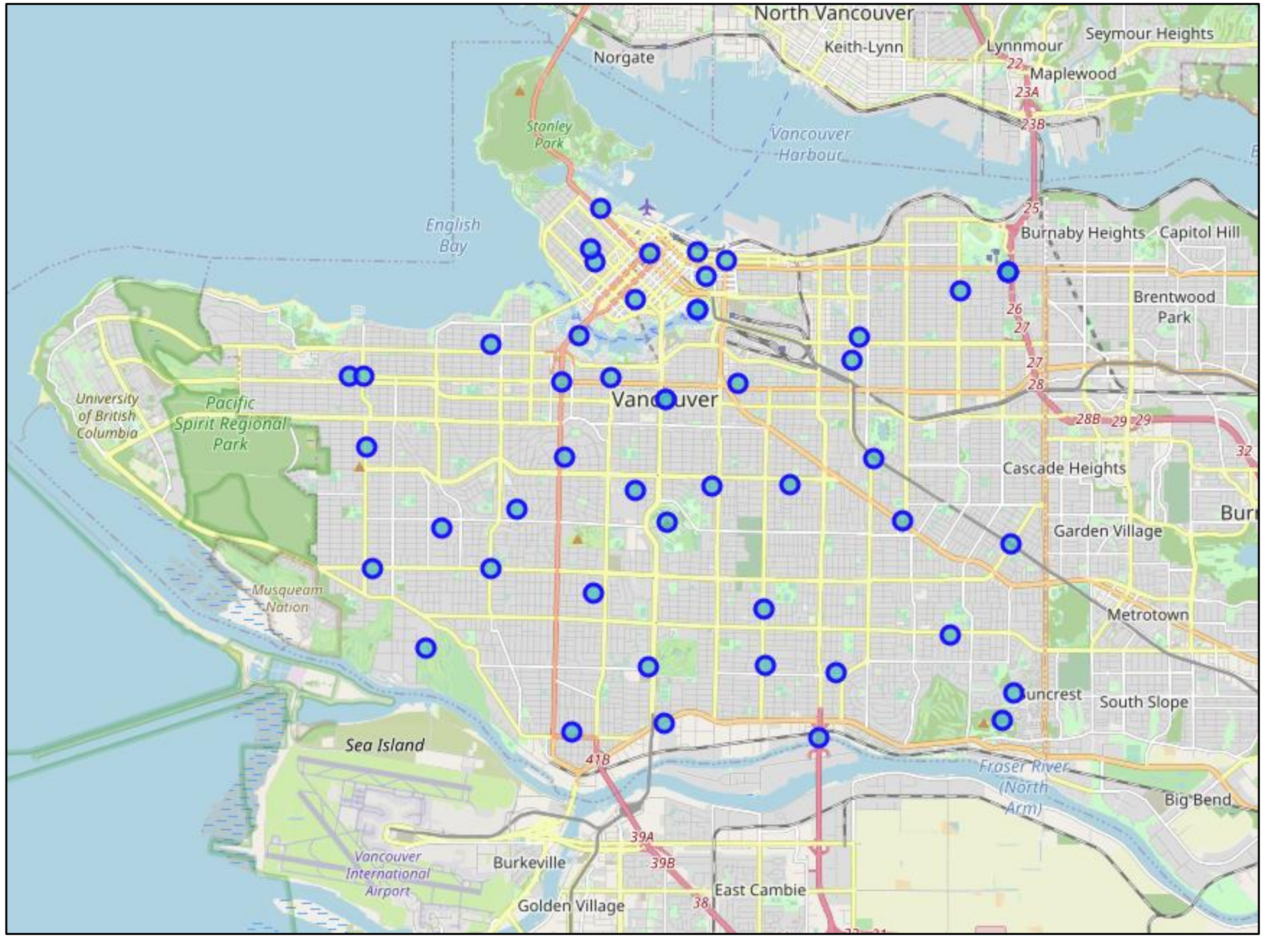
```
[5]: neighborhoods = df_van.values.tolist()

latitude = []
longitude = []
for neighborhood in neighborhoods:
    print('-', end='')
    coord = do_geocode('{}, Vancouver'.format(neighborhood))

    #check to make sure all latitude and longitude values are present in the Nominatim API
    #handles the case where Nominatim returns a 'None' object because the neighborhood does not exist in their API

    if (coord == None):
        latitude.append('0')
        longitude.append('0')
    else:
        latitude.append(coord.latitude)
        longitude.append(coord.longitude)

#add coordinates columns to dataframe
df_van['Latitude'] = latitude
df_van['Longitude'] = longitude

df_van.head()
```

A dataset was created from the combination of two sources.

	Neighborhood	Latitude	Longitude
0	Arbutus Ridge	49.240968	-123.167001
1	Cedar Cottage	49.251622	-123.064548
2	Champlain Heights	49.215266	-123.030915
3	Chinatown	49.279981	-123.104089
4	Coal Harbour	49.290375	-123.129281

Neighborhood Data – Vancouver

A map of Vancouver and its neighborhoods.



Venues Data – Vancouver

- For the city, data that describes the venues of its neighborhoods and the categories of these venues is needed.
- Foursquare API service will be utilized to access and download venues data. To retrieve data from Foursquare using their API, a URL should be prepared and used to request data related a specific location. An example URL is the following:

FOURSQUARE

```
https://api.foursquare.com/v2/venues/search?
&client_id=1234&client_secret=1234&v=20180605&
ll=40.89470517661,-73.84720052054902&radius=500&limit=100
```

Venues Data – Vancouver

The code used to create a function that takes as input the names, latitudes, and longitudes of the neighborhoods, and returns a dataframe with information about each neighborhood and its venues.

```
[9]: # Explore Neighborhoods in Vancouver

LIMIT = 100
def getNearbyVenues(names, latitudes, longitudes, radius=500):

    venues_list=[]
    for name, lat, lng in zip(names, latitudes, longitudes):
        print('.', end='')

        # create the API request URL
        url = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secret={}&v={}&ll={},{}&radius={}&limit={}'.format(
            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([(
            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 = ['Neighborhood',
                  'Neighborhood Latitude',
                  'Neighborhood Longitude',
                  'Venue',
                  'Venue Latitude',
                  'Venue Longitude',
                  'Venue Category']

    return(nearby_venues)
```

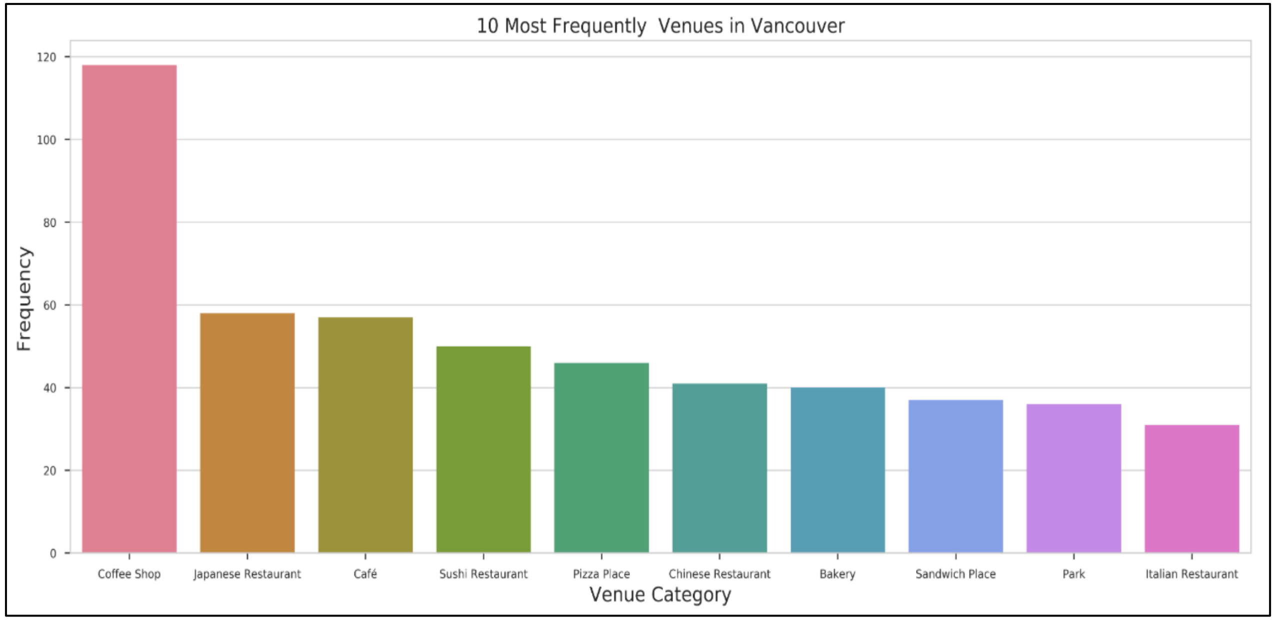
Venues Data – Vancouver

The head of the dataframe returned by the function for Vancouver.

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Arbutus Ridge	49.240968	-123.167001	Butter Baked Goods	49.242209	-123.170381	Bakery
1	Arbutus Ridge	49.240968	-123.167001	The Haven	49.241377	-123.166331	Spa
2	Arbutus Ridge	49.240968	-123.167001	Barktholomews Pet Supplies	49.242746	-123.170193	Pet Store
3	Arbutus Ridge	49.240968	-123.167001	The Dragon's Layer	49.238518	-123.169029	Nightlife Spot
4	Arbutus Ridge	49.240968	-123.167001	The Heights Market	49.237902	-123.170949	Grocery Store
5	Cedar Cottage	49.251622	-123.064548	Commercial Street Cafe	49.252539	-123.068178	Café
6	Cedar Cottage	49.251622	-123.064548	Trout Lake Community Centre	49.255403	-123.065048	Gym
7	Cedar Cottage	49.251622	-123.064548	Trout Lake Fitness Centre	49.255601	-123.065317	Gym / Fitness Center
8	Cedar Cottage	49.251622	-123.064548	The Lower Mainland Childbearing Society	49.252836	-123.068136	Child Care Service
9	Cedar Cottage	49.251622	-123.064548	Flourist Mill & Bakery	49.253881	-123.068209	Bakery

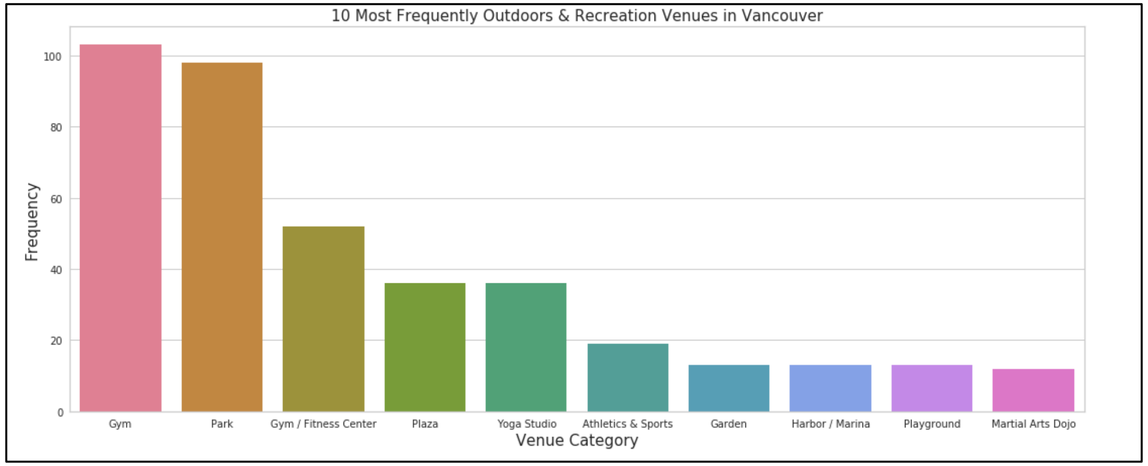
Exploratory data Analysis – Vancouver

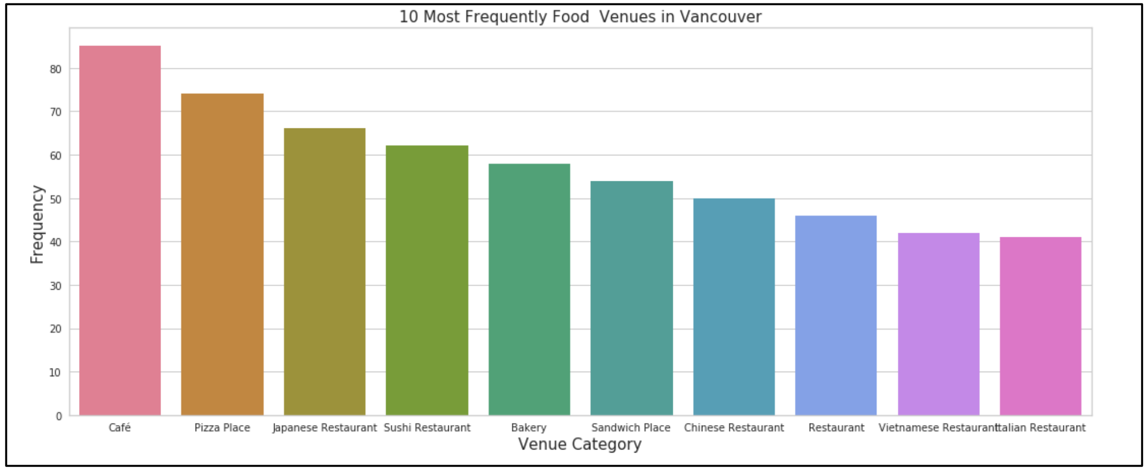
- Most Common Venue Categories (All categories).



Exploratory data Analysis – Vancouver

- Most Common Venue - Outdoors & Recreation Categories.

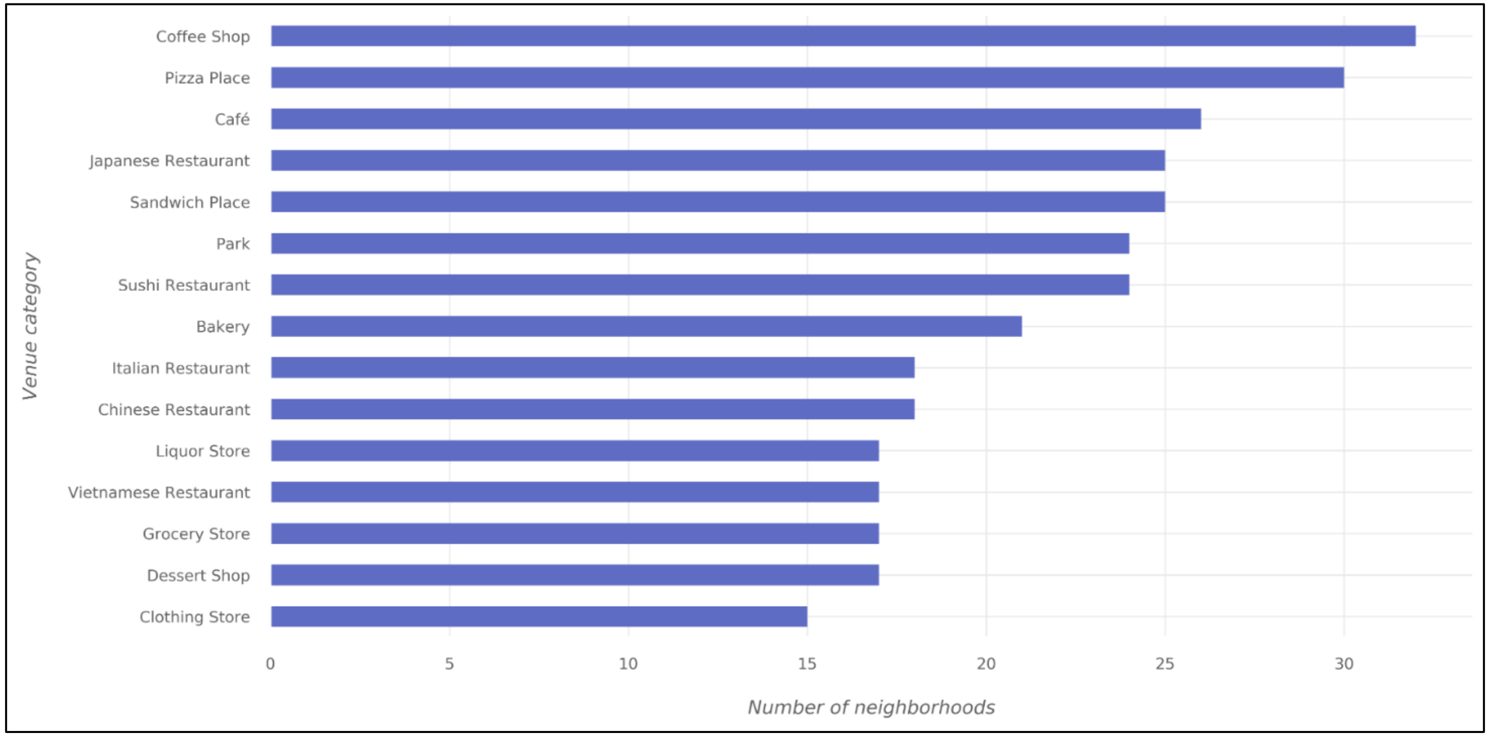




- Most Common Venue - Food Categories.

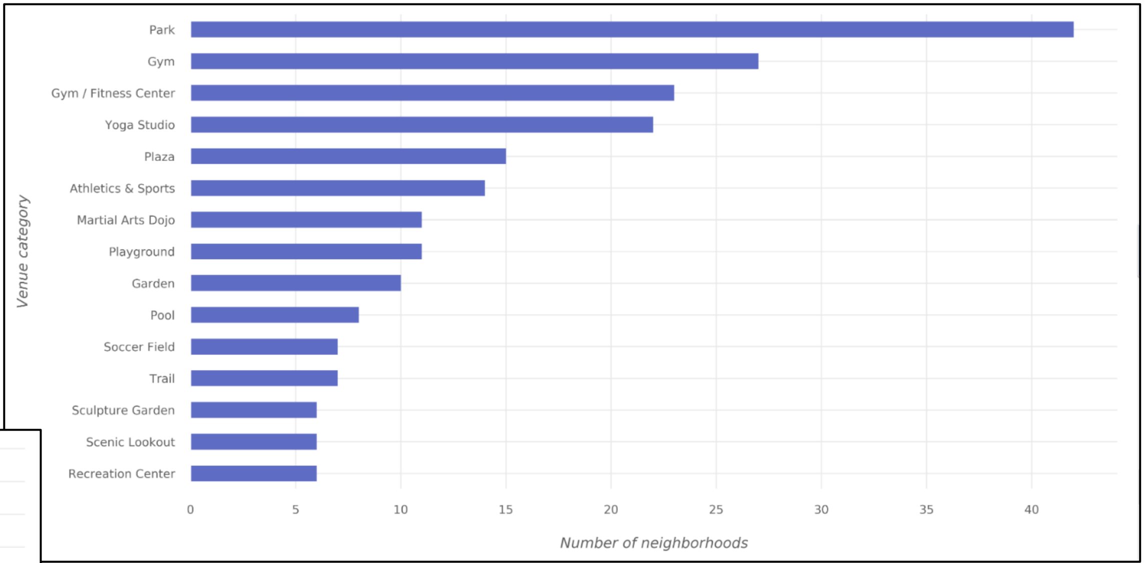
Exploratory data Analysis – Vancouver

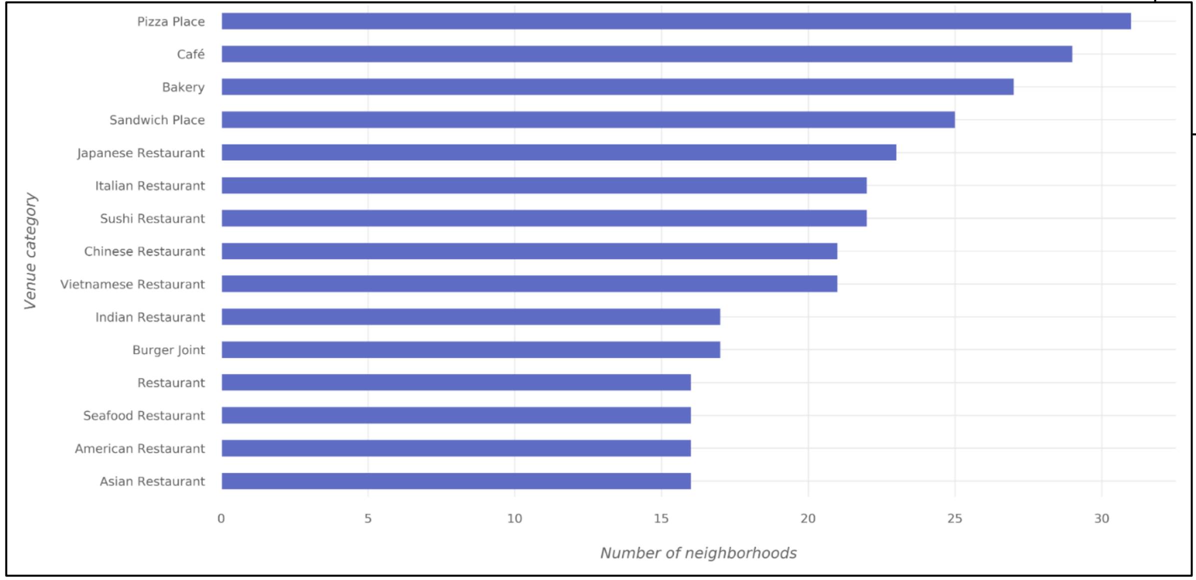
- Most Widespread Venue Categories (All categories).



Exploratory data Analysis – Vancouver

- Most Widespread Venue - Outdoors & Recreation Categories.



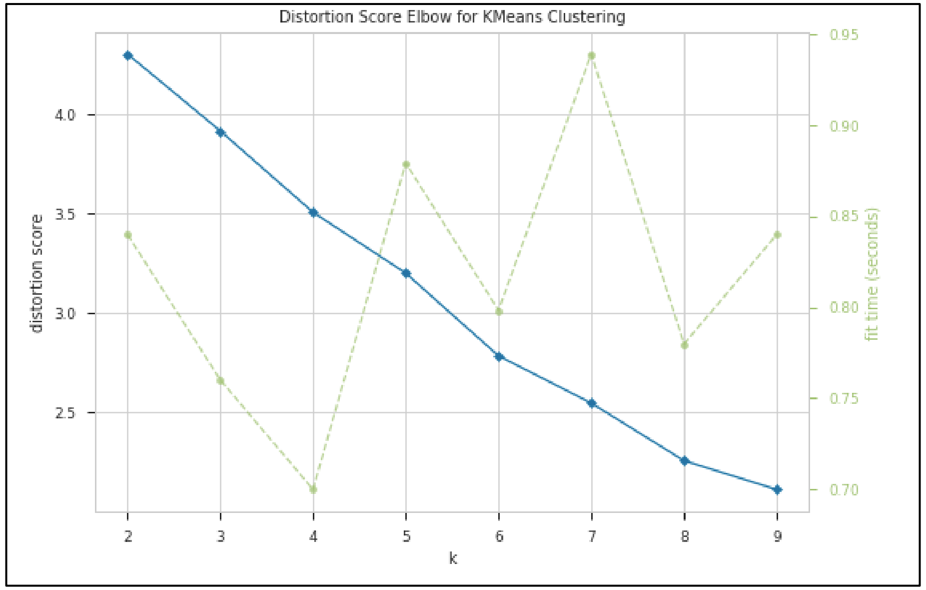


- Most Widespread Venue - Food Categories.

Clustering of Neighborhoods – Vancouver

Clustering will be applied on Vancouver neighborhoods to find similar neighborhoods in the city.. In particular, K-means clustering algorithm of the Scikit-learn Python library will be used.

- Using Elbow method for the value of k.

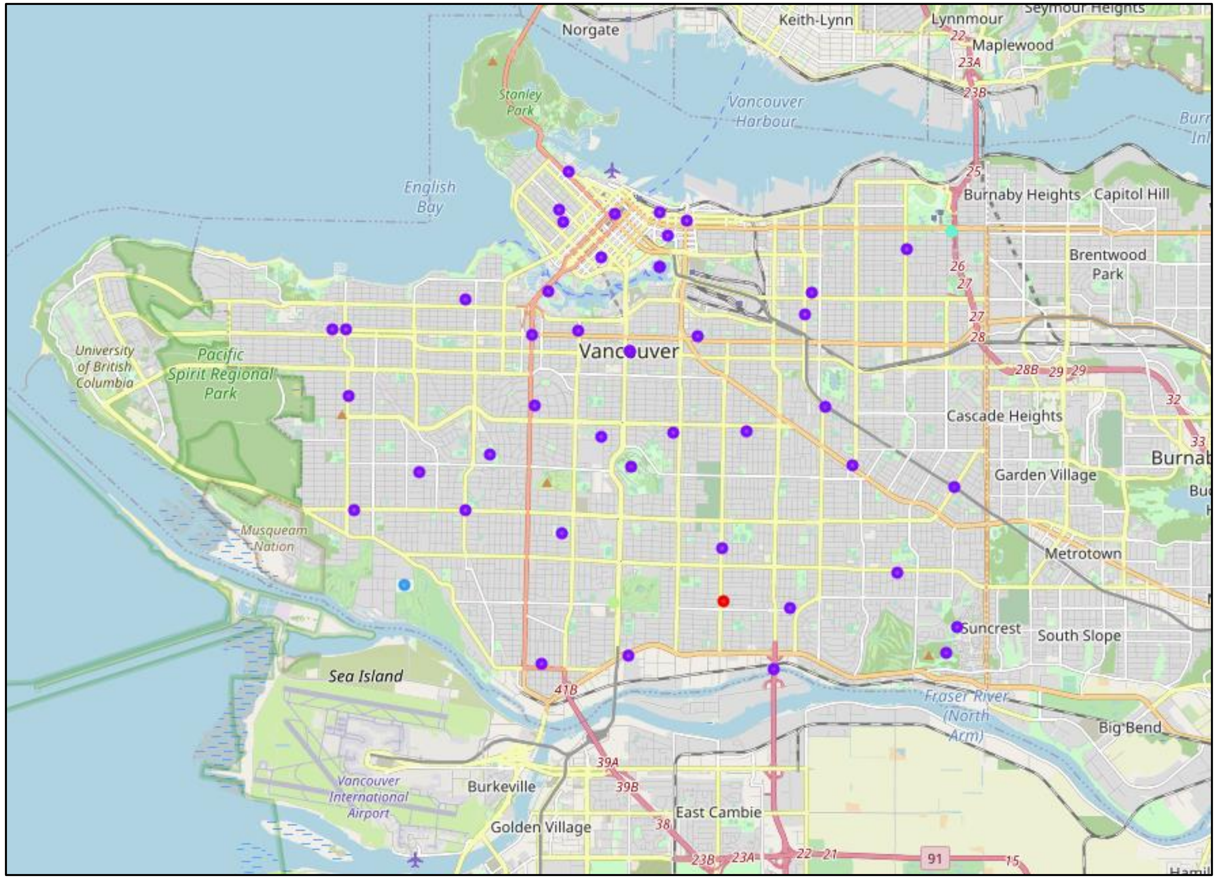


- Dataset with cluster labels.

	Neighborhood	Latitude	Longitude	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue
0	Arbutus Ridge	49.240968	-123.167001	1	Spa	Pet Store	Nightlife Spot	Bakery	Grocery Store	Yoga Studio
1	Cedar Cottage	49.251622	-123.064548	1	Lake	Skating Rink	Child Care Service	Bakery	Park	Bookstore
2	Champlain Heights	49.215266	-123.030915	1	Video Store	Recreation Center	Park	Pizza Place	Bus Stop	Flower Shop
3	Chinatown	49.279981	-123.104089	1	Café	Coffee Shop	Sandwich Place	Pizza Place	Chinese Restaurant	Mexican Restaurant
4	Coal Harbour	49.290375	-123.129281	1	Japanese Restaurant	Coffee Shop	Ramen Restaurant	Dessert Shop	Café	Park

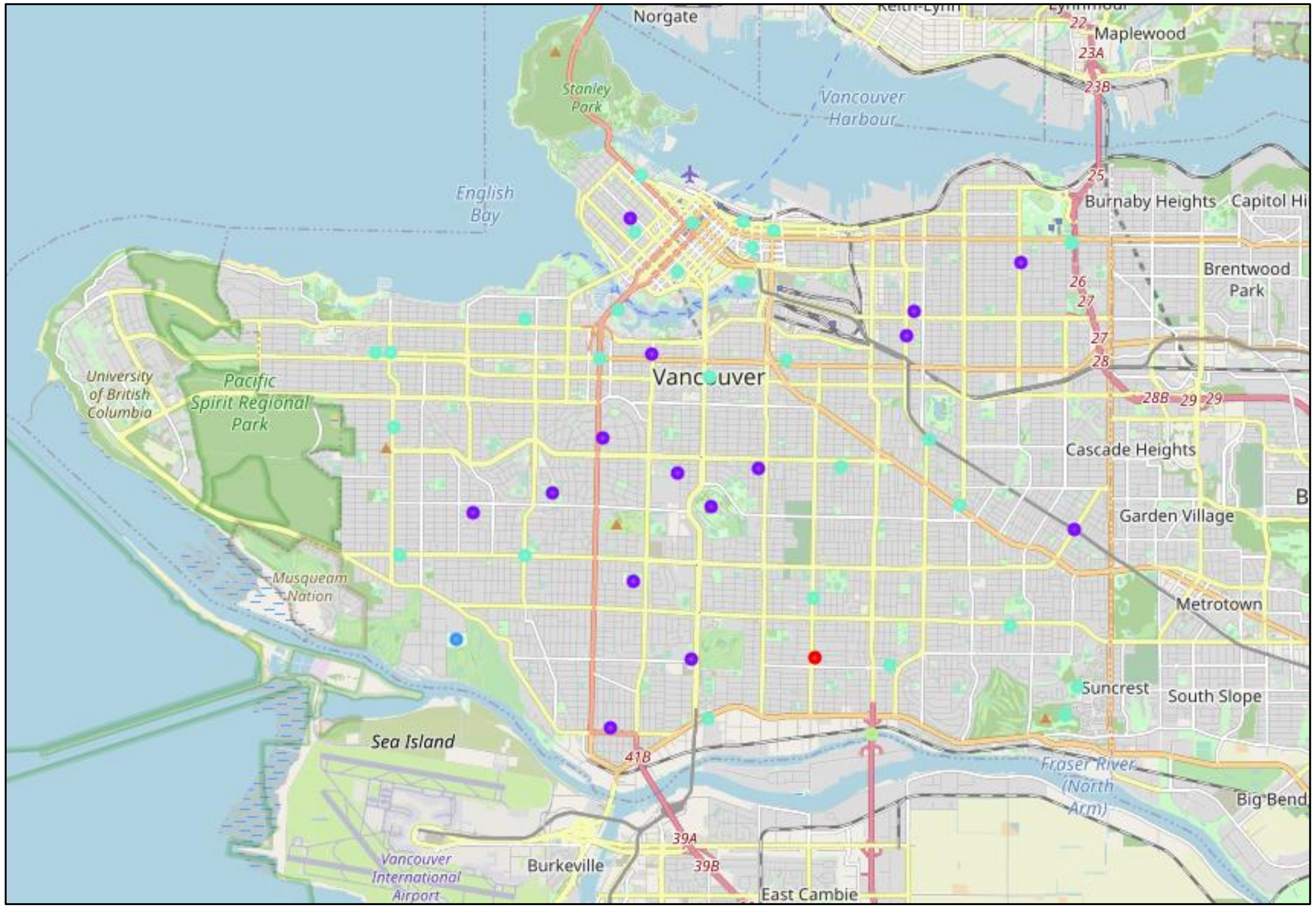
Clustering of Neighborhoods – Vancouver

- Map with the 5 clusters for all venue categories .



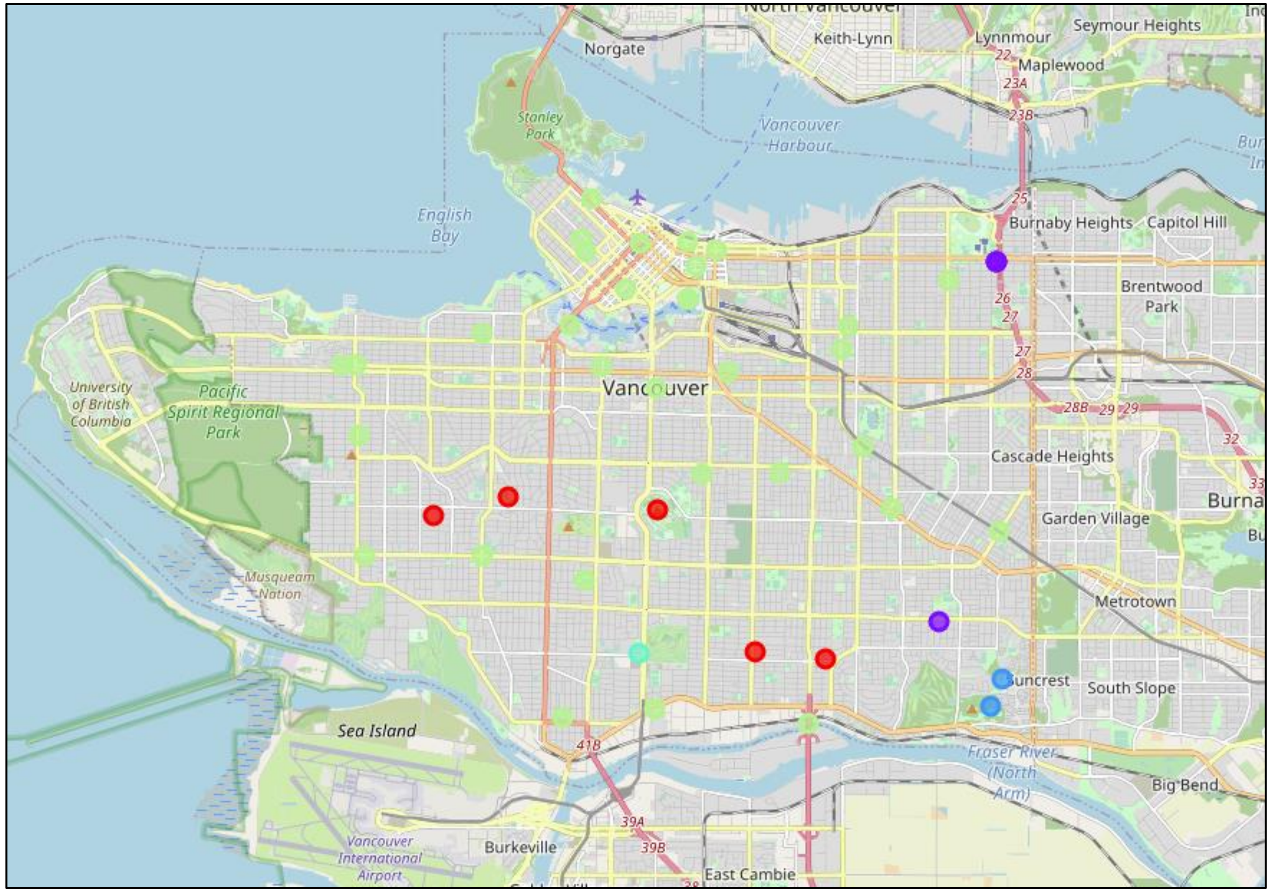
Clustering of Neighborhoods – Vancouver

- Map with the 5 clusters for Outdoors & Recreation categories .



Clustering of Neighborhoods – Vancouver

- Map with the 5 clusters for Food categories .



Conclusions

In this project, the neighborhoods of Vancouver were clustered into multiple groups based on the categories (types) of the venues in these neighborhoods. The results showed that there are venue categories that are more common in some cluster than the others; the most common venue categories differ from one cluster to the other. If a deeper analysis—taking more aspects into account—is performed, it might result in discovering different style in each cluster based on the most common categories in the cluster.