Simplilearn_Python_Project_1_Customer_Service_Requests_Analysis

September 22, 2021

1 Project 1 - Customer Service Requests Analysis.ipynb

DESCRIPTION

Background of Problem Statement : NYC 311's mission is to provide the public with quick and easy access to all New York City government services and information while offering the best customer service. Each day, NYC311 receives thousands of requests related to several hundred types of non-emergency services, including noise complaints, plumbing issues, and illegally parked cars. These requests are received by NYC311 and forwarded to the relevant agencies such as the police, buildings, or transportation. The agency responds to the request, addresses it, and then closes it.

Problem Objective : Perform a service request data analysis of New York City 311 calls. You will focus on the data wrangling techniques to understand the pattern in the data and also visualize the major complaint types. Domain: Customer Service

Analysis Tasks to be performed:

(Perform a service request data analysis of New York City 311 calls)

```
Import a 311 NYC service request.
Read or convert the columns 'Created Date' and Closed Date' to datetime datatype and create a r
Provide major insights/patterns that you can offer in a visual format (graphs or tables); at le
Order the complaint types based on the average 'Request_Closing_Time', grouping them for differ
Perform a statistical test for the following:
```

Please note: For the below statements you need to state the Null and Alternate and then provide a statistical test to accept or reject the Null Hypothesis along with the corresponding 'p-value'.

```
Whether the average response time across complaint types is similar or not (overall)
Are the type of complaint or service requested and location related?
```

Dataset Description : Field Description

Unique Key (Plain text) - Unique identifier for the complaints

Created Date (Date and Time) - The date and time on which the complaint is raised

Closed Date (Date and Time) - The date and time on which the complaint is closed

Agency (Plain text) - Agency code

Agency Name (Plain text) - Name of the agency

Complaint Type (Plain text) - Type of the complaint

Descriptor (Plain text) - Complaint type label (Heating - Heat, Traffic Signal Condition - Controller)

Location Type (Plain text) - Type of the location (Residential, Restaurant, Bakery, etc)

Incident Zip (Plain text) - Zip code for the location

Incident Address (Plain text) - Address of the location

Street Name (Plain text) - Name of the street

Cross Street 1 (Plain text) - Detail of cross street

Cross Street 2 (Plain text) - Detail of another cross street

Intersection Street 1 (Plain text) - Detail of intersection street if any

Intersection Street 2 (Plain text) - Detail of another intersection street if any

Address Type (Plain text) - Categorical (Address or Intersection)

City (Plain text) - City for the location

Landmark (Plain text) - Empty field

Facility Type (Plain text) - N/A

Status (Plain text) - Categorical (Closed or Pending)

Due Date (Date and Time) - Date and time for the pending complaints

Resolution Action Updated Date (Date and Time) - Date and time when the resolution was provided

Community Board (Plain text) - Categorical field (specifies the community board with its code)

Borough (Plain text) - Categorical field (specifies the community board)

X Coordinate (State Plane) (Number)

Y Coordinate (State Plane) (Number)

Park Facility Name (Plain text) - Unspecified

Park Borough (Plain text) - Categorical (Unspecified, Queens, Brooklyn etc)

School Name (Plain text) - Unspecified

School Number (Plain text) - Unspecified

School Region (Plain text) - Unspecified

School Code (Plain text) - Unspecified

School Phone Number (Plain text) - Unspecified

School Address (Plain text) - Unspecified

School City (Plain text) - Unspecified

School State (Plain text) - Unspecified

School Zip (Plain text) - Unspecified

School Not Found (Plain text) - Empty Field

School or Citywide Complaint (Plain text) - Empty Field

Vehicle Type (Plain text) - Empty Field

Taxi Company Borough (Plain text) - Empty Field

Taxi Pick Up Location (Plain text) - Empty Field

Bridge Highway Name (Plain text) - Empty Field

Bridge Highway Direction (Plain text) - Empty Field

Road Ramp (Plain text) - Empty Field

Bridge Highway Segment (Plain text) - Empty Field

Garage Lot Name (Plain text) - Empty Field

Ferry Direction (Plain text) - Empty Field

Ferry Terminal Name (Plain text) - Empty Field

Latitude (Number) - Latitude of the location

Longitude (Number) - Longitude of the location

Location (Location) - Coordinates (Latitude, Longitude)

[236]:
```
import pandas as pd
```

[237]:
```
filepath = '/content/drive/MyDrive/My Projects/Simplilearn/01 Data Science with␣
↪Python/Project 1 - Customer Service Requests Analysis/
↪311_Service_Requests_from_2010_to_Present.csv'
```

[238]:
```
dataset = pd.read_csv(filepath, parse_dates=["Created Date", "Closed Date"],␣
↪low_memory=False)
```

[239]:
```
dataset.shape
```

[239]:
```
(300698, 53)
```

[240]:
```
dataset.columns
```

[240]:
```
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')
```

[241]:
```
dataset.head()
```

[241]:
```
   Unique Key  …                                  Location
0    32310363  …   (40.86568153633767, -73.92350095571744)
1    32309934  …  (40.775945312321085, -73.91509393898605)
2    32309159  …  (40.870324522111424, -73.88852464418646)
3    32305098  …   (40.83599404683083, -73.82837939584206)
4    32306529  …  (40.733059618956815, -73.87416975810375)

[5 rows x 53 columns]
```

[242]:
```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 300698 entries, 0 to 300697
Data columns (total 53 columns):
 #   Column                          Non-Null Count   Dtype
---  ------                          --------------   -----
 0   Unique Key                      300698 non-null  int64
 1   Created Date                    300698 non-null  datetime64[ns]
 2   Closed Date                     298534 non-null  datetime64[ns]
 3   Agency                          300698 non-null  object
 4   Agency Name                     300698 non-null  object
 5   Complaint Type                  300698 non-null  object
 6   Descriptor                      294784 non-null  object
 7   Location Type                   300567 non-null  object
 8   Incident Zip                    298083 non-null  float64
 9   Incident Address                256288 non-null  object
 10  Street Name                     256288 non-null  object
 11  Cross Street 1                  251419 non-null  object
 12  Cross Street 2                  250919 non-null  object
 13  Intersection Street 1           43858 non-null   object
 14  Intersection Street 2           43362 non-null   object
 15  Address Type                    297883 non-null  object
 16  City                            298084 non-null  object
 17  Landmark                        349 non-null     object
 18  Facility Type                   298527 non-null  object
 19  Status                          300698 non-null  object
 20  Due Date                        300695 non-null  object
 21  Resolution Description          300698 non-null  object
 22  Resolution Action Updated Date  298511 non-null  object
```

```
 23  Community Board                 300698 non-null  object
 24  Borough                         300698 non-null  object
 25  X Coordinate (State Plane)      297158 non-null  float64
 26  Y Coordinate (State Plane)      297158 non-null  float64
 27  Park Facility Name              300698 non-null  object
 28  Park Borough                    300698 non-null  object
 29  School Name                     300698 non-null  object
 30  School Number                   300698 non-null  object
 31  School Region                   300697 non-null  object
 32  School Code                     300697 non-null  object
 33  School Phone Number             300698 non-null  object
 34  School Address                  300698 non-null  object
 35  School City                     300698 non-null  object
 36  School State                    300698 non-null  object
 37  School Zip                      300697 non-null  object
 38  School Not Found                300698 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             243 non-null     object
 44  Bridge Highway Direction        243 non-null     object
 45  Road Ramp                       213 non-null     object
 46  Bridge Highway Segment          213 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                        297158 non-null  float64
 51  Longitude                       297158 non-null  float64
 52  Location                        297158 non-null  object
dtypes: datetime64[ns](2), float64(10), int64(1), object(40)
memory usage: 121.6+ MB
```

[243]:
```
dataset.describe()
```

[243]:

	Unique Key	Incident Zip	…	Latitude	Longitude
count	3.006980e+05	298083.000000	…	297158.000000	297158.000000
mean	3.130054e+07	10848.888645	…	40.725885	-73.925630
std	5.738547e+05	583.182081	…	0.082012	0.078454
min	3.027948e+07	83.000000	…	40.499135	-74.254937
25%	3.080118e+07	10310.000000	…	40.669796	-73.972142
50%	3.130436e+07	11208.000000	…	40.718661	-73.931781
75%	3.178446e+07	11238.000000	…	40.781840	-73.876805
max	3.231065e+07	11697.000000	…	40.912869	-73.700760

[8 rows x 11 columns]

[244]:
```
dataset.head()
```

[244]:
```
   Unique Key  …                                  Location
0    32310363  …   (40.86568153633767, -73.92350095571744)
1    32309934  …  (40.775945312321085, -73.91509393898605)
2    32309159  …  (40.870324522111424, -73.88852464418646)
3    32305098  …   (40.83599404683083, -73.82837939584206)
4    32306529  …  (40.733059618956815, -73.87416975810375)

[5 rows x 53 columns]
```

Handling Missing Values

[245]:
```
dataset.isnull().sum()
```

[245]:
```
Unique Key                          0
Created Date                        0
Closed Date                      2164
Agency                              0
Agency Name                         0
Complaint Type                      0
Descriptor                       5914
Location Type                     131
Incident Zip                     2615
Incident Address                44410
Street Name                     44410
Cross Street 1                  49279
Cross Street 2                  49779
Intersection Street 1          256840
Intersection Street 2          257336
Address Type                     2815
City                             2614
Landmark                       300349
Facility Type                    2171
Status                              0
Due Date                            3
Resolution Description              0
Resolution Action Updated Date   2187
Community Board                     0
Borough                             0
X Coordinate (State Plane)       3540
Y Coordinate (State Plane)       3540
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    300698
Vehicle Type                    300698
Taxi Company Borough            300698
Taxi Pick Up Location           300698
Bridge Highway Name             300455
Bridge Highway Direction        300455
Road Ramp                       300485
Bridge Highway Segment          300485
Garage Lot Name                 300698
Ferry Direction                 300697
Ferry Terminal Name             300696
Latitude                          3540
Longitude                         3540
Location                          3540
dtype: int64
```

[246]:
```
dataset.shape
```

[246]:
```
(300698, 53)
```

since the following columns all all empty we are dropping them all.

'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'

The following columns are dropped because they contain large counts of missing data.

['Incident Address', 'Street Name', 'Cross Street 1', 'Cross Street 2', 'Intersection Street 1', 'Intersection Street 2']

[247]:
```
dataset = dataset.drop(columns=['School or Citywide Complaint', 'Vehicle␣
↪Type','Landmark', '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','Incident Address', 'Street Name', 'Cross Street 1', 'Cross Street 2',␣
↪'Intersection Street 1', 'Intersection Street 2'])
```

[248]:
```
dataset.shape
```

[248]:
```
(300698, 35)
```

[249]:
```
dataset = dataset.fillna( method ='ffill')
```

[251]:
```
dataset = pd.DataFrame(dataset)
```

dropping the rows with missing values.

[253]:
```
dataset.isnull().sum()
```

[253]:
```
Unique Key                        0
Created Date                      0
Closed Date                       0
Agency                            0
Agency Name                       0
Complaint Type                    0
Descriptor                        0
Location Type                     0
Incident Zip                      0
Address Type                      0
City                              0
Facility Type                     0
Status                            0
Due Date                          0
Resolution Description            0
Resolution Action Updated Date    0
Community Board                   0
Borough                           0
X Coordinate (State Plane)        0
Y Coordinate (State Plane)        0
Park Facility Name                0
Park Borough                      0
School Name                       0
School Number                     0
School Region                     0
School Code                       0
School Phone Number               0
School Address                    0
School City                       0
School State                      0
School Zip                        0
School Not Found                  0
Latitude                          0
Longitude                         0
Location                          0
dtype: int64
```

[254]:
```
dataset.shape
```

[254]:
```
(300698, 35)
```

Checking for Duplicate Values

[255]:
```
dataset.duplicated().sum()
```

[255]: 0

[256]:
```
import matplotlib.pyplot as plt
```

[257]:
```
dataset['Status'].value_counts()
```

[257]:
```
Closed      298471
Open          1439
Assigned       786
Draft            2
Name: Status, dtype: int64
```

2 Request Closing Time

[258]:
```
dataset['Request_Closing_Time'] =  dataset['Closed Date'] - dataset['Created␣
↪Date']
```

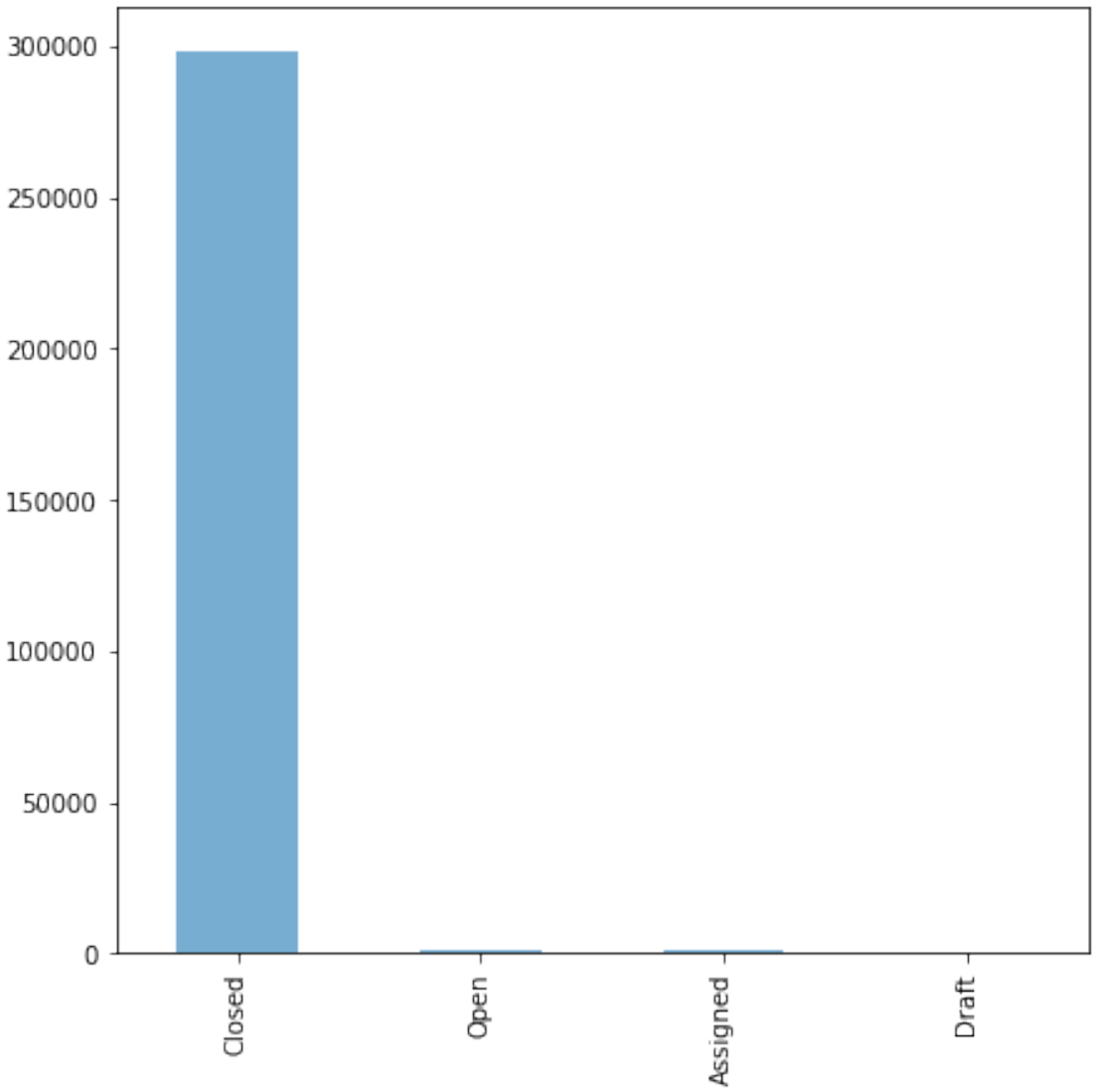
3 Major Insights

3.0.1 Status Types

[259]:
```
dataset['Status'].value_counts()
```

[259]:
```
Closed      298471
Open          1439
Assigned       786
Draft            2
Name: Status, dtype: int64
```

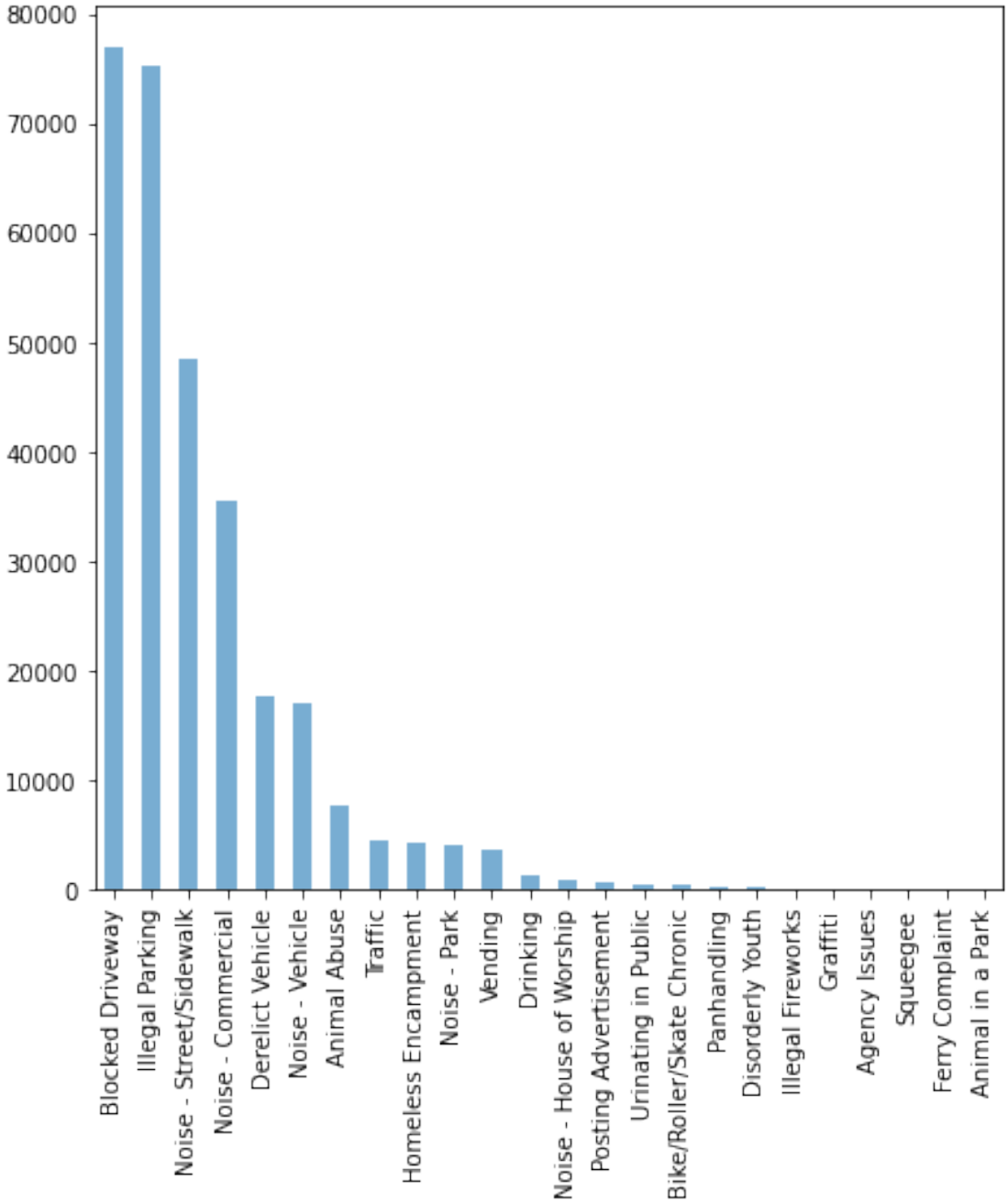
[260]:
```
dataset['Status'].value_counts().plot(kind='bar',alpha=0.6,figsize=(7,7))
plt.show()
```



We can observe that majority of the service requests status is closed and a small fraction of requests are “Open”, “Assigned” or under “Draft”

3.0.2 Top Complaints

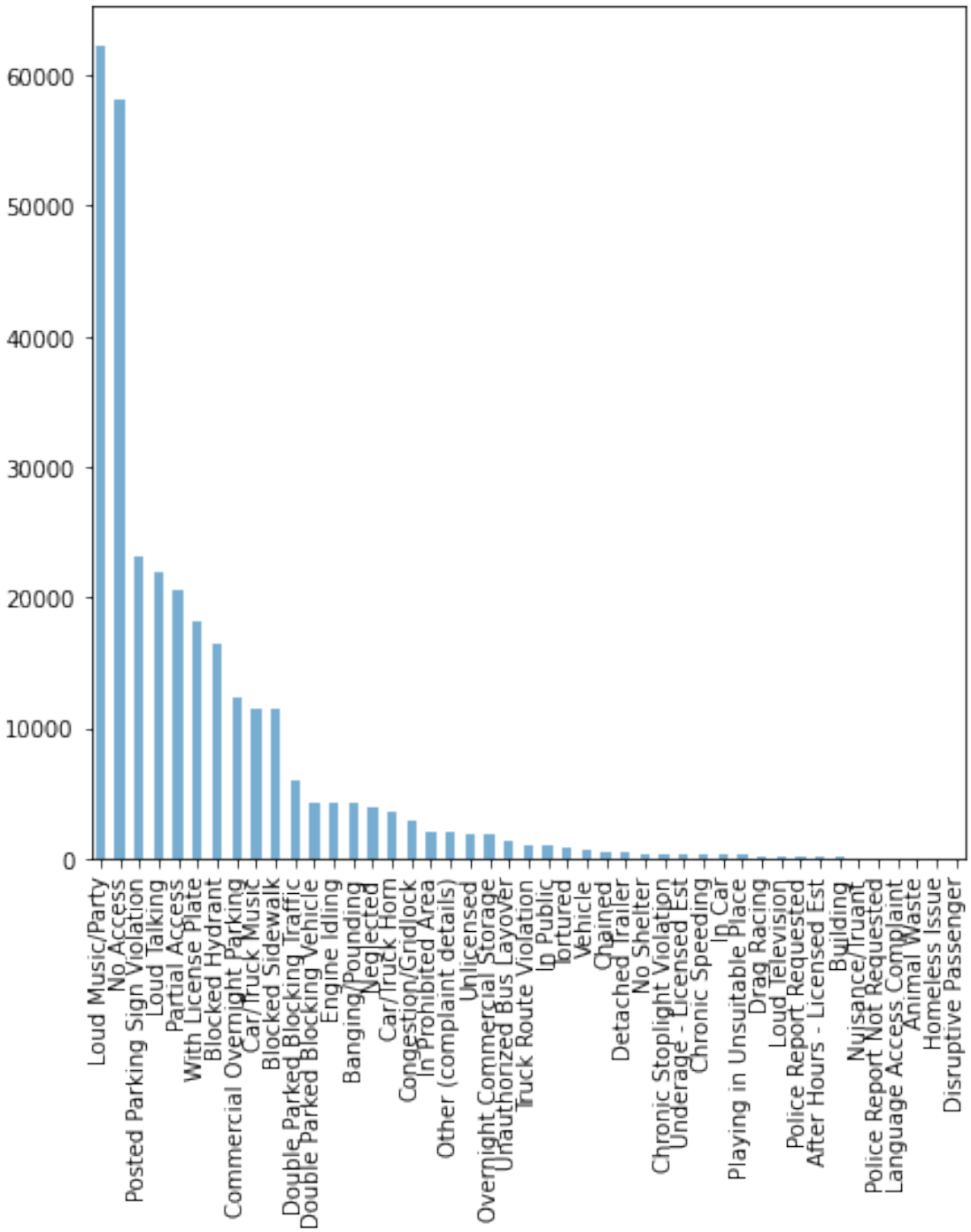
[261]:
```
dataset['Complaint Type'].value_counts().plot(kind='bar',alpha=0.
↪6,figsize=(7,7))
plt.show()
```



We can Observe that majority of the complaint type falls under “Blocked Driveway” or “Illegal Parking” or “Noise - Street/Sidewalk” Category

3.0.3 Descriptor

[262]:
```
dataset['Descriptor'].value_counts().plot(kind='bar',alpha=0.6,figsize=(7,7))
plt.show()
```



We can observe that majority of the compaint description falls under “Loud music/Party” and “No

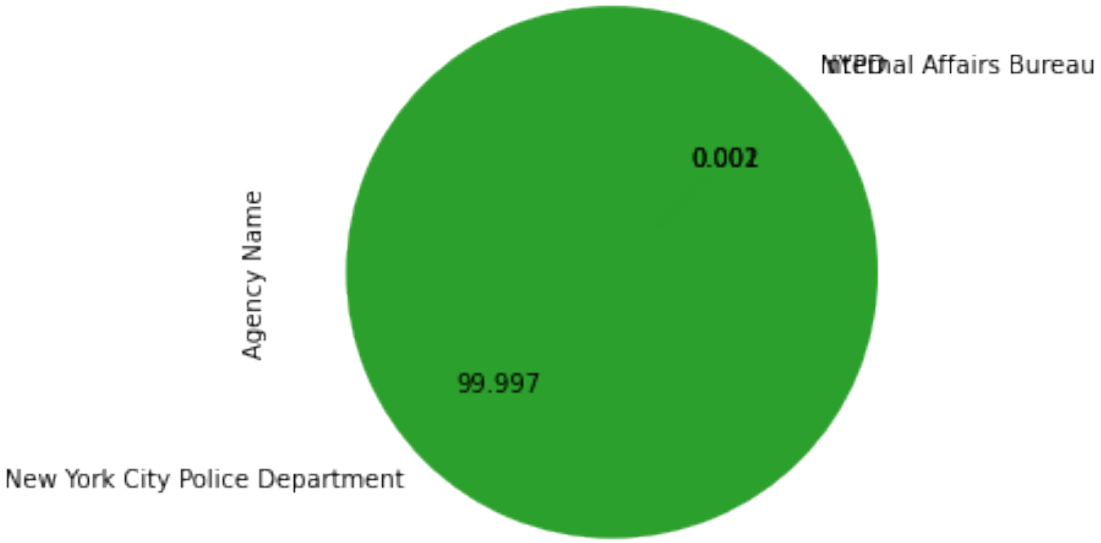
Access" Category

3.0.4 Agency Names

[263]:
```
dataset['Agency Name'].value_counts()
```

[263]:
```
New York City Police Department    300690
Internal Affairs Bureau                 6
NYPD                                    2
Name: Agency Name, dtype: int64
```

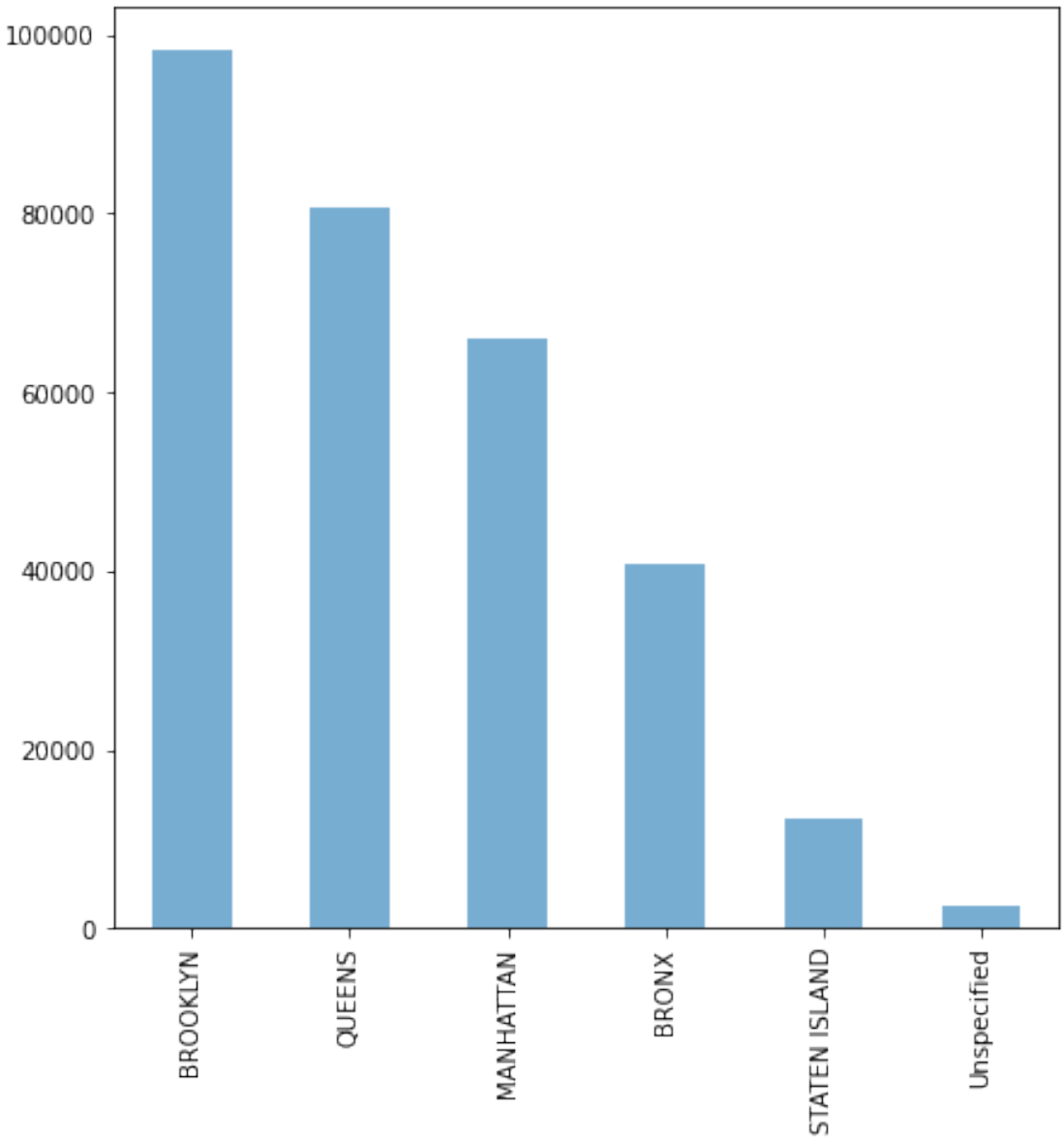
[264]:
```
dataset['Agency Name'].groupby(dataset['Agency Name']).size().plot(kind='pie',␣
↪autopct='%.3f', startangle=45)
plt.show()
```



We can see that there Majority of the complaints are registed with "New York City Police Department" and a fraction of the cases are registered with "Internal Affairs Bureau" and "NYPD"

3.0.5 Borough

[265]:
```
dataset['Borough'].value_counts().plot(kind='bar',alpha=0.6,figsize=(7,7))
plt.show()
```



We can observe that the cases is highest in Brooklyn district

3.1 Request Date vs Frequency

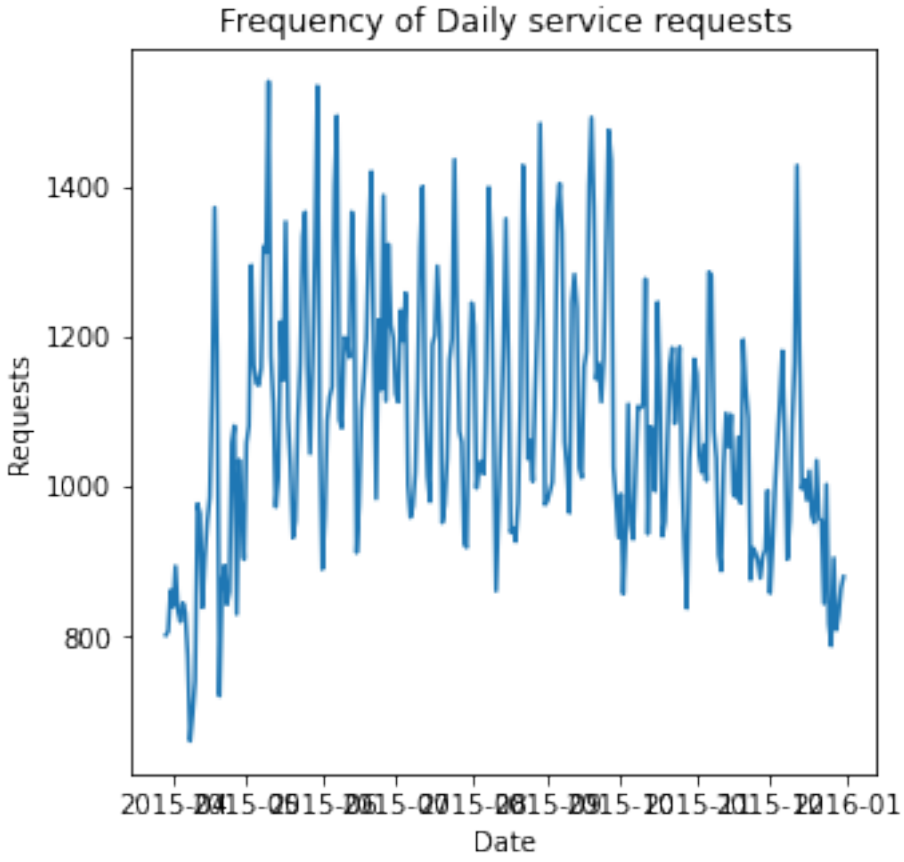
Plotting the frequency of daily service requests

[266]:
```
import datetime as dt
```

[267]:
```
daily_dates = pd.DataFrame(dataset['Created Date'].dt.date).reset_index()
```

[268]:
```
daily_dates = daily_dates.groupby(['Created Date']).count().reset_index()
```

[269]:
```
plt.plot(daily_dates['Created Date'], daily_dates['index'])
plt.rcParams['figure.figsize'] = [15, 15]
plt.title('Frequency of Daily service requests')
plt.xlabel('Date')
plt.ylabel('Requests')
plt.show()
```



We can observed that the frequecy of requests increased during the period of April 2015 and January 2016 and then decreased

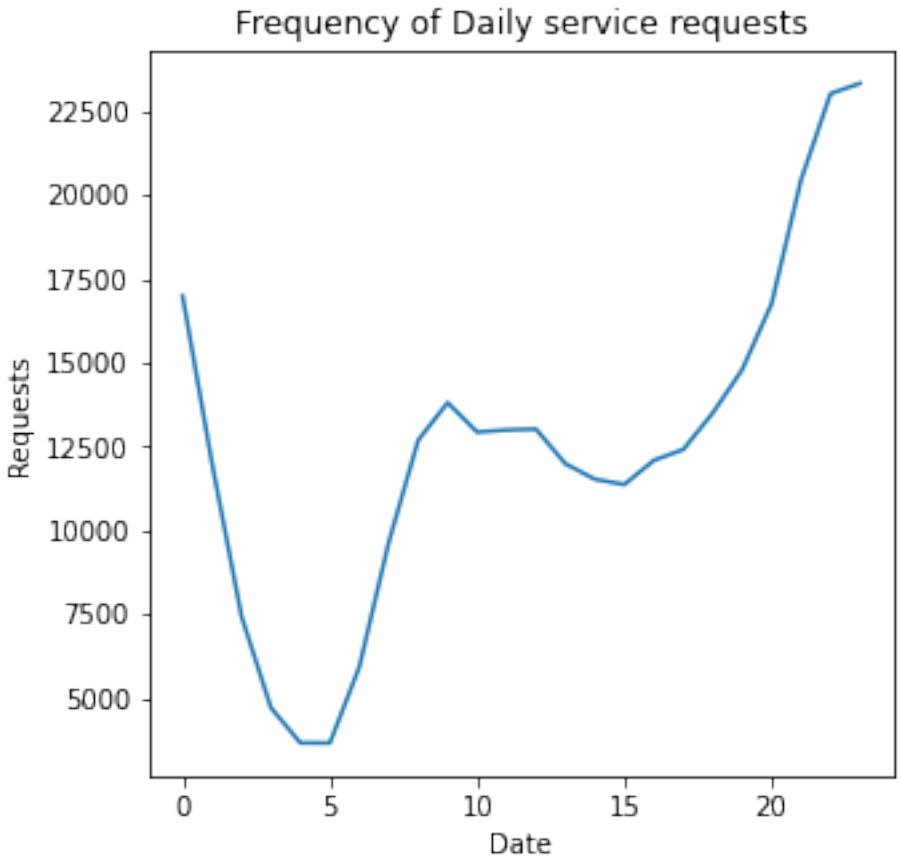
3.2 Created Hour

[270]:
```
created_hour = pd.DataFrame(dataset['Created Date'].dt.hour).reset_index()
```

[271]:
```
created_hour = created_hour.groupby(['Created Date']).count().reset_index()
```

[293]:
```
plt.plot(created_hour['Created Date'], created_hour['index'])
plt.rcParams['figure.figsize'] = [10,8]
plt.title('Frequency of Daily service requests')
plt.xlabel('Date')
```

```
plt.ylabel('Requests')
plt.show()
```



We can observe that the frequency of requests is highest around 12 AM and lowest around 3-5 AM.

4 Average Request Closing Time

[273]:
```
print("Mean Request Closing Time: ", dataset['Request_Closing_Time'].mean())
```

```
Mean Request Closing Time:  0 days 04:18:49.770005121
```

5 Hypothesis Testing

5.1 1. Whether the average response time across complaint types is similar or not (overall)

Converting Request_Closing_Time to minutes for more precise results.

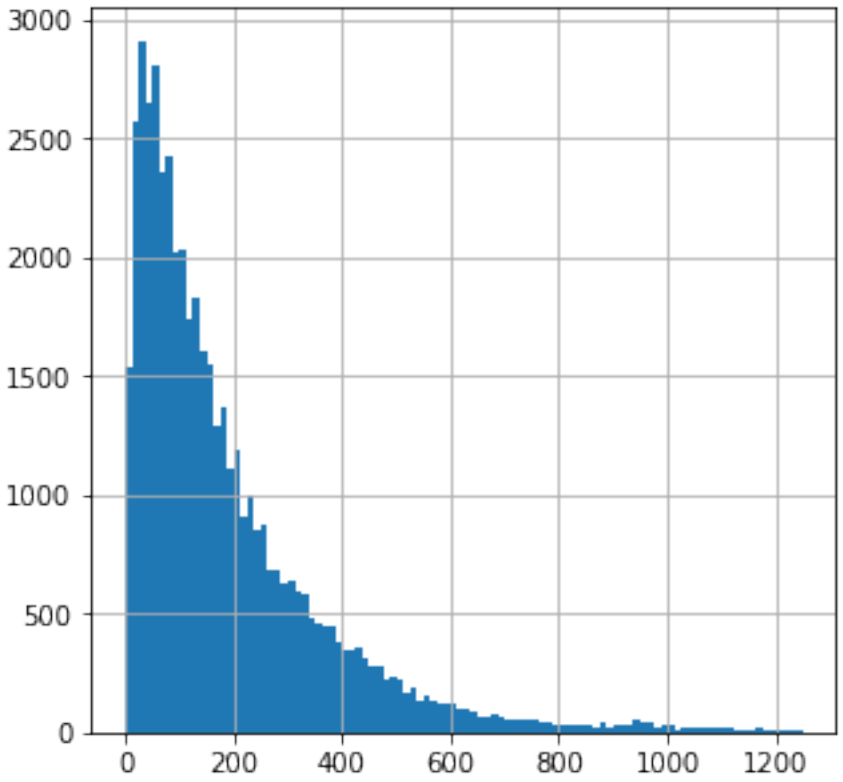
[274]:
```
dataset['Request_Closing_Minutes'] = dataset['Request_Closing_Time'].
↪astype('timedelta64[m]')+1
```

[275]:
```
dataset['Request_Closing_Minutes'].head()
```

[275]:
```
0     56.0
1     87.0
2    292.0
3    466.0
4    208.0
Name: Request_Closing_Minutes, dtype: float64
```

distribution of our Request_Closing_Minutes data for 'Noise - Street/Sidewalk' complaint type.

[276]:
```
dataset = dataset[dataset['Complaint Type'].notnull()]
original = dataset[dataset['Complaint Type']=='Noise - Street/
↪Sidewalk']['Request_Closing_Minutes']
plt.rcParams['figure.figsize'] = [5,5]
original.hist(bins=100,range=(0,1250))
plt.show()
```



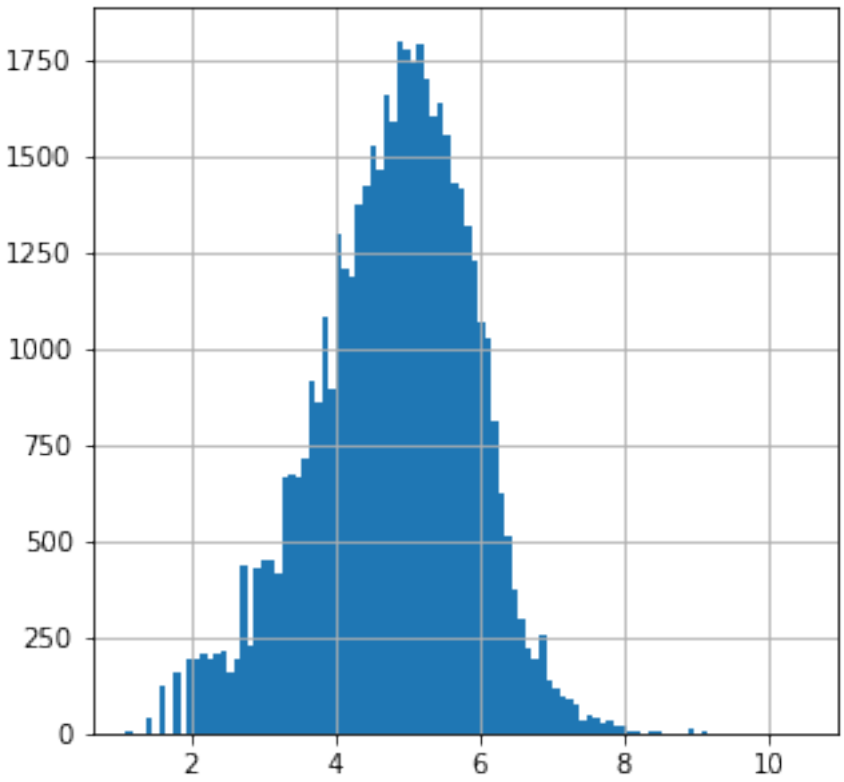
[277]:
```
original.describe()
```

[277]:
```
count    48612.000000
mean       207.739694
std        325.702552
min          3.000000
25%         61.000000
50%        133.000000
75%        260.000000
max      35573.000000
Name: Request_Closing_Minutes, dtype: float64
```

[278]:
```
import numpy as np
```

[279]:
```
data = {}
for complaint in dataset['Complaint Type'].unique():
  data[complaint] = np.log(dataset[dataset['Complaint␣
 ↪Type']==complaint]['Request_Closing_Minutes'])
```

[280]:
```
data['Noise - Street/Sidewalk'].hist(bins=100)
plt.show()
```



[281]:
```
for complaint in data.keys():
    print(complaint, ':\t', data[complaint].std())
```

```
Noise - Street/Sidewalk :        1.0874813672401784
Blocked Driveway :       0.9691386274397016
Illegal Parking :        1.0669597020557937
Derelict Vehicle :       1.2457252611009721
Noise - Commercial :     1.0763091176697792
Noise - House of Worship :       1.157426484535852
Posting Advertisement :  1.193586677415039
Noise - Vehicle :        1.0642536735565946
Animal Abuse :   1.0349692057796114
Vending :        1.0985617859387775
Traffic :        1.1693690064170892
Drinking :       1.0354164293056451
Bike/Roller/Skate Chronic :      1.1559861346703018
Panhandling :    1.0545634443510679
Noise - Park :   1.1040725556180655
Homeless Encampment :    1.0212999697423413
Urinating in Public :    1.0896898895618483
Graffiti :       1.0581967861803852
Disorderly Youth :       1.0276748370244453
Illegal Fireworks :      1.1905406895067956
Ferry Complaint :        0.5624511395192774
Agency Issues :  0.8285353314860184
Squeegee :       0.8469384425802964
Animal in a Park :       nan
```

5.1.1 ANOVA Test (Analysis of Variance)

as we have to compare the means of more than two groups.

Conditions for test:

- All distributions must follow a normal distributions curve. We have verified this after the log transformation.
- Standard deviation for all groups must be same. Above output proves that this is true.
- All samples are drawn independently of each other.

Null Hypothesis: Average response time for all the complaints type is similar.

Alternate Hypothesis: Average response time for all the complaints type is not similar.

if p < alpha(0.05): Reject Null Hypothesis, Average response time for all the complaints type is not similar.

if p > alpha(0.05): Fail to reject Null Hypothesis, Average response time for all the complaints type is similar.

[282]:
```
from scipy.stats import f_oneway
```

[294]:
```
stat, p = f_oneway(data['Blocked Driveway'],
                   data['Illegal Parking'],
                   data['Noise - Street/Sidewalk'],
```

```
                    data['Noise - Commercial'])

print('Statistics=%.3f, p=%.3f\n' % (stat, p))

print("Result: ", end=' ')
if p < 0.05:
     print('Different distributions (reject Null Hypothesis H0)')
else:
    print('Same distributions (fail to reject Null Hypothesis H0)')
```

```
Statistics=2583.941, p=0.000

Result:  Different distributions (reject Null Hypothesis H0)
```

5.2 2. Are the type of complaint or service requested and location related?

To find the correlation between location and complaint types, we will consider below columns

1. Complaint Type
2. Borough
3. Longitude
4. Latitude
5. City

[284]:
```
dataset['City'].isnull().sum()
```

[284]:
```
0
```

[285]:
```
dataset = dataset[dataset['City'].notnull()]
```

[286]:
```
dataset['City'].isnull().sum()
```

[286]:
```
0
```

[287]:
```
sample = dataset[['Complaint Type','Borough','Longitude','Latitude','City']]
```

[288]:
```
sample.head()
```

[288]:

	Complaint Type	Borough	Longitude	Latitude	City
0	Noise - Street/Sidewalk	MANHATTAN	-73.923501	40.865682	NEW YORK
1	Blocked Driveway	QUEENS	-73.915094	40.775945	ASTORIA
2	Blocked Driveway	BRONX	-73.888525	40.870325	BRONX
3	Illegal Parking	BRONX	-73.828379	40.835994	BRONX
4	Illegal Parking	QUEENS	-73.874170	40.733060	ELMHURST

converting the variables to categorical

[289]:
```
from sklearn.preprocessing import LabelEncoder
```

[290]:
```
le = LabelEncoder()
sample['City'] = le.fit_transform(sample['City'])
sample['Complaint Type'] = le.fit_transform(sample['Complaint Type'])
sample['Borough'] = le.fit_transform(sample['Borough'])
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:3:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  This is separate from the ipykernel package so we can avoid doing imports
until
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:4:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  after removing the cwd from sys.path.
```

[291]:
```
sample.head()
```

[291]:

	Complaint Type	Borough	Longitude	Latitude	City
0	16	2	-73.923501	40.865682	33
1	4	3	-73.915094	40.775945	1
2	4	0	-73.888525	40.870325	6
3	12	0	-73.828379	40.835994	6
4	12	3	-73.874170	40.733060	13

[292]:
```
sample.corr(method='pearson')
```

[292]:

	Complaint Type	Borough	Longitude	Latitude	City
Complaint Type	1.000000	-0.057730	-0.182274	0.149257	0.095026
Borough	-0.057730	1.000000	0.021738	-0.239928	0.689435
Longitude	-0.182274	0.021738	1.000000	0.364812	-0.123322
Latitude	0.149257	-0.239928	0.364812	1.000000	0.000602
City	0.095026	0.689435	-0.123322	0.000602	1.000000

Clearly there is no realtion between Location and complaint type.