Untitled2

June 13, 2023

Importing libraries Pandas is a Python library for data manipulation and analysis. Numpy is a package that contains a multidimensional array object and several derivative ones. Matplotlib is a Python visualization package for 2D array plots. Seaborn is built on top of Matplotlib. It's used for exploratory data analysis and data visualization.

[3]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import datetime
from dateutil.relativedelta import relativedelta
from datetime import date

%matplotlib inline
```

[2]:
```
import warnings
warnings.filterwarnings('ignore')
```

1 1. Understand the dataset:

- Import the dataset

[5]:
```
df = pd.read_csv('311_Service_Requests_from_2010_to_Present.csv',␣
↪parse_dates=['Created Date', 'Closed Date'])
```

[6]:
```
df.head()
```

[6]:
```
   Unique Key        Created Date         Closed Date Agency  \
0    32310363 2015-12-31 23:59:45 2016-01-01 00:55:15   NYPD
1    32309934 2015-12-31 23:59:44 2016-01-01 01:26:57   NYPD
2    32309159 2015-12-31 23:59:29 2016-01-01 04:51:03   NYPD
3    32305098 2015-12-31 23:57:46 2016-01-01 07:43:13   NYPD
4    32306529 2015-12-31 23:56:58 2016-01-01 03:24:42   NYPD

                       Agency Name           Complaint Type  \
0  New York City Police Department  Noise - Street/Sidewalk
```

```
1  New York City Police Department         Blocked Driveway
2  New York City Police Department         Blocked Driveway
3  New York City Police Department          Illegal Parking
4  New York City Police Department          Illegal Parking

                    Descriptor    Location Type  Incident Zip  \
0             Loud Music/Party  Street/Sidewalk       10034.0
1                    No Access  Street/Sidewalk       11105.0
2                    No Access  Street/Sidewalk       10458.0
3  Commercial Overnight Parking  Street/Sidewalk       10461.0
4             Blocked Sidewalk  Street/Sidewalk       11373.0

        Incident Address  … Bridge Highway Name Bridge Highway Direction  \
0    71 VERMILYEA AVENUE  …                 NaN                       NaN
1        27-07 23 AVENUE  …                 NaN                       NaN
2  2897 VALENTINE AVENUE  …                 NaN                       NaN
3    2940 BAISLEY AVENUE  …                 NaN                       NaN
4          87-14 57 ROAD  …                 NaN                       NaN

  Road Ramp Bridge Highway Segment Garage Lot Name Ferry Direction  \
0       NaN                    NaN             NaN             NaN
1       NaN                    NaN             NaN             NaN
2       NaN                    NaN             NaN             NaN
3       NaN                    NaN             NaN             NaN
4       NaN                    NaN             NaN             NaN

  Ferry Terminal Name   Latitude  Longitude  \
0                 NaN  40.865682 -73.923501
1                 NaN  40.775945 -73.915094
2                 NaN  40.870325 -73.888525
3                 NaN  40.835994 -73.828379
4                 NaN  40.733060 -73.874170

                                  Location
0   (40.86568153633767, -73.92350095571744)
1  (40.775945312321085, -73.91509393898605)
2  (40.870324522111424, -73.88852464418646)
3   (40.83599404683083, -73.82837939584206)
4  (40.733059618956815, -73.87416975810375)

[5 rows x 53 columns]
```

[7]:
```
df.columns
```

[7]:
```
Index(['Unique Key', 'Created Date', 'Closed Date', 'Agency', 'Agency Name',
       'Complaint Type', 'Descriptor', 'Location Type', 'Incident Zip',
       'Incident Address', 'Street Name', 'Cross Street 1', 'Cross Street 2',
```

```
       'Intersection Street 1', 'Intersection Street 2', 'Address Type',
       'City', 'Landmark', 'Facility Type', 'Status', 'Due Date',
       'Resolution Description', 'Resolution Action Updated Date',
       'Community Board', 'Borough', 'X Coordinate (State Plane)',
       'Y Coordinate (State Plane)', 'Park Facility Name', 'Park Borough',
       'School Name', 'School Number', 'School Region', 'School Code',
       'School Phone Number', 'School Address', 'School City', 'School State',
       'School Zip', 'School Not Found', 'School or Citywide Complaint',
       'Vehicle Type', 'Taxi Company Borough', 'Taxi Pick Up Location',
       'Bridge Highway Name', 'Bridge Highway Direction', 'Road Ramp',
       'Bridge Highway Segment', 'Garage Lot Name', 'Ferry Direction',
       'Ferry Terminal Name', 'Latitude', 'Longitude', 'Location'],
      dtype='object')
```

[8]:
```
df.shape
```

[8]:
```
(364558, 53)
```

[9]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 364558 entries, 0 to 364557
Data columns (total 53 columns):
 #   Column                          Non-Null Count   Dtype
---  ------                          --------------   -----
 0   Unique Key                      364558 non-null  int64
 1   Created Date                    364558 non-null  datetime64[ns]
 2   Closed Date                     362177 non-null  datetime64[ns]
 3   Agency                          364558 non-null  object
 4   Agency Name                     364558 non-null  object
 5   Complaint Type                  364558 non-null  object
 6   Descriptor                      358057 non-null  object
 7   Location Type                   364425 non-null  object
 8   Incident Zip                    361560 non-null  float64
 9   Incident Address                312859 non-null  object
 10  Street Name                     312859 non-null  object
 11  Cross Street 1                  307370 non-null  object
 12  Cross Street 2                  306753 non-null  object
 13  Intersection Street 1           51120 non-null   object
 14  Intersection Street 2           50512 non-null   object
 15  Address Type                    361306 non-null  object
 16  City                            361561 non-null  object
 17  Landmark                        375 non-null     object
 18  Facility Type                   362169 non-null  object
 19  Status                          364558 non-null  object
 20  Due Date                        364555 non-null  object
 21  Resolution Description          364558 non-null  object
```

```
 22  Resolution Action Updated Date  362156 non-null  object
 23  Community Board                 364558 non-null  object
 24  Borough                         364558 non-null  object
 25  X Coordinate (State Plane)      360528 non-null  float64
 26  Y Coordinate (State Plane)      360528 non-null  float64
 27  Park Facility Name              364558 non-null  object
 28  Park Borough                    364558 non-null  object
 29  School Name                     364558 non-null  object
 30  School Number                   364558 non-null  object
 31  School Region                   364557 non-null  object
 32  School Code                     364557 non-null  object
 33  School Phone Number             364558 non-null  object
 34  School Address                  364558 non-null  object
 35  School City                     364558 non-null  object
 36  School State                    364558 non-null  object
 37  School Zip                      364557 non-null  object
 38  School Not Found                364558 non-null  object
 39  School or Citywide Complaint    0 non-null       float64
 40  Vehicle Type                    0 non-null       float64
 41  Taxi Company Borough            0 non-null       float64
 42  Taxi Pick Up Location           0 non-null       float64
 43  Bridge Highway Name             297 non-null     object
 44  Bridge Highway Direction        297 non-null     object
 45  Road Ramp                       262 non-null     object
 46  Bridge Highway Segment          262 non-null     object
 47  Garage Lot Name                 0 non-null       float64
 48  Ferry Direction                 1 non-null       object
 49  Ferry Terminal Name             2 non-null       object
 50  Latitude                        360528 non-null  float64
 51  Longitude                       360528 non-null  float64
 52  Location                        360528 non-null  object
dtypes: datetime64[ns](2), float64(10), int64(1), object(40)
memory usage: 147.4+ MB
```

[10]:
```
df.isnull().sum()
```

[10]:
```
Unique Key                      0
Created Date                    0
Closed Date                  2381
Agency                          0
Agency Name                     0
Complaint Type                  0
Descriptor                   6501
Location Type                 133
Incident Zip                 2998
Incident Address            51699
Street Name                 51699
```

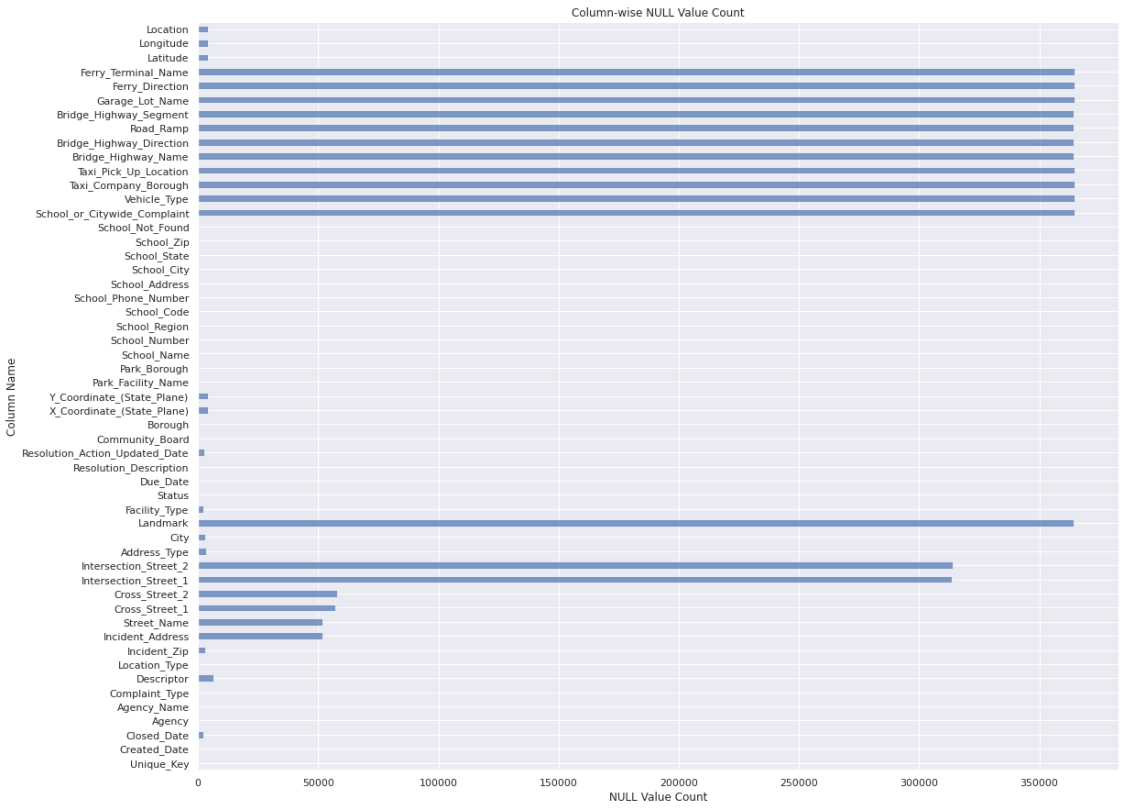
```
Cross Street 1                    57188
Cross Street 2                    57805
Intersection Street 1            313438
Intersection Street 2            314046
Address Type                       3252
City                               2997
Landmark                         364183
Facility Type                      2389
Status                                0
Due Date                              3
Resolution Description                0
Resolution Action Updated Date     2402
Community Board                       0
Borough                               0
X Coordinate (State Plane)         4030
Y Coordinate (State Plane)         4030
Park Facility Name                    0
Park Borough                          0
School Name                           0
School Number                         0
School Region                         1
School Code                           1
School Phone Number                   0
School Address                        0
School City                           0
School State                          0
School Zip                            1
School Not Found                      0
School or Citywide Complaint     364558
Vehicle Type                     364558
Taxi Company Borough             364558
Taxi Pick Up Location            364558
Bridge Highway Name              364261
Bridge Highway Direction         364261
Road Ramp                        364296
Bridge Highway Segment           364296
Garage Lot Name                  364558
Ferry Direction                  364557
Ferry Terminal Name              364556
Latitude                           4030
Longitude                          4030
Location                           4030
dtype: int64
```

2 2. Perform basic data exploratory analysis:

- Replace the special characters that are not needed in the DataFrame

[11]:
```
df.columns= df.columns.str.replace(" ","_")
```

[12]:
```
sns.set()
df.isnull().sum().plot(kind='barh', alpha= 0.7, figsize= (18,15),␣
↪title="Column-wise NULL Value Count")
plt.xlabel('NULL Value Count')
plt.ylabel('Column Name')
plt.show()
```



[17]:
```
df[['Closed_Date', 'Created_Date']].isnull().sum()
```

[17]:
```
Closed_Date     0
Created_Date    0
dtype: int64
```

3 a. Missing value treatment

- Remove the records whose Closed Date values are null

[19]:
```
df = df[pd.notnull(df['Closed_Date'])]
```

4 b. Analyze the date column and remove entries that have an incorrect timeline

- Time elapsed in closed and creation date

[20]:
```
a=(df.Created_Date[0] - df.Closed_Date[0] )
a.seconds
```

[20]:
```
83070
```

[21]:
```
df['Request_Closing_Time']=  df.Closed_Date - df.Created_Date
```

5 > - Convert the calculated date to seconds to get a better representation

[22]:
```
df['Request_Closing_Time']=df['Request_Closing_Time']/np.timedelta64(1,'s')
df['Request_Closing_Time'].head()
```

[22]:
```
0     3330.0
1     5233.0
2    17494.0
3    27927.0
4    12464.0
Name: Request_Closing_Time, dtype: float64
```

6 > - View the descriptive statistics for the newly created column

[23]:
```
df['Request_Closing_Time'].describe()
```

[23]:
```
count    3.621770e+05
mean     1.511330e+04
std      2.110255e+04
min      6.100000e+01
25%      4.533000e+03
50%      9.616000e+03
```

```
75%       1.887800e+04
max       2.134342e+06
Name: Request_Closing_Time, dtype: float64
```

[24]:
```
df['Request_Closing_Time'].mean()
```

[24]:
```
15113.299632500131
```

7 > - Check the number of null values in Complaint_Type and City columns

[25]:
```
df[['City', 'Complaint_Type']].isnull().sum()
```

[25]:
```
City              674
Complaint_Type      0
dtype: int64
```

8 > - Let's impute the NA value with Unknown City

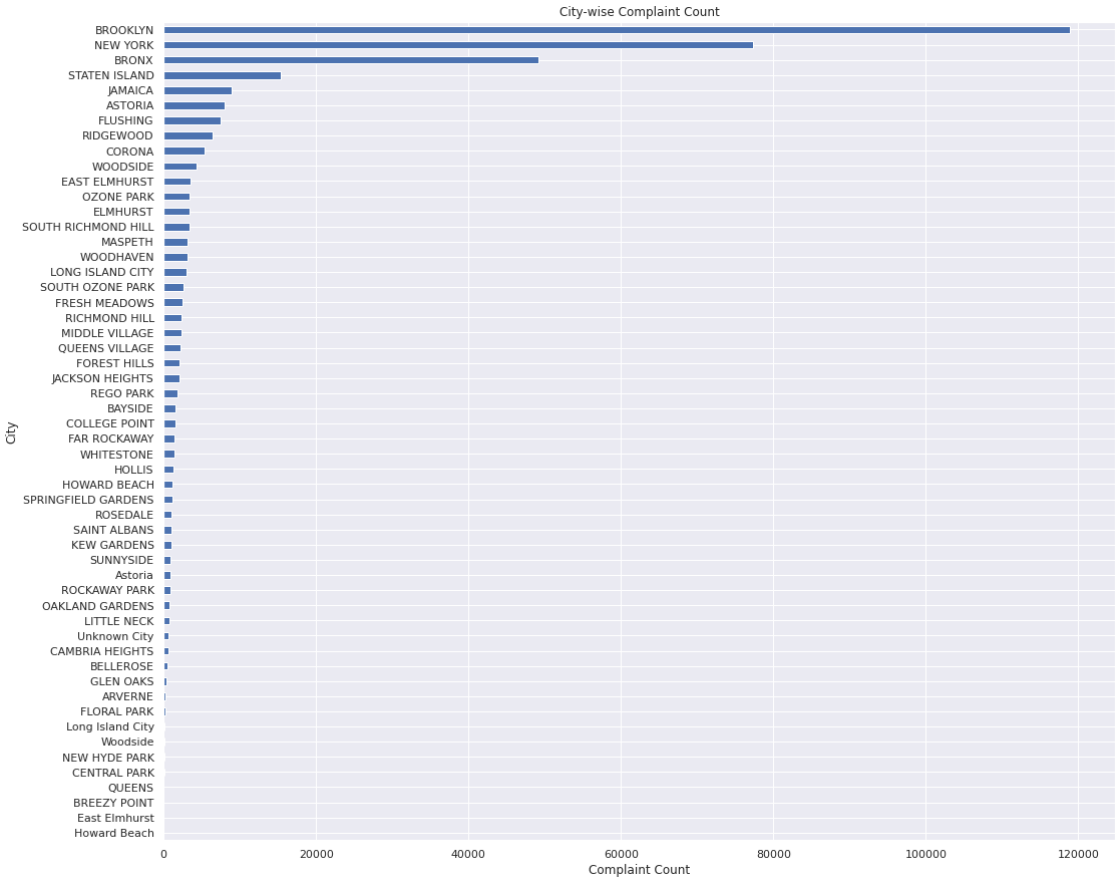
[26]:
```
df['City'].fillna('Unknown City', inplace= True)
```

[27]:
```
df[['City', 'Complaint_Type']].isnull().sum()
```

[27]:
```
City              0
Complaint_Type    0
dtype: int64
```

9 c. Draw a frequency plot for the complaints in each city

[28]:
```
sns.set()
df['City'].value_counts().sort_values(ascending= True).plot(kind= 'barh',␣
 ↪figsize=(17,15), title="City-wise Complaint Count")
plt.xlabel('Complaint Count')
plt.ylabel('City')
plt.show()
```



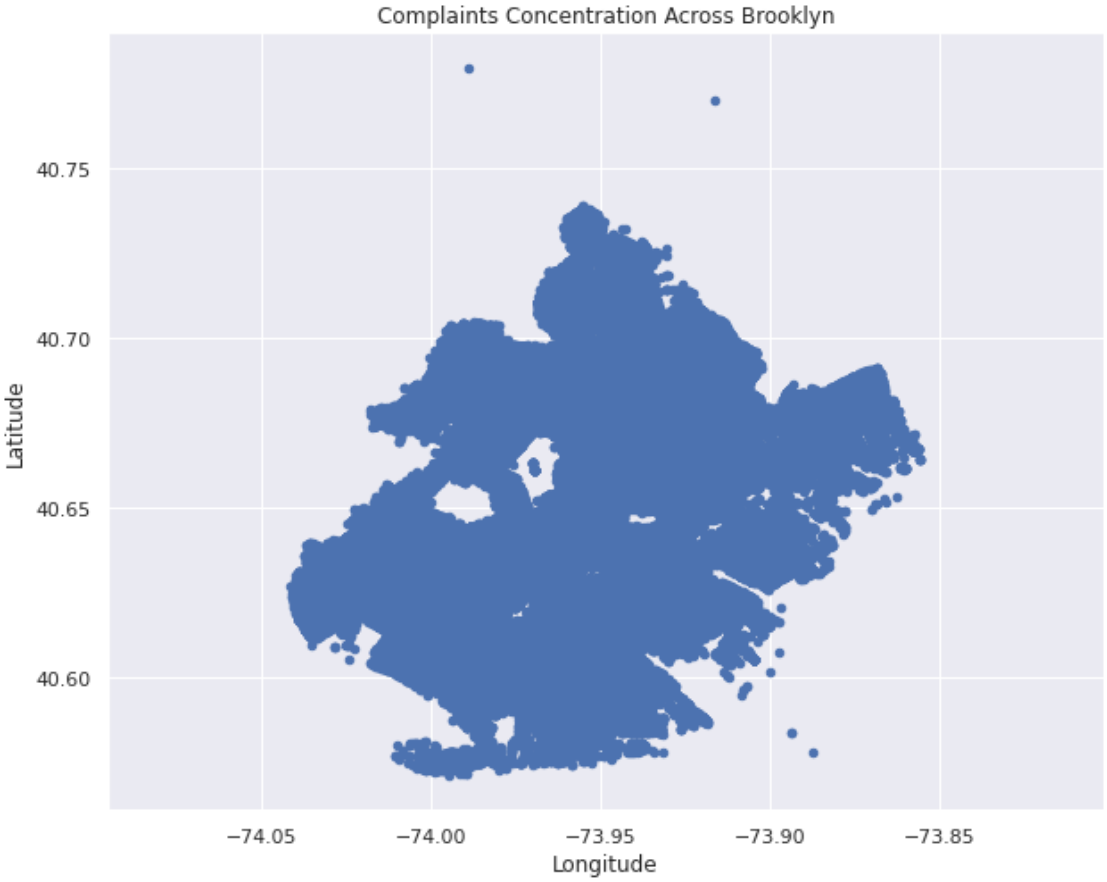
10 Let us review Brooklyn's complaint information

[29]:
```
brooklyn_data = df[df['City']=='BROOKLYN']
```

[ ]:
```
brooklyn_data[['Longitude', 'Latitude']].plot(kind='scatter',
                                              x='Longitude',
                                              y='Latitude',
                                              figsize=(10,8),
                                              title = 'Complaints Concentration␣
 ↪Across Brooklyn'
                                             ).axis('equal')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.show()
```

c argument looks like a single numeric RGB or RGBA sequence, which should be avoided as value-mapping will have precedence in case its length matches with

```
*x* & *y*.  Please use the *color* keyword-argument or provide a 2D array with a
single row if you intend to specify the same RGB or RGBA value for all points.
```



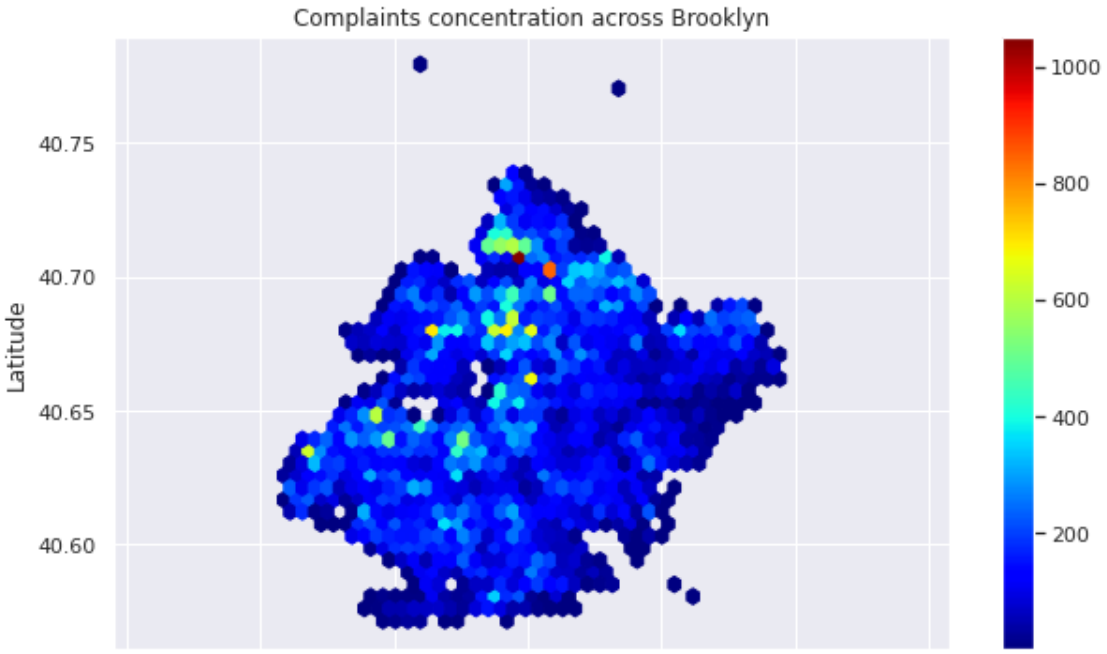
11 Observations:

The scatter plot is inconclusive as it is just one color The hexbin plot is a better indicator of the concentration of complaints in this scenario

12 > - Hexbin plot to visualize the complaint concetration across Brooklyn

[31]:
```
brooklyn_data.plot(kind='hexbin',
                   x='Longitude',
                   y='Latitude',
                   gridsize=40,
                   colormap = 'jet',
```

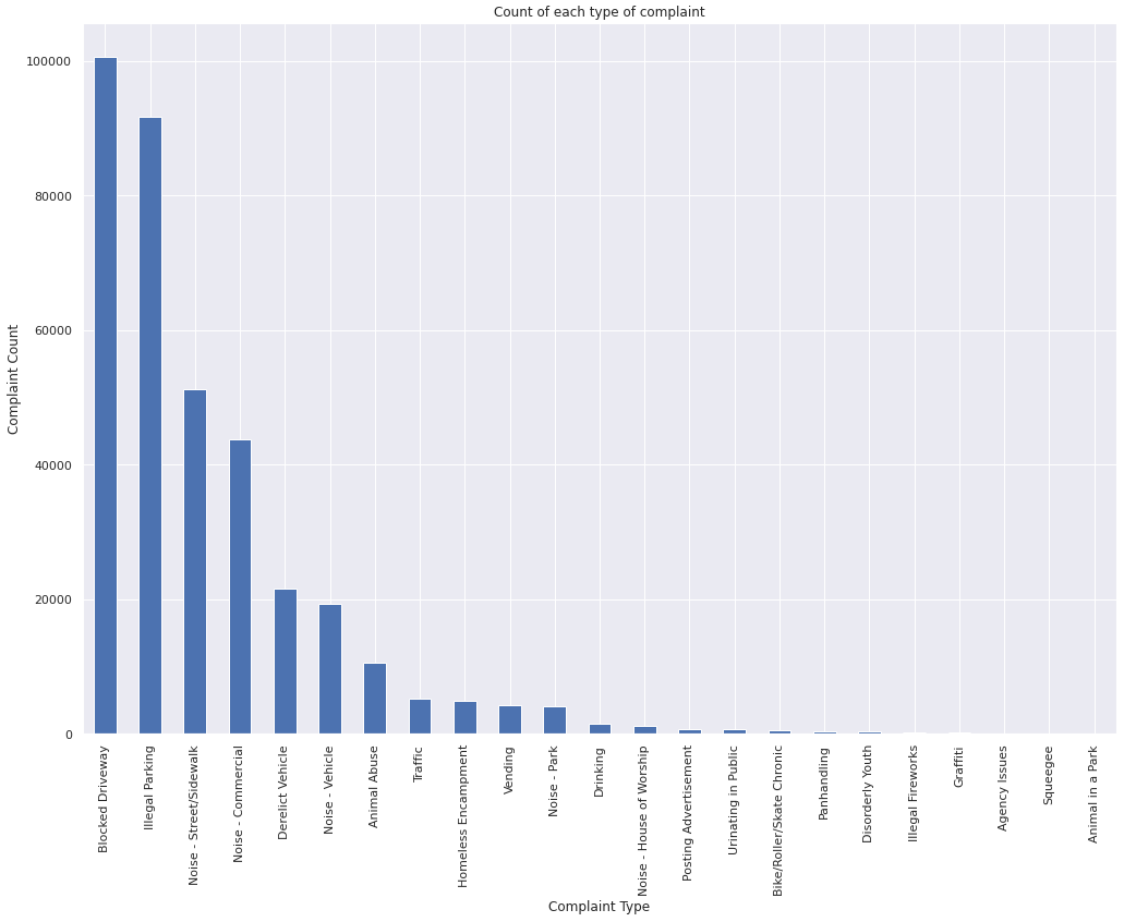
```
                mincnt=1,
                title = 'Complaints concentration across Brooklyn',
                figsize=(10,6)
               ).axis('equal')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.show()
```



13 3. Find major types of complaints:

a. Plot a bar graph to show the types of complaints

[32]:
```
sns.set()
df['Complaint_Type'].value_counts().plot(kind= 'bar', figsize=(17,12),␣
 ↪title="Count of each type of complaint")
plt.xlabel('Complaint Type')
plt.ylabel('Complaint Count')
plt.show()
```



[33]:
```
df['City'].unique()
```

[33]:
```
array(['NEW YORK', 'ASTORIA', 'BRONX', 'ELMHURST', 'BROOKLYN',
       'KEW GARDENS', 'JACKSON HEIGHTS', 'MIDDLE VILLAGE', 'REGO PARK',
       'SAINT ALBANS', 'JAMAICA', 'SOUTH RICHMOND HILL', 'Unknown City',
       'RIDGEWOOD', 'HOWARD BEACH', 'FOREST HILLS', 'STATEN ISLAND',
       'OZONE PARK', 'RICHMOND HILL', 'WOODHAVEN', 'FLUSHING', 'CORONA',
       'QUEENS VILLAGE', 'OAKLAND GARDENS', 'HOLLIS', 'MASPETH',
       'EAST ELMHURST', 'SOUTH OZONE PARK', 'WOODSIDE', 'FRESH MEADOWS',
       'LONG ISLAND CITY', 'ROCKAWAY PARK', 'SPRINGFIELD GARDENS',
       'COLLEGE POINT', 'BAYSIDE', 'GLEN OAKS', 'FAR ROCKAWAY',
       'BELLEROSE', 'LITTLE NECK', 'CAMBRIA HEIGHTS', 'ROSEDALE',
       'SUNNYSIDE', 'WHITESTONE', 'ARVERNE', 'FLORAL PARK',
       'NEW HYDE PARK', 'CENTRAL PARK', 'BREEZY POINT', 'QUEENS',
       'Astoria', 'Long Island City', 'Woodside', 'East Elmhurst',
       'Howard Beach'], dtype=object)
```

14 > - Check the frequency of various types of complaints for New York city

[34]:
```
df.loc[df['City']== 'NEW YORK']['Complaint_Type'].value_counts()
```

[34]:
```
Noise - Street/Sidewalk      22245
Noise - Commercial           18686
Illegal Parking              14549
Noise - Vehicle               6294
Homeless Encampment           3060
Blocked Driveway              2705
Vending                       2638
Animal Abuse                  1941
Traffic                       1769
Noise - Park                  1243
Derelict Vehicle               695
Drinking                       321
Urinating in Public            264
Bike/Roller/Skate Chronic      254
Noise - House of Worship       222
Panhandling                    206
Disorderly Youth                81
Posting Advertisement           49
Illegal Fireworks               38
Graffiti                        25
Squeegee                         4
Name: Complaint_Type, dtype: int64
```

15 b. Find the top 10 complaint types

[35]:
```
df['Complaint_Type'].value_counts()[0:10]
```

[35]:
```
Blocked Driveway           100624
Illegal Parking             91716
Noise - Street/Sidewalk     51139
Noise - Commercial          43751
Derelict Vehicle            21518
Noise - Vehicle             19301
Animal Abuse                10530
Traffic                      5196
Homeless Encampment          4879
Vending                      4185
Name: Complaint_Type, dtype: int64
```

[36]:
```
top10_complaints= np.array(df['Complaint_Type'].value_counts()[0:10].index)
```

16 c. Display the various types of complaints in each city

- Create a DataFrame df_new, which contains cities as columns and complaint types in rows

[37]:
```
df_new= pd.DataFrame()
```

[38]:
```
for i in df['City'].unique():
    df_new[i]= df.loc[df['City']== i]['Complaint_Type'].value_counts()
```

[39]:
```
df_new.head()
```

[39]:

	NEW YORK	ASTORIA	BRONX	ELMHURST	BROOKLYN	\
Noise - Street/Sidewalk	22245	409.0	9144.0	228.0	13982.0	
Noise - Commercial	18686	1653.0	2944.0	85.0	13855.0	
Illegal Parking	14549	1340.0	9889.0	760.0	33532.0	
Noise - Vehicle	6294	236.0	3556.0	69.0	5965.0	
Homeless Encampment	3060	32.0	275.0	34.0	948.0	

	KEW GARDENS	JACKSON HEIGHTS	MIDDLE VILLAGE	\
Noise - Street/Sidewalk	13.0	238.0	38.0	
Noise - Commercial	203.0	619.0	13.0	
Illegal Parking	276.0	240.0	1104.0	
Noise - Vehicle	23.0	75.0	45.0	
Homeless Encampment	5.0	11.0	5.0	

	REGO PARK	SAINT ALBANS	…	FLORAL PARK	\
Noise - Street/Sidewalk	64.0	81.0	…	3.0	
Noise - Commercial	82.0	36.0	…	3.0	
Illegal Parking	640.0	237.0	…	72.0	
Noise - Vehicle	60.0	50.0	…	2.0	
Homeless Encampment	6.0	11.0	…	NaN	

	NEW HYDE PARK	CENTRAL PARK	BREEZY POINT	QUEENS	\
Noise - Street/Sidewalk	NaN	105.0	1.0	6.0	
Noise - Commercial	4.0	NaN	4.0	6.0	
Illegal Parking	32.0	5.0	16.0	10.0	
Noise - Vehicle	2.0	NaN	1.0	2.0	
Homeless Encampment	NaN	NaN	NaN	2.0	

	Astoria	Long Island City	Woodside	East Elmhurst	\
Noise - Street/Sidewalk	145.0	28.0	5.0	NaN	
Noise - Commercial	310.0	19.0	2.0	NaN	
Illegal Parking	277.0	64.0	124.0	28.0	

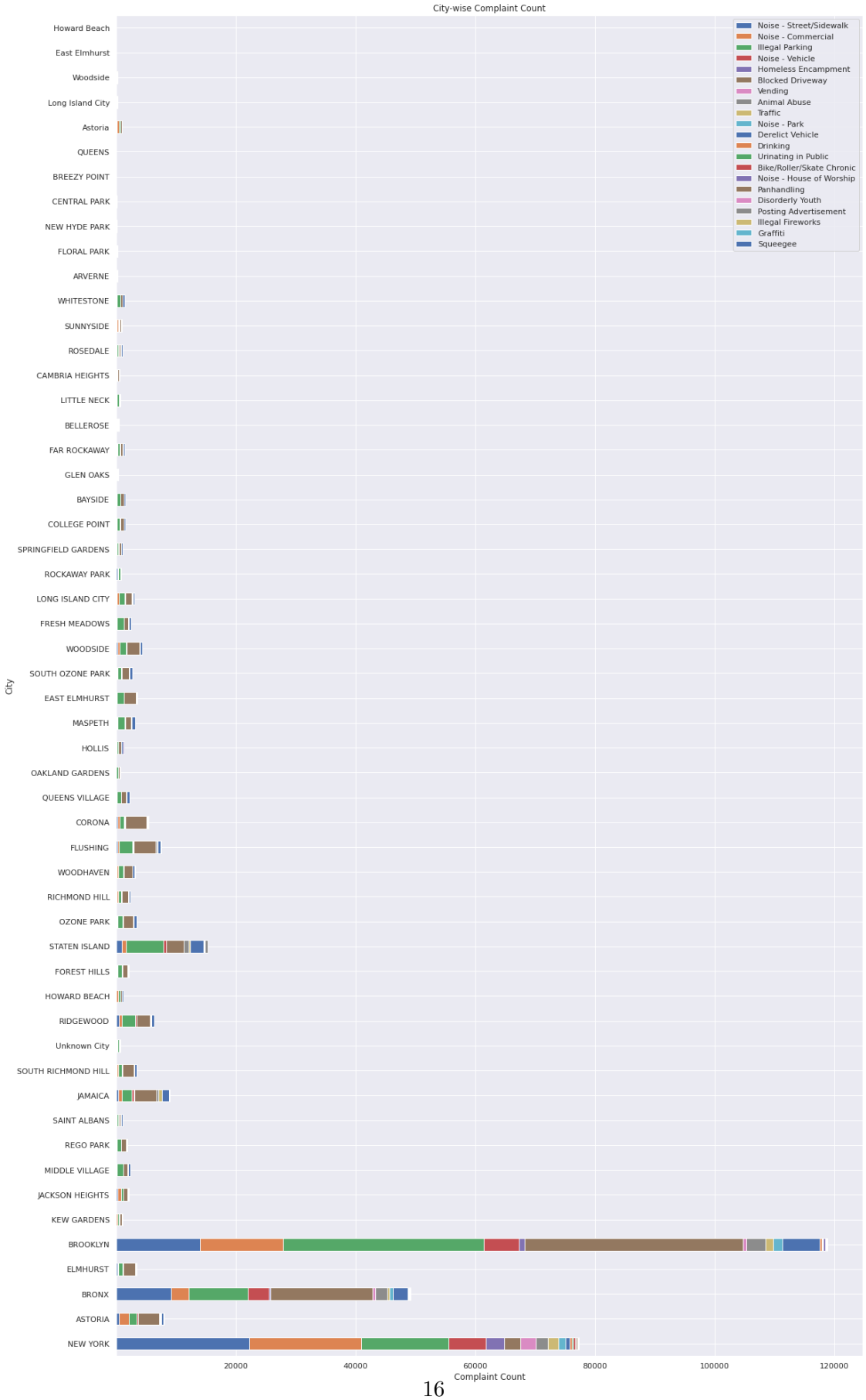
```
Noise - Vehicle              NaN               NaN       NaN            NaN
Homeless Encampment          NaN               NaN       NaN            NaN

                          Howard Beach
Noise - Street/Sidewalk            NaN
Noise - Commercial                 NaN
Illegal Parking                    NaN
Noise - Vehicle                    NaN
Homeless Encampment                NaN

[5 rows x 54 columns]
```

17 4. Visualize the major types of complaints in each city

Draw another chart that shows the types of complaints in each city in a single chart, where diffrent colors show the different types of complaints

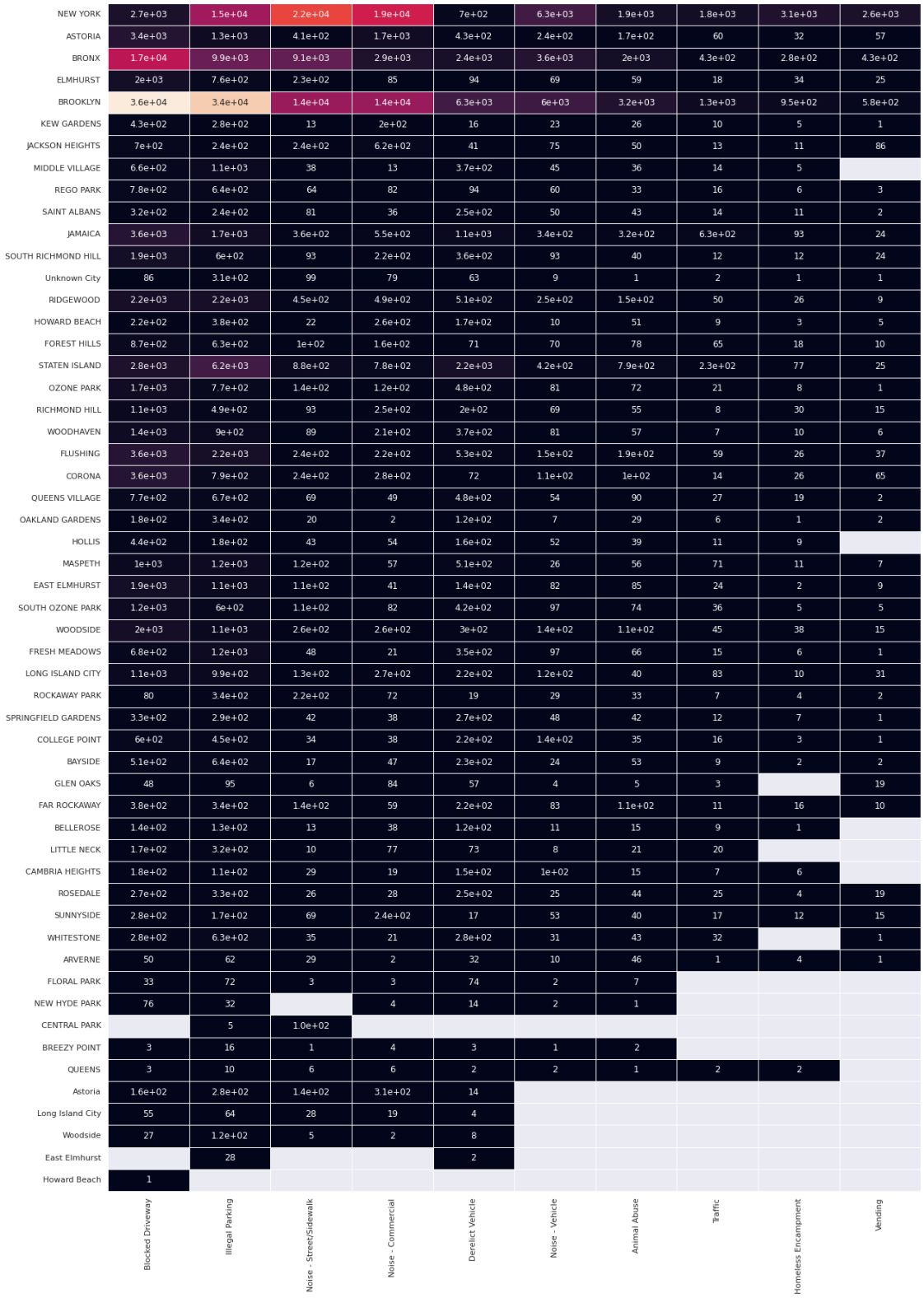
18 > - This type of chart is known as the stacked bar chart

[40]:
```
sns.set()
df_new.T.plot(kind= 'barh', stacked= True, figsize= (19, 35), title="City-wise␣
 ↪Complaint Count")
plt.xlabel('Complaint Count')
plt.ylabel('City')
plt.show()
```


City-wise Complaint Count
Howard Beach
East Elmhurst
Woodside
Long Island City
Astoria
QUEENS
BREEZY POINT
CENTRAL PARK
NEW HYDE PARK
FLORAL PARK
ARVERNE
WHITESTONE
SUNNYSIDE
ROSEDALE
CAMBRIA HEIGHTS
LITTLE NECK
BELLEROSE
FAR ROCKAWAY
GLEN OAKS
BAYSIDE
COLLEGE POINT
SPRINGFIELD GARDENS
ROCKAWAY PARK
LONG ISLAND CITY
FRESH MEADOWS
WOODSIDE
SOUTH OZONE PARK
EAST ELMHURST
MASPETH
HOLLIS
OAKLAND GARDENS
QUEENS VILLAGE
CORONA
FLUSHING
WOODHAVEN
RICHMOND HILL
OZONE PARK
STATEN ISLAND
FOREST HILLS
HOWARD BEACH
RIDGEWOOD
Unknown City
SOUTH RICHMOND HILL
JAMAICA
SAINT ALBANS
REGO PARK
MIDDLE VILLAGE
JACKSON HEIGHTS
KEW GARDENS
BROOKLYN
ELMHURST
BRONX
ASTORIA
NEW YORK
City
20000
40000
60000
80000
100000
120000
Complaint Count
Noise - Street/Sidewalk
Noise - Commercial
Illegal Parking
Noise - Vehicle
Homeless Encampment
Blocked Driveway
Vending
Animal Abuse
Traffic
Noise - Park
Derelict Vehicle
Drinking
Urinating in Public
Bike/Roller/Skate Chronic
Noise - House of Worship
Panhandling
Disorderly Youth
Posting Advertisement
Illegal Fireworks
Graffiti
Squeegee

19 - Heat Map for top 10 types of Complaints in each city

[41]:
```
df_new_T= df_new.T
sns.set()
f, ax = plt.subplots(figsize=(20, 30))
sns.heatmap(df_new_T[top10_complaints], annot=True, cbar=False, linewidths=.8,␣
↪ax=ax)
plt.show()
```


NEW YORK 2.7e+03 1.5e+04 2.2e+04 1.9e+04 7e+02 6.3e+03 1.9e+03 1.8e+03 3.1e+03 2.6e+03
ASTORIA 3.4e+03 1.3e+03 4.1e+02 1.7e+03 4.3e+02 2.4e+02 1.7e+02 60 32 57
BRONX 1.7e+04 9.9e+03 9.1e+03 2.9e+03 2.4e+03 3.6e+03 2e+03 4.3e+02 2.8e+02 4.3e+02
ELMHURST 2e+03 7.6e+02 2.3e+02 85 94 69 59 18 34 25
BROOKLYN 3.6e+04 3.4e+04 1.4e+04 1.4e+04 6.3e+03 6e+03 3.2e+03 1.3e+03 9.5e+02 5.8e+02
KEW GARDENS 4.3e+02 2.8e+02 13 2e+02 16 23 26 10 5 1
JACKSON HEIGHTS 7e+02 2.4e+02 2.4e+02 6.2e+02 41 75 50 13 11 86
MIDDLE VILLAGE 6.6e+02 1.1e+03 38 13 3.7e+02 45 36 14 5
REGO PARK 7.8e+02 6.4e+02 64 82 94 60 33 16 6 3
SAINT ALBANS 3.2e+02 2.4e+02 81 36 2.5e+02 50 43 14 11 2
JAMAICA 3.6e+03 1.7e+03 3.6e+02 5.5e+02 1.1e+03 3.4e+02 3.2e+02 6.3e+02 93 24
SOUTH RICHMOND HILL 1.9e+03 6e+02 93 2.2e+02 3.6e+02 93 40 12 12 24
Unknown City 86 3.1e+02 99 79 63 9 1 2 1 1
RIDGEWOOD 2.2e+03 2.2e+03 4.5e+02 4.9e+02 5.1e+02 2.5e+02 1.5e+02 50 26 9
HOWARD BEACH 2.2e+02 3.8e+02 22 2.6e+02 1.7e+02 10 51 9 3 5
FOREST HILLS 8.7e+02 6.3e+02 1e+02 1.6e+02 71 70 78 65 18 10
STATEN ISLAND 2.8e+03 6.2e+03 8.8e+02 7.8e+02 2.2e+03 4.2e+02 7.9e+02 2.3e+02 77 25
OZONE PARK 1.7e+03 7.7e+02 1.4e+02 1.2e+02 4.8e+02 81 72 21 8 1
RICHMOND HILL 1.1e+03 4.9e+02 93 2.5e+02 2e+02 69 55 8 30 15
WOODHAVEN 1.4e+03 9e+02 89 2.1e+02 3.7e+02 81 57 7 10 6
FLUSHING 3.6e+03 2.2e+03 2.4e+02 2.2e+02 5.3e+02 1.5e+02 1.9e+02 59 26 37
CORONA 3.6e+03 7.9e+02 2.4e+02 2.8e+02 72 1.1e+02 1e+02 14 26 65
QUEENS VILLAGE 7.7e+02 6.7e+02 69 49 4.8e+02 54 90 27 19 2
OAKLAND GARDENS 1.8e+02 3.4e+02 20 2 1.2e+02 7 29 6 1 2
HOLLIS 4.4e+02 1.8e+02 43 54 1.6e+02 52 39 11 9
MASPETH 1e+03 1.2e+03 1.2e+02 57 5.1e+02 26 56 71 11 7
EAST ELMHURST 1.9e+03 1.1e+03 1.1e+02 41 1.4e+02 82 85 24 2 9
SOUTH OZONE PARK 1.2e+03 6e+02 1.1e+02 82 4.2e+02 97 74 36 5 5
WOODSIDE 2e+03 1.1e+03 2.6e+02 2.6e+02 3e+02 1.4e+02 1.1e+02 45 38 15
FRESH MEADOWS 6.8e+02 1.2e+03 48 21 3.5e+02 97 66 15 6 1
LONG ISLAND CITY 1.1e+03 9.9e+02 1.3e+02 2.7e+02 2.2e+02 1.2e+02 40 83 10 31
ROCKAWAY PARK 80 3.4e+02 2.2e+02 72 19 29 33 7 4 2
SPRINGFIELD GARDENS 3.3e+02 2.9e+02 42 38 2.7e+02 48 42 12 7 1
COLLEGE POINT 6e+02 4.5e+02 34 38 2.2e+02 1.4e+02 35 16 3 1
BAYSIDE 5.1e+02 6.4e+02 17 47 2.3e+02 24 53 9 2 2
GLEN OAKS 48 95 6 84 57 4 5 3 19
FAR ROCKAWAY 3.8e+02 3.4e+02 1.4e+02 59 2.2e+02 83 1.1e+02 11 16 10
BELLEROSE 1.4e+02 1.3e+02 13 38 1.2e+02 11 15 9 1
LITTLE NECK 1.7e+02 3.2e+02 10 77 73 8 21 20
CAMBRIA HEIGHTS 1.8e+02 1.1e+02 29 19 1.5e+02 1e+02 15 7 6
ROSEDALE 2.7e+02 3.3e+02 26 28 2.5e+02 25 44 25 4 19
SUNNYSIDE 2.8e+02 1.7e+02 69 2.4e+02 17 53 40 17 12 15
WHITESTONE 2.8e+02 6.3e+02 35 21 2.8e+02 31 43 32 1
ARVERNE 50 62 29 2 32 10 46 1 4 1
FLORAL PARK 33 72 3 3 74 2 7
NEW HYDE PARK 76 32 4 14 2 1
CENTRAL PARK 5 1.0e+02
BREEZY POINT 3 16 1 4 3 1 2
QUEENS 3 10 6 6 2 2 1 2 2
Astoria 1.6e+02 2.8e+02 1.4e+02 3.1e+02 14
Long Island City 55 64 28 19 4
Woodside 27 1.2e+02 5 2 8
East Elmhurst 28 2
Howard Beach 1
Blocked Driveway
Illegal Parking
Noise - Street/Sidewalk
Noise - Commercial
Derelict Vehicle
Noise - Vehicle
Animal Abuse
Traffic
Homeless Encampment
Vending

20 - Sort the complaint types based on the average Request_Closing_Time grouping them for different locations.

[42]:
```
df.groupby(['City', 'Complaint_Type'], sort= True).Request_Closing_Time.mean()
```

[42]:
```
City      Complaint_Type
ARVERNE   Animal Abuse                8399.195652
          Blocked Driveway            8318.840000
          Derelict Vehicle           11394.000000
          Disorderly Youth           12928.500000
          Drinking                     859.000000
                                           …
Woodside  Blocked Driveway           15566.185185
          Derelict Vehicle           19994.500000
          Illegal Parking            17293.459677
          Noise - Commercial          8619.000000
          Noise - Street/Sidewalk    12285.600000
Name: Request_Closing_Time, Length: 792, dtype: float64
```

21 5. See whether the average response time across complaint types is similar or not (overall)

[43]:
```
df.groupby(['Complaint_Type'], sort= True).Request_Closing_Time.mean()
```

[43]:
```
Complaint_Type
Agency Issues                1.828912e+04
Animal Abuse                 1.803256e+04
Animal in a Park             1.212634e+06
Bike/Roller/Skate Chronic    1.312369e+04
Blocked Driveway             1.623252e+04
Derelict Vehicle             2.535960e+04
Disorderly Youth             1.236375e+04
Drinking                     1.382130e+04
Graffiti                     2.327634e+04
Homeless Encampment          1.545138e+04
Illegal Fireworks            1.011348e+04
Illegal Parking              1.565044e+04
Noise - Commercial           1.108576e+04
Noise - House of Worship     1.139109e+04
Noise - Park                 1.222606e+04
Noise - Street/Sidewalk      1.223130e+04
Noise - Vehicle              1.256180e+04
Panhandling                  1.585355e+04
Posting Advertisement        7.286256e+03
```

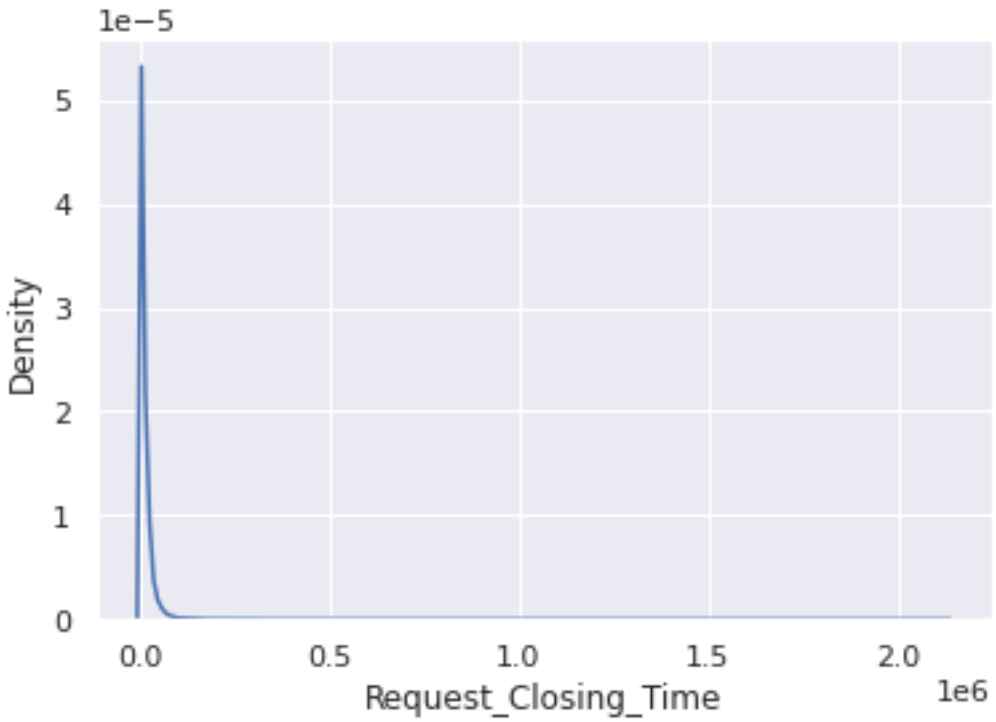
```
Squeegee                      1.456025e+04
Traffic                       1.230912e+04
Urinating in Public           1.295929e+04
Vending                       1.436628e+04
Name: Request_Closing_Time, dtype: float64
```

[44]:
```
df.Request_Closing_Time.describe()
```

[44]:
```
count    3.621770e+05
mean     1.511330e+04
std      2.110255e+04
min      6.100000e+01
25%      4.533000e+03
50%      9.616000e+03
75%      1.887800e+04
max      2.134342e+06
Name: Request_Closing_Time, dtype: float64
```

22 > - Let us visualize graphically the average Request_Closing_Time

[45]:
```
sns.set()
sns.distplot(df.Request_Closing_Time, hist= False)
plt.show()
```



23 6. Identify significant variables by performing a statistical analysis using p-values

Shapiro- Wilk Test of Normality > - H0: Request_Closing_Time feature is normal.

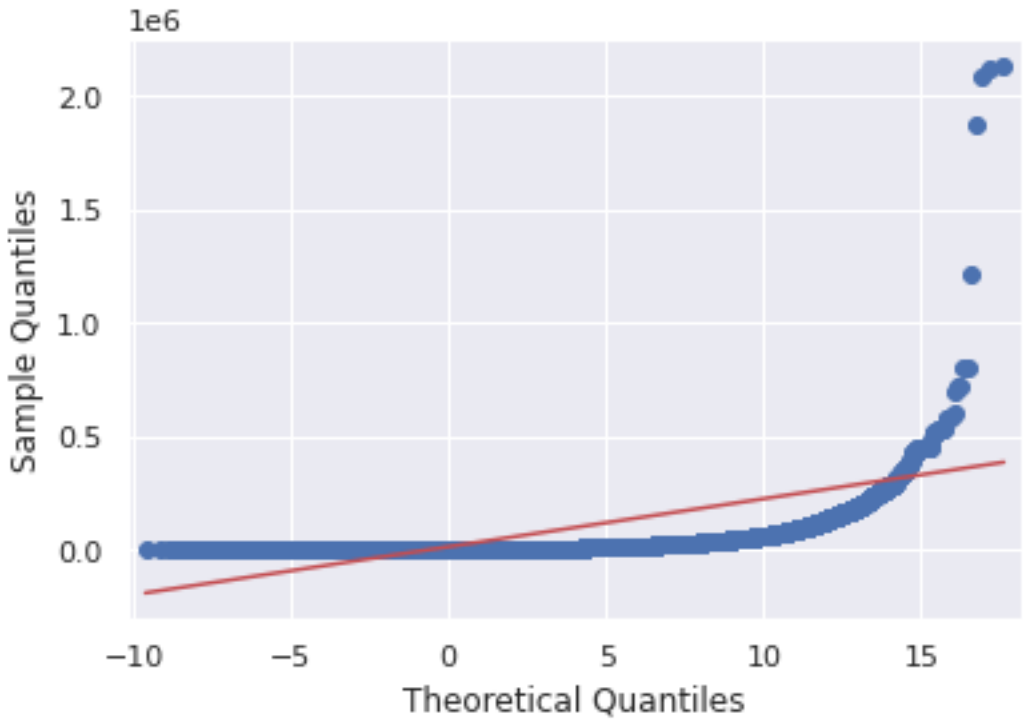
[46]:
```
#Check for nomality of Age
# Shapiro- Wilk Test
from scipy.stats import shapiro
stat, p= shapiro(df.Request_Closing_Time)
print(stat, p)
```

```
0.5136188268661499 0.0
```

24 Observation:

- If the P value (0.0) returned is less than 0.05, then the null hypothesis is rejected and there is evidence that the data is not from a normally distributed population. Now let's check for QQ Plots.

[47]:
```
import statsmodels.api as sm
import pylab
sm.qqplot(df.Request_Closing_Time, loc = 4, scale = 3, line='s')
pylab.show()
```



25 Kruskal Wallis H Test

Fail to Reject H0: All sample distributions are equal. Reject H0: One or more sample distributions are not equal.

[48]:
```
Complaint_Types= df.Complaint_Type.unique()
```

[49]:
```
len(Complaint_Types)
```

[49]: 23

[50]:
```
new= []
for i in Complaint_Types:
  new.append(df.loc[df['Complaint_Type']== i]['Request_Closing_Time'].ravel())
```

[51]:
```
new[20]
```

[51]:
```
array([27090., 37381.,  9736., 24686.,  4073., 10628., 30818.,  1901.])
```

[52]:
```
from scipy.stats import kruskal
# compare samples
stat, p = kruskal(new[0],new[1], new[2],new[3], new[4], new[5], new[6], new[7],␣
 ↪new[8], new[9], new[10], new[11], new[12], new[13], new[14], new[15],␣
 ↪new[16], new[17], new[18], new[19],new[20], new[21], new[22])
print('Statistics=%.3f, p=%.3f' % (stat, p))
# interpret
alpha = 0.05
if p > alpha:
    print('Same distributions (fail to reject H0)')
else:
    print('Different distributions (reject H0)')
```

```
Statistics=11988.269, p=0.000
Different distributions (reject H0)
```

26 Observation:

So, we reject the H0, that means average response time across complaint types are not similar ### 1.Chi-Squared Test of independence ### 2.Let us see whether the type of complaint or service requested and location related ### 3.H0:There is no statistically significant relationship between City and Complaint Type. ### 4.Ha:There is a statistically significant relationship between City and Complaint Type

[53]:
```
#contingency Table for complaints
contingency_table = pd.crosstab(df['City'],df['Complaint_Type'], margins= True)
contingency_table.head()
```

[53]:

Complaint_Type	Agency Issues	Animal Abuse	Animal in a Park	\
City				
ARVERNE	0	46	0	
ASTORIA	0	170	0	
Astoria	0	0	0	
BAYSIDE	0	53	0	
BELLEROSE	0	15	0	

Complaint_Type	Bike/Roller/Skate Chronic	Blocked Driveway	Derelict Vehicle	\
City				
ARVERNE	0	50	32	
ASTORIA	16	3436	426	
Astoria	0	159	14	
BAYSIDE	0	514	231	
BELLEROSE	1	138	120	

```
Complaint_Type  Disorderly Youth  Drinking  Graffiti  Homeless Encampment  \
City
ARVERNE                        2         1         1                    4
ASTORIA                        5        43         4                   32
Astoria                        0         0         0                    0
BAYSIDE                        2         1         3                    2
BELLEROSE                      2         1         0                    1

Complaint_Type  …  Noise - Park  Noise - Street/Sidewalk  Noise - Vehicle  \
City            …
ARVERNE         …             2                       29               10
ASTORIA         …            64                      409              236
Astoria         …             0                      145                0
BAYSIDE         …             4                       17               24
BELLEROSE       …             1                       13               11

Complaint_Type  Panhandling  Posting Advertisement  Squeegee  Traffic  \
City
ARVERNE                   1                      0         0        1
ASTORIA                   2                      3         0       60
Astoria                   0                      0         0        0
BAYSIDE                   0                      0         0        9
BELLEROSE                 1                      1         0        9

Complaint_Type  Urinating in Public  Vending   All
City
ARVERNE                           1        1   259
ASTORIA                          10       57  7991
Astoria                           0        0   905
BAYSIDE                           0        2  1550
BELLEROSE                         1        0   487

[5 rows x 24 columns]
```

[54]:
```
contingency_table.shape
```

[54]:
```
(55, 24)
```

[55]:
```
contingency_table.iloc[0:5][0:24].values
```

[55]:
```
array([[   0,   46,    0,    0,   50,   32,    2,    1,    1,    4,    0,
          62,    2,   14,    2,   29,   10,    1,    0,    0,    1,    1,
           1,  259],
       [   0,  170,    0,   16, 3436,  426,    5,   43,    4,   32,    4,
        1340, 1653,   21,   64,  409,  236,    2,    3,    0,   60,   10,
          57, 7991],
       [   0,    0,    0,    0,  159,   14,    0,    0,    0,    0,    0,
```

```
       277,  310,    0,    0,  145,    0,    0,    0,    0,    0,    0,
         0,  905],
       [   0,   53,    0,    0,  514,  231,    2,    1,    3,    2,    0,
       638,   47,    3,    4,   17,   24,    0,    0,    0,    9,    0,
         2, 1550],
       [   0,   15,    0,    1,  138,  120,    2,    1,    0,    1,    1,
       132,   38,    1,    1,   13,   11,    1,    1,    0,    9,    1,
         0,  487]])
```

[56]:
```
f_obs= []
for i in range(0, contingency_table.shape[0]-1):
    f_obs.append(contingency_table.iloc[i][0:24].values)
f_obs= np.array(f_obs)
f_obs[0:5]
```

[56]:
```
array([[   0,   46,    0,    0,   50,   32,    2,    1,    1,    4,    0,
          62,    2,   14,    2,   29,   10,    1,    0,    0,    1,    1,
           1,  259],
       [   0,  170,    0,   16, 3436,  426,    5,   43,    4,   32,    4,
        1340, 1653,   21,   64,  409,  236,    2,    3,    0,   60,   10,
          57, 7991],
       [   0,    0,    0,    0,  159,   14,    0,    0,    0,    0,    0,
       277,  310,    0,    0,  145,    0,    0,    0,    0,    0,    0,
         0,  905],
       [   0,   53,    0,    0,  514,  231,    2,    1,    3,    2,    0,
       638,   47,    3,    4,   17,   24,    0,    0,    0,    9,    0,
         2, 1550],
       [   0,   15,    0,    1,  138,  120,    2,    1,    0,    1,    1,
       132,   38,    1,    1,   13,   11,    1,    1,    0,    9,    1,
         0,  487]])
```

[57]:
```
from scipy import stats
stats.chi2_contingency(f_obs)[0:3]
```

[57]:
```
(145971.80461890678, 0.0, 1219)
```

27 P-value is approximately zero. So we have evidence against the null hypothesis.

With a p-value < 0.05 , we can reject the null hypothesis at 95% confidence Interval. There is definitely some sort of relationship between 'City' and the 'Complaint_Type' column. We don't know what this relationship is, but we do know that these two variables are not independent of each other.

[ ]:
```

```