1) Introduction: Business Problem

The aim of this project is to find a safe and secure location for opening of commercial establishments in Vancouver, Canada. Specifically, this report will be targeted to stakeholders interested in opening any business place like **Grocery Store** in **Vancouver City**, Canada.

The first task would be to **choose the safest borough** by analysing crime data for opening a grocery store and **short listing a neighbourhood**, where grocery stores are not amongst the most commom venues, and yet **as close to the city as possible**.

We will make use of our data science tools to analyse data and focus on the safest borough and explore its neighborhoods and the 10 most common venues in each neighborhood so that the best neighborhood where grocery store is not amongst the most common venue can be selected.

2) Data

Based on definition of our problem, factors that will influence our decission are:

- finding the safest borough based on crime statistics
- finding the most common venues
- choosing the right neighbourhood within the borough

We will be using the geographical coordinates of Vancouver to plot neighbourhoods in a borough that is safe and in the city's vicinity, and finally cluster our neighborhoods and present our findings.

Following data sources will be needed to extract/generate the required information:

- **Part 1**: Using a real world data set from Kaggle containing the Vancouver Crimes from 2003 to 2019: A dataset consisting of the crime statistics of each Neighbourhoof in Vancouver along with type of crime, recorded year, month and hour.
- **Part 2**: Gathering additional information of the list of officially categorized boroughs in Vancouver from Wikipedia.: Borough information will be used to map the existing data where each neighbourhood can be assigned with the right borough.
- **Part 3**: Creating a new consolidated dataset of the Neighborhoods, along with their boroughs, crime data and the respective Neighbourhood's co-ordinates.: This data will be fetched using OpenCage Geocoder to find the safest borough and explore the neighbourhood by plotting it on maps using Folium and perform exploratory data analysis.
- **Part 4**: Creating a new consolidated dataset of the Neighborhoods, boroughs, and the most common venues and the respective Neighbourhood along with co-ordinates.: This data will be fetched using Four Square API to explore the neighbourhood venues and to apply machine learning algorithm to cluster the neighbourhoods and present the findings by plotting it on maps using Folium.

Part 1: Using a real world data set from Kaggle containing the Vancouver Crimes from 2003 to 2019

Vancouver Crime Report

Properties of the Crime Report

- TYPE - Crime type
- YEAR - Recorded year
- MONTH - Recorded month
- DAY - Recorded day
- HOUR - Recorded hour
- MINUTE - Recorded minute
- HUNDRED_BLOCK - Recorded block
- NEIGHBOURHOOD - Recorded neighborhood
- X - GPS longtitude
- Y - GPS latitude

Data set URL: https://www.kaggle.com/agilesifaka/vancouver-crime-report/version/2 (https://www.kaggle.com/agilesifaka/vancouver-crime-report/version/2)

import numpy as np import pandas as pd #Command to install OpenCage Geocoder for fetching Lat and Lng of Neighborhood !pip install opencage #Importing OpenCage Geocoder from opencage.geocoder import OpenCageGeocode # use the inline backend to generate the plots within the browser %matplotlib inline #Importing Matplot lib and associated packages to perform Data Visualisation and Exploratory Data Analysis import matplotlib as mpl import matplotlib.pyplot as plt mpl.style.use('ggplot') # optional: for ggplot-like style # check for latest version of Matplotlib print ('Matplotlib version: ', mpl.__version__) # >= 2.0.0 # Matplotlib and associated plotting modules import matplotlib.cm as cm import matplotlib.colors as colors #Importing folium to visualise Maps and plot based on Lat and Lng import folium #Requests to request web pages by making get requests to FourSquare REST Client import requests #To normalise data returned by FourSquare API from pandas.io.json import json_normalize #Importing KMeans from SciKit library to Classify neighborhoods into clusters from sklearn.cluster import KMeans print('Libraries imported')

vnc_crime_df = pd.read_csv('https://raw.githubusercontent.com/RamanujaSVL/Coursera_Capstone/master/vancouver_crime_reco (https://raw.githubusercontent.com/RamanujaSVL/Coursera_Capstone/master/vancouver_crime_records_2018.cs index_col=None)

Dropping X,Y which represents Lat, Lng data as Coordinates, the data seems to be corrupt

vnc_crime_df.drop(['Unnamed: 0','MINUTE', 'HUNDRED_BLOCK', 'X', 'Y'], axis = 1, inplace = True)

vnc_crime_df.columns

vnc_crime_df.head()

In [26]:
```
vnc_crime_df.columns = ['Type', 'Year','Month','Day','Hour','Neighbourhood']
vnc_crime_df.head()
```

Out[26]:

	Type	Year	Month	Day	Hour	Neighbourhood
0	Break and Enter Commercial	2018	3	2	6	West End
1	Break and Enter Commercial	2018	6	16	18	West End
2	Break and Enter Commercial	2018	12	12	0	West End
3	Break and Enter Commercial	2018	4	9	6	Central Business District
4	Break and Enter Commercial	2018	10	2	18	Central Business District

In [27]:
```
vnc_crime_df['Neighbourhood'].value_counts()
```

Out[27]:
```
Central Business District    10857
West End                      3031
Mount Pleasant                2396
Strathcona                    1987
Kitsilano                     1802
Fairview                      1795
Renfrew-Collingwood           1762
Grandview-Woodland            1761
Kensington-Cedar Cottage      1391
Hastings-Sunrise              1270
Sunset                         967
Riley Park                     866
Marpole                        828
Victoria-Fraserview            600
Killarney                      565
Oakridge                       499
Dunbar-Southlands              474
Kerrisdale                     417
Shaughnessy                    414
West Point Grey                372
Arbutus Ridge                  311
South Cambie                   292
Stanley Park                   154
Musqueam                        17
Name: Neighbourhood, dtype: int64
```

Part 2: Gathering additional information about the Neighborhood from Wikipedia

As part of data set Borough which the neighborhood was part of was not categorized, so we will create a dictionary of Neighborhood and based on data in the following [Wikipedia page (https://en.wikipedia.org/wiki/List_of_neighbourhoods_in_Vancouver)](https://en.wikipedia.org/wiki/List_of_neighbourhoods_in_Vancouver).

In [28]:
```
# define the dataframe columns
column_names = ['Neighbourhood', 'Borough']

# instantiate the dataframe
vnc_neigh_bor = pd.DataFrame(columns=column_names)

vnc_neigh_bor['Neighbourhood'] = vnc_crime_df['Neighbourhood'].unique()

neigh_bor_dict = {'Central Business District':'Central', 'West End':'Central',
'Stanley Park':'Central', 'Victoria-Fraserview':'South Vancouver',
                  'Killarney':'South Vancouver', 'Musqueam':'South Vancouver',
'Mount Pleasant':'East Side', 'Strathcona':'East Side',
                  'Renfrew-Collingwood':'East Side', 'Grandview-Woodland':'Eas
t Side', 'Kensington-Cedar Cottage':'East Side', 'Hastings-Sunrise':'East Sid
e',
                  'Sunset':'East Side', 'Riley Park':'East Side', 'Kitsilano':
'West Side', 'Fairview':'West Side',
                  'Marpole':'West Side', 'Oakridge':'West Side', 'Dunbar-South
lands':'West Side', 'Kerrisdale':'West Side',
                  'Shaughnessy':'West Side', 'West Point Grey':'West Side', 'A
rbutus Ridge':'West Side', 'South Cambie':'West Side'}

for row, neigh in zip(neigh_bor_dict, vnc_neigh_bor['Neighbourhood']):
  vnc_neigh_bor.loc[vnc_neigh_bor.Neighbourhood == row, 'Borough'] = neigh_bor
_dict.get(row)

vnc_neigh_bor.dropna(inplace=True)

print("Total Neighbourhood Count",len(vnc_neigh_bor['Neighbourhood']),"Borough
Count",len(vnc_neigh_bor['Borough'].unique()))

vnc_neigh_bor.head()
```

```
Total Neighbourhood Count 24 Borough Count 4
```

Out[28]:

	Neighbourhood	Borough
0	West End	Central
1	Central Business District	Central
2	Hastings-Sunrise	East Side
3	Grandview-Woodland	East Side
4	Mount Pleasant	East Side

In [29]:
```
vnc_boroughs_crime = pd.merge(vnc_crime_df,vnc_neigh_bor, on='Neighbourhood')

vnc_boroughs_crime.head()
```

Out[29]:

	Type	Year	Month	Day	Hour	Neighbourhood	Borough
0	Break and Enter Commercial	2018	3	2	6	West End	Central
1	Break and Enter Commercial	2018	6	16	18	West End	Central
2	Break and Enter Commercial	2018	12	12	0	West End	Central
3	Break and Enter Commercial	2018	3	2	3	West End	Central
4	Break and Enter Commercial	2018	3	17	11	West End	Central

In [30]:
```
vnc_boroughs_crime.dropna(inplace=True)
vnc_boroughs_crime['Borough'].value_counts()
```

Out[30]:
```
Central             14042
East Side           12400
West Side            7204
South Vancouver      1182
Name: Borough, dtype: int64
```

3) Methodology

Categorized the methodologysection into two parts:

- **Exploratory Data Analysis:** Visualise the crime repots in different Vancouver boroughs to idenity the safest borough and normalise the neighborhoods of that borough. We will Use the resulting data and find 10 most common venues in each neighborhood.

- **Modelling:** To help stakeholders choose the right neighborhood within a borough we will be clustering similar neighborhoods using K - means clustering which is a form of unsupervised machine learning algorithm that clusters data based on predefined cluster size. We will use K-Means clustering to address this problem so as to group data based on existing venues which will help in the decision making process.

In [31]:
```
vnc_crime_cat = pd.pivot_table(vnc_boroughs_crime,
                               values=['Year'],
                               index=['Borough'],
                               columns=['Type'],
                               aggfunc=len,
                               fill_value=0,
                               margins=True)
vnc_crime_cat
```

Out[31]:

	Year							
Type	Break and Enter Commercial	Break and Enter Residential/Other	Mischief	Other Theft	Theft from Vehicle	Theft of Bicycle	Theft of Vehicle	Vehicle Collision or Pedestrian Struck (with Fatality)
Borough								
Central	787	198	2280	2489	6871	857	245	1
East Side	786	1043	2192	1674	4754	678	605	8
South Vancouver	49	156	187	88	483	36	71	1
West Side	403	1000	1062	696	2838	588	225	3
All	2025	2397	5721	4947	14946	2159	1146	13

In [32]:
```
vnc_crime_cat.reset_index(inplace = True)
vnc_crime_cat.columns = vnc_crime_cat.columns.map(''.join)
vnc_crime_cat.rename(columns={'YearAll':'Total'}, inplace=True)
# To ignore bottom All in Borough
vnc_crime_cat = vnc_crime_cat.head(4)
vnc_crime_cat
```

Out[32]:

	Borough	YearBreak and Enter Commercial	YearBreak and Enter Residential/Other	YearMischief	YearOther Theft	YearTheft from Vehicle	YearTheft of Bicycle	Year Ve
0	Central	787	198	2280	2489	6871	857	
1	East Side	786	1043	2192	1674	4754	678	
2	South Vancouver	49	156	187	88	483	36	
3	West Side	403	1000	1062	696	2838	588	

In [33]:
```
vnc_crime_neigh = pd.pivot_table(vnc_boroughs_crime,
                               values=['Year'],
                               index=['Neighbourhood'],
                               columns=['Type'],
                               aggfunc=len,
                               fill_value=0,
                               margins=True)
vnc_crime_neigh
```

Out[33]:

	Year							
Type	**Break and Enter Commercial**	**Break and Enter Residential/Other**	**Mischief**	**Other Theft**	**Theft from Vehicle**	**Theft of Bicycle**	**Theft of Vehicle**	**Vehicl Collisi or Pedes Struck (with Fatalit**
Neighbourhood								
Arbutus Ridge	12	78	49	18	111	12	12	
Central Business District	551	124	1812	2034	5301	640	165	
Dunbar-Southlands	8	106	81	31	199	16	9	
Fairview	138	73	233	297	692	245	55	
Grandview-Woodland	148	162	304	215	634	110	123	
Hastings-Sunrise	48	117	195	107	607	52	74	
Kensington-Cedar Cottage	62	145	255	148	541	69	71	
Kerrisdale	24	97	49	9	172	13	11	
Killarney	34	72	90	31	240	19	33	
Kitsilano	106	165	320	154	755	189	51	
Marpole	44	125	134	75	290	34	39	
Mount Pleasant	205	124	353	493	822	232	67	
Musqueam	0	4	3	0	4	2	2	
Oakridge	19	123	64	63	164	18	18	
Renfrew-Collingwood	91	156	243	472	569	37	92	
Riley Park	35	122	140	53	378	52	39	
Shaughnessy	12	120	41	0	187	10	11	
South Cambie	22	42	41	38	111	19	8	
Stanley Park	6	2	8	0	109	14	3	
Strathcona	160	124	527	81	821	108	76	
Sunset	37	93	175	105	382	18	63	
Victoria-Fraserview	15	80	94	57	239	15	36	
West End	230	72	460	455	1461	203	77	
West Point Grey	18	71	50	11	157	32	11	
All	2025	2397	5721	4947	14946	2159	1146	

In [34]:
```
vnc_crime_neigh.reset_index(inplace = True)
vnc_crime_neigh.columns = vnc_crime_neigh.columns.map(''.join)
vnc_crime_neigh.rename(columns={'YearAll':'Total'}, inplace=True)

vnc_crime_neigh.head()
```

Out[34]:

	Neighbourhood	YearBreak and Enter Commercial	YearBreak and Enter Residential/Other	YearMischief	YearOther Theft	YearTheft from Vehicle	YearTheft of Bicycle
0	Arbutus Ridge	12	78	49	18	111	12
1	Central Business District	551	124	1812	2034	5301	640
2	Dunbar-Southlands	8	106	81	31	199	16
3	Fairview	138	73	233	297	692	245
4	Grandview-Woodland	148	162	304	215	634	110

In [35]:
```
vnc_crime_cat.describe()
```

Out[35]:

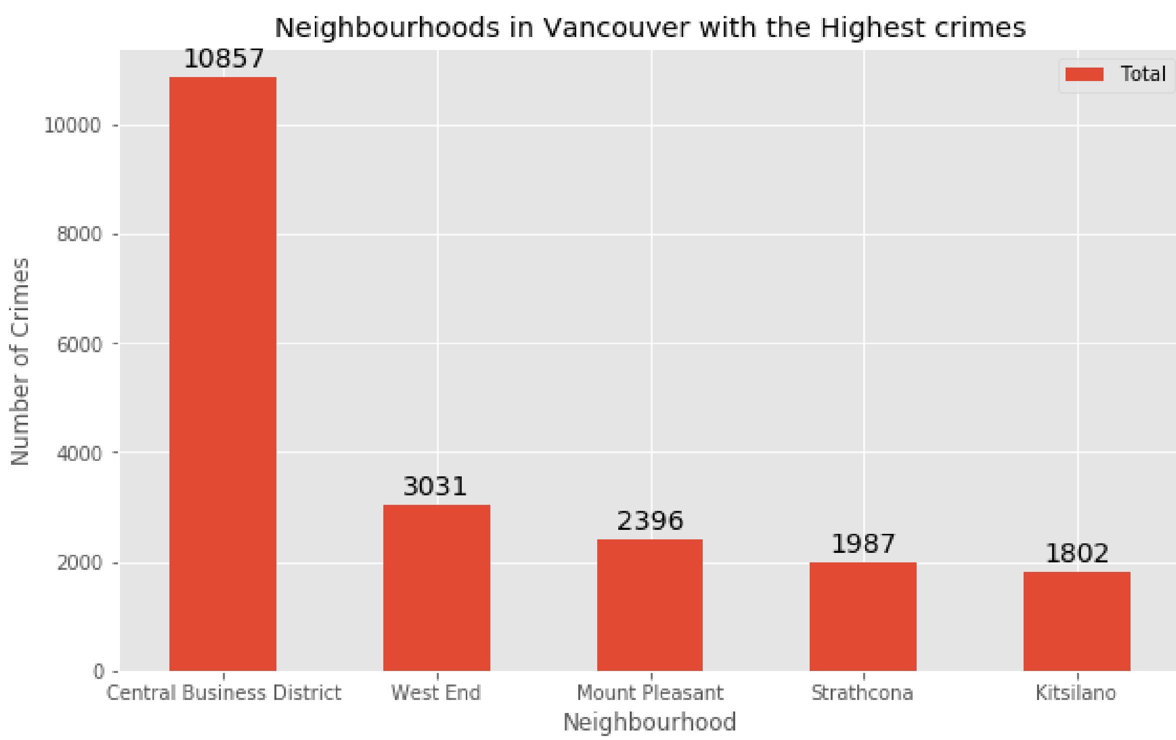
	YearBreak and Enter Commercial	YearBreak and Enter Residential/Other	YearMischief	YearOther Theft	YearTheft from Vehicle	YearTheft of Bicycle	Year of Ve
count	4.000000	4.000000	4.00000	4.000000	4.000000	4.000000	4.0(
mean	506.250000	599.250000	1430.25000	1236.750000	3736.500000	539.750000	286.5(
std	354.409721	488.189427	997.26572	1060.087221	2723.536977	353.955153	226.1
min	49.000000	156.000000	187.00000	88.000000	483.000000	36.000000	71.0(
25%	314.500000	187.500000	843.25000	544.000000	2249.250000	450.000000	186.5(
50%	594.500000	599.000000	1627.00000	1185.000000	3796.000000	633.000000	235.0(
75%	786.250000	1010.750000	2214.00000	1877.750000	5283.250000	722.750000	335.0(
max	787.000000	1043.000000	2280.00000	2489.000000	6871.000000	857.000000	605.0(

In [36]:
```
vnc_crime_neigh.sort_values(['Total'], ascending = False, axis = 0, inplace =
True )

crime_neigh_top5 = vnc_crime_neigh.iloc[1:6]
crime_neigh_top5
```

Out[36]:

	Neighbourhood	YearBreak and Enter Commercial	YearBreak and Enter Residential/Other	YearMischief	YearOther Theft	YearTheft from Vehicle	YearTheft of Bicycle
1	Central Business District	551	124	1812	2034	5301	640
22	West End	230	72	460	455	1461	203
11	Mount Pleasant	205	124	353	493	822	232
19	Strathcona	160	124	527	81	821	108
9	Kitsilano	106	165	320	154	755	189

In [37]:
```
per_neigh = crime_neigh_top5[['Neighbourhood','Total']]

per_neigh.set_index('Neighbourhood',inplace = True)

ax = per_neigh.plot(kind='bar', figsize=(10, 6), rot=0)

ax.set_ylabel('Number of Crimes')
ax.set_xlabel('Neighbourhood')
ax.set_title('Neighbourhoods in Vancouver with the Highest crimes')

for p in ax.patches:
    ax.annotate(np.round(p.get_height(),decimals=2),
                (p.get_x()+p.get_width()/2., p.get_height()),
                ha='center',
                va='center',
                xytext=(0, 10),
                textcoords='offset points',
                fontsize = 14,
               )

plt.show()
```



In [38]:
```
vnc_crime_cat = pd.pivot_table(vnc_boroughs_crime,
                               values=['Year'],
                               index=['Borough'],
                               columns=['Type'],
                               aggfunc=len,
                               fill_value=0,
                               margins=True)
vnc_crime_cat
```

Out[38]:

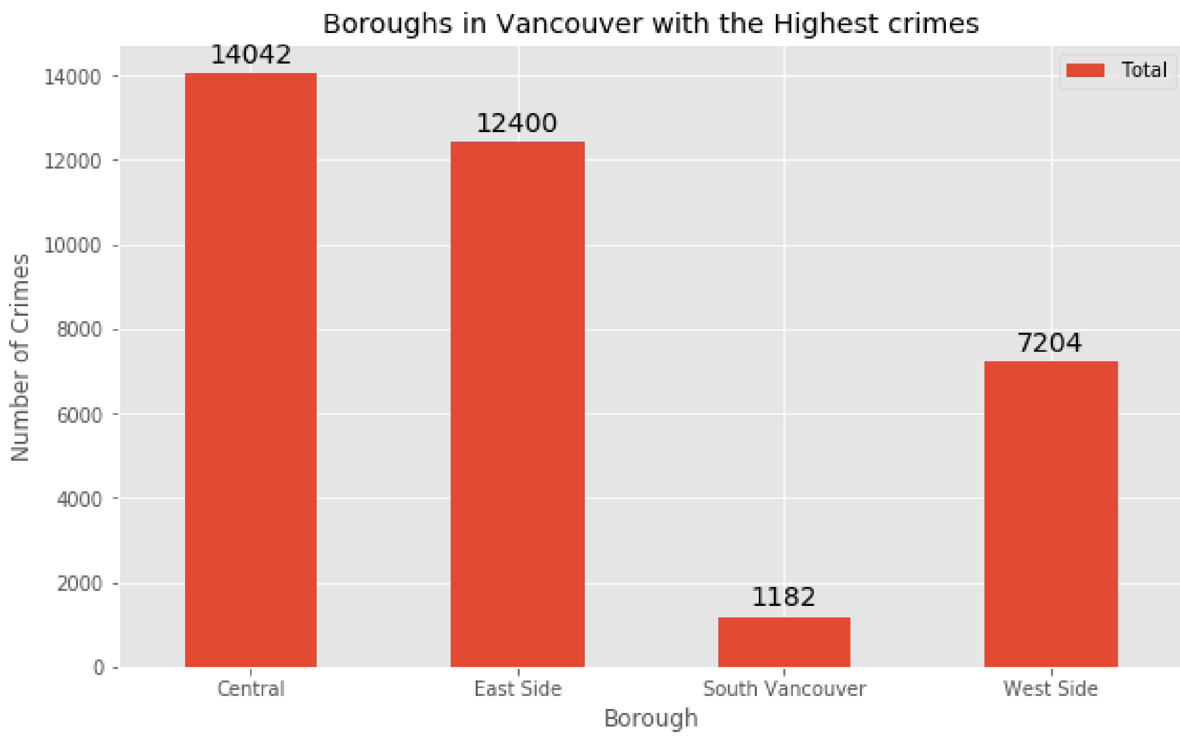
	Year							
Type	**Break and Enter Commercial**	**Break and Enter Residential/Other**	**Mischief**	**Other Theft**	**Theft from Vehicle**	**Theft of Bicycle**	**Theft of Vehicle**	**Vehicle Collision or Pedestrian Struck (with Fatality)**
Borough								
Central	787	198	2280	2489	6871	857	245	1
East Side	786	1043	2192	1674	4754	678	605	8
South Vancouver	49	156	187	88	483	36	71	1
West Side	403	1000	1062	696	2838	588	225	3
All	2025	2397	5721	4947	14946	2159	1146	13

In [39]:
```
vnc_crime_cat.reset_index(inplace = True)
vnc_crime_cat.columns = vnc_crime_cat.columns.map(''.join)
vnc_crime_cat.rename(columns={'YearAll':'Total',
                              'YearBreak and Enter Commercial' : 'Break and En
ter Commercial',
                              'YearBreak and Enter Residential/Other' : 'Break
and Enter Residential',
                              'YearMischief' : 'Mischief',
                              'YearOther Theft' : 'Other',
                              'YearTheft from Vehicle' : 'Theft from Vehicle',
                              'YearTheft of Bicycle' : 'Theft of Bicycle',
                              'YearTheft of Vehicle' : 'Theft of Vehicle',
                              'YearVehicle Collision or Pedestrian Struck (wit
h Fatality)' : 'Vehicle Collision or Pedestrian Struck (with Fatality)',
                              'YearVehicle Collision or Pedestrian Struck (wit
h Injury)' : 'Vehicle Collision or Pedestrian Struck (with Injury)'}, inplace=
True)
# To ignore bottom All in Borough
vnc_crime_cat = vnc_crime_cat.head(4)
vnc_crime_cat
```

Out[39]:

	Borough	Break and Enter Commercial	Break and Enter Residential	Mischief	Other	Theft from Vehicle	Theft of Bicycle	Theft of Vehicle	Vehicle Collision or Pedestrian Struck (with Fatality)	P
0	Central	787	198	2280	2489	6871	857	245	1	
1	East Side	786	1043	2192	1674	4754	678	605	8	
2	South Vancouver	49	156	187	88	483	36	71	1	
3	West Side	403	1000	1062	696	2838	588	225	3	

In [40]:
```
per_borough = vnc_crime_cat[['Borough','Total']]

per_borough.set_index('Borough',inplace = True)

ax = per_borough.plot(kind='bar', figsize=(10, 6), rot=0)

ax.set_ylabel('Number of Crimes')
ax.set_xlabel('Borough')
ax.set_title('Boroughs in Vancouver with the Highest crimes')

for p in ax.patches:
    ax.annotate(np.round(p.get_height(),decimals=2),
                (p.get_x()+p.get_width()/2., p.get_height()),
                ha='center',
                va='center',
                xytext=(0, 10),
                textcoords='offset points',
                fontsize = 14,
               )

plt.show()
```



Part 3: Creating a new consolidated dataset of the Neighborhoods, along with their boroughs, crime data and the respective Neighbourhood's co-ordinates.:

This data will be fetched using OpenCage Geocoder to find the safest borough and explore the neighbourhood by plotting it on maps using Folium and perform exploratory data analysis.

In [ ]:

```
vnc_ws_df = vnc_crime_cat[vnc_crime_cat['Borough'] == 'West Side']

 vnc_ws_df = vnc_ws_df.sort_values(['Total'], ascending = True, axis = 0)

vnc_ws = vnc_ws_df[['Borough','Theft of Vehicle', 'Break and Enter Commercial'
,'Break and Enter Residential','Mischief','Other',
                 'Theft from Vehicle','Vehicle Collision or Pedestrian Struck
 (with Fatality)','Theft of Bicycle',
                 'Vehicle Collision or Pedestrian Struck (with Injury)']]


vnc_ws.set_index('Borough',inplace = True)

ax = vnc_ws.plot(kind='bar', figsize=(10, 6), rot=0)

ax.set_ylabel('Number of Crimes')
ax.set_xlabel('Borough')
ax.set_title('Different Kind of Crimes in West Side Borough')

for p in ax.patches:
    ax.annotate(np.round(p.get_height(),decimals=3),
                (p.get_x()+p.get_width()/3., p.get_height()),
                ha='center',
                va='center',
                xytext=(5, 10),
                textcoords='offset points',
                fontsize = 14
               )
    ax.legend(loc='upper left', bbox_to_anchor=(1.00, 0.5))

plt.show()
```

In [43]:

```
vnc_ws_neigh = vnc_boroughs_crime

#vnc_ws_neigh.drop(['Type','Year', 'Month', 'Day', 'Hour'], axis = 1, inplace
 = True)
vnc_ws_neigh = vnc_ws_neigh[vnc_ws_neigh['Borough'] == 'West Side']
vnc_ws_neigh.reset_index(inplace=True, drop=True)

print('Number of Neighbourhoods in West Side Borough', len(vnc_ws_neigh['Neigh
bourhood'].unique()))

vnc_ws_neigh['Neighbourhood'].unique()
```

```
Number of Neighbourhoods in West Side Borough 10
```

Out[43]:

```
array(['Shaughnessy', 'Fairview', 'Oakridge', 'Marpole', 'Kitsilano',
       'Kerrisdale', 'West Point Grey', 'Arbutus Ridge', 'South Cambie',
       'Dunbar-Southlands'], dtype=object)
```

In [44]:
```
Latitude = []
Longitude = []
Borough = []
Neighbourhood = vnc_ws_neigh['Neighbourhood'].unique()



key = '830323b5ca694362904814ff0a11b803'
geocoder = OpenCageGeocode(key)

for i in range(len(Neighbourhood)):
    address = '{}, Vancouver, BC, Canada'.format(Neighbourhood[i])
    location = geocoder.geocode(address)
    Latitude.append(location[0]['geometry']['lat'])
    Longitude.append(location[0]['geometry']['lng'])
    Borough.append('West Side')
print(Latitude, Longitude)

#print('The geographical coordinate of Vancouver City are {}, {}.'.format(latit
ude, longitude))
```

```
[49.2518626, 49.2641128, 49.2308288, 49.2092233, 49.2694099, 49.2346728, 49.2
644843, 49.2409677, 49.2466847, 49.2534601] [-123.1380226, -123.1268352, -12
3.1311342, -123.1361495, -123.155267, -123.1553893, -123.1854326, -123.167000
8, -123.120915, -123.1850439]
```

In [45]:
```
ws_neig_dict = {'Neighbourhood': Neighbourhood,'Borough':Borough,'Latitude': L
atitude,'Longitude':Longitude}
ws_neig_geo = pd.DataFrame(data=ws_neig_dict, columns=['Neighbourhood', 'Borou
gh', 'Latitude', 'Longitude'], index=None)

ws_neig_geo
```

Out[45]:

	Neighbourhood	Borough	Latitude	Longitude
0	Shaughnessy	West Side	49.251863	-123.138023
1	Fairview	West Side	49.264113	-123.126835
2	Oakridge	West Side	49.230829	-123.131134
3	Marpole	West Side	49.209223	-123.136150
4	Kitsilano	West Side	49.269410	-123.155267
5	Kerrisdale	West Side	49.234673	-123.155389
6	West Point Grey	West Side	49.264484	-123.185433
7	Arbutus Ridge	West Side	49.240968	-123.167001
8	South Cambie	West Side	49.246685	-123.120915
9	Dunbar-Southlands	West Side	49.253460	-123.185044

In [46]:
```
address = 'Vancouver, BC, Canada'

location = geocoder.geocode(address)
latitude = location[0]['geometry']['lat']
longitude = location[0]['geometry']['lng']

print('The geograpical coordinate of Vancouver, Canada are {}, {}.'.format(lat
itude, longitude))
```

```
The geograpical coordinate of Vancouver, Canada are 49.2608724, -123.1139529.
```

In [ ]:
```
van_map = folium.Map(location=[latitude, longitude], zoom_start=12)

# add markers to map
for lat, lng, borough, neighborhood in zip(ws_neig_geo['Latitude'], ws_neig_ge
o['Longitude'], ws_neig_geo['Borough'], ws_neig_geo['Neighbourhood']):
    label = '{}, {}'.format(neighborhood, borough)
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [lat, lng],
        radius=5,
        popup=label,
        color='red',
        fill=True,
        fill_color='#3186cc',
        fill_opacity=0.7,
        parse_html=False).add_to(van_map)

van_map
```

Part 4: Creating a new consolidated dataset of the Neighborhoods, boroughs, and the most common venues and the respective Neighbourhood along with co-ordinates.:

This data will be fetched using Four Square API to explore the neighbourhood venues and to apply machine learning algorithm to cluster the neighbourhoods and present the findings by plotting it on maps using Folium.

In [48]:
```
#Four Square Credentials

CLIENT_ID = 'XVY0YGK3DX5QGHMN2TGSK2EWA55P3JNPIVC5QVW5SGIGUI2L'
CLIENT_SECRET = 'T53Z3HT4W5DVALRIPBK2DPD4NFOCISMUTMNBLNW13KEJTAIJ'
VERSION = '20191101'
LIMIT = 100

print('Your credentials:')
print('CLIENT_ID: ' + CLIENT_ID)
print('CLIENT_SECRET:' + CLIENT_SECRET)
```

```
Your credentials:
CLIENT_ID: XVY0YGK3DX5QGHMN2TGSK2EWA55P3JNPIVC5QVW5SGIGUI2L
CLIENT_SECRET:T53Z3HT4W5DVALRIPBK2DPD4NFOCISMUTMNBLNW13KEJTAIJ
```

In [49]:
```
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={}&clie
nt_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']['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 = ['Neighbourhood',
                  'Neighborhood Latitude',
                  'Neighborhood Longitude',
                  'Venue',
                  'Venue Category']

    return(nearby_venues)
```

In [50]:
```
vnc_ws_venues = getNearbyVenues(names=ws_neig_geo['Neighbourhood'],
                                   latitudes=ws_neig_geo['Latitude'],
                                   longitudes=ws_neig_geo['Longitude']
                                  )
```

```
Shaughnessy
Fairview
Oakridge
Marpole
Kitsilano
Kerrisdale
West Point Grey
Arbutus Ridge
South Cambie
Dunbar-Southlands
```

In [51]:
```
print(vnc_ws_venues.shape)
vnc_ws_venues.head()
```

```
(222, 5)
```

Out[51]:

	Neighbourhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Category
0	Shaughnessy	49.251863	-123.138023	Angus Park	Park
1	Shaughnessy	49.251863	-123.138023	Crepe & Cafe	French Restaurant
2	Fairview	49.264113	-123.126835	Gyu-Kaku Japanese BBQ	BBQ Joint
3	Fairview	49.264113	-123.126835	CRESCENT nail and spa	Nail Salon
4	Fairview	49.264113	-123.126835	Charleson Park	Park

In [52]:
```
vnc_ws_venues.groupby('Neighbourhood').count().drop(['Neighborhood Latitude',
'Neighborhood Longitude','Venue Category'], axis = 1)
```

Out[52]:

	Venue
Neighbourhood	
Arbutus Ridge	5
Dunbar-Southlands	8
Fairview	26
Kerrisdale	39
Kitsilano	47
Marpole	30
Oakridge	7
Shaughnessy	2
South Cambie	16
West Point Grey	42

In [53]:
```
print('There are {} uniques categories.'.format(len(vnc_ws_venues['Venue Categ
ory'].unique())))
```

```
There are 87 uniques categories.
```

4) Result

In [54]:
```
# one hot encoding
vnc_onehot = pd.get_dummies(vnc_ws_venues[['Venue Category']], prefix="", pref
ix_sep="")

# add neighborhood column back to dataframe
vnc_onehot['Neighbourhood'] = vnc_ws_venues['Neighbourhood']

# move neighborhood column to the first column
fixed_columns = [vnc_onehot.columns[-1]] + list(vnc_onehot.columns[:-1])
vnc_onehot = vnc_onehot[fixed_columns]

vnc_onehot.head()
```

Out[54]:

	Neighbourhood	American Restaurant	Asian Restaurant	BBQ Joint	Bakery	Bank	Bar	Beach	Bookstore	Boutiq
0	Shaughnessy	0	0	0	0	0	0	0	0	
1	Shaughnessy	0	0	0	0	0	0	0	0	
2	Fairview	0	0	1	0	0	0	0	0	
3	Fairview	0	0	0	0	0	0	0	0	
4	Fairview	0	0	0	0	0	0	0	0	

5) Discussion

The objective of the business problem was to help stakeholders identify one of the safest borough in Vancouver, and an appropriate neighborhood within the borough to set up a commercial establishment especially a Grocery store. This has been achieved by first making use of Vancouver crime data to identify a safe borugh with considerable number of neighborhood for any business to be viable. After selecting the borough it was imperative to choose the right neighborhood where grocery shops were not among venues in a close proximity to each other. We achieved this by grouping the neighborhoods into clusters to assist the stakeholders by providing them with relavent data about venues and safety of a given neighborhood.

6) Conclusion

We have explored the crime data to understand different types of crimes in all neighborhoods of Vancouver and later categorized them into different boroughs, this helped us group the neighborhoods into boroughs and choose the safest borough first. Once we confirmed the borough the number of neighborhoods for consideration also comes down, we further shortlist the neighborhoods based on the common venues, to choose a neighborhood which best suits the business problem.

In [ ]:

```

```