<u>Analysis of prospective New Office locations in the city of Vancouver, Canada.</u>

<u>Introduction</u> -

One of the key challenges faced by businesses today is to identify prospective locations to set up offices. Various factors come into play in taking this decision. The issue is especially demanding for the MNCs that frequently need to set up offices at new locations across the globe. Although the decision making process is highly subjective, the process largely includes a common format -
"Businesses need the data regarding the amenities at different areas in a target city !!"
Getting such data is of use particularly for international businesses that have never been players in that city/area. Getting this data will help in having an idea about the different factors that affect business. For an example, a Travel / Tourism start-up might prefer a location that is close to a major bus terminus, railway station, airport or a place where often tourism fairs are organised. A restaurant chain might look to expand its business in another part of the city where other providers are available and the footfall is high!! They might try to lure in customers of a competitive business by opening a branch nearby. But for all these to happen, they need authentic up-to-date information on the particulars of an area.
In this document, we will discuss one such case study where we develop a model to offer strategic business locations for businesses - both new and existing to choose a particular area (Postal Code) in the city of Vancouver, Canada, as a prospective site for them to open office and expand their business.

<u>The data used</u> –

As discussed in the last paragraph, we concluded that we will develop a model that is able to showcase the characteristics of a particular area in the city of Vancouver. For that to happen, we need a list of all the areas in that city. In our case, our first aim is to obtain the postal code addresses of the city of Vancouver from "www.zip-codes.com". We then convert the relevant data into a Data Frame using pandas. Now, we need to obtain the latitude and longitude values of these postal areas. For that we use Google's Geocoding API. We add the newly fetched latitude and longitude values into the Data Frame against relevant Postal Codes and pass each such row to the Foursquare API to obtain the Venues around those locations. In such a way we are able to gain insight about the venues in the postal area.

We can then cluster the Data Frame into groups based on the data pertaining to the venues and when the user inserts his priority (such as location near banks, rental spaces, event spaces, or restaurants), the best locations against the respective clusters are displayed, thus helping him decide the venue for his business.

Methodology –

To start the process, the first step is to obtain the names, neighbourhoods and values of the Postal Codes in the city of Vancouver. In this specific case, we use "www.zip-codes.com" to obtain this data.

Reading the Vancouver Postal Code data from zip-codes.com and creating a List of Zip Codes

```
In [2]: from bs4 import BeautifulSoup
        #Reading HTML page
        url='https://www.zip-codes.com/canadian/city.asp?city=vancouver&province=bc'
        response=requests.get(url)
        html = response.text
        page = BeautifulSoup(html,'lxml')
```

```
In [3]: contents=page.find_all('a')
```

```
In [4]: l = []
        for link in page.findAll('a'):
            l.append(link.string)
```

```
In [5]: list_of_VC = []
        for i in range(27,427):
            list_of_VC.append(l[i])
        print(list_of_VC)

        ['V5K 0A1', 'V5K 0A2', 'V5K 0A3', 'V5K 0A4', 'V5K 0A5', 'V5K 0A6', 'V5K 0A7', 'V5K 0A8', 'V5K 0A9', 'V5K 0
        B1', 'V5K 0B2', 'V5K 0B3', 'V5K 0B4', 'V5K 0B5', 'V5K 0B6', 'V5K 0B7', 'V5K 0B8', 'V5K 0B9', 'V5K 0C1', 'V
        5K 0C2', 'V5K 0C3', 'V5K 0C4', 'V5K 0C5', 'V5K 0C6', 'V5K 0C7', 'V5K 0C9', 'V5K 0E1', 'V5K 0E2', 'V5K 0E
        3', 'V5K 0E4', 'V5K 0E5', 'V5K 0E6', 'V5K 0E7', 'V5K 1A1', 'V5K 1A4', 'V5K 1A5', 'V5K 1A6', 'V5K 1A7', 'V5
        K 1A8', 'V5K 1A9', 'V5K 1B1', 'V5K 1B2', 'V5K 1B3', 'V5K 1B4', 'V5K 1B5', 'V5K 1B6', 'V5K 1B7', 'V5K 1B8',
        'V5K 1B9', 'V5K 1C1', 'V5K 1C3', 'V5K 1C4', 'V5K 1C5', 'V5K 1C6', 'V5K 1C7', 'V5K 1C8', 'V5K 1C9', 'V5K 1E
        1', 'V5K 1E2', 'V5K 1E3', 'V5K 1E4', 'V5K 1E5', 'V5K 1E6', 'V5K 1E7', 'V5K 1E8', 'V5K 1E9', 'V5K 1G1', 'V5
        K 1G2', 'V5K 1G3', 'V5K 1G4', 'V5K 1G5', 'V5K 1G6', 'V5K 1G7', 'V5K 1G8', 'V5K 1G9', 'V5K 1H1', 'V5K 1H2',
```

After obtaining the Postal Codes for the city of Vancouver, Canada from "www.zip-codes.com", we need to obtain the latitude and longitude values against each such Postal Area in order for us to get details / specifics about a particular Postal Area from the Foursquare API. For this task, we take the help of the Google Geocoding API. We pass each and every Post Code from the data frame we created to the Google API and obtain the respective co-ordinates.

```
In [11]: for i in list_of_VC:
             #print("-----"+i+"-----")
             url=base_url + i + '&key=' + my_key
             #print(url)
             result= requests.get(url)
             s=result.json()
             neigh=s['results'][0]['address_components'][1]['long_name']
             lat  =s['results'][0]['geometry']['location']['lat']
             lng  =s['results'][0]['geometry']['location']['lng']
             df = df.append({'postalcode': i, 'neighborhood': neigh, 'latitude': lat, 'longitude': lng}, ignore_ind
         ex=True)
```

```
In [12]: df.shape

Out[12]: (400, 4)
```

The next step is to obtain the details/specifics pertaining to each such Postal Area – such as the most popular venues around each location. For this, we pass the latitude and longitude co-ordinates to the Foursquare API and obtain the resulting JSON file.

```
In [15]: def getNearbyVenues(names, latitudes, longitudes, radius=500):

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

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

                 #print(url)

                 # 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)
```

```
In [16]: LIMIT=30
         Vancouver_Venues_List = getNearbyVenues(names=df['postalcode'],
                                          latitudes=df['latitude'],
                                          longitudes=df['longitude']
                                          )
```

```
In [17]: Vancouver_Venues_List.shape
Out[17]: (5445, 7)
```

Now that we have the various venues listed in our data frame, we need to group them by the Postal Areas and take the mean value to find the frequency of each such venue.

In [22]:
```
Vancouver_grouped = Vancouver_onehot.groupby('Postal Areas').mean().reset_index()
Vancouver_grouped
```

Out[22]:

	Postal Areas	American Restaurant	Amphitheater	Asian Restaurant	Athletics & Sports	BBQ Joint	Bakery	Bank	Bar	Beach	...	Sushi Restaurant	Te R
0	V5K 0A1	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0	0.25	...	0.000000	0.
1	V5K 0A2	0.000000	0.000000	0.000000	0.090909	0.000000	0.000000	0.000000	0.0	0.00	...	0.000000	0.
2	V5K 0A3	0.000000	0.000000	0.066667	0.000000	0.000000	0.000000	0.000000	0.0	0.00	...	0.066667	0.
3	V5K 0A4	0.000000	0.000000	0.000000	0.000000	0.000000	0.033333	0.033333	0.0	0.00	...	0.066667	0.

Let’s list the top venues at each postal code area –

In [24]:
```
num_top_venues = 5

for hood in Vancouver_grouped['Postal Areas']:
    print("----"+hood+"----")
    temp = Vancouver_grouped[Vancouver_grouped['Postal Areas'] == hood].T.reset_index()
    temp.columns = ['venue','freq']
    temp = temp.iloc[1:]
    temp['freq'] = temp['freq'].astype(float)
    temp = temp.round({'freq': 2})
    print(temp.sort_values('freq', ascending=False).reset_index(drop=True).head(num_top_venues))
    print('\n')
```

```
----V5K 0A1----
                  venue  freq
0                  Pool  0.25
1                  Pier  0.25
2                  Park  0.25
3                 Beach  0.25
4  Marijuana Dispensary  0.00


----V5K 0A2----
                  venue  freq
0     Convenience Store  0.09
1           Bridal Shop  0.09
2    Italian Restaurant  0.09
```

Now, let’s add the columns in the data frame to list the most popular venues at each Postal Code area.

In [25]:
```python
def return_most_common_venues(row, num_top_venues):
    row_categories = row.iloc[1:]
    row_categories_sorted = row_categories.sort_values(ascending=False)

    return row_categories_sorted.index.values[0:num_top_venues]
```

In [26]:
```python
num_top_venues = 10

indicators = ['st', 'nd', 'rd']

# create columns according to number of top venues
columns = ['Postal Areas']
for ind in np.arange(num_top_venues):
    try:
        columns.append('{}{} Most Common Venue'.format(ind+1, indicators[ind]))
    except:
        columns.append('{}th Most Common Venue'.format(ind+1))

# create a new dataframe
neighborhoods_venues_sorted = pd.DataFrame(columns=columns)
neighborhoods_venues_sorted['Postal Areas'] = Vancouver_grouped['Postal Areas']

for ind in np.arange(Vancouver_grouped.shape[0]):
    neighborhoods_venues_sorted.iloc[ind, 1:] = return_most_common_venues(Vancouver_grouped.iloc[ind, :],
num_top_venues)

neighborhoods_venues_sorted.head()
```

Out[26]:

	Postal Areas	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Con Ven
0	V5K 0A1	Pool	Park	Beach	Pier	Vietnamese Restaurant	Donut Shop	College Gym	Convenience Store	Deli / Bodega	Des Sho
1	V5K 0A2	Coffee Shop	Italian Restaurant	Bus Stop	Bus Station	Bridal Shop	Park	Fast Food Restaurant	Convenience Store	Hotel	Athl Spo
2	V5K 0A3	Soccer Field	Café	Fast Food Restaurant	Chinese Restaurant	Bus Stop	Bus Station	Pub	Restaurant	College Gym	Coff Sho

It's now time to cluster the Postal Areas based on the venues and list the top popular venues in that area on the map of Vancouver city. We accomplish this using the K-Means clustering machine learning algorithm –

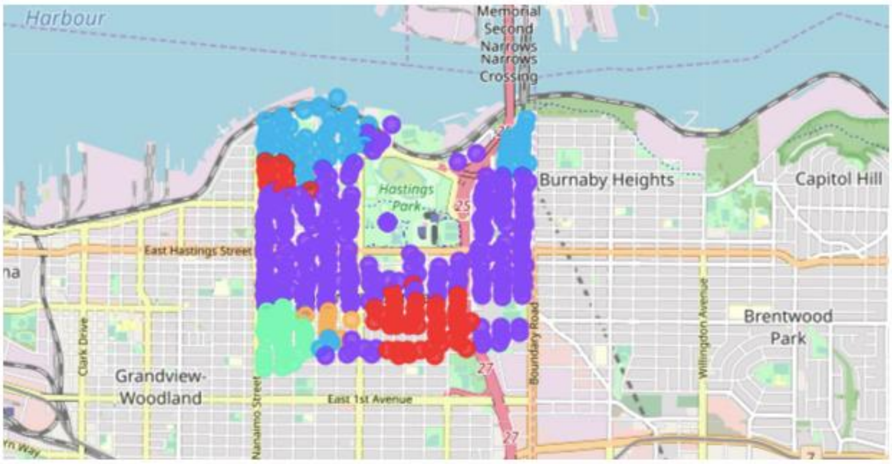
In [27]:
```python
# set number of clusters
kclusters = 5

Vancouver_grouped_clustering = Vancouver_grouped.drop('Postal Areas', 1)

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

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

Out[27]: array([1, 1, 1, 1, 3, 2, 2, 1, 1, 1], dtype=int32)

In [28]:
```python
# add clustering labels

neighborhoods_venues_sorted.insert(0, 'Cluster Labels', kmeans.labels_)

Vancouver_merged = df

# merge toronto_grouped with toronto_data to add latitude/longitude for each neighborhood
Vancouver_merged = df.join(neighborhoods_venues_sorted.set_index('Postal Areas'), on='postalcode')
```

```
address = 'Vancouver, British Columbia, Canada'

geolocator = Nominatim(user_agent="ny_explorer")
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print('The geograpical coordinate of Vancouver are {}, {}.'.format(latitude, longitude))

map_clusters = folium.Map(location=[latitude, longitude], zoom_start=13)

# 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(Vancouver_merged['latitude'], Vancouver_merged['longitude'], Vancouver_m
erged['postalcode'], Vancouver_merged['Cluster Labels']):
    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=0.7).add_to(map_clusters)

map_clusters
```

Out[35]:



So now we are able to plot the most popular venues in the city of Vancouver on the map. However, merely having this data is insufficient to help out businesses who aim to set up offices and expand their business in this city! Therefore, to help them, we first ned to understand the priority of the user. If it's a restaurant business that is trying to expand in this part of the globe, then it needs to know the areas where existing restaurant chains are operating. It can then choose to establish a base of operations nearer to those existing places and in a way aim to draw a portion of their customer base that frequents that area or choose an alternate strategy based on other factors – such as amusement parks, malls, vegetable stores, etc. – to selectively establish their base of operations at places that does not have good restaurant options.

Similarly, if it's an advertising company looking to set foot in Vancouver, a locality that has good Rental Services, Office space availability, Auditoriums to organise seminars and hotels to host parties and events might be a good choice. Since the priorities change with each Business, it is better to know it first.

Hence, we change the model to incorporate the user's priority that will be searched in each cluster, and plotted on the map of the city as follows-

```
In [104]: pr1=input("What is your first priority ? NOTE: Enter the words as in the list - data is case sensitive
!!")
display(pr1)



# create map

address = 'Vancouver, British Columbia, Canada'

geolocator = Nominatim(user_agent="ny_explorer")
location = geolocator.geocode(address)
latitude = location.latitude + 0.02
longitude = location.longitude + 0.08
print('The geograpical coordinate of Vancouver are {}, {}.'.format(latitude, longitude))

map_clusters = folium.Map(location=[latitude, longitude], zoom_start=13)

# 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]


clus=Vancouver_merged.loc[Vancouver_merged['Cluster Labels'] == 0, :]
clusa=clus[clus['1st Most Common Venue']==pr1]
clusb=clus[clus['2nd Most Common Venue']==pr1]
clusc=clus[clus['3rd Most Common Venue']==pr1]

a=clusa.append(clusb.append(clusc))
print("The Cluster 0 Data based on User's Priority -")
print(a[a.columns[[0]+[5]+[6]+[7]+[8]+[9]]])
print("\n")
```

```
# add markers to the map
markers_colors = []
for lat, lon, poi, cluster in zip(a['latitude'], a['longitude'], a['postalcode'], a['Cluster Labels']):
    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=0.7).add_to(map_clusters)
```

In this way, we map the Postal Areas retrieved (based on user's priority) from each cluster and thus gives the Business lead a fair idea about which areas might be prospective targets for them to set up their new base of Operations.

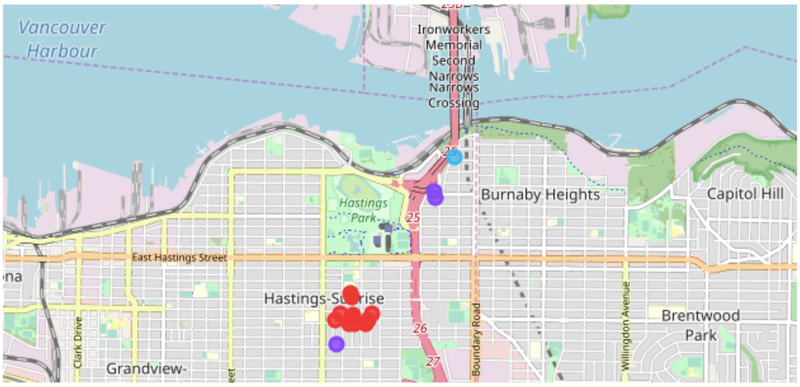
Results -

```
What is your first priority ? NOTE: Enter the words as in the list - data is case sensitive !!Office
'Office'
```

```
The geograpical coordinate of Vancouver are 49.2808724, -123.0339529.
The Cluster 0 Data based on User's Priority -
    postalcode 1st Most Common Venue 2nd Most Common Venue  \
316    V5K 2S4          Pizza Place               Office
265    V5K 2K7          Pizza Place          Event Space
291    V5K 2N6          Pizza Place          Event Space
312    V5K 2R9          Pizza Place          Bridal Shop
313    V5K 2S1          Pizza Place          Bridal Shop
330    V5K 2T9          Pizza Place          Bridal Shop
331    V5K 2V1          Pizza Place          Bridal Shop
332    V5K 2V2          Pizza Place          Bridal Shop
333    V5K 2V3          Pizza Place          Bridal Shop
334    V5K 2V4          Pizza Place          Bridal Shop
335    V5K 2V5          Pizza Place          Bridal Shop

     3rd Most Common Venue  4th Most Common Venue  5th Most Common Venue
316  Portuguese Restaurant              BBQ Joint                   Café
265                 Office               Gun Shop  Portuguese Restaurant
291                 Office               Gun Shop  Portuguese Restaurant
312                 Office  Portuguese Restaurant                   Café
313                 Office  Portuguese Restaurant                   Café
330                 Office  Portuguese Restaurant                   Café
331                 Office  Portuguese Restaurant                   Café
332                 Office  Portuguese Restaurant                   Café
333                 Office  Portuguese Restaurant                   Café
334                 Office                   Park  Portuguese Restaurant
335                 Office                   Park  Portuguese Restaurant


The Cluster 1 Data based on User's Priority -
    postalcode 1st Most Common Venue 2nd Most Common Venue  \
111    V5K 1M2               Tunnel               Office
184    V5K 1Y3               Tunnel               Office
386    V5K 3B3             Gun Shop               Office

    3rd Most Common Venue 4th Most Common Venue 5th Most Common Venue
111                   Bar                  Road               Dog Run
184                   Bar                  Farm         Deli / Bodega
386             BBQ Joint                Bakery                  Café
```

```
The Cluster 2 Data based on User's Priority -
   postalcode 1st Most Common Venue 2nd Most Common Venue  \
69    V5K 1G4                  Park                 Trail

   3rd Most Common Venue  4th Most Common Venue 5th Most Common Venue
69                Office  Vietnamese Restaurant           Event Space


The Cluster 3 Data based on User's Priority -
Empty DataFrame
Columns: [postalcode, 1st Most Common Venue, 2nd Most Common Venue, 3rd Most Common Venue, 4th Most Common
Venue, 5th Most Common Venue]
Index: []


The Cluster 4 Data based on User's Priority -
Empty DataFrame
Columns: [postalcode, 1st Most Common Venue, 2nd Most Common Venue, 3rd Most Common Venue, 4th Most Common
Venue, 5th Most Common Venue]
Index: []
```

Out[107]:



In the above example, the user entered "Office" as his priority, and the system returned all those Postal Areas where Office is in the 1^{st}, 2^{nd} and 3^{rd} most popular venues in all the clusters (0,1,2,3 and 4). A map is plotted with the resulting data and the client is now able to visualize the locations having the most popular Office destinations.

<u>Discussions</u> -

Also note that along with this information, two (02) additional columns as the 4^{th} and 5^{th} most popular destinations have been shown to acquaint users with other popular facilities in that Postal Area and help them in taking a wise choice. For example, in the above result, Office spaces in areas having a Café might be a wiser choice in order to let employees have some occasional relaxation amidst their workload. Similarly, an ice cream / dessert shop adjoining to a restaurant without dessert options might make it a happening spot.

<u>Conclusion</u> -

Doing business is a complicated task and more complex is the process to identify prospective locations to expand the business. An entire venture may be ruined if a poor selection is made as a choice for the base of operations. Hence, pre analysing prospective locations has been a hotbed of debate and research for decades. Any sound help in this area will be welcomed by any prospective start-up or an existing business looking to expand. Although we did this exercise based on the data pertaining to the city of Vancouver, Canada, this methodology may be applied for almost any city in the world as long as we are able to fetch the Foursquare data for that location.

P.S. - I do not claim to be an expert in the field and this model may be improved in a myriad of ways. As such, if the reader identifies any loop holes and suggests any improvement, it will help me become a better analyser. Please feel free to mail your valuable feedback at – arunabeshc@gmail.com

Acknowledgements –

1) www.zip-codes.com

2) Google Geocoding API

3) Foursquare API

4) Coursera Teaching Staff