Toronto Police

Major Crime Indicators (MCI)

SCS 3253-061 – Machine Learning

Instructor: *Saeid Abolfazli*

Team members:

Caio Gasparine | Fábio Queiroz | Olivier Sangam | Illidan Yuan

https://github.com/caiogasparine/SCS_3253_061-Machine-Learning

April 2023

Toronto Police

Source of Data: Major Crime Indicators (MCI) Historical

This dataset includes all **Major Crime Indicators (MCI)** occurrences by reported date and related offences from **2014 to June 30, 2022.**

https://data.torontopolice.on.ca/

Toronto Police Service
PUBLIC SAFETY DATA PORTAL

https://data.torontopolice.on.ca/pages/major-crime-indicators

Toronto Police

Data

Major Crime Indicators

Toronto Police Service | TorontoPoliceService

This dataset includes all Major **Crime** Indicators (MCI) occurrences by reported date and related offences from 2014 to June 30, 2022.Major Crime Indicators (MCI) Dashboard -Download…

Type: Feature Layer
Last Updated: December 5, 2022

Rows: 301,233
Tags: MCI, **Crime**, **Crimes**, Major **Crime** Indicators, Assault, …

Toronto Police Service
PUBLIC SAFETY DATA PORTAL

This dataset includes all **Major Crime Indicators (MCI)** occurrences by reported date and related offences. The MCI categories include Assault, Break and Enter, Auto Theft, Robbery, and Theft Over. This data is provided at the offence and/or victim level, therefore one occurrence number may have several records associated with the various MCIs used to categorize the occurrence. This data does not include occurrences that have been deemed unfounded. The definition of unfounded according to Statistics Canada is: “It has been determined through police investigation that the offence reported did not occur, nor was it attempted” (Statistics Canada, 2020).4

Format: CSV, XML, SHP

https://data.torontopolice.on.ca/datasets/TorontoPS::major-crime-indicators-1/about

Major Crime Indicators

Private Member

Toronto Police Service

View Full Details

Download

Details

Dataset
Feature Layer

December 5, 2022
Info Updated

December 5, 2022
Data Updated

September 27, 2022
Published Date

301,233 Records
View data table

Public
Anyone can see this content

Custom License
View license details

I want to use this

Showing 25 of 301,233 rows

Close Table

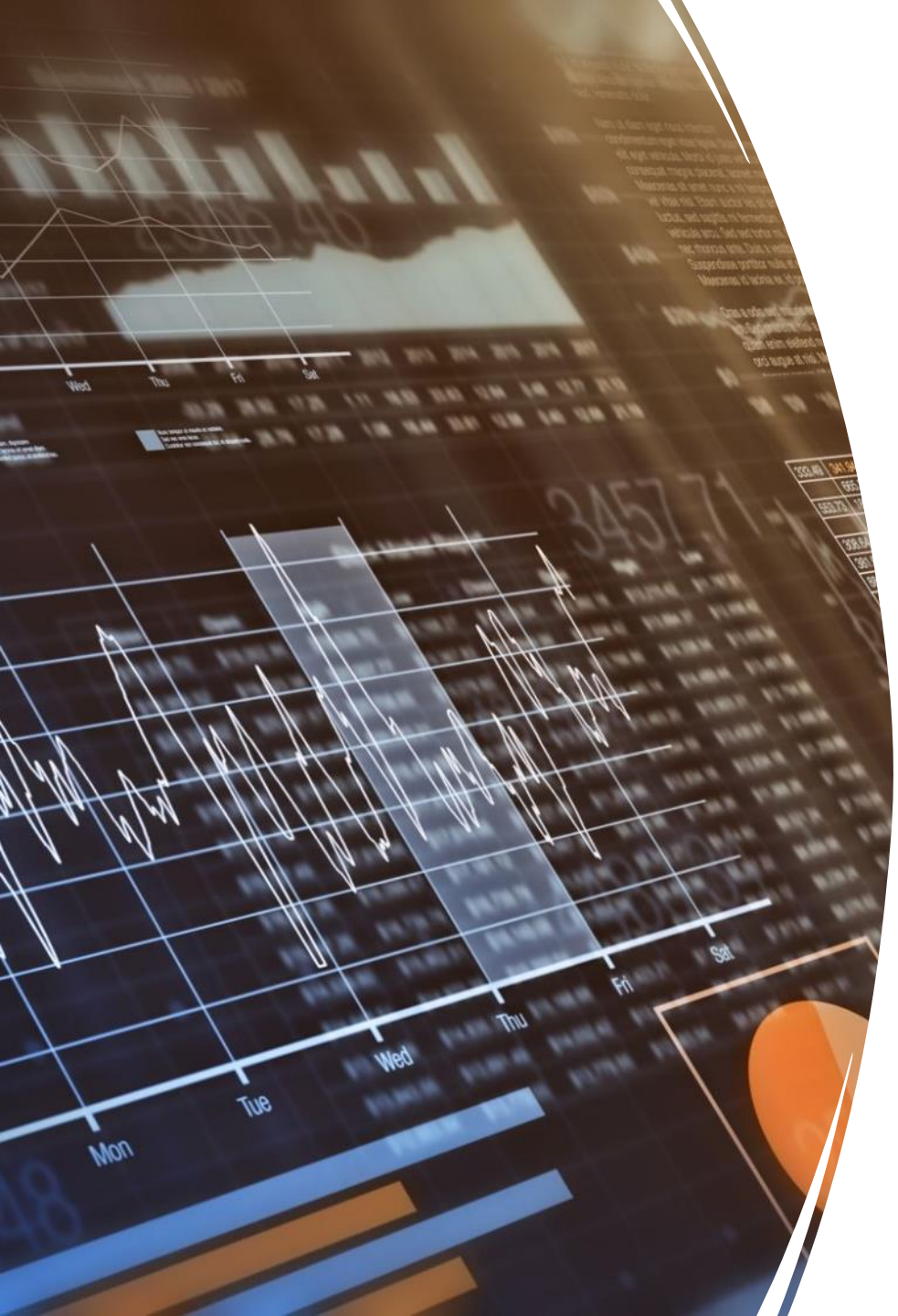
Index	event_unique_id	Division	occurrencedate	reporteddate	location_type
201	GO-20141273318	D31	1/3/2014, 12:00 AM	1/3/2014, 12:00 AM	Apartment (Rooming House,
202	GO-20141274349	D42	1/3/2014, 12:00 AM	1/3/2014, 12:00 AM	Single Home, House (Attach .
203	GO-20141274052	D22	1/3/2014, 12:00 AM	1/3/2014, 12:00 AM	Open Areas (Lakes, Parks, Riv
204	GO-20141276966	D53	1/3/2014, 12:00 AM	1/3/2014, 12:00 AM	Other Commercial / Corpora
205	GO-20141274457	D22	1/3/2014, 12:00 AM	1/3/2014, 12:00 AM	Convenience Stores
206	GO-20141273151	D51	1/2/2014, 12:00 AM	1/3/2014, 12:00 AM	Convenience Stores
207	GO-20141274747	D33	1/2/2014, 12:00 AM	1/3/2014, 12:00 AM	Other Commercial / Corpora
208	GO-20141275836	D14	12/31/2013, 12:00 AM	1/3/2014, 12:00 AM	Parking Lots (Apt., Commerci
209	GO-20141275598	D13	1/3/2014, 12:00 AM	1/3/2014, 12:00 AM	Apartment (Rooming House,
210	GO-2014946295	D11	1/3/2014, 12:00 AM	1/3/2014, 12:00 AM	Apartment (Rooming House,
211	GO-20141275653	D42	1/3/2014, 12:00 AM	1/3/2014, 12:00 AM	Single Home, House (Attach .
212	GO-20141276157	D42	1/3/2014, 12:00 AM	1/3/2014, 12:00 AM	Single Home, House (Attach .
213	GO-20141274285	D12	1/3/2014, 12:00 AM	1/3/2014, 12:00 AM	Open Areas (Lakes, Parks, Riv

The Problem

- Crime rates are skyrocketing very fast in Toronto and two main challenges are part of this situation:
- (1) identify what would be the probability of a new occurrence considering your geolocation, based on the number of occurrences in certain location.
- (2) determine where would be the optimized location (clusters) for new police patrol based on the current crime rates and location.

Solution Scope

Develop a solution to address the problem (1) and (2) and give possible alternatives to the problem, based on the historical data analysis.



Assumptions

- Consider only years with a **complete set of data** (2014 – 2021).
- The **occurrence date** (day, month, year) of the events will the major driver for all the evaluations.
- Explore visuals solutions to facilitate the data interpretability.
- Real Time **OR** Near Real-Time Solution.
- The enterprise solution architecture must consider automatic **data ingestion** (and be prepared to scale in an RT/NRT data scenario).
- Complete solution (*End-2-End*), considering the ML model and training + enterprise cloud architecture for future implementation.
- Utilization of the current Microsoft **Azure** infra.

Methodology

1. Big picture / Frame the problem
2. Get the data / develop the pipeline
3. Explore and visualize the data to gain insights
4. Prepare the data for the ML algorithm
5. Select a model and train it
6. Fine-tune your model
7. Present your solution
8. Launch, monitor, and maintain your system (MLOps)

1. Big picture / Frame the problem

(1) identify what would be the probability of a new occurrence considering your geolocation.

Fields available:

- **Long + Lat**
- **Location Type**
- **Premises Type**
- **Hood_ID**
- **Neighborhood**

(2) determine where would be the optimized location (clusters) for new police patrols based on the current crime rates and location. **> Same fields + number of occurrences and critically**

2. Get the data / develop the pipeline

METHODOLOGY

```
[1] ### Toronto Police / MCI indicators
    ### Created on February 21, 2023 / Last update April 3rd, 2023
    ### Important assumptions: occurrenceyear will be used to base all the statistics and evaluations (reportedyear)
    ### This is to reflect WHEN the event really happened

    ### Loading the main used libraries
    from datetime import date
    import numpy as np
    import pandas as pd
    import statsmodels.formula.api as sm
    import seaborn as sns
    import matplotlib.pyplot as plt
    import warnings                    #These 2 lines remove all the warnigns in the code
    warnings.filterwarnings("ignore")  #These 2 lines remove all the warnigns in the code
    %matplotlib inline
    from pandas.plotting import scatter_matrix
    from sklearn.model_selection import train_test_split
    from sklearn.neighbors import KNeighborsClassifier
    from sklearn.tree import DecisionTreeClassifier
    from sklearn.linear_model import LogisticRegression
    from sklearn.linear_model import LinearRegression

    from google.colab import drive
    from google.colab import data_table        #Mar 16, 2023 - Interactive Tables for DataFrames
    data_table.enable_dataframe_formatter()    #Mar 16, 2023 - Interactive Tables for Dataframes
```

```
[2] ### Change the code here, depending on your dataset location
    ### data = pd.read_csv('Major_Crime_Indicators.csv', header=0)
    drive.mount('/content/drive')
    data = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/Major_Crime_Indicators.csv', header=0)

    Mounted at /content/drive
```

2.1 Data Description

Field	Field Name	Description
1	Index	Unique Identifier
2	event_unique_id	Offence Number
3	Division	Police Division where Offence Occurred
4	occurrence_date	Date of Offence
5	reporteddate	Date Offence was Reported
6	location_type	Location Type of Offence
7	premises_type	Premises Type of Offence
8	ucr_code	UCR Code for Offence
9	ucr_ext	UCR Extension for Offence
10	Offence	Title of Offence
11	reportedyear	Year Offence was Reported
12	reportedmonth	Month Offence was Reported
13	reportedday	Day of the Month Offence was Reported
14	reporteddayofyear	Day of the Year Offence was Reported
15	reporteddayofweek	Day of the Week Offence was Reported
16	reportedhour	Hour Offence was Reported
17	occurrenceyear	Year Offence Occurred
18	occurrencemonth	Month Offence Occurred
19	occurrenceday	Day of the Month Offence Occurred
20	occurrencedayofyear	Day of the Year Offence Occurred
21	occurrencedayofweek	Day of the Week Offence Occurred
22	occurrencehour	Hour Offence Occurred
23	MCI	MCI Category of Occurrence
24	Hood_ID	Identifier of Neighbourhood
25	Neighbourhood	Name of Neighbourhood
26	Long	Longitude Coordinates (Offset to nearest intersection)
27	Lat	Latitude Coordinates (Offset to nearest intersection)

This dataset includes all **Major Crime Indicators (MCI)** occurrences by reported date and related offences. The MCI categories include Assault, Break and Enter, Auto Theft, Robbery, and Theft Over. This data is provided at the offence and/or victim level, therefore one occurrence number may have several records associated with the various MCIs used to categorize the occurrence. This data does not include occurrences that have been deemed unfounded. The definition of unfounded according to Statistics Canada is: “It has been determined through police investigation that the offence reported did not occur, nor was it attempted” (Statistics Canada, 2020).4

Format: CSV, XML, SHP

```
### Exploring the data
data.shape
```

```
(281153, 30)
```

3. Explore and visualize the data to gain insights

METHODOLOGY

```
### Defining our scope (Assumption) considering years with complete data 2014 to 2021 // March 7, 2023
### ocurrencedate min = 9/6/1966 and year = NULL AND ocurrencedata max = 30/06/2022
### Clearning the dataset to have the full data for a complete year 2014-2021

data = data[data.occurrenceyear >=2014]
data = data[data.occurrenceyear <=2021]
```

3 - Explore and visualize the data to gain insights + Cleaning

```
[4] ### Exploring the data to gain insigths // (lines, colunms) OR (instances, features)
    data.shape
```

```
(281153, 30)
```

```
[5] ### Exploring the data to gain insigths // visualizing all the columns available
    data.columns
```

```
Index(['X', 'Y', 'Index_', 'event_unique_id', 'Division', 'occurrencedate',
       'reporteddate', 'location_type', 'premises_type', 'ucr_code', 'ucr_ext',
       'offence', 'reportedyear', 'reportedmonth', 'reportedday',
       'reporteddayofyear', 'reporteddayofweek', 'reportedhour',
       'occurrenceyear', 'occurrencemonth', 'occurrenceday',
       'occurrencedayofyear', 'occurrencedayofweek', 'occurrencehour',
       'mci_category', 'Hood_ID', 'Neighbourhood', 'Longitude', 'Latitude',
       'ObjectId'],
      dtype='object')
```

```
[6] ### Exploring the data to gain insigths // visualizing the data - 5 first instances
    data.head(5)
```

3. Explore and visualize the data to gain insights

METHODOLOGY

```
[10] ### Exploring the data to gain insigths // Looking for duplicated values
     data.loc[data.duplicated()]
```

```
Warning: Total number of columns (30) exceeds max_columns (20). Falling back to pandas display.
```

X	Y	Index_	event_unique_id	Division	occurrencedate	reporteddate	location_type	premises_type	ucr_code	...

0 rows × 30 columns

```
[11] ### Exploring the data to gain insigths // count of values for a specific column/feature
     data['mci_category'].value_counts()
```

```
Assault            151290
Break and Enter     56446
Auto Theft          36415
Robbery             27786
Theft Over           9216
Name: mci_category, dtype: int64
```

```
[12] ### Exploring the data to gain insigths // count of values for a specific column/feature
     data['location_type'].value_counts()
```

```
Apartment (Rooming House, Condo)                                67263
Single Home, House (Attach Garage, Cottage, Mobile)             50460
Streets, Roads, Highways (Bicycle Path, Private Road)           44757
Other Commercial / Corporate Places (For Profit, Warehouse, Corp. Bldg    32460
Parking Lots (Apt., Commercial Or Non-Commercial)               23310
Bar / Restaurant                                                11324
Open Areas (Lakes, Parks, Rivers)                                6022
Schools During Supervised Activity                               4321
Convenience Stores                                               3598
Ttc Subway Station                                               3316
Commercial Dwelling Unit (Hotel, Motel, B & B, Short Term Rental)        3256
Schools During Un-Supervised Activity                            2512
```

3. Explore and visualize the data to gain insights

METHODOLOGY

```
[14] ### Exploring the data to gain insigths /
     data['premises_type'].value_counts()
```

```
Outside        74890
Apartment      67263
Commercial     56924
House          50460
Other          16367
Transit         7887
Educational     7362
Name: premises_type, dtype: int64
```

```
[15] ### Exploring the data to gain insigths /
     data['offence'].value_counts()
```

```
Assault                          103291
B&E                               47650
Theft Of Motor Vehicle            36415
Assault With Weapon               25223
Robbery - Mugging                  8149
B&E W'Intent                       6973
Assault Bodily Harm                6652
Theft Over                         5112
Assault Peace Officer              4855
Robbery - Other                    4609
Robbery With Weapon                4538
Robbery - Business                 3975
Assault - Resist/ Prevent Seiz     2894
Aggravated Assault                 2366
Theft From Motor Vehicle Over      2261
Robbery - Swarming                 2239
Discharge Firearm With Intent      1912
Unlawfully In Dwelling-House       1701
Discharge Firearm - Recklessly     1326
Theft From Mail / Bag / Key        1116
Robbery - Home Invasion            1074
Pointing A Firearm                 1037
Robbery - Purse Snatch             1011
Robbery - Financial Institute       860
Robbery - Vehicle Jacking           658
Assault Peace Officer Wpn/Cbh       506
Theft Over - Shoplifting            459
```

```
[17] ### Exploring the data to gain insigths // Descriptive
     ### of a dataset's distribution, excluding NaN values
     data.describe()
```

index	X	Y	Index_	ucr_c
count	281153.0	281153.0	281153.0	
mean	-8725778.781201236	5351167.191005955	141581.91726924485	1700.83282
std	991171.8912079454	607858.9101760103	81452.61592346136	325.535649
min	-8910331.11926327	0.0	2.0	
25%	-8846627.41966955	5412954.53870776	71137.0	
50%	-8837968.21489199	5419022.15720652	141570.0	
75%	-8829773.01090277	5426993.22386777	212067.0	
max	0.0	5517227.73472418	301144.0	

Show 25 per page

```
[18] ### Exploring the data to gain insigths // count of va
     data['occurrenceyear'].value_counts()
```

```
2019.0    39967
2018.0    37440
2017.0    35501
2020.0    34973
2021.0    34277
2016.0    33608
2015.0    32920
2014.0    32467
Name: occurrenceyear, dtype: int64
```

```
[19] ### Exploring the data to gain insigths / Check if any
     data.isnull().any()
```

```
X                    False
Y                    False
```

3. Explore and visualize the data to gain insights

METHODOLOGY

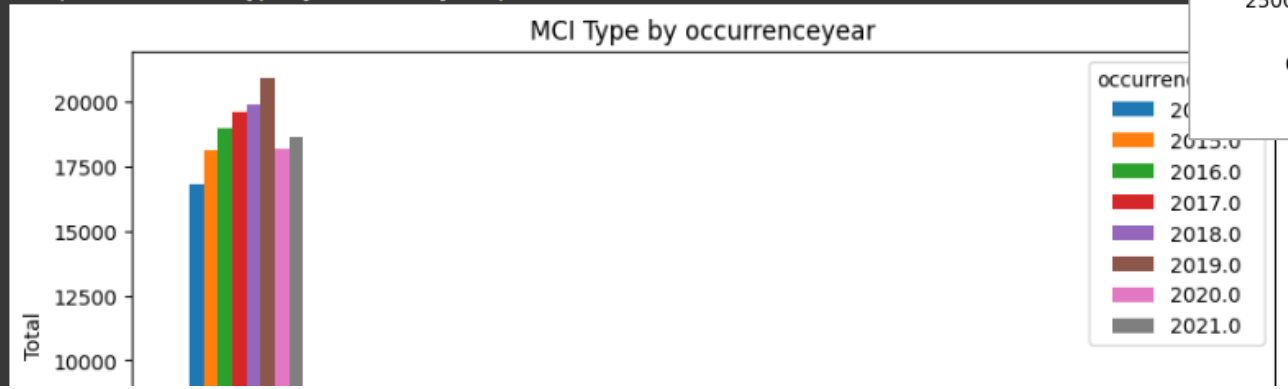
```
[23] ### Exploring the data to gain insigths
     ### Using pandas crosstab // Feb 24, 2023
     data_ct = pd.crosstab(data.mci_category, data.occurrenceyear, margins=True)
     data_ct
```

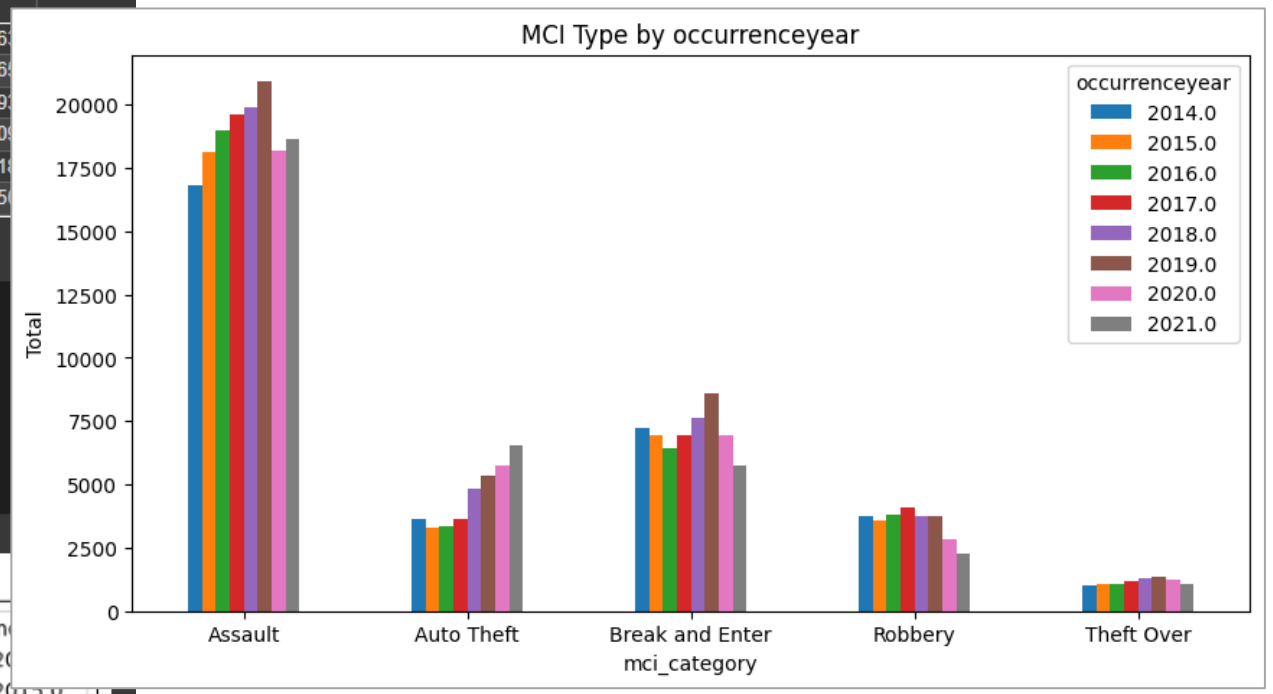
mci_category	2014.0	2015.0	2016.0	2017.0
Assault	16820	18128	19009	196
Auto Theft	3645	3265	3346	36
Break and Enter	7231	6939	6435	69
Robbery	3757	3545	3776	40
Theft Over	1014	1043	1042	11
All	32467	32920	33608	355

Show 25 per page

```
[24] ### Exploring the data to gain insigths
     ### Using pandas crosstab // Feb 24, 2023
     data_ct_plot = pd.crosstab(data.mci_category, data.occurrenceyear)
     data_ct_plot.plot(kind='bar', stacked=False, rot = 0, figsize=(10, 5));
     plt.ylabel('Total')
     plt.title('MCI Type by occurrenceyear')
```

```
Text(0.5, 1.0, 'MCI Type by occurrenceyear')
```

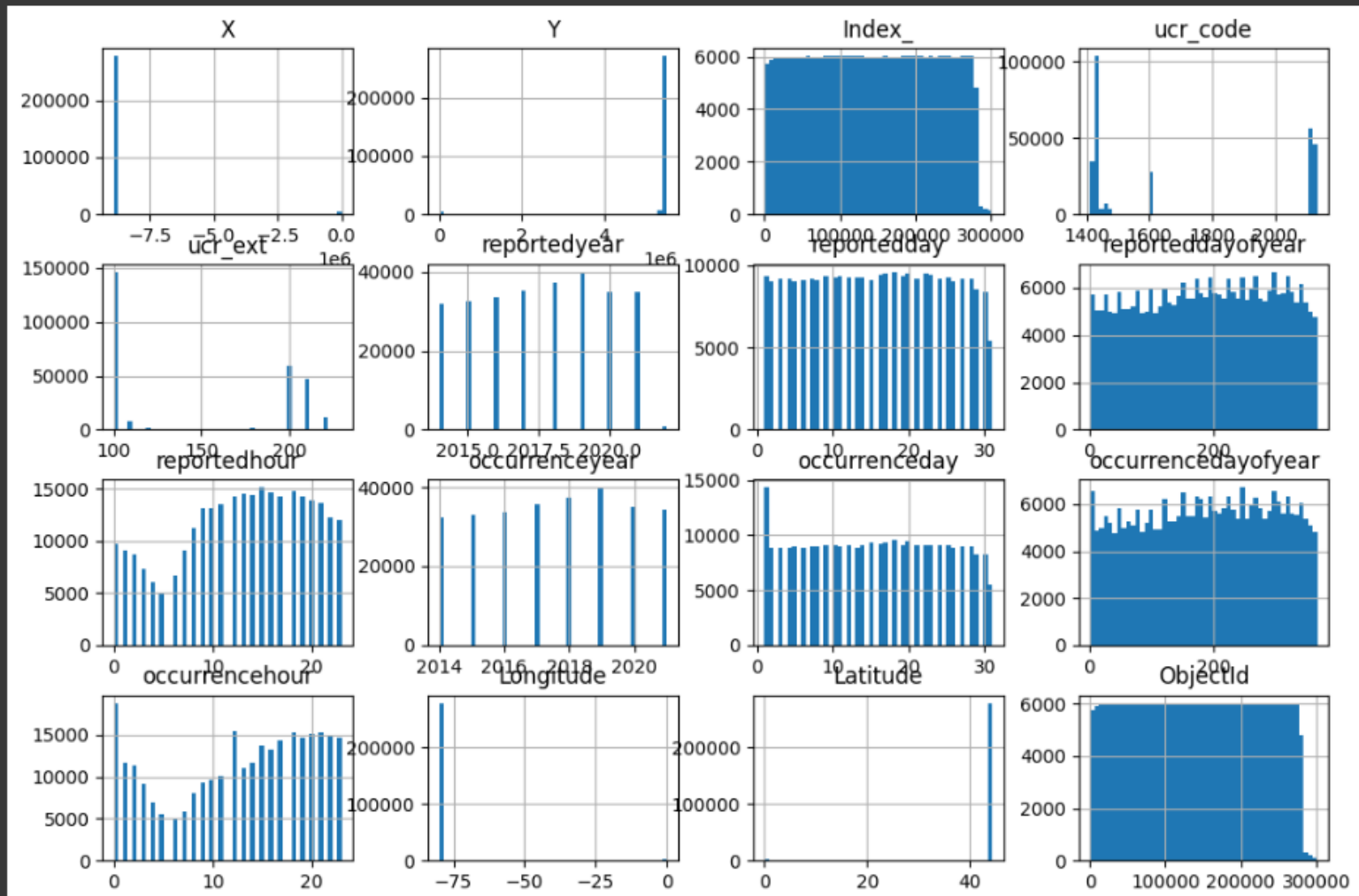




3. Explore and visualize the data to gain insights

METHODOLOGY

```
### Exploring the data to gain insigths / Histogram for each numerical atribute
import matplotlib.pyplot as plt
data.hist(bins=50, figsize=(12,8))
plt.show()
```

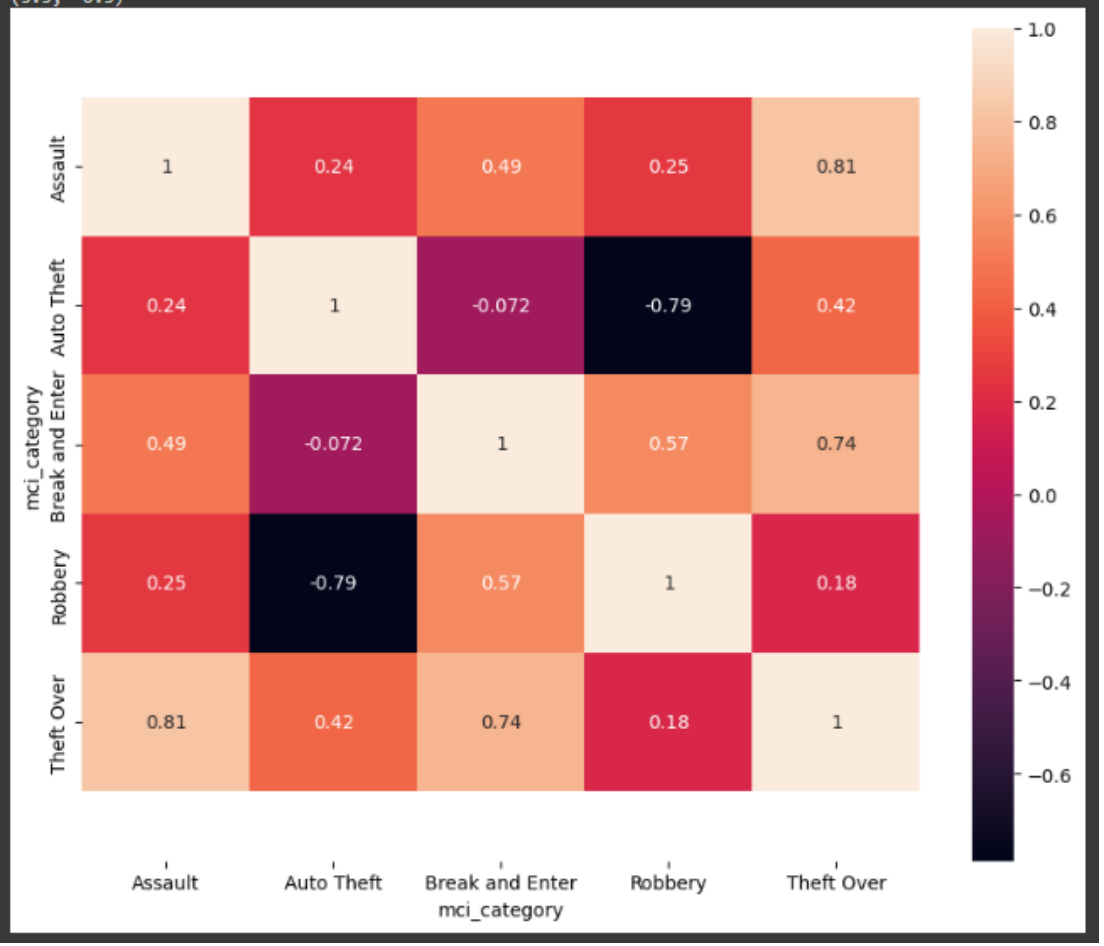


3. Explore and visualize the data to gain insights

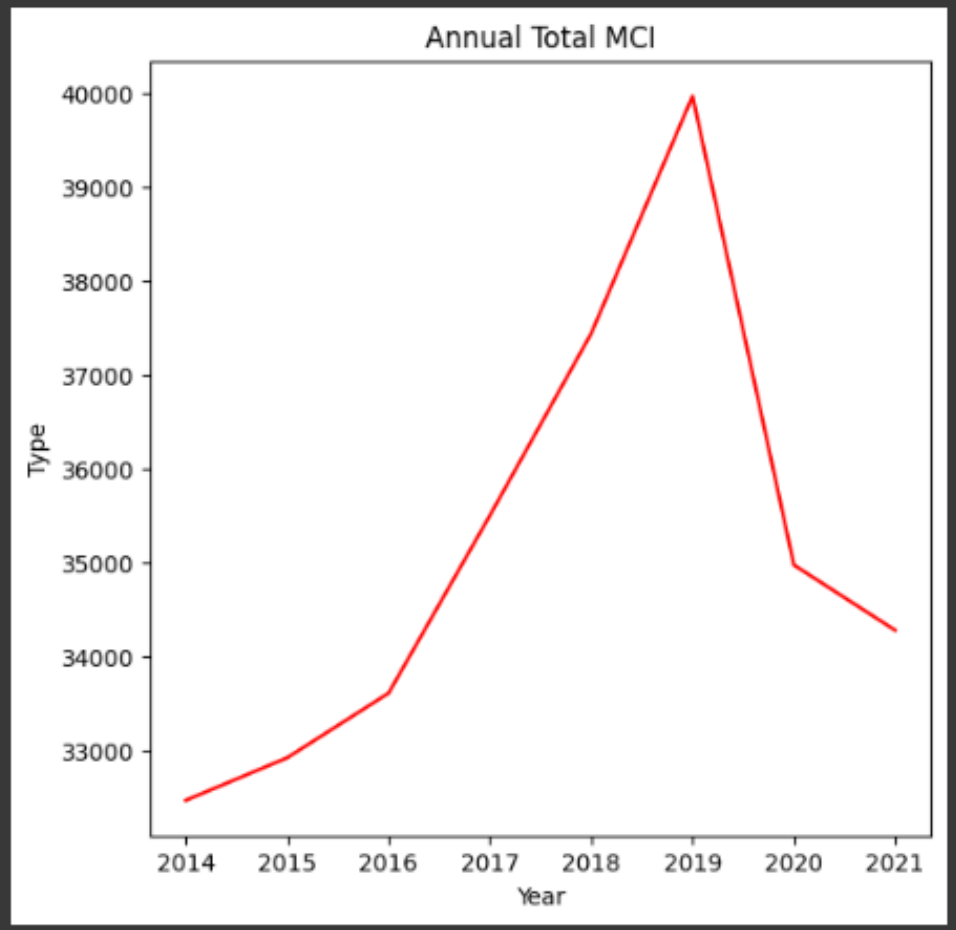
METHODOLOGY

```
[32] ### Exploring the data to gain insigths
     ### Trying to find data correlation using sns seaborn // Feb 24, 2023
     plt.figure(figsize=(10,8))
     ax = sns.heatmap(type_corr, annot=True)
     bottom, top = ax.get_ylim()
     ax.set_ylim(bottom + 0.5, top - 0.5)
```

```
(5.5, -0.5)
```




```
[36] ### Exploring the data to gain insigths / MCI per year analysis /
     fig, ax = plt.subplots(figsize=(6,6))
     sns.lineplot(x='Year', y='Type', data=data_annual, color='r')
     ax.set_title('Annual Total MCI')
     plt.show()
```

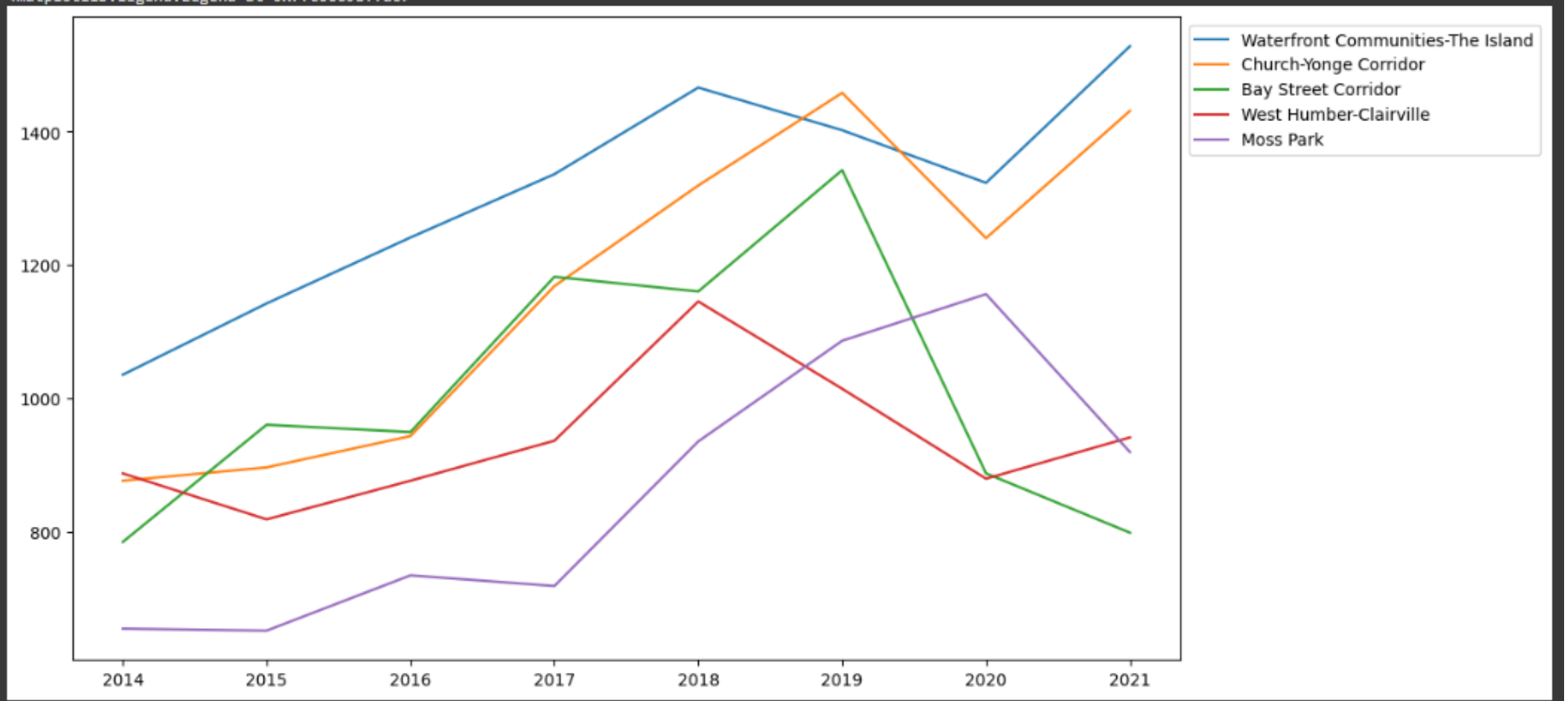


3. Explore and visualize the data to gain insights

METHODOLOGY

```
[38] ### Exploring the data to gain insigths / MCI by most dangerous neighbourhood // March 7, 2023
     top5danger = data_hood_ct.iloc[1:6,:8].T
     fig, ax = plt.subplots(1,1,figsize=(12,7))
     ax.plot(top5danger)
     ax.legend(top5danger.columns.to_list(), loc="best", bbox_to_anchor = (1,1))
```

```
<matplotlib.legend.Legend at 0x7fc06098ffa0>
```



3. Explore and visualize the data to gain insights

METHODOLOGY

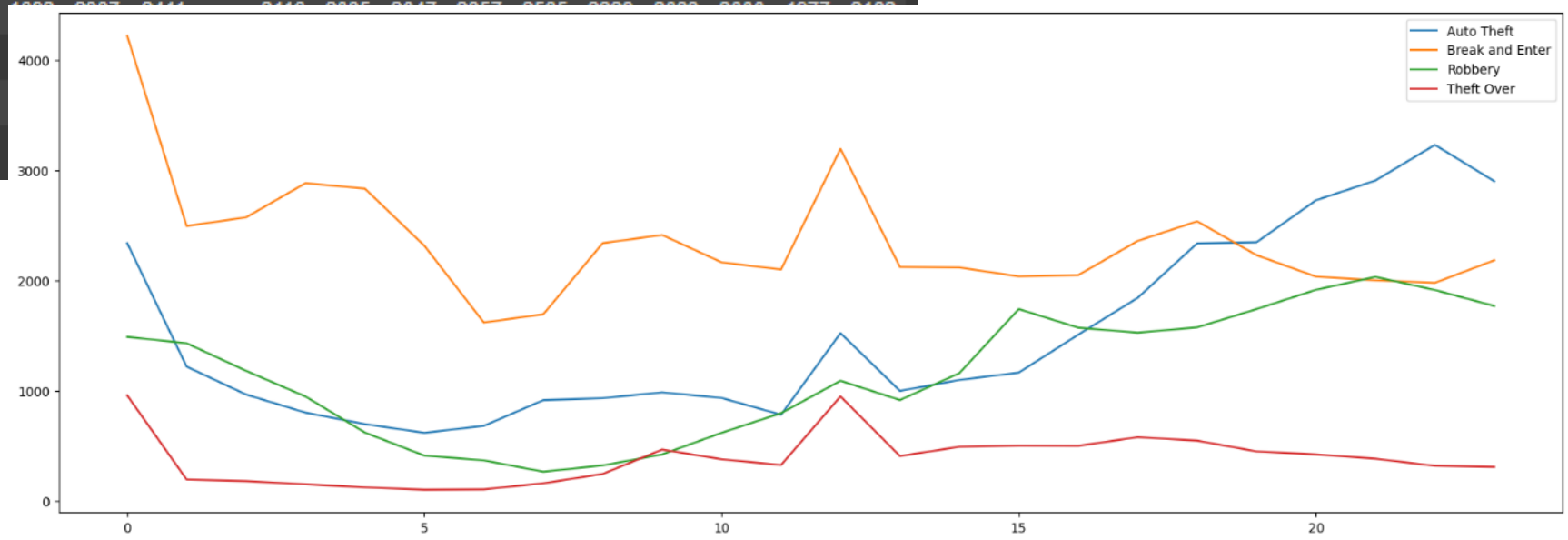
3.3 Most dangerous hours

```
[41] ### Exploring the data to gain insigths
     ### MCI by hour of the day // March 21, 2023
     data_hourly_ct = pd.crosstab(data.mci_category, data.occurrencehour, margins=False)
     #data_hourly_ct = pd.crosstab(data.occurrencehour, data.mci_category,margins=False)
     data_hourly_ct.head()
```

```
Warning: Total number of columns (24) exceeds max_columns (20). Falling back to pandas display.
```

occurrencehour	0	1	2	3	4	5	6	7	8	9	...	14	15	16	17	18	19	20	21	22	23
mci_category																					
Assault	9830	6290	6491	4352	2773	2057	2112	2784	4257	5077	...	6898	8327	7702	7992	8320	7980	8062	7988	7545	7586
Auto Theft	2337	1217	964	800	696	617	680	913	931	984	...	1096	1163	1507	1841	2335	2345	2726	2905	3228	2898
Break and Enter	4217	2492	2571	2881	2831	2313	1617														
Robbery	1487	1429	1179	946	619	410	367														
Theft Over	956	193	179	150	122	101	104														

5 rows × 24 columns

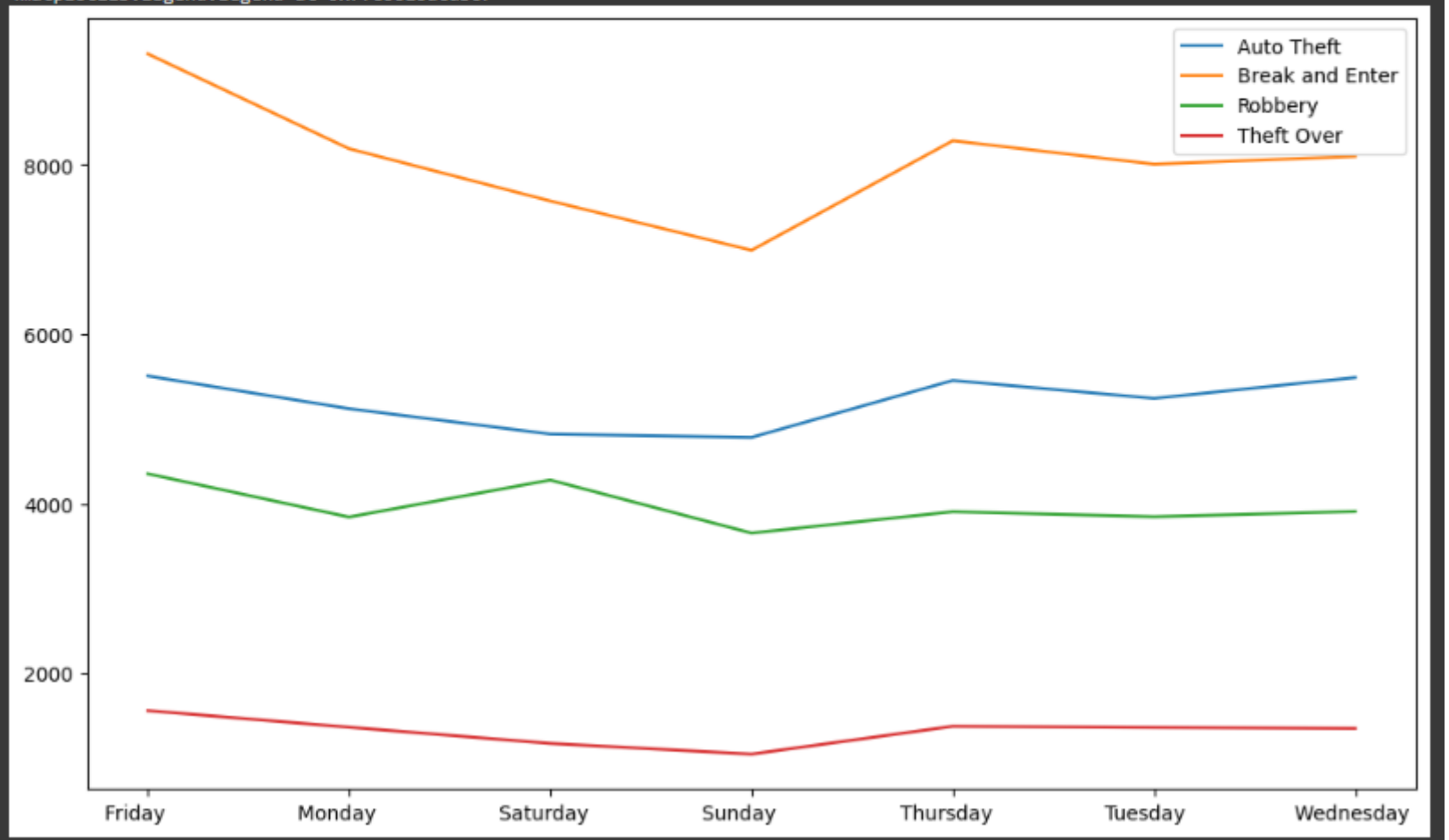


3. Explore and visualize the data to gain insights

METHODOLOGY

```
[44] ### Exploring the data to gain insigths / MCI by day of the week // March 21, 2023
     wdays_analysis = data_wdays_ct.iloc[1:23].T
     fig, ax = plt.subplots(1,1,figsize=(12,7))
     ax.plot(wdays_analysis)
     ax.legend(wdays_analysis.columns.to_list(), loc="best", bbox_to_anchor = (1,1))
```

```
<matplotlib.legend.Legend at 0x7fc0610dcd30>
```



3. Explore and visualize the data to gain insights

METHODOLOGY

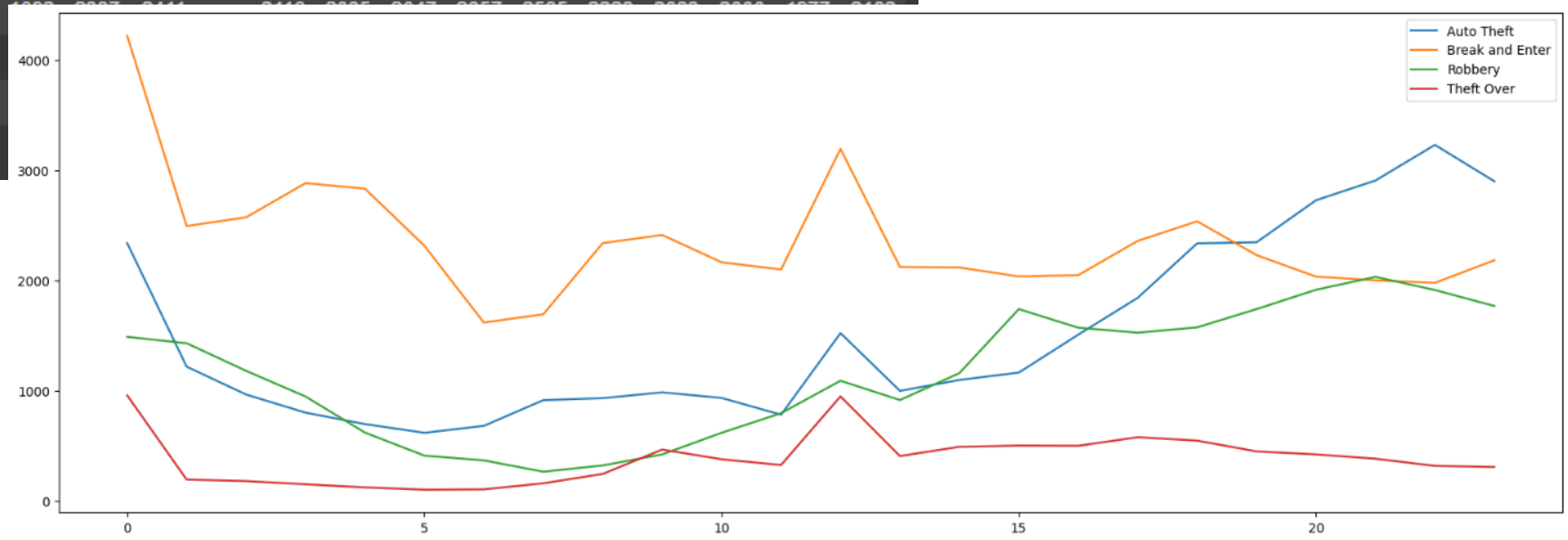
3.3 Most dangerous hours

```
[41] ### Exploring the data to gain insigths
     ### MCI by hour of the day // March 21, 2023
     data_hourly_ct = pd.crosstab(data.mci_category, data.occurrencehour, margins=False)
     #data_hourly_ct = pd.crosstab(data.occurrencehour, data.mci_category,margins=False)
     data_hourly_ct.head()
```

```
Warning: Total number of columns (24) exceeds max_columns (20). Falling back to pandas display.
```

occurrencehour	0	1	2	3	4	5	6	7	8	9	...	14	15	16	17	18	19	20	21	22	23
mci_category																					
Assault	9830	6290	6491	4352	2773	2057	2112	2784	4257	5077	...	6898	8327	7702	7992	8320	7980	8062	7988	7545	7586
Auto Theft	2337	1217	964	800	696	617	680	913	931	984	...	1096	1163	1507	1841	2335	2345	2726	2905	3228	2898
Break and Enter	4217	2492	2571	2881	2831	2313	1617														
Robbery	1487	1429	1179	946	619	410	367														
Theft Over	956	193	179	150	122	101	104														

5 rows × 24 columns

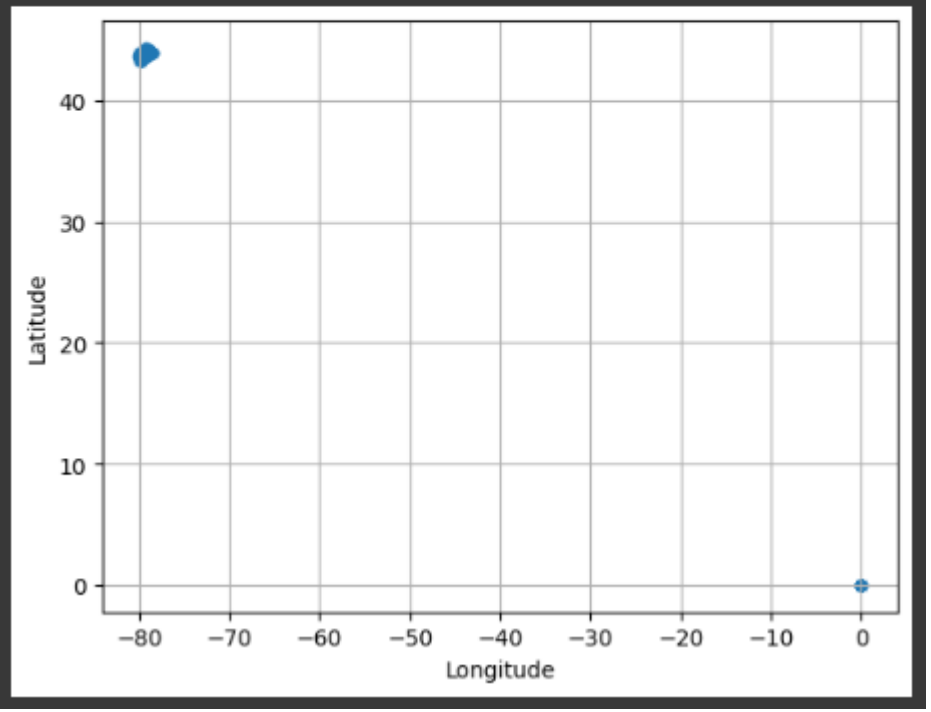


3. Explore and visualize the data to gain insights

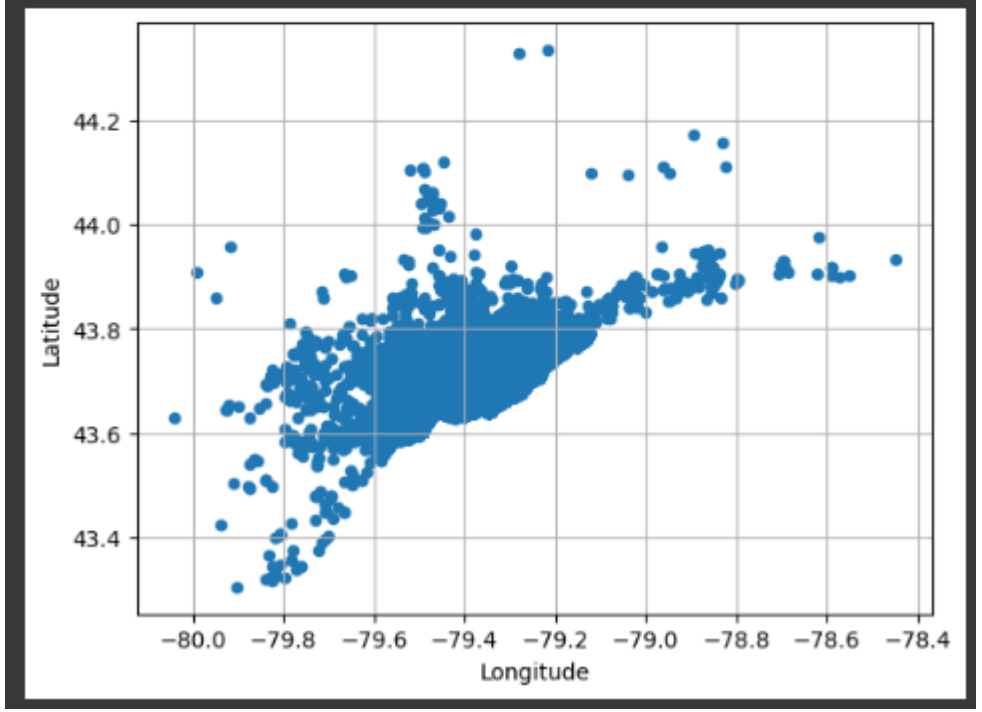
METHODOLOGY

3.5 Latitute and Longitude

```
### Exploring the data to gain insigths / Visualizing Goeg
import matplotlib.pyplot as plt
data.plot(kind="scatter", x="Longitude", y="Latitude", gri
plt.show()
```




```
### Replace ZERO values for NaN = "Not a Number"
### data.replace(0, np.nan, inplace=True) ### To replace all
cols = ["Latitude","Longitude"]
data[cols] = data[cols].replace(['0', 0], np.nan)
import matplotlib.pyplot as plt
data.plot(kind="scatter", x="Longitude", y="Latitude", grid=
plt.show()
```

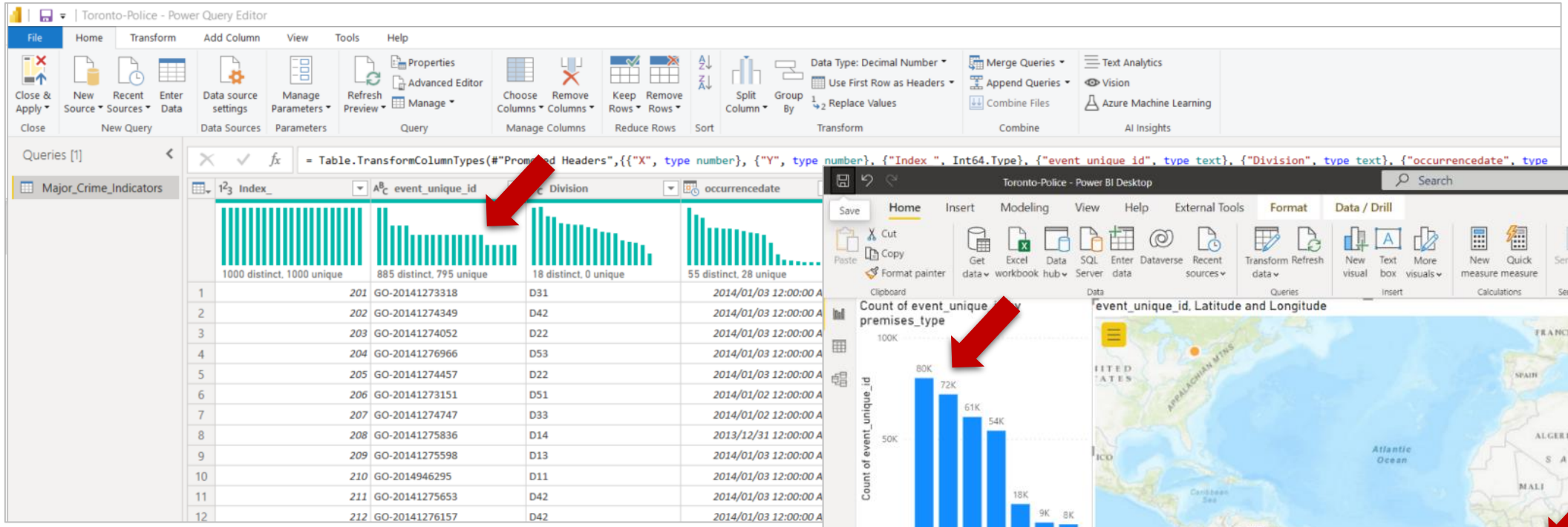


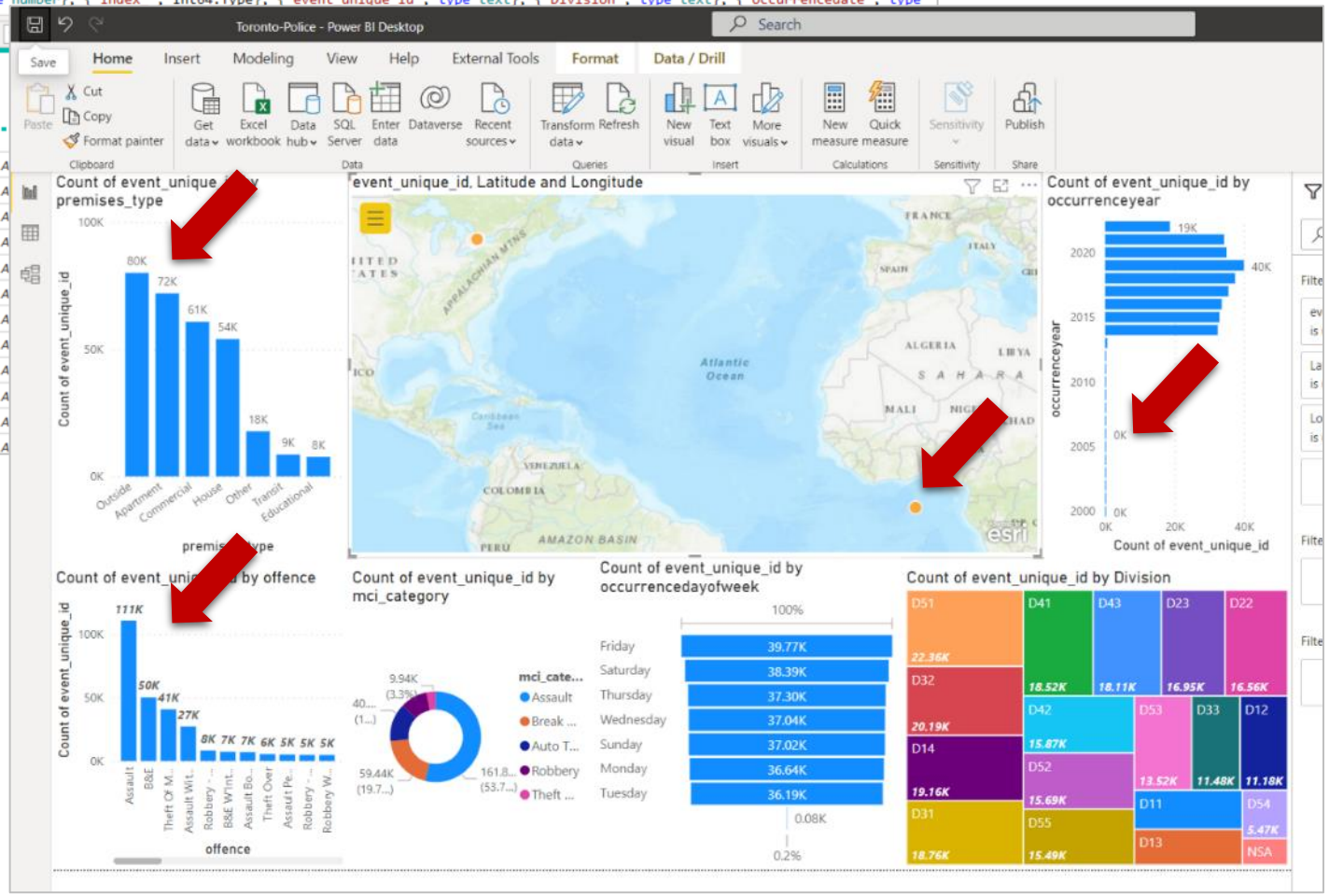

```
### Replace ZERO values for NaN = "Not a Number"
### data.replace(0, np.nan, inplace=True) ### To replace all the columns
cols = ["Latitude","Longitude"]
data[cols] = data[cols].replace(['0', 0], np.nan)
```

3. Explore and visualize the data to gain insights

METHODOLOGY

Microsoft PowerBI





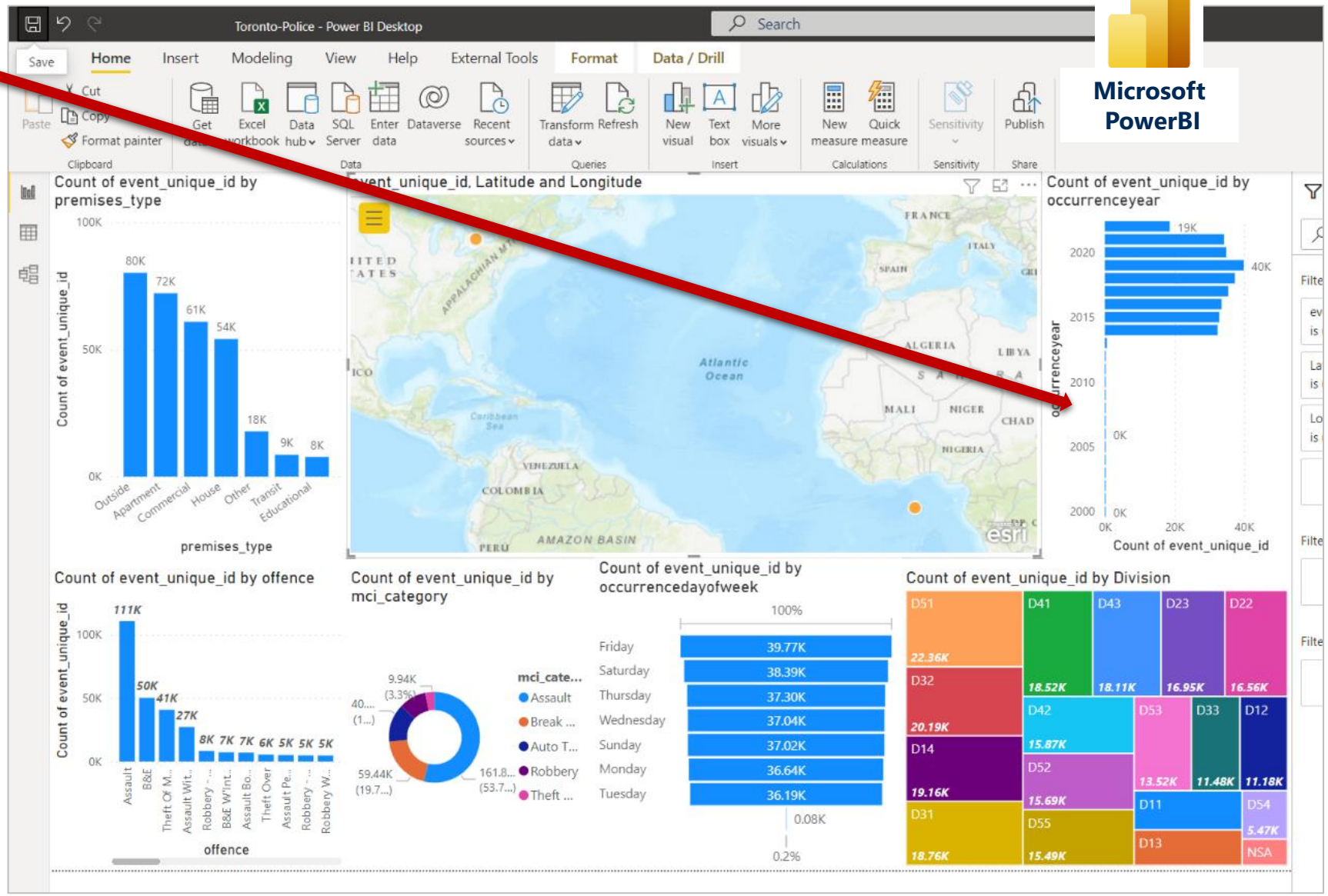
Additionally, we used a visualization tool (PowerBI) to explore, visualize and gain insights from the dataset. **Easy, fast, and visual**.

3. Explore and visualize the data to gain insights

METHODOLOGY

```
data['occurrenceyear'].value_counts()
```

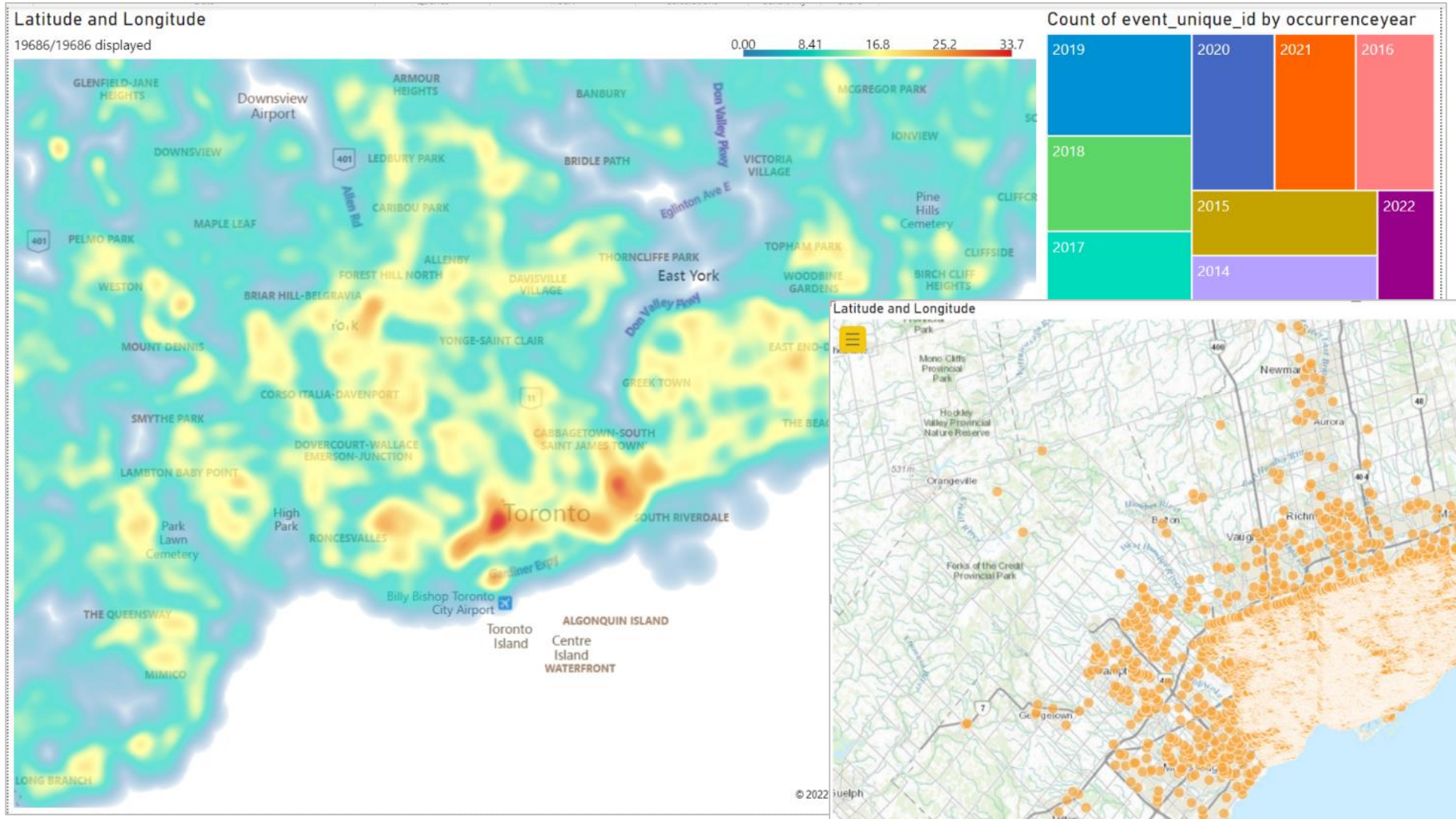
```
2019.0    39967
2018.0    37440
2017.0    35501
2020.0    34973
2021.0    34277
2016.0    33608
2015.0    32920
2014.0    32467
2022.0    18675
2013.0      596
2012.0      186
2011.0      130
2010.0       96
2009.0       73
2008.0       47
2007.0       34
2004.0       27
2000.0       26
2005.0       25
2001.0       21
2002.0       17
2003.0       15
2006.0       12
Name: occurrenceyear, dtype: int64
```



3. Explore and visualize the data to gain insights

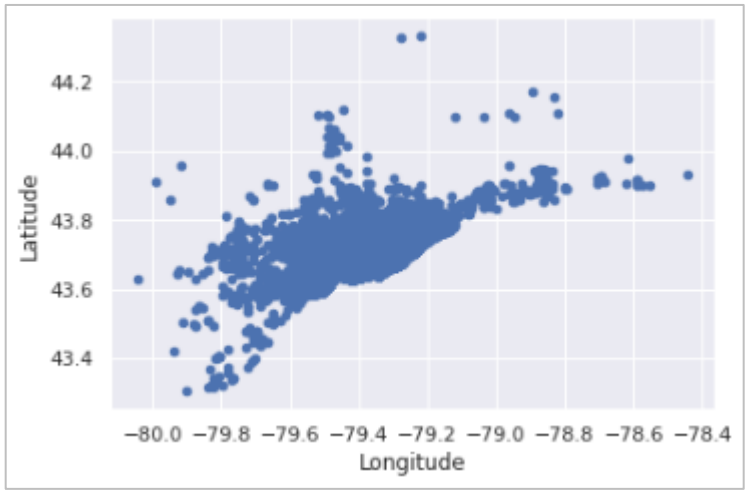
METHODOLOGY

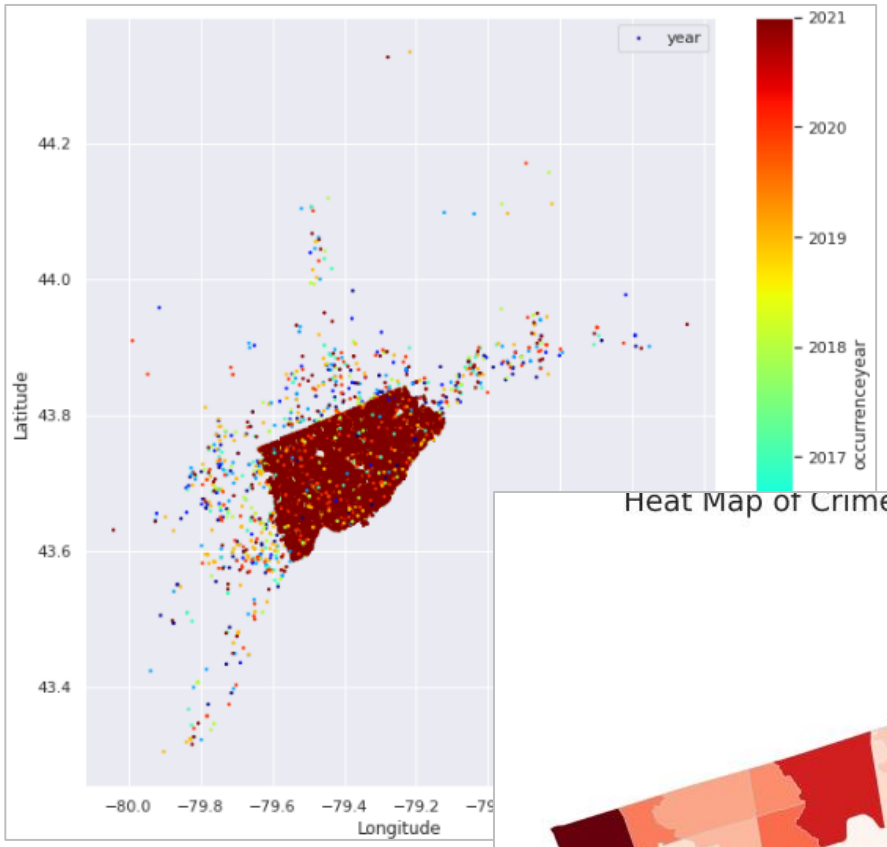
Microsoft PowerBI

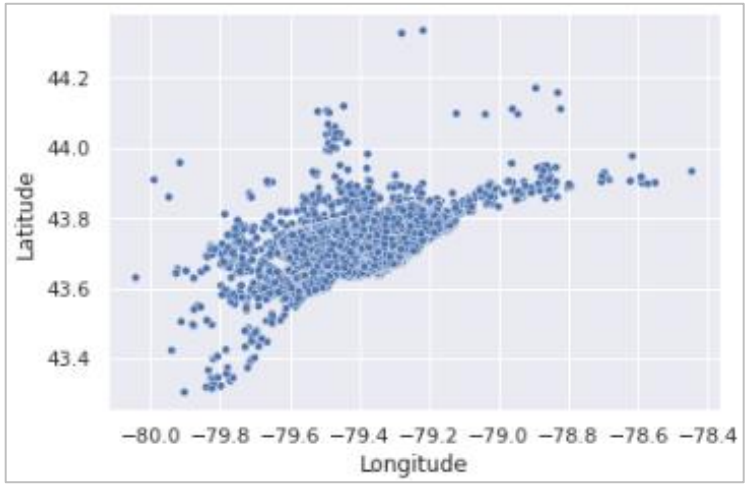


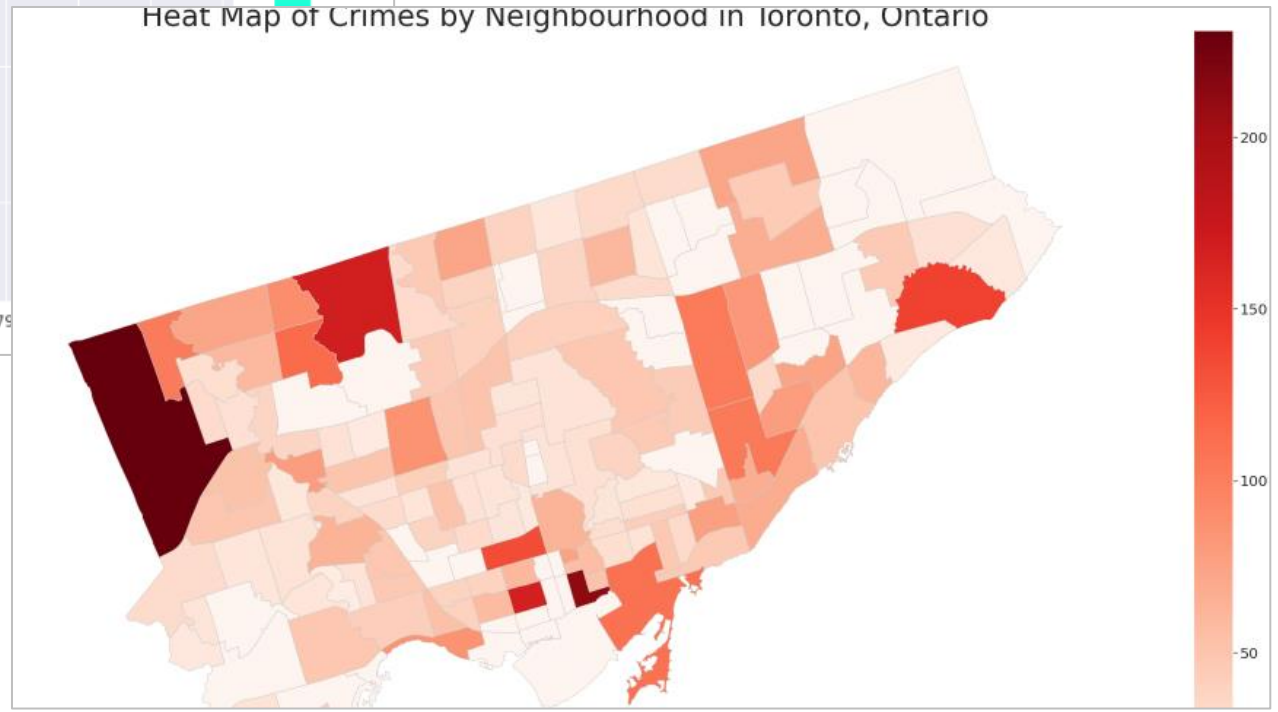
3. Explore and visualize the data to gain insights

METHODOLOGY









Google Collab

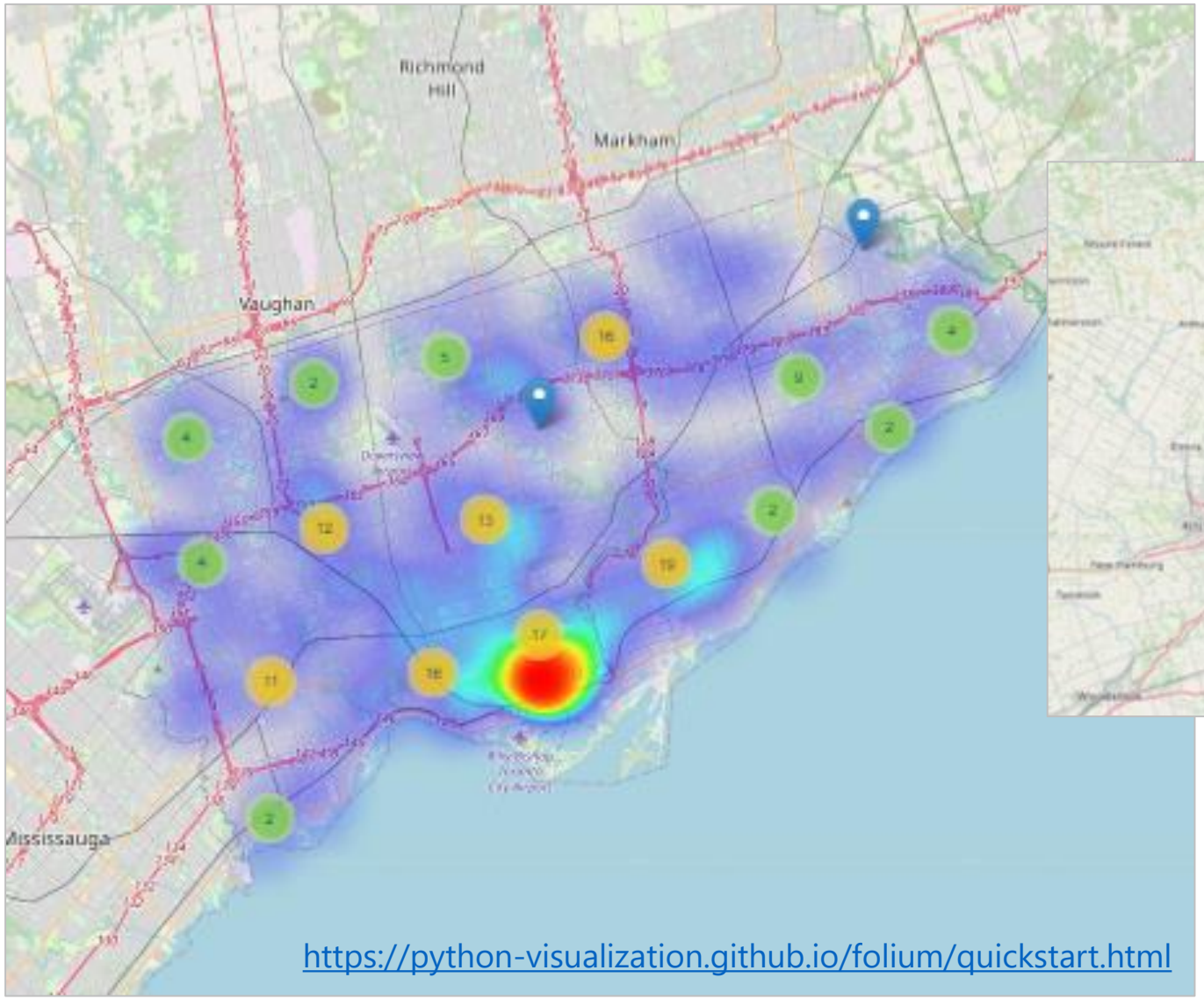
Powerful data visualizations features

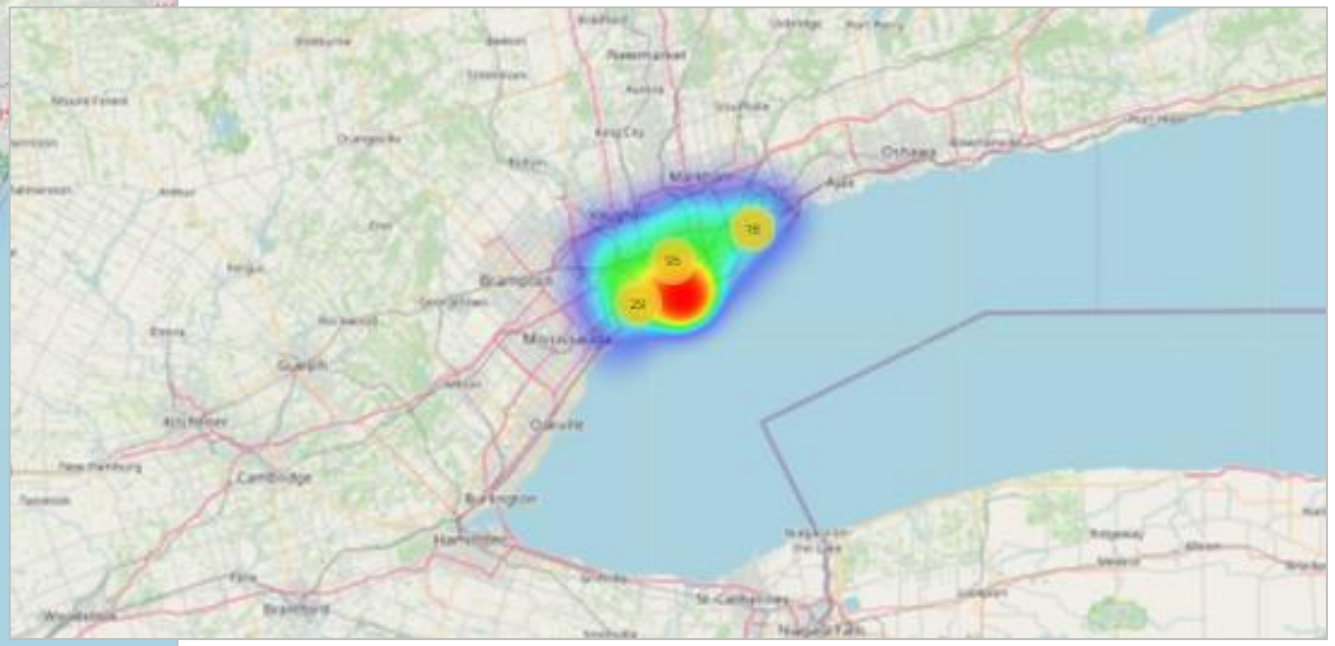
Using scatterplots, maps, and shape files…

3. Explore and visualize the data to gain insights

METHODOLOGY

Google Collab





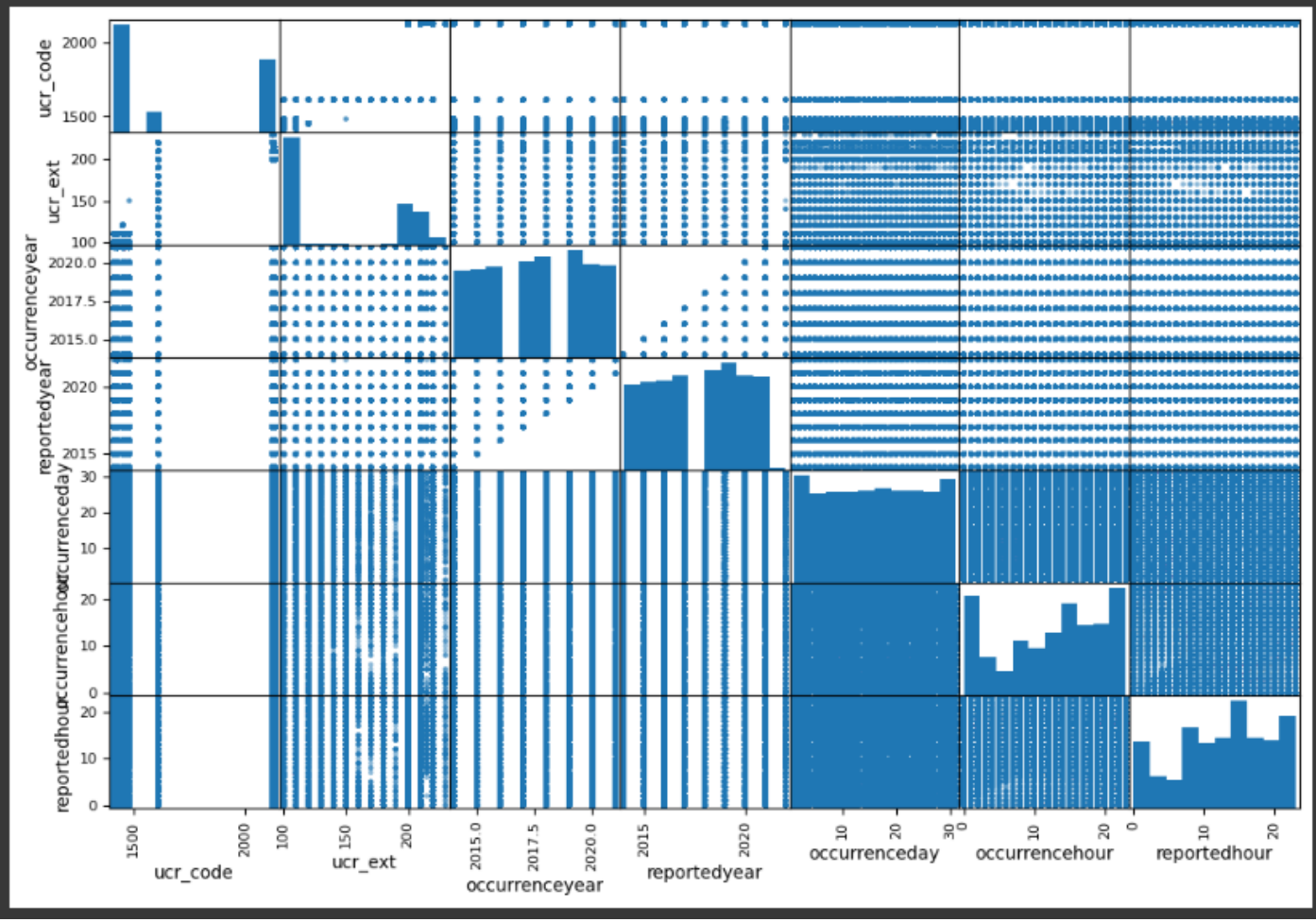
https://python-visualization.github.io/folium/quickstart.html

And also more **advanced resources** like Folium with zoom features and etc.

3. Explore and visualize the data to gain insights

METHODOLOGY

```
[54] ### Exploring the data to gain insigths / Looking for correlations - using pandas Scatter Matrix
     ### Scatter Matrix plots every numerical atribute against every other attribite, plus a histogram of each numerical atribu
     from pandas.plotting import scatter_matrix
     attributes = ["ucr_code", "ucr_ext", "occurrenceyear", "reportedyear", "occurrenceday", "occurrencehour", "reportedhour"]
     scatter_matrix(data[attributes], figsize=(12,8))
     plt.show()
```



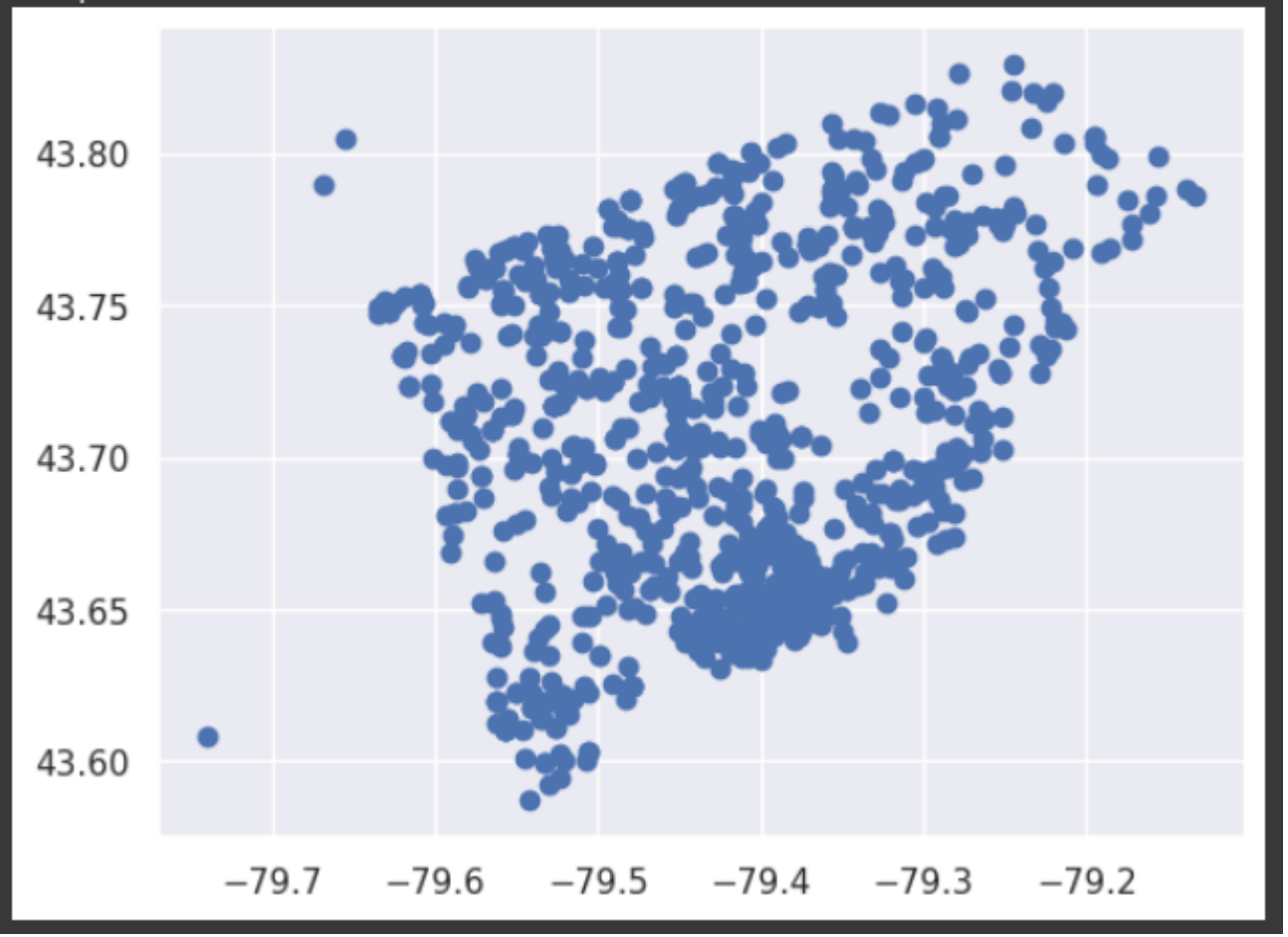

We need more data…

4. Prepare the data for the ML algorithm

METHODOLOGY

```
lat_long = df[['Latitude','Longitude']]
lat, longg = df.Latitude, df.Longitude
plt.scatter(longg, lat)
```

```
<matplotlib.collections.PathCollection at 0x7f5f2064d7f0>
```

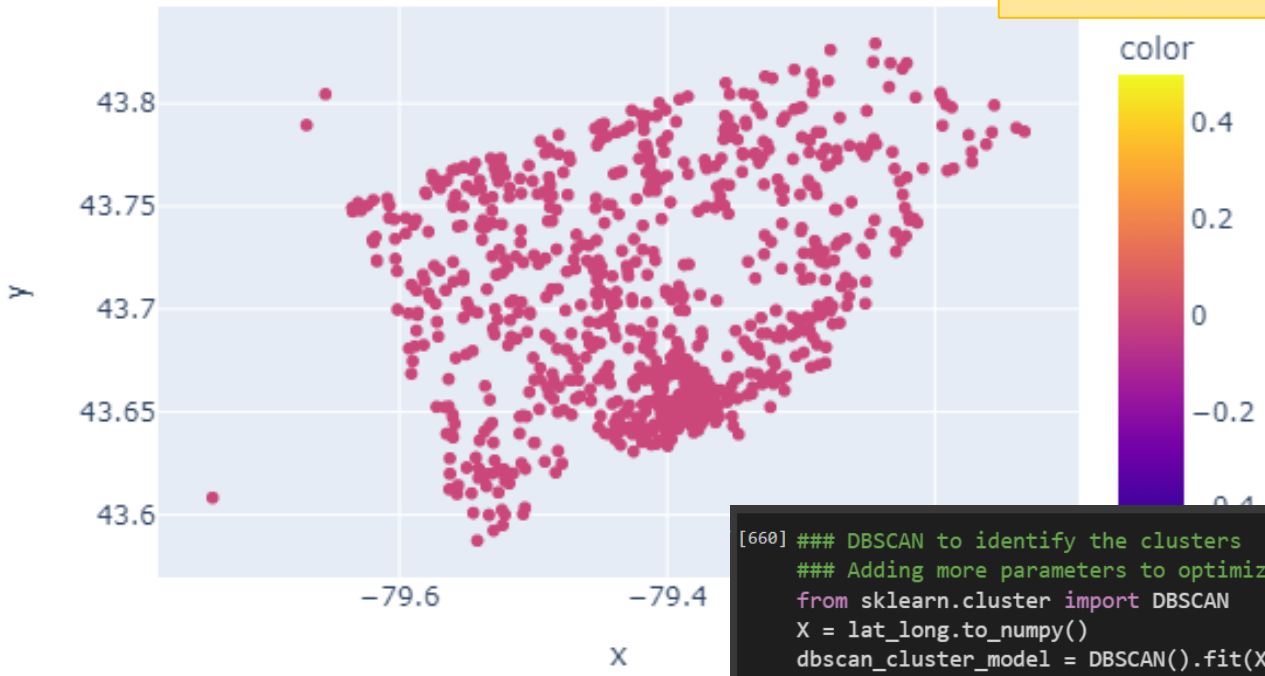


Plotting Latitude and **Longitude** values

5. Select a model and train it

METHODOLOGY

DBSCAN using **Latitude** and **Longitude** features




```
[660] ### DBSCAN to identify the clusters
      ### Adding more parameters to optimize DBSCAN
      from sklearn.cluster import DBSCAN
      X = lat_long.to_numpy()
      dbscan_cluster_model = DBSCAN().fit(X)
      dbscan_cluster_model.labels_
```

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

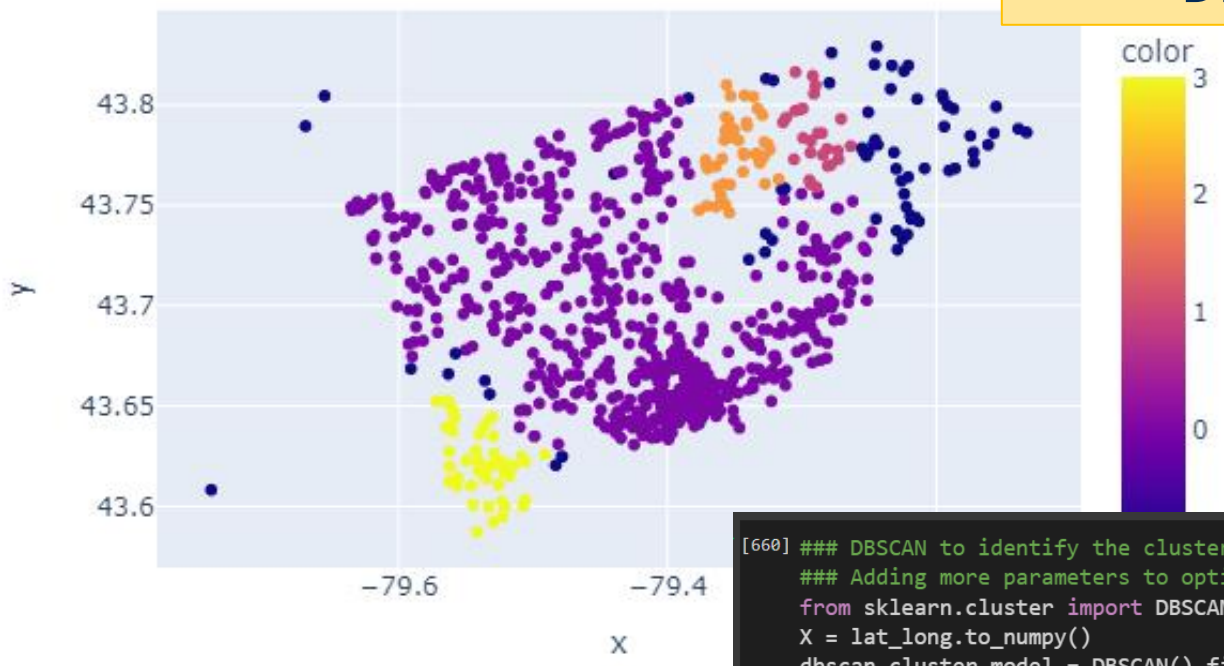
```
[661] df['cluster'] = dbscan_cluster_model.labels_
      df.head()
```

1 to 5 of 5 entries Filter

index	mci_category	occurrenceyear	premises_type	Latitude	Longitude	cluster
246327	Theft Over	2021.0	Apartment	43.69437813	-79.30056835	0
246328	Theft Over	2021.0	Outside	43.71844401	-79.60169835	0
246692	Theft Over	2021.0	Apartment	43.75760637	-79.40850721	0
246747	Theft Over	2021.0	Outside	43.6880498	-79.47154503	0
246944	Theft Over	2021.0	Apartment	43.60326759	-79.50627079	0

5. Select a model and train it

DBSCAN using **Latitude** and **Longitude** features




```
### Calculating the Silhouette Score // From -1 to 1
### This score represents how well the data points has b
### while negative points mean your K-means algorithm ha
from sklearn.metrics import silhouette_score as ss
ss(X, df['cluster'])
```

```
0.03517664393802231
```

```
[660] ### DBSCAN to identify the clusters
      ### Adding more parameters to optimize DBSCAN
      from sklearn.cluster import DBSCAN
      X = lat_long.to_numpy()
      dbscan_cluster_model = DBSCAN().fit(X)
      dbscan_cluster_model.labels_
```

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

```
[661] df['cluster'] = dbscan_cluster_model.labels_
      df.head()
```

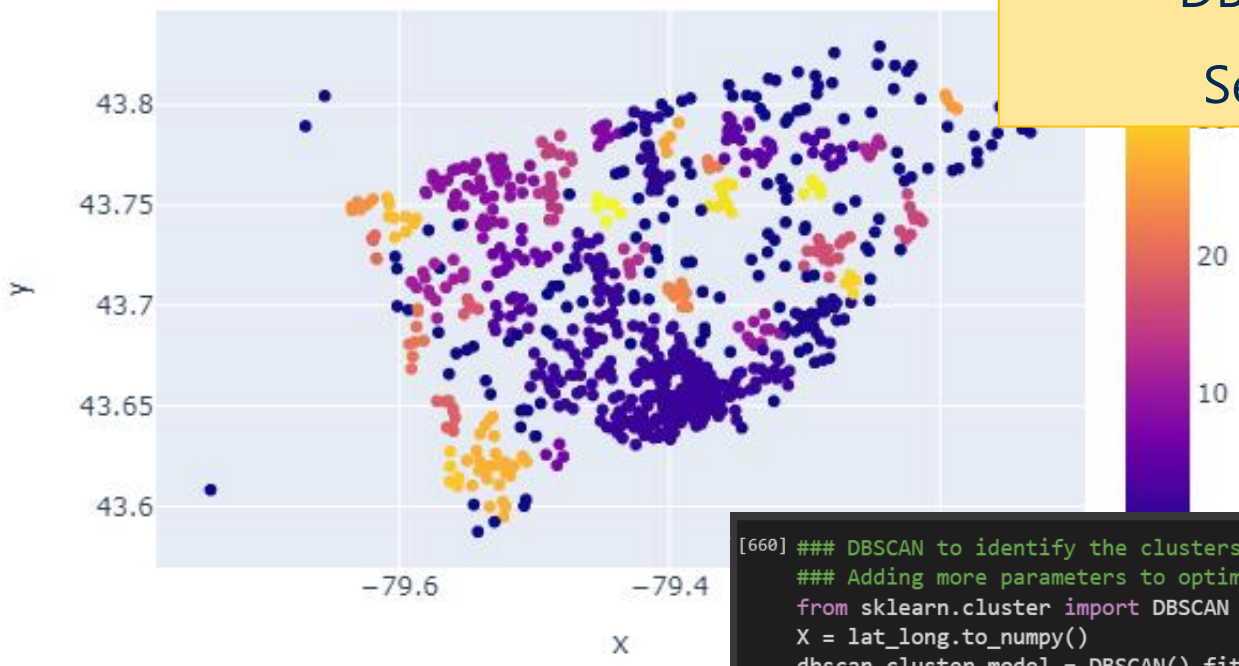
1 to 5 of 5 entries Filter

index	mci_category	occurrenceyear	premises_type	Latitude	Longitude	cluster
246327	Theft Over	2021.0	Apartment	43.69437813	-79.30056835	0
246328	Theft Over	2021.0	Outside	43.71844401	-79.60169835	0
246692	Theft Over	2021.0	Apartment	43.75760637	-79.40850721	0
246747	Theft Over	2021.0	Outside	43.6880498	-79.47154503	0
246944	Theft Over	2021.0	Apartment	43.60326759	-79.50627079	0

6. Fine-tune your model

DBSCAN using **Latitude** and **Longitude** features

Several combinations for *eps* and min_samples




```
### Calculating the Silhouette Score // From -1 to 1
### This score represents how well the data points has
### while negative points mean your K-means algorithm
ss(X, df['cluster'])
```

```
0.0681281414634729
```

```
[660] ### DBSCAN to identify the clusters
      ### Adding more parameters to optimize DBSCAN
      from sklearn.cluster import DBSCAN
      X = lat_long.to_numpy()
      dbscan_cluster_model = DBSCAN().fit(X)
      dbscan_cluster_model.labels_
```

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

```
[661] df['cluster'] = dbscan_cluster_model.labels_
      df.head()
```

1 to 5 of 5 entries Filter

index	mci_category	occurrenceyear	premises_type	Latitude	Longitude	cluster
246327	Theft Over	2021.0	Apartment	43.69437813	-79.30056835	0
246328	Theft Over	2021.0	Outside	43.71844401	-79.60169835	0
246692	Theft Over	2021.0	Apartment	43.75760637	-79.40850721	0
246747	Theft Over	2021.0	Outside	43.6880498	-79.47154503	0
246944	Theft Over	2021.0	Apartment	43.60326759	-79.50627079	0

6. Fine-tune your model

METHODOLOGY

DBSCAN using **Latitude** and **Longitude** features

Adding new features (**OHE** > mci_category & premises)

```
[92] ### One Hot Encoder for mci_category and premises_type field
     ohe = OneHotEncoder()
     ohe_data = pd.get_dummies(df, columns = ['mci_category','premises_type'])
     ohe_data.columns
```

```
Index(['occurrenceyear', 'Latitude', 'Longitude', 'cluster',
       'mci_category_Theft Over', 'premises_type_Apartment',
       'premises_type_Commercial', 'premises_type_Educational',
       'premises_type_House', 'premises_type_Other', 'premises_type_Outside',
       'premises_type_Transit'],
      dtype='object')
```

```
[93] ### One Hot Encoder for mci_category and premises_type field
     ### Merge data - ohe_data + df
     df = pd.concat([ohe_data], axis = 1)
     df.head()
```

index	occurrenceyear	Latitude	Longitude	cluster	mci_category_Theft Over	premises_type_Apartment	premise
246327	2021.0	43.69437813	-79.30056835	0	1	1	
246328	2021.0	43.71844401	-79.60169835	-1	1	0	
246692	2021.0	43.75760637	-79.40850721	1	1	1	
246747	2021.0	43.6880498	-79.47154503	2	1	0	
246944	2021.0	43.60326759	-79.50627079	-1	1	1	

Show 25 per page

6. Fine-tune your model

METHODOLOGY

Silhouette Analysis using k-means to find the ideal # of clusters

For 3 clusters the AVG Silhouette Score = 0.6312005687664468 Next...
For 6 clusters the AVG Silhouette Score = 0.6242680845731877 Next...
For 9 clusters the AVG Silhouette Score = 0.6219071008417552 Next...
For 12 clusters the AVG Silhouette Score = 0.6039122205183359 Next...
For 15 clusters the AVG Silhouette Score = 0.6502589731015673 Next...
For 18 clusters the AVG Silhouette Score = 0.694167706152834 Next...
For 21 clusters the AVG Silhouette Score = 0.7511084295804148 Next...
For 24 clusters the AVG Silhouette Score = 0.7810515666285528 Next...
For 27 clusters the AVG Silhouette Score = 0.823988304607556 Next...
For 30 clusters the AVG Silhouette Score = 0.8435800167012812 Next...
For 33 clusters the AVG Silhouette Score = 0.8571568601223338 Next...
For 36 clusters the AVG Silhouette Score = 0.8761368349346094 Next...
For 39 clusters the AVG Silhouette Score = 0.8956240293835462 Next...
For 42 clusters the AVG Silhouette Score = 0.9090275448838063 Next...
For 45 clusters the AVG Silhouette Score = 0.9224878434813757 Next...
For 48 clusters the AVG Silhouette Score = 0.9337492605790139 Next...
For 51 clusters the AVG Silhouette Score = 0.9430669405354508 Next...
For 54 clusters the AVG Silhouette Score = 0.9109089527785104 Next...
For 57 clusters the AVG Silhouette Score = 0.8847501953993314 Next...
For 60 clusters the AVG Silhouette Score = 0.859151251299482 Next...
For 63 clusters the AVG Silhouette Score = 0.8013646650548201 Next...
For 66 clusters the AVG Silhouette Score = 0.7511701226832084 Next...
For 69 clusters the AVG Silhouette Score = 0.7358522217798414 Next...
For 72 clusters the AVG Silhouette Score = 0.7271377592154833 Next...
For 75 clusters the AVG Silhouette Score = 0.7130372623601136 Next...
For 78 clusters the AVG Silhouette Score = 0.6836004200751056 Next...
For 81 clusters the AVG Silhouette Score = 0.6936645475506825 Next...
For 84 clusters the AVG Silhouette Score = 0.6629023565053651 Next...
For 87 clusters the AVG Silhouette Score = 0.6312766149553531 Next...
For 90 clusters the AVG Silhouette Score = 0.6521264511513082 Next...
For 93 clusters the AVG Silhouette Score = 0.6574746989205946 Next...
For 96 clusters the AVG Silhouette Score = 0.6453992305767153 Next...
For 99 clusters the AVG Silhouette Score = 0.6196287186575458 Next...

7. Conclusions

- Before you fine tune your model try to find relevant features for you data and expected results

Results and Model Performance

3 tested scenarios:

(1) DBSCAN

Silhouette score: N/A only 1 cluster

(2) DBSCAN with parameters

Silhouette score: `0.0351766439380223 1`

(3) DBSCAN several times

Silhouette score: `0.0681281414634729`

(4) DBSCAN added features // OneHotEncoder

Silhouette score: `0.822814785408203 1`

```
### Calculating the Silhouette Score // From -1 to 1
### This score represents how well the data points has
 been clustered, and scores above 0 are seen as good,
### while negative points mean your algorithm has put
that data point in the wrong cluster.
```

Why did we choose DBSCAN? Because we don't want to define the number of clusters. We would like the algo to define it for us.

Challenges

- Team communication and collaboration (e-mail, WhatsApp, etc.)
- Share the code with the whole team and collaborate (GitHub)
- Make sure you know the data you are working on. Do not make wrong assumptions about your data – **occurrence-date** – **occurrence-year**
- Working with **date fields** is always challenging – transformation, calculation, etc.
- **Time consuming** process (CPU/GPU time)

Your session crashed after using all available RAM. If you are interested in access to high-RAM runtimes, you may want to check out Colab Pro.

View runtime logs

CHALLENGES

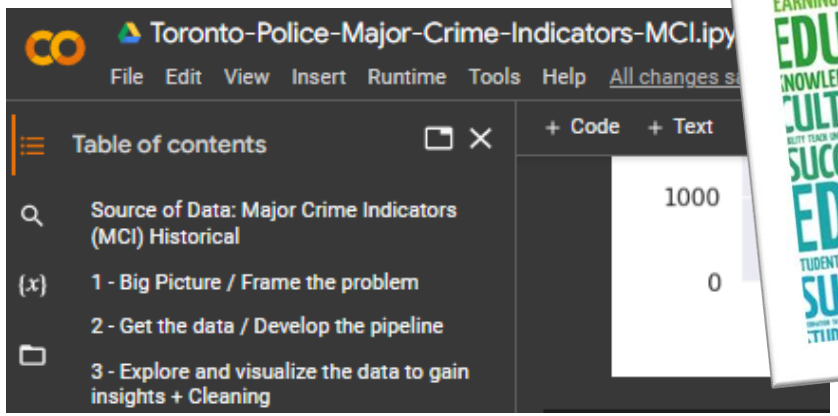
- Really great tool and easy to install
- Documentation is good with good code samples (several links are broken)
- Speed up the DS analysis and process, so you can dedicate your time to other tasks, like presenting the data, results, etc.
- https://github.com/caiogasparine/SCS_3253_061-Machine-Learning/blob/main/Toronto_Police_MCI_with_pyCaret.ipynb

Lessons Learned (1)

- Tools are important, but team **collaboration and team interaction** are key components in the analysis and scenario validation.
- **Reproducibility** – only if we keep the data cleaning process documented in the code will be possible to reproduce exactly the same result utilizing the original dataset. Avoid manual data cleaning directly to the dataset.
- **Quick reading** – The visualization tool (PowerBI) allowed us to do a "quick reading" on the available data, generating quick and easy insights (simple ones).
- **Correlation** – Get rid of highly correlated features because they don't add any value to your model
- **Data cleaning** – Be prepared for the data cleaning process. It is not a process but an iteration with several cycles.
- **Code** – Be familiar with your IDE (e.g., Collab) and use its features. It will make your life much easier (e.g., *Sections* and *Table of contents* to organize the code)

Lessons Learned (2)

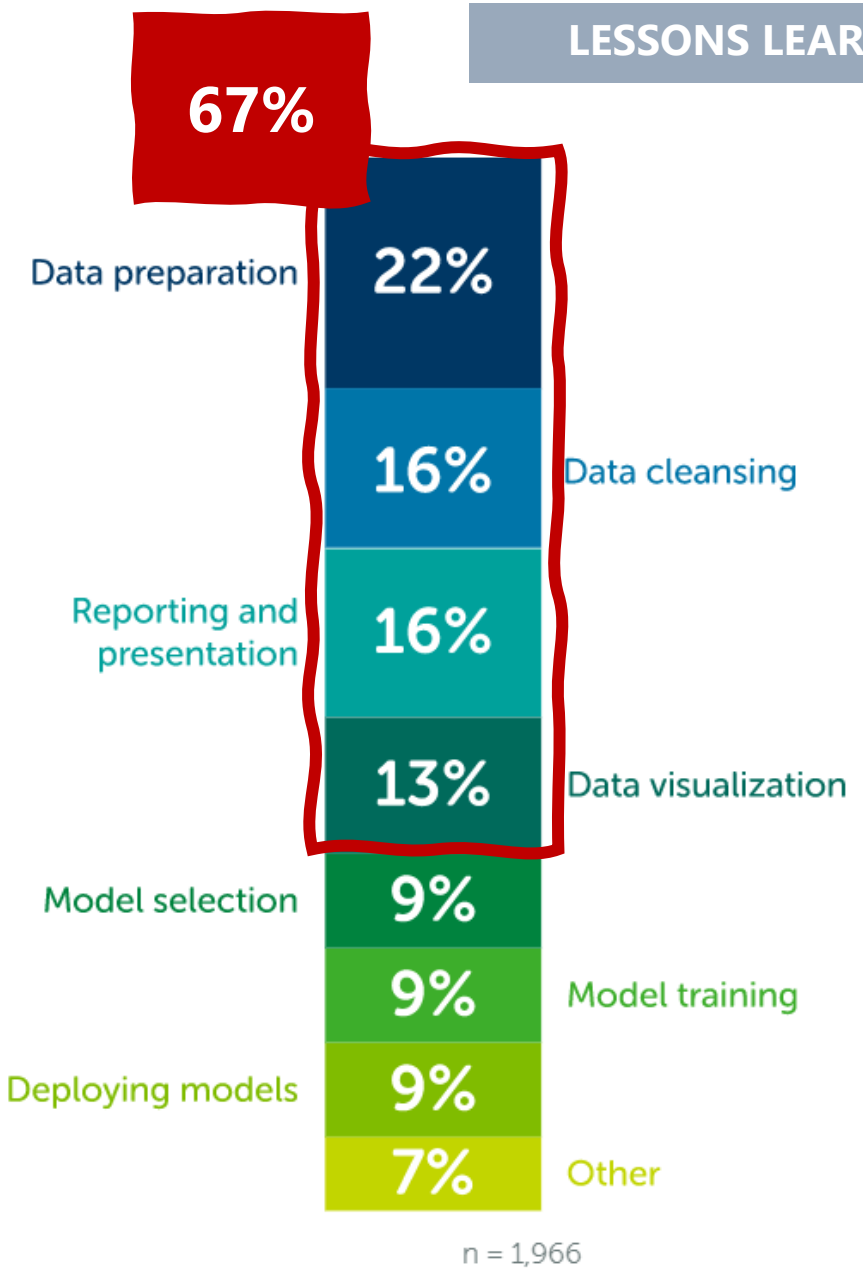
- **New best friends** – "Stack Overflow" and "Google"
- **Table of contents** – Use it to help you navigate in your code. It will be very helpful when your code grows.
- **Clear problem** – Have a clear problem defined before you start working on your data, otherwise, you will be moving in circles.
- **Data cleansing** – The earlier - *in the process* - you clear your dataset the better will be your vision for the important data…
- **Organize your code** – Using sections will help you a lot when moving through your code…



DATA PROFESSIONALS AT WORK

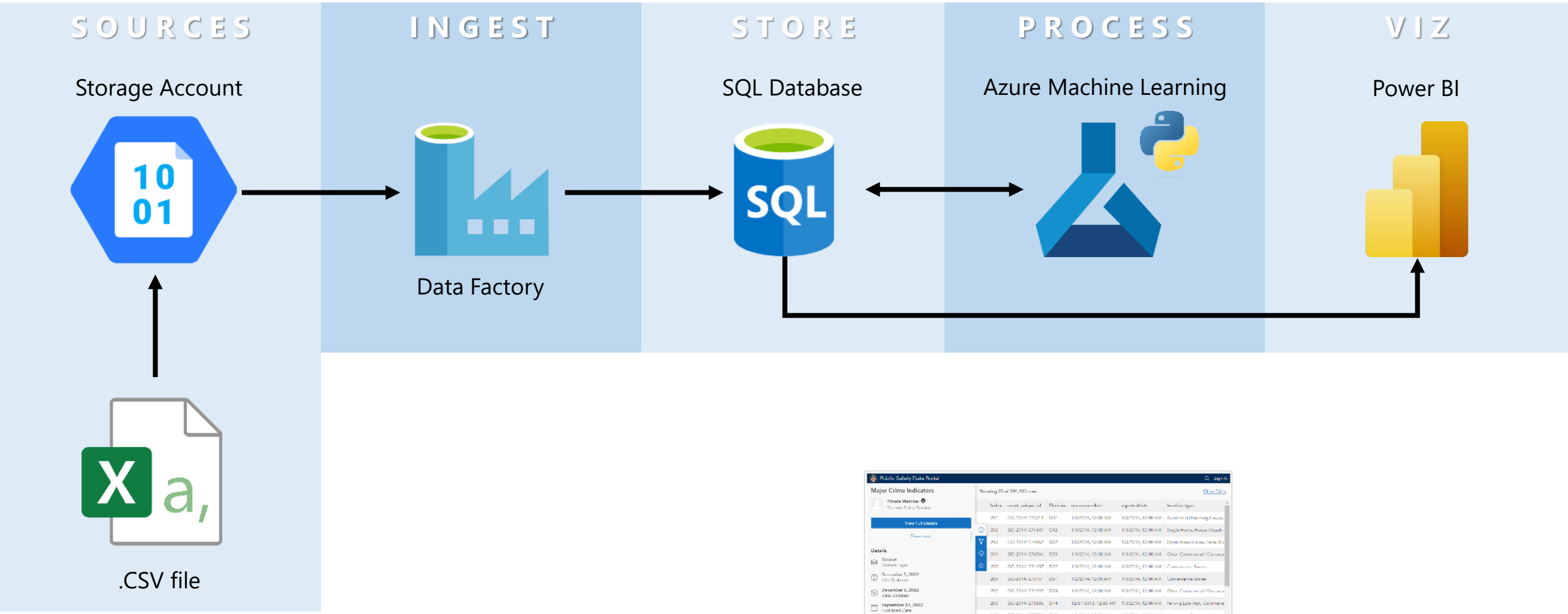
How do data scientists spend their time?

Data professionals spend their time on a variety of tasks that require diverse technical and non-technical skills. Respondents indicated they spend about 37.75% of their time on data preparation and cleansing. Beyond preparing and cleaning data, interpreting results remains critical. **Data visualization** (12.99%) and demonstrating data's value through reporting and presentation (16.20%) are essential steps toward making data actionable and providing answers to critical questions. Working with models through selection, training, and deployment takes about 26.44% of respondents' time (-8.56% YoY).



https://www.anaconda.com/state-of-data-science-report-2022

Enterprise Solution Architecture



https://data.torontopolice.on.ca/pages/major-crime-indicators

Team Members

Illidan Yuan

www.linkedin

yilan_Illdian@hotmail.com

Caio Gasparine

IT Project Manager

Data & Analytics

https://www.linkedin.com/in/caiogasparine/

caiogasparine@gmail.com

Olivier Sangam

Electrical Designer

https://www.linkedin.com/in/olivier-mawaba-sangam-46a973108/

mawoliv@gmail.com

Fábio Queiroz

Sr. Data Engineer

https://www.linkedin.com/in/fabiomsq/

fabio.maia82@gmail.com

Tools we used...

Google Collab

GitHub

Microsoft PowerBI

Arc GIS

Python

Microsoft Azure

stackoverflow

PYCARET

References

- Géron, Aurélien 2022, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*, 3rd Edition, O'Reilly Media.
- Toronto Police Service, https://data.torontopolice.on.ca/
- Toronto Police Service, Public Safety Data Portal, February 2023, MCI documentation, https://data.torontopolice.on.ca/datasets/TorontoPS::major-crime-indicators-1/about
- City of Toronto, Open data, Neighbourhoods, https://open.toronto.ca/dataset/neighbourhoods/
- Mukund Vemuri 2019, *Create a geographic heat map of the City of Toronto in Python*, https://medium.com/@m_vemuri/create-a-geographic-heat-map-of-the-city-of-toronto-in-python-cd2ae0f8be55
- Blob Storage, https://learn.microsoft.com/en-us/azure/storage/blobs/storage-blobs-introduction
- Data Factory, https://learn.microsoft.com/en-us/azure/data-factory/
- Machine Learning, https://learn.microsoft.com/en-us/azure/machine-learning/
- Power BI, https://learn.microsoft.com/en-us/power-bi/fundamentals/videos
- Azure icons, https://learn.microsoft.com/en-us/azure/architecture/icons/

* - references consulted between January 24, 2023, and April 6, 2023.

References (2)

- Anaconda website, 2023, https://www.anaconda.com/state-of-data-science-report-2022
- Jiri Stodulka 2023, *Toronto Crime and Folium*, website, https://www.jiristodulka.com/post/toronto-crime/
- Folium, QuickStart, website, 2023, https://python-visualization.github.io/folium/quickstart.html
- Pycaret, GitHub, 2023, https://github.com/pycaret/pycaret/blob/master/tutorials/Tutorial%20-%20Clustering.ipynb
- Statistics Canada, UCR Code Uniform Crime Reporting Survey (UCR), website, 2023, https://www23.statcan.gc.ca/imdb/p2SV.pl?Function=getSurvey&SDDS=3302

* - references consulted between January 24, 2023, and April 6, 2023.

Thank you! ;-)

SCS 3253-061 – Machine Learning

Instructor: *Saeid Abolfazli*

Team members:

Caio Gasparine | Fábio Queiroz | Olivier Sangam | Illidan Yuan

https://github.com/caiogasparine/SCS_3253_061-Machine-Learning