Python for Data Science: Final Project

In this project, an open dataset from the site Crime in Vancouver (https://www.kaggle.com/wosaku/crime-in-vancouver) is being used.

Exploratory data analysis

Preliminary explore the records (rows) and fields (columns)

In [1]:
```
import math
import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns

from collections import Counter

import folium
from folium import Circle, Marker
from folium.plugins import HeatMap, MarkerCluster

from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
```

In [2]:
```
data_raw = pd.read_csv('./final_project_data/1929_6405_bundle_archive/crime.csv')
print(data_raw.shape)

data = data_raw.dropna()
print(len(data))

data.head(2)
```

```
(530652, 12)
474015
```

Out[2]:

	TYPE	YEAR	MONTH	DAY	HOUR	MINUTE	HUNDRED_BLOCK	NEIGHBOURHOOD	X	Y	Latitude	Longitude
0	Other Theft	2003	5	12	16.0	15.0	9XX TERMINAL AVE	Strathcona	493906.5	5457452.47	49.269802	-123.083763
1	Other Theft	2003	5	7	15.0	20.0	9XX TERMINAL AVE	Strathcona	493906.5	5457452.47	49.269802	-123.083763

In [3]:
```
# all type of crimes
print(len(data.TYPE.unique()))
Counter(data.TYPE)
```

```
9
```

Out[3]:
```
Counter({'Other Theft': 52160,
         'Break and Enter Residential/Other': 60856,
         'Mischief': 70157,
         'Break and Enter Commercial': 33841,
         'Theft from Vehicle': 170889,
         'Vehicle Collision or Pedestrian Struck (with Injury)': 21887,
         'Vehicle Collision or Pedestrian Struck (with Fatality)': 254,
         'Theft of Vehicle': 38351,
         'Theft of Bicycle': 25620})
```

In [4]:
```
# number of records containing vehicle collision and pedestrian struck
data[data.TYPE.str.contains('Vehicle Collision or Pedestrian Struck')].shape
```

Out[4]: (22141, 12)

In [5]:
```
# all neighbourhoods
print(len(data.NEIGHBOURHOOD.unique()))
Counter(data.NEIGHBOURHOOD)
```

```
24
```

Out[5]:
```
Counter({'Strathcona': 20917,
         'Kerrisdale': 7447,
         'Dunbar-Southlands': 7746,
         'Grandview-Woodland': 27180,
         'Sunset': 17395,
         'West End': 41352,
         'Central Business District': 110945,
         'Hastings-Sunrise': 18126,
         'Victoria-Fraserview': 10818,
         'Fairview': 32161,
         'Kensington-Cedar Cottage': 24941,
         'West Point Grey': 5870,
         'Shaughnessy': 5426,
         'Renfrew-Collingwood': 26761,
         'Killarney': 10475,
         'Riley Park': 12520,
         'Arbutus Ridge': 6066,
         'Musqueam': 532,
         'Mount Pleasant': 30534,
         'Kitsilano': 26698,
         'Stanley Park': 3775,
         'South Cambie': 5212,
         'Marpole': 13083,
         'Oakridge': 8035})
```

In [6]:
```
# unique street block of all crime incidents
len(data.HUNDRED_BLOCK.unique())
```

Out[6]: 21192

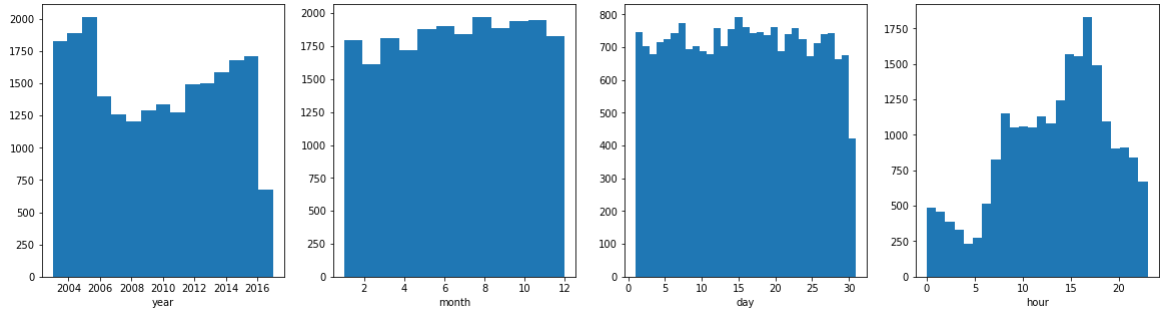
In [7]:
```
# unique street block of vehicle collision and pedestrian struck
len(data[data.TYPE.str.contains('Vehicle Collision or Pedestrian Struck')].HUNDRED_BLOCK.unique())
```

Out[7]: 7411

Explore patterns of vehicle collision and pedestrian struck relative to year, month, day, hour

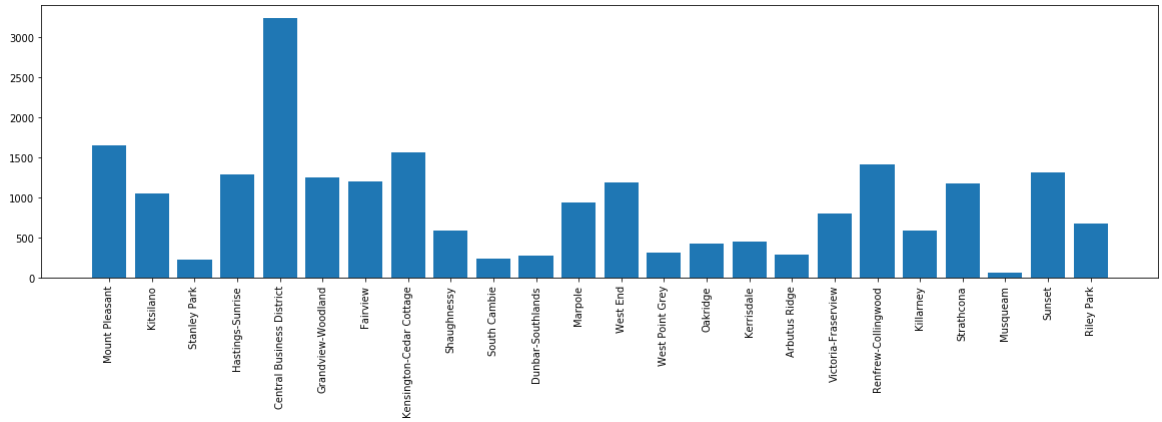
In [8]:
```
# from now on, focus only on the collision with fatality or injury
data = data[data.TYPE.str.contains('Vehicle Collision or Pedestrian Struck')]

fig, axes = plt.subplots(nrows = 1, ncols = 4, figsize = (20, 5))

axes[0].hist(data.YEAR, bins = 15)
axes[0].set_xlabel('year')

axes[1].hist(data.MONTH, bins = 12)
axes[1].set_xlabel('month')

axes[2].hist(data.DAY, bins = 31)
axes[2].set_xlabel('day')

axes[3].hist(data.HOUR, bins = 24)
axes[3].set_xlabel('hour')

fig.show()
```

```
/home/ornwipa/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.py:18: UserWarning: Matplotlib is currently using module://ipykernel.p
ylab.backend_inline, which is a non-GUI backend, so cannot show the figure.
```



In [9]:
```
# how many incidents in each neighbourhood
print(type(Counter(data.NEIGHBOURHOOD)))
print(Counter(data.NEIGHBOURHOOD).keys())
print(Counter(data.NEIGHBOURHOOD).values())

plt.figure(figsize = (20,5))
plt.bar(Counter(data.NEIGHBOURHOOD).keys(), Counter(data.NEIGHBOURHOOD).values())
plt.xticks(rotation = 90)
plt.show()
```

```
<class 'collections.Counter'>
dict_keys(['Mount Pleasant', 'Kitsilano', 'Stanley Park', 'Hastings-Sunrise', 'Central Business District', 'Grandview-Woodland', 'Fairview',
'Kensington-Cedar Cottage', 'Shaughnessy', 'South Cambie', 'Dunbar-Southlands', 'Marpole', 'West End', 'West Point Grey', 'Oakridge', 'Kerris
dale', 'Arbutus Ridge', 'Victoria-Fraserview', 'Renfrew-Collingwood', 'Killarney', 'Strathcona', 'Musqueam', 'Sunset', 'Riley Park'])
dict_values([1642, 1053, 223, 1288, 3229, 1249, 1194, 1555, 586, 239, 272, 941, 1190, 309, 420, 450, 288, 796, 1413, 582, 1174, 60, 1313, 67
5])
```



In [10]:
```
# focus on the rush hour from 6 am to 6 pm, every day, month, year and street
data = data[data.HOUR.isin(range(8,18))]
data.shape
```

Out[10]: (12726, 12)

Create interactive maps

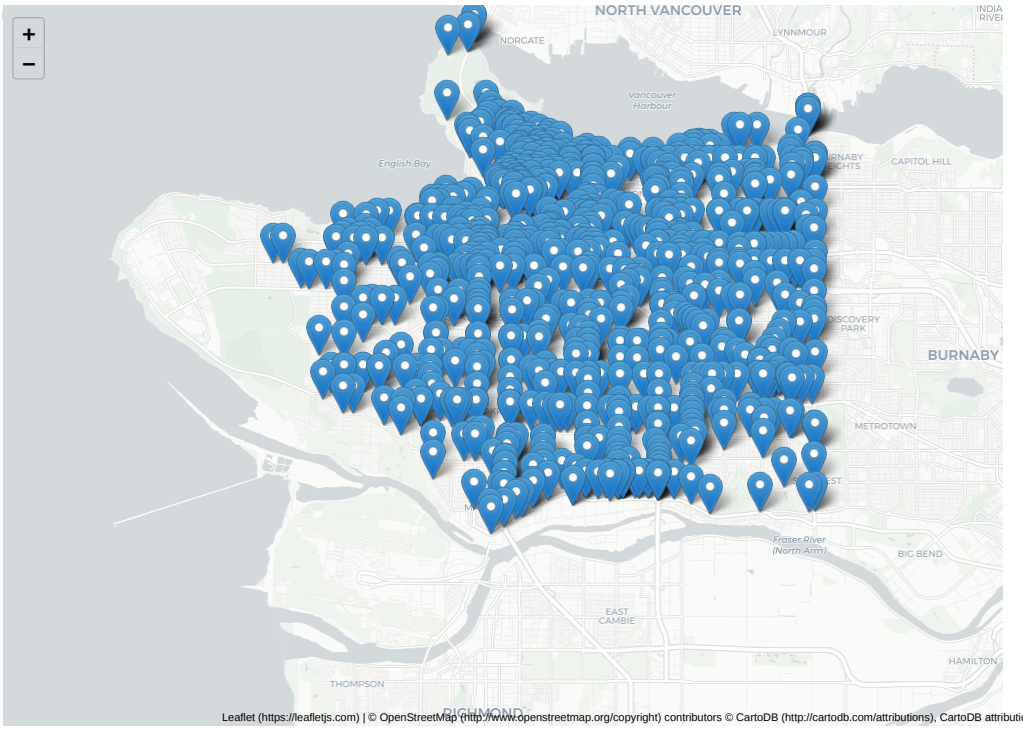
The original data set contains coordinates in UTM Zone 10 (columns X and Y) as well as the Latitude and Longitude.

The maps represent only the latest full year (2016) for simplicity.

In [11]:
```
print(len(data[data.YEAR == 2016]))

m_2 = folium.Map(location = [49.255707, -123.135152], tiles = 'cartodbpositron', zoom_start = 12)

for idx, row in data[data.YEAR == 2016].iterrows():
    Marker([row['Latitude'], row['Longitude']]).add_to(m_2)

m_2
# simple marker might not well indicate the cluster of the location as ones are on top of the others
```

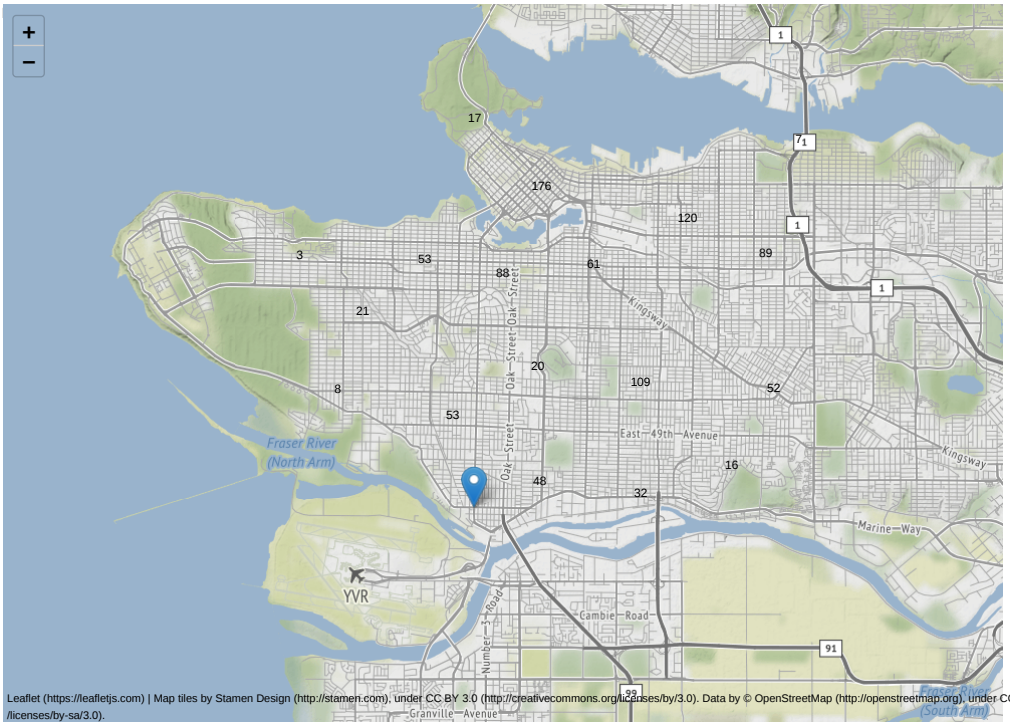
974

Out[11]:



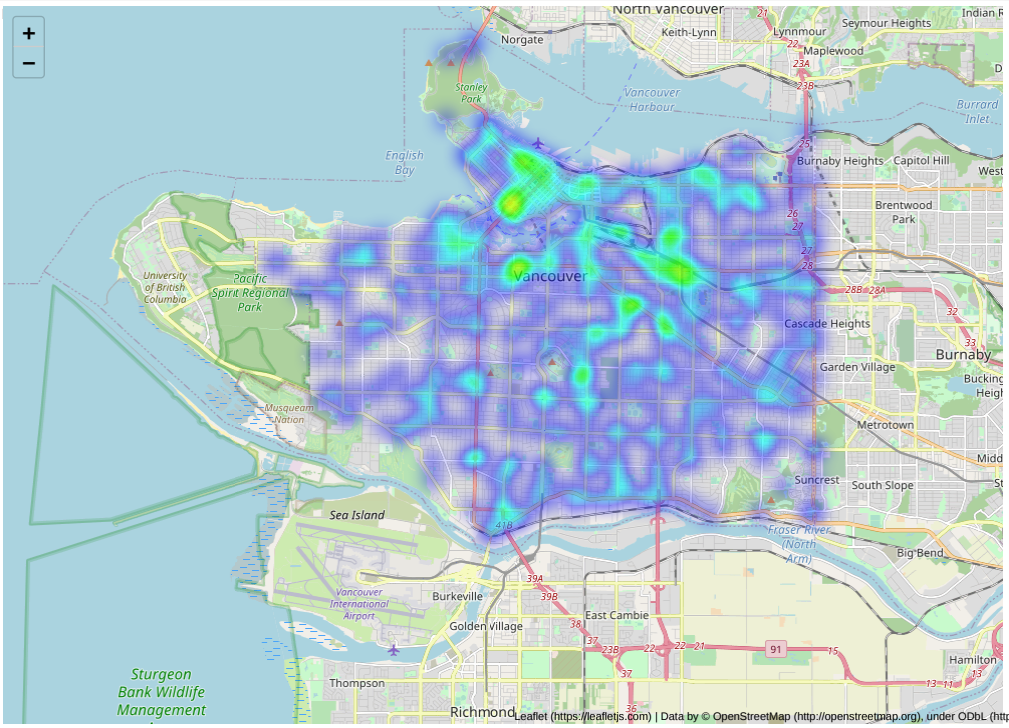
In [12]:
```
m_3 = folium.Map(location = [49.255707, -123.135152], tiles = 'stamenterrain', zoom_start = 12)

mc = MarkerCluster()
for idx, row in data[data.YEAR == 2016].iterrows():
    if not math.isnan(row['Latitude']) and not math.isnan(row['Longitude']):
        mc.add_child(Marker([row['Latitude'], row['Longitude']]))
m_3.add_child(mc)

m_3
```

Out[12]:



In [13]:
```
m_5 = folium.Map(location = [49.255707, -123.135152], tiles = 'openstreetmap', zoom_start = 12)

HeatMap(data = data[data.YEAR == 2016][['Latitude', 'Longitude']], radius=10).add_to(m_5)

m_5
# heat map indicates some dangerous intersections
```

Out[13]:



Unsupervised machine learning for hotspot analysis

Scale latitude and longitude features

In [14]:
```
print(data.columns)
features = ['Latitude', 'Longitude']
unscaled_features = data[features]
unscaled_features[-2:]
```

```
Index(['TYPE', 'YEAR', 'MONTH', 'DAY', 'HOUR', 'MINUTE', 'HUNDRED_BLOCK',
       'NEIGHBOURHOOD', 'X', 'Y', 'Latitude', 'Longitude'],
      dtype='object')
```

Out[14]:

	Latitude	Longitude
530638	49.262208	-123.112923
530651	49.215706	-123.134512

In [15]:
```
X = StandardScaler().fit_transform(unscaled_features)
X[-2:]
```

Out[15]:
```
array([[ 0.23648504, -0.30041215],
       [-1.72157896, -0.83507895]])
```

Train the model with K-means clustering

In [16]:
```
kmeans = KMeans(n_clusters = 7)
model = kmeans.fit(X)
print("model\n", model)
print(model.labels_)
print(model.labels_.shape)
print((type(model.labels_)))
```

```
model
 KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
       n_clusters=7, n_init=10, n_jobs=None, precompute_distances='auto',
       random_state=None, tol=0.0001, verbose=0)
[2 0 0 ... 2 6 1]
(12726,)
<class 'numpy.ndarray'>
```

In [17]:
```
# show the location of all clusters
centers = model.cluster_centers_
centers
```

Out[17]:
```
array([[ 0.6551013 ,  1.13512643],
       [-1.40565528, -0.86735859],
       [ 1.09949512, -0.41616711],
       [-0.88038261,  1.28146152],
       [-1.50228308,  0.38455362],
       [ 0.15962936, -1.45782746],
       [ 0.13362391, -0.04502364]])
```

In [18]:
```
# how many data points each cluster
Counter(model.labels_)
```

Out[18]:
```
Counter({2: 2707, 0: 2312, 6: 1969, 5: 1711, 4: 1208, 1: 1391, 3: 1428})
```

In [19]:
```
cluster = pd.DataFrame(model.labels_)
cluster
```

Out[19]:

	0
0	2
1	0
2	0
3	2
4	6
...	...
12721	2
12722	4
12723	2
12724	6
12725	1

12726 rows × 1 columns

In [20]:
```
data
```

Out[20]:

	TYPE	YEAR	MONTH	DAY	HOUR	MINUTE	HUNDRED_BLOCK	NEIGHBOURHOOD	X	Y	Latitude	Longitude
501	Vehicle Collision or Pedestrian Struck (with I...	2003	7	4	14.0	8.0	X STANLEY PARK CSWY	Stanley Park	490050.59	5460277.25	49.295161	-123.136838
511	Vehicle Collision or Pedestrian Struck (with I...	2003	8	14	14.0	52.0	X HWY / E 1ST AV	Hastings-Sunrise	497925.00	5457402.00	49.269375	-123.028523
512	Vehicle Collision or Pedestrian Struck (with I...	2003	7	23	16.0	0.0	X HWY / E 1ST AV	Hastings-Sunrise	497925.00	5457402.00	49.269375	-123.028523
514	Vehicle Collision or Pedestrian Struck (with I...	2003	2	15	14.0	40.0	X EXPO BLVD	Central Business District	492342.00	5458331.00	49.277687	-123.105286
516	Vehicle Collision or Pedestrian Struck (with I...	2003	4	1	15.0	11.0	X BLOCK W 1ST AVE	Mount Pleasant	492318.00	5457475.00	49.269987	-123.105600
...	...	...	...	...	...	...	...	...	...	...	...	...
530577	Vehicle Collision or Pedestrian Struck (with I...	2017	3	17	17.0	31.0	DUNSMUIR ST / CAMBIE ST	Central Business District	491886.00	5458643.00	49.280488	-123.111562
530583	Vehicle Collision or Pedestrian Struck (with I...	2017	1	4	13.0	30.0	X SE MARINE DR	Sunset	492266.00	5451052.00	49.212210	-123.106191
530594	Vehicle Collision or Pedestrian Struck (with I...	2017	5	4	14.0	24.0	DUNSMUIR ST / RICHARDS ST	Central Business District	491675.00	5458852.00	49.282365	-123.114467
530638	Vehicle Collision or Pedestrian Struck (with I...	2017	7	10	9.0	20.0	YUKON ST / W 10TH AVE	Mount Pleasant	491784.00	5456611.00	49.262208	-123.112923
530651	Vehicle Collision or Pedestrian Struck (with I...	2017	6	6	17.0	38.0	13XX BLOCK PARK DR	Marpole	490204.00	5451444.00	49.215706	-123.134512

12726 rows × 12 columns

In [21]:
```
new_index = np.array([x for x in range(0,len(cluster))])
data = data.reset_index()
data
```

Out[21]:

	index	TYPE	YEAR	MONTH	DAY	HOUR	MINUTE	HUNDRED_BLOCK	NEIGHBOURHOOD	X	Y	Latitude	Longitude
0	501	Vehicle Collision or Pedestrian Struck (with I...	2003	7	4	14.0	8.0	X STANLEY PARK CSWY	Stanley Park	490050.59	5460277.25	49.295161	-123.136838
1	511	Vehicle Collision or Pedestrian Struck (with I...	2003	8	14	14.0	52.0	X HWY / E 1ST AV	Hastings-Sunrise	497925.00	5457402.00	49.269375	-123.028523
2	512	Vehicle Collision or Pedestrian Struck (with I...	2003	7	23	16.0	0.0	X HWY / E 1ST AV	Hastings-Sunrise	497925.00	5457402.00	49.269375	-123.028523
3	514	Vehicle Collision or Pedestrian Struck (with I...	2003	2	15	14.0	40.0	X EXPO BLVD	Central Business District	492342.00	5458331.00	49.277687	-123.105286
4	516	Vehicle Collision or Pedestrian Struck (with I...	2003	4	1	15.0	11.0	X BLOCK W 1ST AVE	Mount Pleasant	492318.00	5457475.00	49.269987	-123.105600
...	...	...	...	...	...	...	...	...	...	...	...	...	...
12721	530577	Vehicle Collision or Pedestrian Struck (with I...	2017	3	17	17.0	31.0	DUNSMUIR ST / CAMBIE ST	Central Business District	491886.00	5458643.00	49.280488	-123.111562
12722	530583	Vehicle Collision or Pedestrian Struck (with I...	2017	1	4	13.0	30.0	X SE MARINE DR	Sunset	492266.00	5451052.00	49.212210	-123.106191
12723	530594	Vehicle Collision or Pedestrian Struck (with I...	2017	5	4	14.0	24.0	DUNSMUIR ST / RICHARDS ST	Central Business District	491675.00	5458852.00	49.282365	-123.114467
12724	530638	Vehicle Collision or Pedestrian Struck (with I...	2017	7	10	9.0	20.0	YUKON ST / W 10TH AVE	Mount Pleasant	491784.00	5456611.00	49.262208	-123.112923
12725	530651	Vehicle Collision or Pedestrian Struck (with I...	2017	6	6	17.0	38.0	13XX BLOCK PARK DR	Marpole	490204.00	5451444.00	49.215706	-123.134512

12726 rows × 13 columns

In [22]:
```
# data frames of features and cluster can be joined
data_with_cluster = data.join(cluster)
data_with_cluster = data_with_cluster.rename(columns={0: 'cluster'})
data_with_cluster[-2:]
```

Out[22]:

	index	TYPE	YEAR	MONTH	DAY	HOUR	MINUTE	HUNDRED_BLOCK	NEIGHBOURHOOD	X	Y	Latitude	Longitude	cluster
12724	530638	Vehicle Collision or Pedestrian Struck (with I...	2017	7	10	9.0	20.0	YUKON ST / W 10TH AVE	Mount Pleasant	491784.0	5456611.0	49.262208	-123.112923	6
12725	530651	Vehicle Collision or Pedestrian Struck (with I...	2017	6	6	17.0	38.0	13XX BLOCK PARK DR	Marpole	490204.0	5451444.0	49.215706	-123.134512	1

In [23]:
```
# map out only the year 2016
clustered_data = data_with_cluster[data_with_cluster.YEAR == 2016].reindex()
clustered_data[:2]
```

Out[23]:

	index	TYPE	YEAR	MONTH	DAY	HOUR	MINUTE	HUNDRED_BLOCK	NEIGHBOURHOOD	X	Y	Latitude	Longitude	cluster
11377	495270	Vehicle Collision or Pedestrian Struck (with F...	2016	5	17	16.0	53.0	85XX STANLEY PARK DR	Stanley Park	489106.0	5460343.0	49.295736	-123.149831	2
11378	495286	Vehicle Collision or Pedestrian Struck (with F...	2016	3	31	11.0	20.0	RUPERT ST / E BROADWAY AVE	Renfrew-Collingwood	497539.0	5456571.0	49.261898	-123.033824	0

In [24]:
```
m_4 = folium.Map(location = [49.255707, -123.135152], tiles = 'stamentoner', zoom_start = 12)

def color_producer(val):
    if val == 0:
        return 'purple'
    elif val == 1:
        return 'forestgreen'
    elif val == 2:
        return 'mediumblue'
    elif val == 3:
        return 'teal'
    elif val == 4:
        return 'orange'
    elif val == 5:
        return 'fuchsia'
    else:
        return 'firebrick'

for i in range(0, len(clustered_data) ):
    Circle(location = [ clustered_data.iloc[i]['Latitude'], clustered_data.iloc[i]['Longitude'] ], radius = 20,
        color = color_producer(clustered_data.iloc[i]['cluster'])).add_to(m_4)

m_4
```

Out[24]: