Project4

May 2, 2022

1 Project 4

Faraz Karim

UID: 117088397

1.1 Part 1: Getting Some Data

The dataset that I chose for this project comes from https://data.baltimorecity.gov/datasets/baltimore::part1-crime-data/, containing data on "Part 1" crimes within Baltimore, MD dating all the way back to 1963. Part 1 crimes are major crimes including homicide, shooting, robbery, aggravated assault, etc.

[134]:
```
import folium
import requests
import pandas as pd

data = pd.read_csv('part1CrimeData.csv', low_memory = False)
data
```

[134]:

	X	Y	RowID	CrimeDateTime	CrimeCode	\
0	1.424850e+06	569777.328405	1	2022/04/23 04:12:45+00	9S	
1	1.442033e+06	611635.789459	2	2022/04/21 11:34:27+00	9S	
2	1.421723e+06	592309.927233	3	2022/04/21 00:55:03+00	9S	
3	1.408272e+06	586720.112426	4	2022/04/20 16:01:00+00	6D	
4	1.434514e+06	592840.753893	5	2022/04/20 15:29:00+00	6F	
…	…	…	…	…	…	
518413	1.400822e+06	604541.096449	518414	1973/07/01 23:00:00+00	2A	
518414	1.409359e+06	598671.099707	518415	1970/06/15 00:01:00+00	2A	
518415	1.415457e+06	616506.081314	518416	1969/07/20 21:00:00+00	2A	
518416	1.394549e+06	593520.411508	518417	1966/01/01 01:00:00+00	2A	
518417	1.396465e+06	604781.500706	518418	1963/10/30 00:00:00+00	2A	

	Location	Description	Inside_Outside	Weapon	Post	\
0	4100 6TH ST	SHOOTING	Outside	FIREARM	913	
1	5900 BELAIR RD	SHOOTING	Outside	FIREARM	425	
2	300 SAINT PAUL ST	SHOOTING	Outside	FIREARM	111	
3	2700 WILKENS AVE	LARCENY FROM AUTO	NaN	NaN	834	

```
4       3500 E FAIRMOUNT AVE            LARCENY            NaN      NaN  223
…                       …                 …              …       …   …
518413   4000 SPRINGDALE AVE               RAPE              I    OTHER  621
518414   2400 ST STEPHENS CT               RAPE              I    OTHER  731
518415       5400 ROLAND AVE               RAPE            NaN    OTHER  534
518416       900 STAMFORD RD               RAPE              I    OTHER  823
518417     3100 FERNDALE AVE               RAPE              I    OTHER  622

          District         Neighborhood  Latitude  Longitude  \
0        SOUTHERN             BROOKLYN   39.2305   -76.6028
1       NORTHEAST              CEDMONT   39.3452   -76.5414
2         CENTRAL             DOWNTOWN   39.2924   -76.6135
3       SOUTHWEST             MILLHILL   39.2772   -76.6611
4       SOUTHEAST  BALTIMORE HIGHLANDS   39.2937   -76.5683
…             …                   …       …         …
518413  NORTHWEST  CENTRAL FOREST PARK   39.3262   -76.6872
518414    WESTERN            MONDAWMIN   39.3100   -76.6571
518415   NORTHERN          ROLAND PARK   39.3589   -76.6353
518416  SOUTHWEST           WEST HILLS   39.2960   -76.7095
518417  NORTHWEST          HOWARD PARK   39.3269   -76.7026

               GeoLocation            Premise VRIName  Total_Incidents
0       (39.2305,-76.6028)        PUBLIC AREA     NaN                1
1       (39.3452,-76.5414)             STREET     NaN                1
2       (39.2924,-76.6135)           HOSPITAL     NaN                1
3       (39.2772,-76.6611)                NaN     NaN                1
4       (39.2937,-76.5683)                NaN     NaN                1
…                      …                  …     …                  …
518413  (39.3262,-76.6872)  ROW/TOWNHOUSE-OCC     NaN                1
518414    (39.31,-76.6571)  ROW/TOWNHOUSE-OCC     NaN                1
518415  (39.3589,-76.6353)                NaN     NaN                1
518416   (39.296,-76.7095)  ROW/TOWNHOUSE-OCC     NaN                1
518417  (39.3269,-76.7026)  ROW/TOWNHOUSE-OCC     NaN                1

[518418 rows x 18 columns]
```

[135]:
```
print("# of GeoLocation values: ", data[pd.
 ↪notnull(data["GeoLocation"])]["GeoLocation"].count())
print("# of Latitude values: ", data[pd.notnull(data["Latitude"])]["Latitude"].
 ↪count())
print("# of Longitude values: ", data[pd.
 ↪notnull(data["Longitude"])]["Longitude"].count())
```

```
# of GeoLocation values:  518418
# of Latitude values:  517530
# of Longitude values:  517530
```

[136]:
```
null_data = data[data['Latitude'].isnull()]
null_data
```

[136]:

	X	Y	RowID	CrimeDateTime	CrimeCode	\
846	NaN	NaN	847	2022/04/10 14:00:00+00	2A	
5203	NaN	NaN	5204	2022/02/21 08:21:00+00	4B	
6716	NaN	NaN	6717	2022/02/05 11:00:00+00	6D	
10574	NaN	NaN	10575	2021/12/26 10:49:00+00	6D	
16053	NaN	NaN	16054	2021/11/06 15:47:00+00	3AJO	
…	..	..	…	…	…	
517247	NaN	NaN	517248	2011/01/09 00:00:00+00	6E	
517788	NaN	NaN	517789	2011/01/04 11:27:00+00	6G	
517816	NaN	NaN	517817	2011/01/04 11:27:00+00	6G	
517841	NaN	NaN	517842	2011/01/04 19:30:00+00	4E	
517890	NaN	NaN	517891	2011/01/04 19:30:00+00	4E	

	Location	Description	Inside_Outside	\
846	3500 Carriage Hill Court	RAPE	NaN	
5203	5000 BALTIMORE NATIONAL PIKE	AGG. ASSAULT	Outside	
6716	NaN	LARCENY FROM AUTO	NaN	
10574	NaN	LARCENY FROM AUTO	NaN	
16053	NaN	ROBBERY - CARJACKING	NaN	
…	…	…	…	
517247	2800 OLD FRANKLIN ST	LARCENY	O	
517788	100 VILLAGE SQUARE	LARCENY	I	
517816	100 VILLAGE SQUARE	LARCENY	I	
517841	1100 MONDAWMIN MA	COMMON ASSAULT	I	
517890	1100 MONDAWMIN MA	COMMON ASSAULT	I	

	Weapon	Post	District	Neighborhood	Latitude	\
846	NaN	NaN	NaN	NaN	NaN	
5203	KNIFE_CUTTING_INSTRUMENT	NaN	NaN	NaN	NaN	
6716	NaN	NaN	NaN	NaN	NaN	
10574	NaN	NaN	NaN	NaN	NaN	
16053	OTHER	NaN	NaN	NaN	NaN	
…	…	…	…	…	…	
517247	NaN	NaN	NaN	NaN	NaN	
517788	NaN	NaN	NaN	NaN	NaN	
517816	NaN	NaN	NaN	NaN	NaN	
517841	NaN	NaN	NaN	NaN	NaN	
517890	NaN	NaN	NaN	NaN	NaN	

	Longitude	GeoLocation	Premise	VRIName	Total_Incidents
846	NaN	(,)	NaN	NaN	1
5203	NaN	(,)	OTHER/RESIDENTIAL	NaN	1
6716	NaN	(,)	NaN	NaN	1
10574	NaN	(,)	NaN	NaN	1

```
16053         NaN         (,)                    NaN     NaN                1
…           …         …                      …     …             …
517247        NaN         (,)  STREET                    NaN                1
517788        NaN         (,)  BARBER/BEAUTY SHOP       NaN                1
517816        NaN         (,)  BARBER/BEAUTY SHOP       NaN                1
517841        NaN         (,)  SHOPPING MALLS/CNTR      NaN                1
517890        NaN         (,)  SHOPPING MALLS/CNTR      NaN                1

[888 rows x 18 columns]
```

[137]:
```
data = data.dropna(subset = ['Latitude', 'Longitude'])
data
```

[137]:
```
                  X              Y   RowID           CrimeDateTime CrimeCode  \
0       1.424850e+06  569777.328405       1  2022/04/23 04:12:45+00        9S
1       1.442033e+06  611635.789459       2  2022/04/21 11:34:27+00        9S
2       1.421723e+06  592309.927233       3  2022/04/21 00:55:03+00        9S
3       1.408272e+06  586720.112426       4  2022/04/20 16:01:00+00        6D
4       1.434514e+06  592840.753893       5  2022/04/20 15:29:00+00        6F
…             …           …     …                      …       …
518413  1.400822e+06  604541.096449  518414  1973/07/01 23:00:00+00        2A
518414  1.409359e+06  598671.099707  518415  1970/06/15 00:01:00+00        2A
518415  1.415457e+06  616506.081314  518416  1969/07/20 21:00:00+00        2A
518416  1.394549e+06  593520.411508  518417  1966/01/01 01:00:00+00        2A
518417  1.396465e+06  604781.500706  518418  1963/10/30 00:00:00+00        2A

                    Location        Description Inside_Outside   Weapon Post  \
0                4100 6TH ST           SHOOTING        Outside  FIREARM  913
1             5900 BELAIR RD           SHOOTING        Outside  FIREARM  425
2          300 SAINT PAUL ST           SHOOTING        Outside  FIREARM  111
3           2700 WILKENS AVE  LARCENY FROM AUTO            NaN      NaN  834
4        3500 E FAIRMOUNT AVE            LARCENY            NaN      NaN  223
…                        …                 …             …       …  …
518413   4000 SPRINGDALE AVE              RAPE              I    OTHER  621
518414   2400 ST STEPHENS CT              RAPE              I    OTHER  731
518415       5400 ROLAND AVE              RAPE            NaN    OTHER  534
518416       900 STAMFORD RD              RAPE              I    OTHER  823
518417     3100 FERNDALE AVE              RAPE              I    OTHER  622

          District         Neighborhood  Latitude  Longitude  \
0        SOUTHERN             BROOKLYN   39.2305   -76.6028
1       NORTHEAST              CEDMONT   39.3452   -76.5414
2         CENTRAL             DOWNTOWN   39.2924   -76.6135
3       SOUTHWEST             MILLHILL   39.2772   -76.6611
4       SOUTHEAST  BALTIMORE HIGHLANDS   39.2937   -76.5683
…             …                   …       …         …
518413  NORTHWEST  CENTRAL FOREST PARK   39.3262   -76.6872
```

```
518414    WESTERN            MONDAWMIN   39.3100   -76.6571
518415   NORTHERN          ROLAND PARK   39.3589   -76.6353
518416  SOUTHWEST           WEST HILLS   39.2960   -76.7095
518417  NORTHWEST          HOWARD PARK   39.3269   -76.7026

               GeoLocation            Premise VRIName  Total_Incidents
0       (39.2305,-76.6028)        PUBLIC AREA     NaN                1
1       (39.3452,-76.5414)             STREET     NaN                1
2       (39.2924,-76.6135)           HOSPITAL     NaN                1
3       (39.2772,-76.6611)                NaN     NaN                1
4       (39.2937,-76.5683)                NaN     NaN                1
…                      …                  …     …                  …
518413  (39.3262,-76.6872)  ROW/TOWNHOUSE-OCC     NaN                1
518414    (39.31,-76.6571)  ROW/TOWNHOUSE-OCC     NaN                1
518415  (39.3589,-76.6353)                NaN     NaN                1
518416   (39.296,-76.7095)  ROW/TOWNHOUSE-OCC     NaN                1
518417  (39.3269,-76.7026)  ROW/TOWNHOUSE-OCC     NaN                1

[517530 rows x 18 columns]
```

I downloaded the .csv file, uploaded it to the Jupyter notebook, and read it using pandas. Just based on the 'GeoLocation' column, it seems like there is a location for every row. However if we look at the 'Latitude' and 'Longitude' columns, we can see there are 88 missing values that are being marked as (,) in the 'GeoLocation' column. It doesn't make sense to impute location's since they are not dependent on the data, so I am removing rows with missing locations.

1.2 Part 2: Making a Map

[138]:
```
map_osm = folium.Map(location=[39.29, -76.61], zoom_start=13)
map_osm
```

[138]:
```
<folium.folium.Map at 0xffff289aac10>
```

Since my dataset is based in Baltimore, I passed the coordinates of the center of Baltimore into the folium.Map function.

1.3 Part 3

[139]:
```
data.columns
```

[139]:
```
Index(['X', 'Y', 'RowID', 'CrimeDateTime', 'CrimeCode', 'Location',
       'Description', 'Inside_Outside', 'Weapon', 'Post', 'District',
       'Neighborhood', 'Latitude', 'Longitude', 'GeoLocation', 'Premise',
       'VRIName', 'Total_Incidents'],
      dtype='object')
```

[140]:
```
crime_description = data['Description'].unique()
crime_description
```

[140]:
```
array(['SHOOTING', 'LARCENY FROM AUTO', 'LARCENY', 'AUTO THEFT',
       'AGG. ASSAULT', 'COMMON ASSAULT', 'ROBBERY - CARJACKING',
       'ROBBERY - COMMERCIAL', 'ROBBERY - STREET', 'ROBBERY - RESIDENCE',
       'HOMICIDE', 'BURGLARY', 'ARSON', 'RAPE'], dtype=object)
```

[141]:
```
start_date = '2022/04/17 00:00:00+00'
end_date = '2022/04/23 23:59:59+00'

mask = (data['CrimeDateTime'] >= start_date) & (data['CrimeDateTime'] <=␣
 ↪end_date)
data_subset = data.loc[mask]
data_subset
```

[141]:

	X	Y	RowID	CrimeDateTime	CrimeCode	\
0	1.424850e+06	569777.328405	1	2022/04/23 04:12:45+00	9S	
1	1.442033e+06	611635.789459	2	2022/04/21 11:34:27+00	9S	
2	1.421723e+06	592309.927233	3	2022/04/21 00:55:03+00	9S	
3	1.408272e+06	586720.112426	4	2022/04/20 16:01:00+00	6D	
4	1.434514e+06	592840.753893	5	2022/04/20 15:29:00+00	6F	
..	…	…	…	…	…	
271	1.409879e+06	581007.669603	272	2022/04/17 02:00:00+00	6D	
272	1.396770e+06	615236.193327	273	2022/04/17 21:30:00+00	3NF	
273	1.441232e+06	590651.552545	274	2022/04/17 22:40:00+00	4E	
274	1.419009e+06	577874.917843	275	2022/04/17 10:14:00+00	4E	
275	1.416162e+06	595856.495217	276	2022/04/17 20:45:39+00	9S	

	Location	Description	Inside_Outside	\
0	4100 6TH ST	SHOOTING	Outside	
1	5900 BELAIR RD	SHOOTING	Outside	
2	300 SAINT PAUL ST	SHOOTING	Outside	
3	2700 WILKENS AVE	LARCENY FROM AUTO	NaN	
4	3500 E FAIRMOUNT AVE	LARCENY	NaN	
..	…	…	…	
271	2800 DELMONT AVE	LARCENY FROM AUTO	NaN	
272	6400 REISTERSTOWN RD	ROBBERY - STREET	NaN	
273	6000 EASTERN AVE	COMMON ASSAULT	NaN	
274	2400 SEABURY RD	COMMON ASSAULT	NaN	
275	1500 PENNSYLVANIA AVE	SHOOTING	Inside	

	Weapon	Post	District	Neighborhood	Latitude	\
0	FIREARM	913	SOUTHERN	BROOKLYN	39.2305	
1	FIREARM	425	NORTHEAST	CEDMONT	39.3452	
2	FIREARM	111	CENTRAL	DOWNTOWN	39.2924	
3	NaN	834	SOUTHWEST	MILLHILL	39.2772	

```
4                   NaN  223  SOUTHEAST  BALTIMORE HIGHLANDS   39.2937
..                  …  …        …                  …       …
271                 NaN  831  SOUTHWEST         MORRELL PARK   39.2615
272  AUTOMATIC_HANDGUN  631  NORTHWEST                 GLEN   39.3556
273   PERSONAL_WEAPONS  232  SOUTHEAST              BAYVIEW   39.2876
274   PERSONAL_WEAPONS  922   SOUTHERN           CHERRY HILL   39.2528
275           FIREARM  123    CENTRAL                 UPTON   39.3022

     Longitude         GeoLocation      Premise  VRIName  Total_Incidents
0     -76.6028  (39.2305,-76.6028)  PUBLIC AREA      NaN                1
1     -76.5414  (39.3452,-76.5414)       STREET      NaN                1
2     -76.6135  (39.2924,-76.6135)     HOSPITAL      NaN                1
3     -76.6611  (39.2772,-76.6611)          NaN      NaN                1
4     -76.5683  (39.2937,-76.5683)          NaN      NaN                1
..         …                 …           …      …              …
271   -76.6555  (39.2615,-76.6555)          NaN      NaN                1
272   -76.7014  (39.3556,-76.7014)          NaN      NaN                1
273   -76.5446  (39.2876,-76.5446)          NaN      NaN                1
274   -76.6233  (39.2528,-76.6233)          NaN      NaN                1
275   -76.6331  (39.3022,-76.6331)     DWELLING  Central                1

[276 rows x 18 columns]
```

[142]:
```python
def find_color(case):
    if case == "SHOOTING": return "red"
    elif case == "LARCENY FROM AUTO": return "blue"
    elif case == "LARCENY": return "blue"
    elif case =="AUTO THEFT": return "blue"
    elif case == "AGG. ASSAULT": return "green"
    elif case == "COMMON ASSAULT": return "green"
    elif case == "ROBBERY - CARJACKING": return "darkpurple"
    elif case == "ROBBERY - COMMERCIAL": return "darkpurple"
    elif case == "ROBBERY - STREET": return "darkpurple"
    elif case == "ROBBERY - RESIDENCE": return "darkpurple"
    elif case == "HOMICIDE": return "lightgray"
    elif case == "BURGLARY": return "orange"
    elif case == "ARSON": return "black"
    elif case == "RAPE": return "pink"

for index, location_info in data_subset.iterrows():
    folium.Marker(
        [location_info["Latitude"],location_info["Longitude"]],
        popup=location_info["Description"],
        icon = folium.Icon(color = find_color(location_info["Description"]))
    ).add_to(map_osm)
```

[143]:
```
def add_legend(folium_map, title, colors, labels):
    if len(colors) != len(labels):
        raise ValueError("colors and labels must have the same length.")

    color_by_label = dict(zip(labels, colors))

    legend_categories = ""
    for label, color in color_by_label.items():
        legend_categories += f"<li><span style='background:{color}'></
↪span>{label}</li>"

    legend_html = f"""
    <div id='maplegend' class='maplegend'>
      <div class='legend-title'>{title}</div>
      <div class='legend-scale'>
        <ul class='legend-labels'>
        {legend_categories}
        </ul>
      </div>
    </div>
    """
    script = f"""
        <script type="text/javascript">
        var oneTimeExecution = (function() {{
                    var executed = false;
                    return function() {{
                        if (!executed) {{
                             var checkExist = setInterval(function() {{
                                       if ((document.
↪getElementsByClassName('leaflet-top leaflet-right').length) || (!executed))␣
↪{{
                                          document.
↪getElementsByClassName('leaflet-top leaflet-right')[0].style.display = "flex"
                                          document.
↪getElementsByClassName('leaflet-top leaflet-right')[0].style.flexDirection =␣
↪"column"
                                          document.
↪getElementsByClassName('leaflet-top leaflet-right')[0].innerHTML +=␣
↪`{legend_html}`;
                                          clearInterval(checkExist);
                                          executed = true;
                                       }}
                                    }}, 100);
                        }}
                    }};
                }})();
        oneTimeExecution()
```

```
      </script>
    """


css = """

<style type='text/css'>
  .maplegend {
    z-index:9999;
    float:right;
    background-color: rgba(255, 255, 255, 1);
    border-radius: 5px;
    border: 2px solid #bbb;
    padding: 10px;
    font-size:12px;
    positon: relative;
  }
  .maplegend .legend-title {
    text-align: left;
    margin-bottom: 5px;
    font-weight: bold;
    font-size: 90%;
    }
  .maplegend .legend-scale ul {
    margin: 0;
    margin-bottom: 5px;
    padding: 0;
    float: left;
    list-style: none;
    }
  .maplegend .legend-scale ul li {
    font-size: 80%;
    list-style: none;
    margin-left: 0;
    line-height: 18px;
    margin-bottom: 2px;
    }
  .maplegend ul.legend-labels li span {
    display: block;
    float: left;
    height: 16px;
    width: 30px;
    margin-right: 5px;
    margin-left: 0;
    border: 0px solid #ccc;
    }
  .maplegend .legend-source {
```

```
        font-size: 80%;
        color: #777;
        clear: both;
        }
      .maplegend a {
        color: #777;
        }
    </style>
    """

    folium_map.get_root().header.add_child(folium.Element(script + css))

    return folium_map
```

[144]:
```
map_osm = add_legend(map_osm, 'Legend',
                             colors = ['#d73e29', '#37aadd', '#71b025',␣
 ↪'#5b3a6b', '#a3a3a3', '#f2952f', '#000000', '#ff8de7'],
                           labels = ['Shooting', 'Larceny', 'Assault',␣
 ↪'Robbery', 'Homicide', 'Burglary', 'Arson', 'Rape'])
map_osm
```

[144]:
```
<folium.folium.Map at 0xffff289aac10>
```

Looking at the columns of the dataset, I decded to categorize the data based on the type of crime. I decided to group crimes into groups since there were so many types. For example, the four types of robbery became one robbery. I then color-coded the groups and made the markers, where clicking on the marker gives the specific type of crime (not just the group). However because there are so many entries in the dataset, I had to limit the markers to just the last week of crime, or else the points took over the map. Then, since folium only has functionality for creating a numerical legend, I used HTML to create a legend just for ease when viewing the map.

This map shows the Part 1 crime that has occured in Baltimore over the span of the last week. My initial goal with this dataset was to see if different types of crime are more commmon in different parts of Baltimore, which doesn't seem to be distinguishable from this map (especially when just showing a week's worth of data). However, we are able to see where crime is most concentrated (Central Baltimore) as well as which types of crime are most common (Assault, Larceny). It is interesting to see how relatively little crime occurs in the outskirts of Baltimore compared to the center. Additonally, despite seeing things like shootings on the news in Baltimore constantly, it is nothing compared to the 100s of cases of assault and larceny that occur during the week. Also, the sheer amount of crime that occurs during the week is absurd, and I was surprised that there were 276 cases of crime in just one week. It would be interesting to see how a normal week like this compares to amount of crime in the holiday season or other times of the year, and if the time of year affects the type of crime as well.