Applied Data Science with Python

Course-end Project

Customer Service Requests Analysis

Objectives:

- To assess the data and prepare a fresh dataset for training and prediction
- To plot a bar graph to identify the relationship between two variables
- To visualize the major types of complaints in each city

Problem Statement:

You've been asked to analyze data on service request (311) calls from New York City. You've also been asked to utilize data wrangling techniques to understand the patterns in the data and visualize the major types of complaints.

Domain: Technology

Content: Dataset name: 311-service-requests-nyc.csv

Fields in the data:

- Unique Key : The unique identification number
- Created Date: The date when the request was created
- Closed Date : The date when the request was closed
- Agency : The agency that handled the case
- Agency Name : The full name of the agency that handled the case
- Complaint Type : The type of complaint received
- Descriptor : The description of the complaint
- Location Type : The type of location where the incident occurred
- Incident Zip : The zip code of the location
- Incident Address : The location at which the incident occurred
- Street Name : The name of the street
- Cross Street 1 : The cross of the street 1
- Cross Street 2 : The cross of street 2
- Intersection Street 1 : The first point of intersection of both streets
- Intersection Street 2 : The second point of intersection of both streets
- Address Type : The type of the address
- City : The city where the incident occurred
- Landmark : The landmark near the incident that occurred
- Facility Type : The type of the facility
- Status :The status of the complaint
- Due Date : The due date of the complaint
- Resolution Description : The resolution provided by the police department

- Resolution Action Updated Date : The date at which the resolution was provided
- Community Board : The location of the community board
- Borough : The town, or area inside a large town, that has some form of local government
- X Coordinate (State Plane) : The X coordinate of the plane
- Y Coordinate (State Plane) : The Y coordinate of the plane
- Park Facility Name : The name of the park facility
- Park Borough : The park town, or area inside a large town, that has some form of local government
- School Name : The name of the school (optional)
- School Number : Number of the school (optional)
- School Region : Region of the school (optional)
- School Code : Code of the school (optional)
- School Phone Number : Contact information of the school (optional)
- School Address : Address of the school (optional)
- School City : City at which the school is located (optional)
- School State : State in which the school is located (optional)
- School Zip : Zip code of the school (optional)
- School Not Found : Valid if the school is not found (optional)
- School or Citywide Complaint : Contains the complaint of the school (optional)
- Vehicle Type : Type of vehicle used (optional)
- Taxi Company Borough : Information on the taxi company (optional)
- Taxi Pick Up Location : Pick up location of the taxi (optional)
- Bridge Highway Name : Name of the highway bridge (optional)
- Bridge Highway Direction : Direction of the highway bridge (optional)
- Road Ramp : Information on the road ramp (optional)
- Bridge Highway Segment : Segment of the bridge (optional)
- Garage Lot Name : Name of the garage (optional)
- Ferry Direction : Ferry direction information (optional)
- Ferry Terminal Name : Name of the ferry terminal (optional)
- Latitude: Latitude value
- Longitude : Longitude value
- Location : Location information

Steps to perform:

1. Understand the dataset:
   1.1 Identify the shape of the dataset
   1.2 Identify variables with null values

2. Perform basic data exploratory analysis:
   2.1 Utilize missing value treatment
   2.2 Analyze the date column and remove the entries if it has an incorrect timeline
      2.2.1 Draw a frequency plot for city-wise complaints
      2.2.2 Draw scatter and hexbin plots for complaint concentration across Brooklyn
3. Find major types of complaints:
   3.1 Plot a bar graph of count vs. complaint types
   3.2 Find the top 10 types of complaints
   3.3 Display the types of complaints in each city in a separate dataset
4. Visualize the major types of complaints in each city
5. Check if the average response time across various types of complaints

Solution

1. Import Libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

2. Load the data file

```
df=pd.read_csv("/Users/david/Downloads/Customer_Service_Requests
_Analysis_Dataset/Customer_Service_Requests_Analysis_Dataset/311
-service-requests-
nyc/311_Service_Requests_from_2010_to_Present.csv",low_memory=Fa
lse) 1.
```

3. Identify the shape of the dataset

```
df.shape
```

```
#Identify the shape of the dataset
df.shape
```

```
(364558, 53)
```

4. Identify variables with null values

```
df.isnull().sum()
```

```
#Identify variables with null values
df.isnull().sum()
```

```
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
```

5. Perform basic data exploratory analysis:

a. Utilize missing value treatment

- `df.isnull().sum()/len(df)*100`

```
#Utilize missing value treatment

df.isnull().sum()/len(df)*100
```

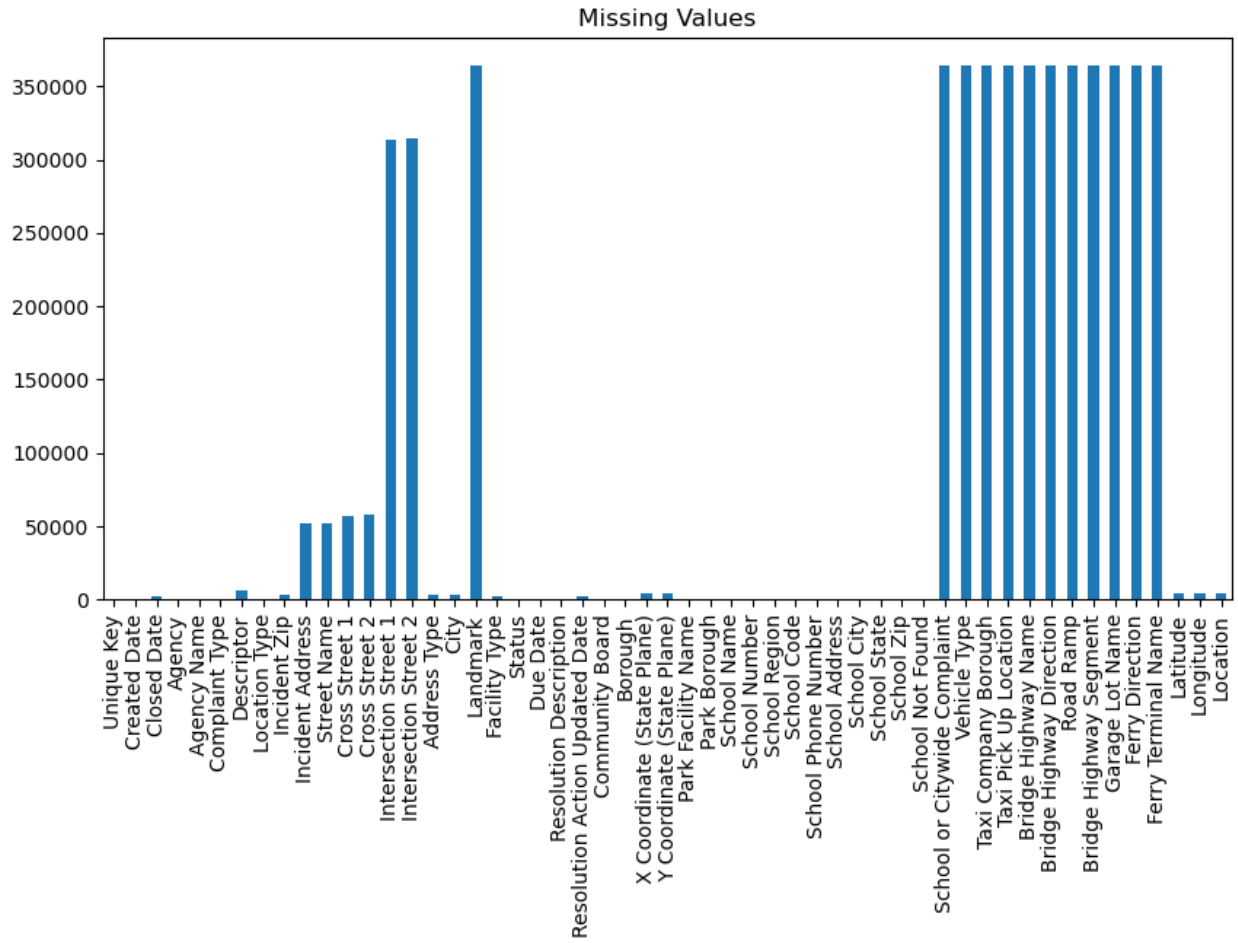
```
Unique Key                         0.000000
Created Date                       0.000000
Closed Date                        0.653120
Agency                             0.000000
Agency Name                        0.000000
Complaint Type                     0.000000
Descriptor                         1.783255
Location Type                      0.036483
Incident Zip                       0.822366
Incident Address                  14.181283
Street Name                       14.181283
Cross Street 1                    15.686941
Cross Street 2                    15.856187
Intersection Street 1             85.977540
Intersection Street 2             86.144317
Address Type                       0.892039
City                               0.822091
Landmark                          99.897136
Facility Type                      0.655314
Status                             0.000000
Due Date                           0.000823
Resolution Description             0.000000
Resolution Action Updated Date     0.658880
```

- **To plot a bar graph of missing values**

```
df.isnull().sum().plot(kind="bar",figsize=(10,5),title=("Missing Values"))
```

```
df.isnull().sum().plot(kind="bar",figsize=(10,5),title=("Missing Values"))
```

```
<Axes: title={'center': 'Missing Values'}>
```



- **Dropping columns that are not needed**

```
un_col=['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','Landmark','Intersection Street 2',
       'Intersection Street 1']
```

```
un_col=['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','Landmark','Intersection Street 2',
       'Intersection Street 1']
```

df.drop(un_col,axis=1,inplace=True)

```
df.drop(un_col,axis=1,inplace=True)
```

df.isnull().sum()/len(df)*100

```
df.isnull().sum()/len(df)*100
```

```
Unique Key                        0.000000
Created Date                      0.000000
Closed Date                       0.653120
Agency                            0.000000
Agency Name                       0.000000
Complaint Type                    0.000000
Descriptor                        1.783255
Location Type                     0.036483
Incident Zip                      0.822366
Incident Address                 14.181283
Street Name                      14.181283
Cross Street 1                   15.686941
Cross Street 2                   15.856187
Address Type                      0.892039
City                              0.822091
Facility Type                     0.655314
Status                            0.000000
Due Date                          0.000823
Resolution Description            0.000000
Resolution Action Updated Date    0.658880
Community Board                   0.000000
Borough                           0.000000
X Coordinate (State Plane)        1.105448
Y Coordinate (State Plane)        1.105448
Park Facility Name                0.000000
Park Borough                      0.000000
School Name                       0.000000
School Number                     0.000000
```

```
School Region                   0.000274
School Code                     0.000274
School Phone Number             0.000000
School Address                  0.000000
School City                     0.000000
School State                    0.000000
School Zip                      0.000274
School Not Found                0.000000
Latitude                        1.105448
Longitude                       1.105448
Location                        1.105448
dtype: float64
```

```
df=df[['Unique Key','Created Date','Closed Date','Agency',
      'Complaint Type','Descriptor','Location Type','Incident
Zip','City','Status',
      'Resolution
Description','Borough','Latitude','Longitude','Location']]

df.info()
```

```
df=df[['Unique Key','Created Date','Closed Date','Agency',
      'Complaint Type','Descriptor','Location Type','Incident Zip','City','Status',
      'Resolution Description','Borough','Latitude','Longitude','Location']]

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 364558 entries, 0 to 364557
Data columns (total 15 columns):
 #   Column                  Non-Null Count   Dtype
---  ------                  --------------   -----
 0   Unique Key              364558 non-null  int64
 1   Created Date            364558 non-null  object
 2   Closed Date             362177 non-null  object
 3   Agency                  364558 non-null  object
 4   Complaint Type          364558 non-null  object
 5   Descriptor              358057 non-null  object
 6   Location Type           364425 non-null  object
 7   Incident Zip            361560 non-null  float64
 8   City                    361561 non-null  object
 9   Status                  364558 non-null  object
 10  Resolution Description  364558 non-null  object
 11  Borough                 364558 non-null  object
 12  Latitude                360528 non-null  float64
 13  Longitude               360528 non-null  float64
 14  Location                360528 non-null  object
dtypes: float64(3), int64(1), object(11)
memory usage: 41.7+ MB
```

- **Finding the missing value columns**

```
df.isnull().sum()
```

```
df.isnull().sum()
```

```
Unique Key                      0
Created Date                    0
Closed Date                  2381
Agency                          0
Complaint Type                  0
Descriptor                   6501
Location Type                 133
Incident Zip                 2998
City                         2997
Status                          0
Resolution Description          0
Borough                         0
Latitude                     4030
Longitude                    4030
Location                     4030
dtype: int64
```

df.isnull().sum()/len(df)*100

```
df.isnull().sum()/len(df)*100
```

```
Unique Key                   0.000000
Created Date                 0.000000
Closed Date                  0.653120
Agency                       0.000000
Complaint Type               0.000000
Descriptor                   1.783255
Location Type                0.036483
Incident Zip                 0.822366
City                         0.822091
Status                       0.000000
Resolution Description       0.000000
Borough                      0.000000
Latitude                     1.105448
Longitude                    1.105448
Location                     1.105448
dtype: float64
```

df.dropna(inplace=True)
df.isnull().sum()/len(df)*100

```
df.dropna(inplace=True)
df.isnull().sum()/len(df)*100
```

```
Unique Key                  0.0
Created Date                0.0
Closed Date                 0.0
Agency                      0.0
Complaint Type              0.0
Descriptor                  0.0
Location Type               0.0
Incident Zip                0.0
City                        0.0
Status                      0.0
Resolution Description      0.0
Borough                     0.0
Latitude                    0.0
Longitude                   0.0
Location                    0.0
dtype: float64
```

Dropped missing values has they are less than 5 %

5. Analyze the date column and remove the entries if it has an incorrect timeline

```
df.head()
```

```
df.head()
```

	Unique Key	Created Date	Closed Date	Agency	Complaint Type	Descriptor	Location Type	Incident Zip	City	Status	Resolution Description	Borough	Latitude	Lo
0	32310363	12/31/2015 11:59:45 PM	01/01/2016 12:55:15 AM	NYPD	Noise - Street/Sidewalk	Loud Music/Party	Street/Sidewalk	10034.0	NEW YORK	Closed	The Police Department responded and upon arriv...	MANHATTAN	40.865682	-73
1	32309934	12/31/2015 11:59:44 PM	01/01/2016 01:26:57 AM	NYPD	Blocked Driveway	No Access	Street/Sidewalk	11105.0	ASTORIA	Closed	The Police Department responded to the complai...	QUEENS	40.775945	-73
2	32309159	12/31/2015 11:59:29 PM	01/01/2016 04:51:03 AM	NYPD	Blocked Driveway	No Access	Street/Sidewalk	10458.0	BRONX	Closed	The Police Department responded and upon arriv...	BRONX	40.870325	-73
3	32305098	12/31/2015 11:57:46 PM	01/01/2016 07:43:13 AM	NYPD	Illegal Parking	Commercial Overnight Parking	Street/Sidewalk	10461.0	BRONX	Closed	The Police Department responded to the complai...	BRONX	40.835994	-73
4	32306529	12/31/2015 11:56:58 PM	01/01/2016 03:24:42 AM	NYPD	Illegal Parking	Blocked Sidewalk	Street/Sidewalk	11373.0	ELMHURST	Closed	The Police Department responded and upon arriv...	QUEENS	40.733060	-73

```
df['Created Date']=pd.to_datetime(df['Created Date'])
df['Closed Date']=pd.to_datetime(df['Closed Date'])
df.info()
```

```
df['Created Date']=pd.to_datetime(df['Created Date'])
df['Closed Date']=pd.to_datetime(df['Closed Date'])
df.info()
```

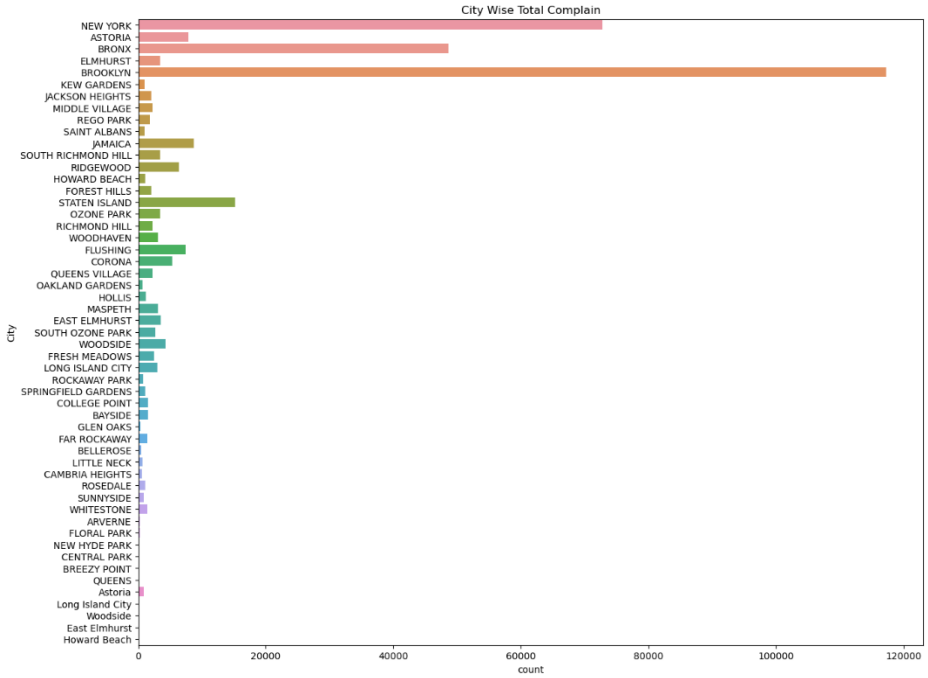
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 353891 entries, 0 to 364557
Data columns (total 15 columns):
 #   Column                  Non-Null Count   Dtype
---  ------                  --------------   -----
 0   Unique Key              353891 non-null  int64
 1   Created Date            353891 non-null  datetime64[ns]
 2   Closed Date             353891 non-null  datetime64[ns]
 3   Agency                  353891 non-null  object
 4   Complaint Type          353891 non-null  object
 5   Descriptor              353891 non-null  object
 6   Location Type           353891 non-null  object
 7   Incident Zip            353891 non-null  float64
 8   City                    353891 non-null  object
 9   Status                  353891 non-null  object
 10  Resolution Description  353891 non-null  object
 11  Borough                 353891 non-null  object
 12  Latitude                353891 non-null  float64
 13  Longitude               353891 non-null  float64
 14  Location                353891 non-null  object
dtypes: datetime64[ns](2), float64(3), int64(1), object(9)
memory usage: 43.2+ MB
```

5.1 Draw a frequency plot for city-wise complaints

```
plt.figure(figsize=(15,12))
sns.countplot(data=df,y='City')
plt.title("City Wise Total Complain")
plt.show()
```

```
#Draw a frequency plot for city-wise complaints

plt.figure(figsize=(15,12))
sns.countplot(data=df,y='City')
plt.title("City Wise Total Complain")
plt.show()
```



5.2 Draw scatter and hexbin plots for complaint concentration across Brooklyn

```
df['Borough'].unique()
```

```
df['Borough'].unique()
```

```
array(['MANHATTAN', 'QUEENS', 'BRONX', 'BROOKLYN', 'STATEN ISLAND',
       'Unspecified'], dtype=object)
```

```
df_brooklyn=df[df['Borough']=='BROOKLYN']
df_brooklyn
```

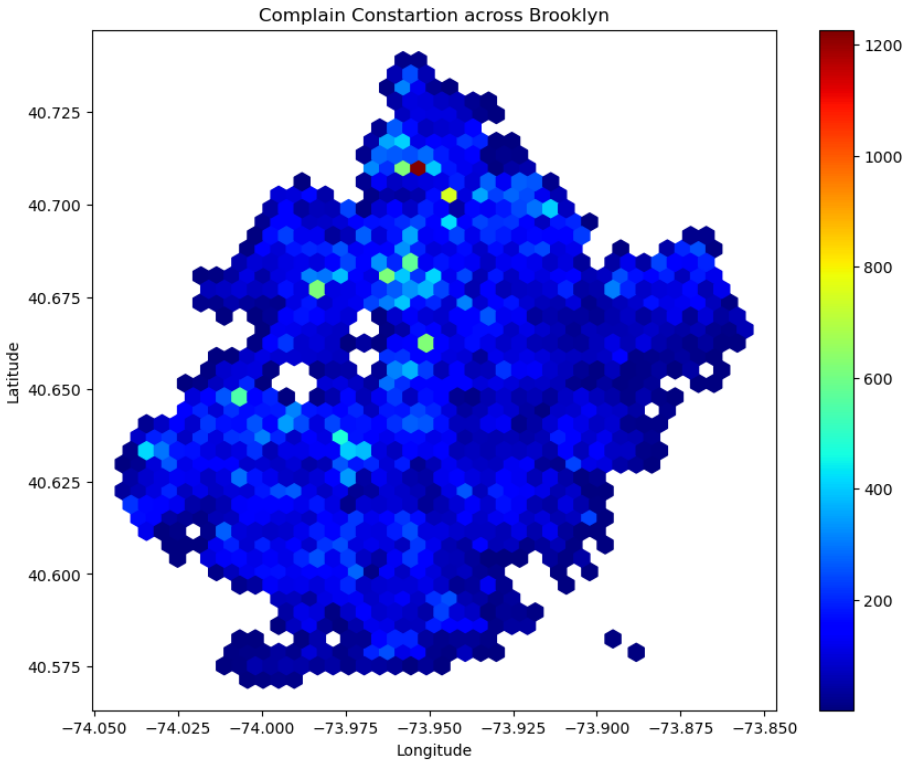
```
df_brooklyn=df[df['Borough']=='BROOKLYN']
df_brooklyn
```

	Unique Key	Created Date	Closed Date	Agency	Complaint Type	Descriptor	Location Type	Incident Zip	City	Status	Resolution Description	Borough	Latitude	L
5	32306554	2015-12-31 23:56:30	2016-01-01 01:50:11	NYPD	Illegal Parking	Posted Parking Sign Violation	Street/Sidewalk	11215.0	BROOKLYN	Closed	The Police Department responded and upon arriv...	BROOKLYN	40.660823	-7
9	32308391	2015-12-31 23:53:58	2016-01-01 01:17:40	NYPD	Blocked Driveway	No Access	Street/Sidewalk	11219.0	BROOKLYN	Closed	The Police Department responded and upon arriv...	BROOKLYN	40.623793	-7
13	32305074	2015-12-31 23:47:58	2016-01-01 08:18:47	NYPD	Illegal Parking	Posted Parking Sign Violation	Street/Sidewalk	11208.0	BROOKLYN	Closed	The Police Department responded to the complai...	BROOKLYN	40.687511	-7
17	32310273	2015-12-31 23:44:52	2016-01-01 00:36:10	NYPD	Noise - Commercial	Loud Music/Party	Club/Bar/Restaurant	11217.0	BROOKLYN	Closed	The Police Department responded to the complai...	BROOKLYN	40.679154	-7
18	32306617	2015-12-31 23:40:59	2016-01-01 02:37:28	NYPD	Noise - Commercial	Loud Music/Party	Club/Bar/Restaurant	11234.0	BROOKLYN	Closed	The Police Department responded to the complai...	BROOKLYN	40.616550	-7
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

```
df_brooklyn[['Latitude','Longitude']].plot(kind='hexbin',x='Long
itude',y='Latitude',
                                           title="Complain
Constartion across Brooklyn",
                                           figsize=(10,8),
                                           gridsize=40,
                                           colormap='jet',
                                           mincnt=1)
```

```
: df_brooklyn[['Latitude','Longitude']].plot(kind='hexbin',x='Longitude',y='Latitude',
                                             title="Complain Constartion across Brooklyn",
                                             figsize=(10,8),
                                             gridsize=40,
                                             colormap='jet',
                                             mincnt=1)
```

```
: <Axes: title={'center': 'Complain Constartion across Brooklyn'}, xlabel='Longitude', ylabel='Latitude'>
```



6. Find major types of complaints:

6.1 Plot a bar graph of count vs. complaint types

```
df['Complaint Type'].value_counts()
```

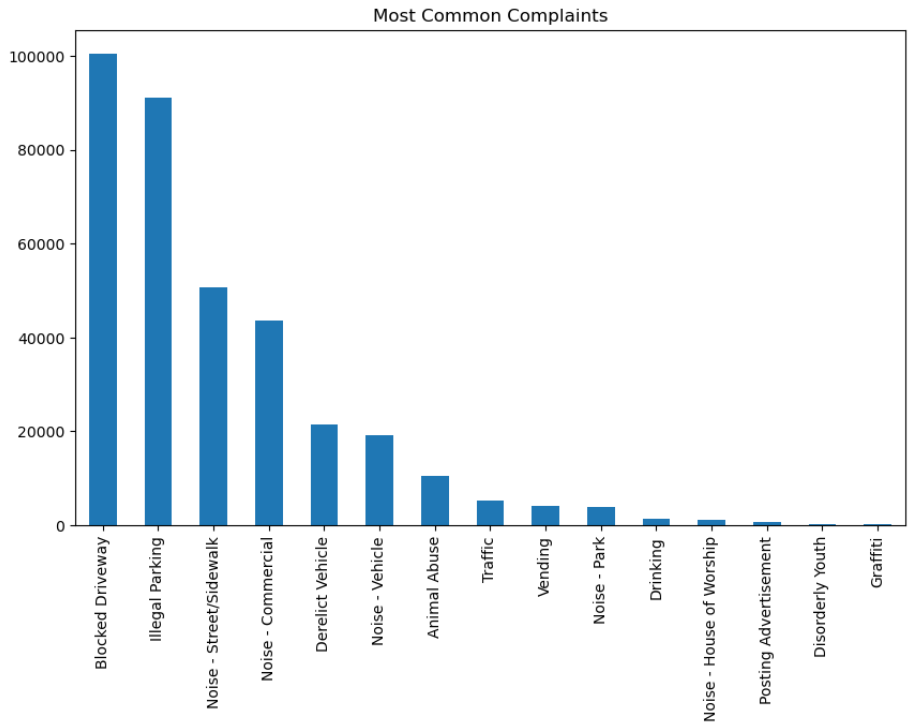
```
df['Complaint Type'].value_counts()
```

```
Blocked Driveway            100455
Illegal Parking              91057
Noise - Street/Sidewalk      50791
Noise - Commercial           43623
Derelict Vehicle             21419
Noise - Vehicle              19122
Animal Abuse                 10500
Traffic                       5161
Vending                       4162
Noise - Park                  3994
Drinking                      1399
Noise - House of Worship      1059
Posting Advertisement          678
Disorderly Youth               314
Graffiti                       157
Name: Complaint Type, dtype: int64
```

```
df['Complaint
Type'].value_counts().plot(kind="bar",figsize=(10,6),title="Most
Common Complaints")
```

```
df['Complaint Type'].value_counts().plot(kind="bar",figsize=(10,6),title="Most Common Complaints")
```

```
<Axes: title={'center': 'Most Common Complaints'}>
```



Highest complaint type is Blocked Driveway from graph.

6.2 Find the top 10 types of complaints

```
df['Complaint Type'].value_counts().sort_values(ascending=False)[:10]
```

```
df['Complaint Type'].value_counts().sort_values(ascending=False)[:10]
```

```
Blocked Driveway           100455
Illegal Parking             91057
Noise - Street/Sidewalk     50791
Noise - Commercial          43623
Derelict Vehicle            21419
Noise - Vehicle             19122
Animal Abuse                10500
Traffic                      5161
Vending                      4162
Noise - Park                 3994
Name: Complaint Type, dtype: int64
```

6.3 Display the types of complaints in each city in a separate dataset

```
df2=df.groupby(['City','Complaint
Type']).size().unstack().fillna(0)

df2
```

```
df2=df.groupby(['City','Complaint Type']).size().unstack().fillna(0)
```

```
df2
```

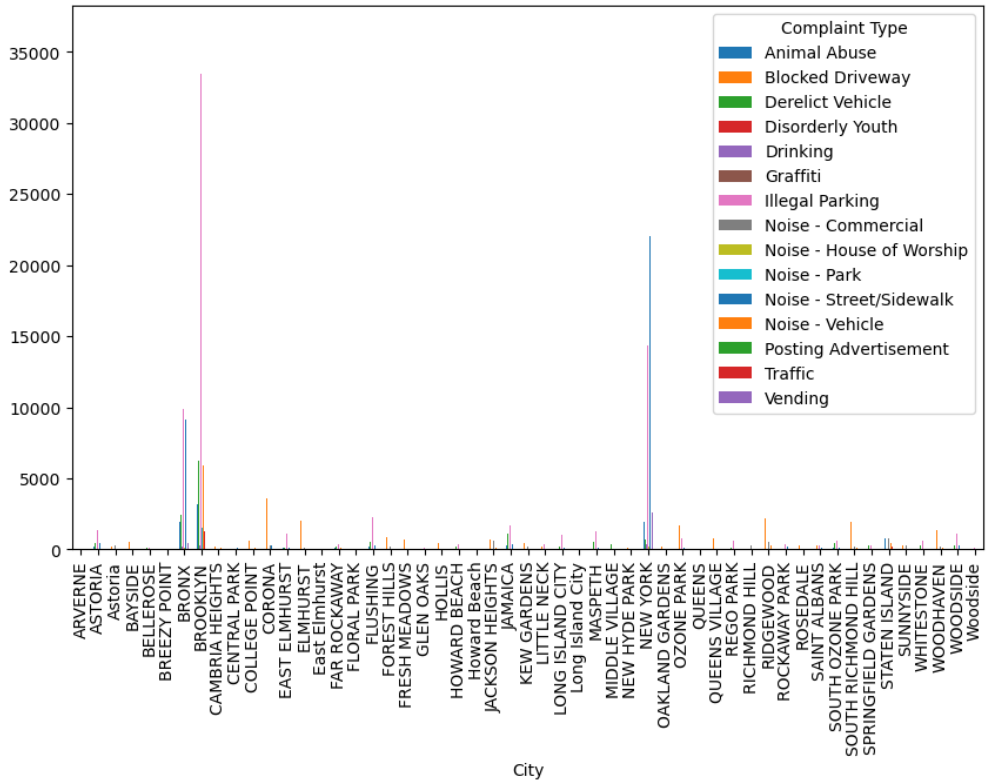
Complaint Type City	Animal Abuse	Blocked Driveway	Derelict Vehicle	Disorderly Youth	Drinking	Graffiti	Illegal Parking	Noise - Commercial	Noise - House of Worship	Noise - Park	Noise - Street/Sidewalk	Noise - Vehicle	Posting Advertisement	Traf
ARVERNE	46.0	50.0	32.0	2.0	1.0	1.0	62.0	2.0	14.0	2.0	29.0	9.0	0.0	1
ASTORIA	170.0	3436.0	426.0	5.0	43.0	4.0	1337.0	1640.0	21.0	64.0	408.0	236.0	3.0	60
Astoria	0.0	159.0	14.0	0.0	0.0	0.0	277.0	310.0	0.0	0.0	145.0	0.0	0.0	0
BAYSIDE	53.0	513.0	231.0	2.0	1.0	3.0	635.0	47.0	3.0	3.0	17.0	24.0	0.0	9
BELLEROSE	15.0	138.0	120.0	2.0	1.0	0.0	131.0	38.0	1.0	1.0	13.0	11.0	1.0	9
BREEZY POINT	2.0	3.0	3.0	0.0	1.0	0.0	16.0	4.0	0.0	0.0	1.0	1.0	0.0	0
BRONX	1966.0	17048.0	2398.0	66.0	205.0	15.0	9853.0	2941.0	90.0	523.0	9118.0	3544.0	17.0	425
BROOKLYN	3185.0	36414.0	6242.0	79.0	291.0	60.0	33446.0	13847.0	387.0	1557.0	13943.0	5932.0	58.0	1253

7. Visualize the major types of complaints in each city

```
df.groupby(['City','Complaint
Type']).size().unstack().fillna(0).plot(kind='bar',figsize=(10,6
))
```

```
df.groupby(['City','Complaint Type']).size().unstack().fillna(0).plot(kind='bar',figsize=(10,6))
```

```
<Axes: xlabel='City'>
```



8. Check if the average response time across various types of complaints

```
df['Resolution_Time']=(df['Closed Date']-df['Created
Date']).dt.days
(df['Closed Date']-df['Created Date']).dt.days
```

```
df['Resolution_Time']=(df['Closed Date']-df['Created Date']).dt.days
(df['Closed Date']-df['Created Date']).dt.days
```

```
0         0
1         0
2         0
3         0
4         0
         ..
364553    0
364554    0
364555    0
364556    0
364557    0
Length: 353891, dtype: int64
```

```
df['Resolution_Time'].value_counts()
```

```
df['Resolution_Time'].value_counts()
```

```
0      350061
1        3172
2         445
3         126
5          34
4          30
6          13
8           3
24          3
9           2
21          1
7           1
Name: Resolution_Time, dtype: int64
```

```
df.groupby('Complaint
Type')['Resolution_Time'].mean().plot(kind='bar')
```

```
df.groupby('Complaint Type')['Resolution_Time'].mean().plot(kind='bar')
```

```
<Axes: xlabel='Complaint Type'>
```