**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)

1. Import a 311 NYC service request.
2. Read or convert the columns ‘Created Date’ and Closed Date’ to datetime datatype and create a new column ‘Request\_Closing\_Time’ as the time elapsed between request creation and request closing. (Hint: Explore the package/module datetime)
3. Provide major insights/patterns that you can offer in a visual format (graphs or tables); at least 4 major conclusions that you can come up with after generic data mining.
4. Order the complaint types based on the average ‘Request\_Closing\_Time’, grouping them for different locations.
5. 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?

**Solution for this problem statement**

1. ### import libraries

#for linear algebra

import numpy as np

# for data processing session

import pandas as pd

#for data visualization

import matplotlib.pyplot as plt

from matplotlib import style

#seaborn is also data visualization library built on top of the matplotlib

import seaborn as sns

#now we use matplotlib inline which is used for the output of plotting commands is displayed inline within frontends like jupitar notebook.

#Mainly used for inline plotting

%matplotlib inline

Graphical user interface, text, application

Description automatically generated

2) #Now import the csv file

Service\_Request\_csv = pd.read\_csv("311\_Service\_Requests\_from\_2010\_to\_Present.csv")

Graphical user interface

Description automatically generated with medium confidence

* 1. #now showing the first five rows of the service request csv

Service\_Request\_csv.head()

Text

Description automatically generated with low confidence

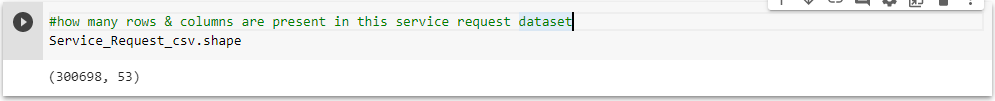
* 1. #now showing the last five rows of the service request csv

Service\_Request\_csv.tail()

A picture containing table

Description automatically generated

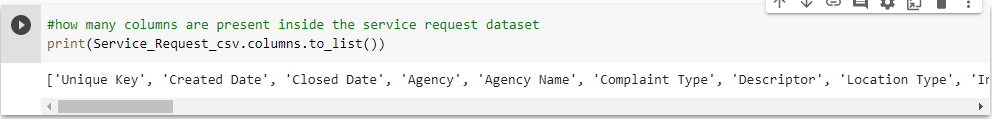
* 1. #how many rows & columns are present in this service request dataset

Service\_Request\_csv.shape

* 1. #how many columns are present inside the service request dataset

print(Service\_Request\_csv.columns.to\_list())

['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']



* 1. #how many unique columns are present inside service request dataset for the column name "Complaint Type"

#The unique() function is used to find the unique elements of an array

Service\_Request\_csv["Complaint Type"].unique()

Text

Description automatically generated

* 1. #how many unique columns are present inside service request dataset for the column name "Descriptor"

#The unique() function is used to find the unique elements of an array

Service\_Request\_csv["Descriptor"].unique()

array(['Loud Music/Party', 'No Access', 'Commercial Overnight Parking',

'Blocked Sidewalk', 'Posted Parking Sign Violation',

'Blocked Hydrant', 'With License Plate', 'Partial Access',

'Unauthorized Bus Layover', 'Double Parked Blocking Vehicle',

'Double Parked Blocking Traffic', 'Vehicle', 'Loud Talking',

'Banging/Pounding', 'Car/Truck Music', 'Tortured',

'In Prohibited Area', 'Congestion/Gridlock', 'Neglected',

'Car/Truck Horn', 'In Public', 'Other (complaint details)', nan,

'No Shelter', 'Truck Route Violation', 'Unlicensed',

'Overnight Commercial Storage', 'Engine Idling',

'After Hours - Licensed Est', 'Detached Trailer',

'Underage - Licensed Est', 'Chronic Stoplight Violation',

'Loud Television', 'Chained', 'Building', 'In Car',

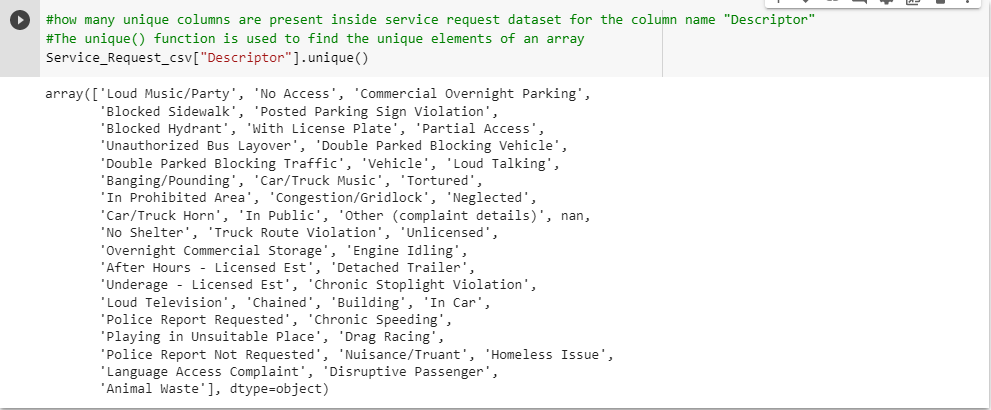
'Police Report Requested', 'Chronic Speeding',

'Playing in Unsuitable Place', 'Drag Racing',

'Police Report Not Requested', 'Nuisance/Truant', 'Homeless Issue',

'Language Access Complaint', 'Disruptive Passenger',

'Animal Waste'], dtype=object)



* 1. #how many missing values are present or not inside this dataset

#using isNa function

Service\_Request\_csv.isna().any()

Unique Key False

Created Date False

Closed Date True

Agency False

Agency Name False

Complaint Type False

Descriptor True

Location Type True

Incident Zip True

Incident Address True

Street Name True

Cross Street 1 True

Cross Street 2 True

Intersection Street 1 True

Intersection Street 2 True

Address Type True

City True

Landmark True

Facility Type True

Status False

Due Date True

Resolution Description False

Resolution Action Updated Date True

Community Board False

Borough False

X Coordinate (State Plane) True

Y Coordinate (State Plane) True

Park Facility Name False

Park Borough False

School Name False

School Number False

School Region True

School Code True

School Phone Number False

School Address False

School City False

School State False

School Zip True

School Not Found False

School or Citywide Complaint True

Vehicle Type True

Taxi Company Borough True

Taxi Pick Up Location True

Bridge Highway Name True

Bridge Highway Direction True

Road Ramp True

Bridge Highway Segment True

Garage Lot Name True

Ferry Direction True

Ferry Terminal Name True

Latitude True

Longitude True

Location True

dtype: bool

* 1. #total missing values are present inside this dataset

Service\_Request\_csv.isna().sum()

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

* 1. #information of the Service\_Request\_csv dataframe

Service\_Request\_csv.info()

Data columns (total 53 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Unique Key 300698 non-null int64

1 Created Date 300698 non-null object

2 Closed Date 298534 non-null object

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: float64(10), int64(1), object(42)

memory usage: 121.6+ MB

* 1. #check the datatype of the dataset

Service\_Request\_csv.dtypes

Unique Key int64

Created Date object

Closed Date object

Agency object

Agency Name object

Complaint Type object

Descriptor object

Location Type object

Incident Zip float64

Incident Address object

Street Name object

Cross Street 1 object

Cross Street 2 object

Intersection Street 1 object

Intersection Street 2 object

Address Type object

City object

Landmark object

Facility Type object

Status object

Due Date object

Resolution Description object

Resolution Action Updated Date object

Community Board object

Borough object

X Coordinate (State Plane) float64

Y Coordinate (State Plane) float64

Park Facility Name object

Park Borough object

School Name object

School Number object

School Region object

School Code object

School Phone Number object

School Address object

School City object

School State object

School Zip object

School Not Found object

School or Citywide Complaint float64

Vehicle Type float64

Taxi Company Borough float64

Taxi Pick Up Location float64

Bridge Highway Name object

Bridge Highway Direction object

Road Ramp object

Bridge Highway Segment object

Garage Lot Name float64

Ferry Direction object

Ferry Terminal Name object

Latitude float64

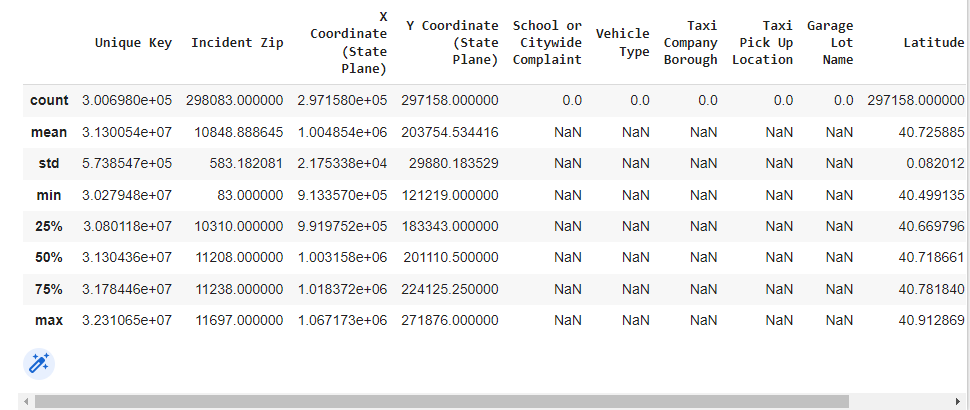
Longitude float64

Location object

dtype: object

* 1. #computes and displays summary statistics for a Service\_Request\_csv dataframe

Service\_Request\_csv.describe()



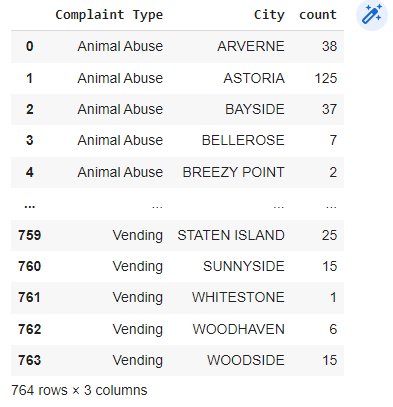
* 1. #create a custom dataframe with Complaint Type & City

customDataObject={'count':Service\_Request\_csv.groupby(['Complaint Type','City']).size()}

ComplaintTypeCity = pd.DataFrame(customDataObject).reset\_index()

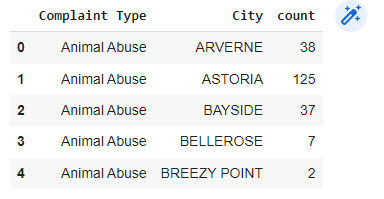
* 1. #show the custom complaint type city

ComplaintTypeCity



* 1. #just show first five rows from the dataset

ComplaintTypeCity.head()

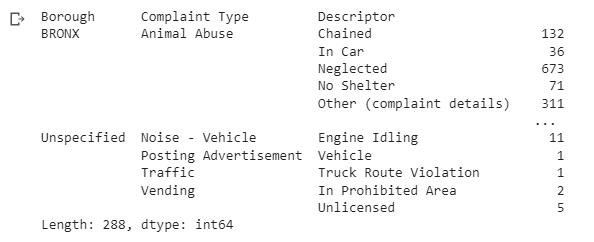


* 1. #get the individual column size for the column name Borough, Complaint Type & Descriptor

#get an array of column name

getColumnArray=["Borough","Complaint Type","Descriptor"]

Service\_Request\_csv.groupby(getColumnArray).size()



**Python Dates** A date in Python is not a data type of its own, but we can import a module named datetime to work with dates as date objects.

import datetime

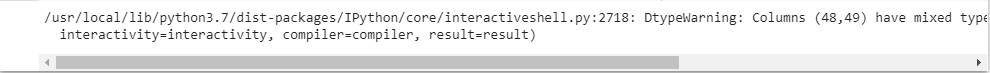
* 1. #Create a dataframe with parsed date

#If True and parse\_dates is enabled, pandas will attempt to infer the format of the datetime strings in the columns.

Service\_Request\_csv\_withParsedDate=pd.read\_csv(

    "311\_Service\_Requests\_from\_2010\_to\_Present.csv",

    parse\_dates=["Created Date","Closed Date"])



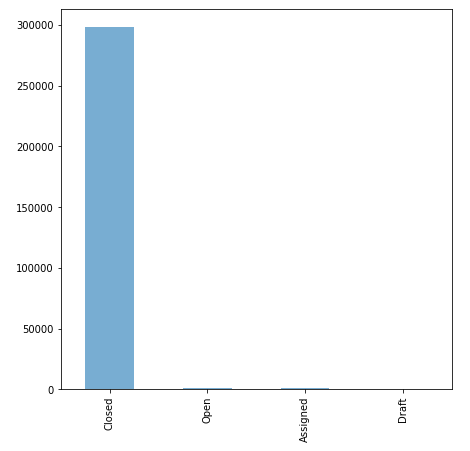
* 1. #calculate the Request Closing Time

Service\_Request\_csv\_withParsedDate["Request\_Closing\_Time"] = Service\_Request\_csv\_withParsedDate["Closed Date"] - Service\_Request\_csv\_withParsedDate["Created Date"]

* 1. #Visualize the status of the ticket

Service\_Request\_csv\_withParsedDate["Status"].value\_counts().plot(kind='bar',alpha=0.6,figsize=(7,7))

plt.show()

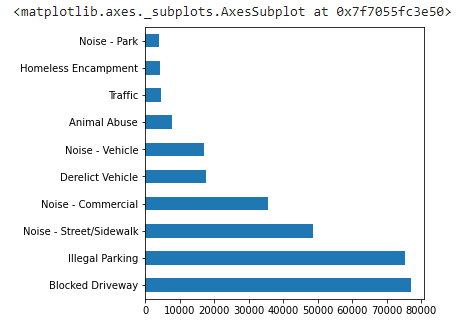


Matplotlib is a library in Python and it is numerical – mathematical extension for NumPy library.The figure module provides the top-level Artist, the Figure, which contains all the plot elements.This module is used to control the default spacing of the subplots and top level container for all plot elements.

* 1. #Complaint type Breakdown with bar plot to figure out majority of complaint types and top 10 complaints

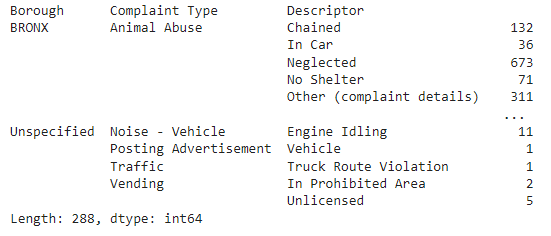
#Matplotlib is a library in Python and it is numerical – mathematical extension for NumPy library.

Service\_Request\_csv["Complaint Type"].value\_counts().head(10).plot(kind='barh',figsize=(5,5))



* 1. #column wise groupby & size

Service\_Request\_csv.groupby(["Borough","Complaint Type","Descriptor"]).size()



* 1. #calculate the major complaint type

#subset: It's an array which limits the dropping process to passed rows/columns through list.

majorComplaints = Service\_Request\_csv.dropna(subset=["Complaint Type"])

majorComplaints = majorComplaints.groupby("Complaint Type")

#Pandas sort\_values() function sorts a data frame in Ascending or Descending order of passed Column.

sortedComplaintType = majorComplaints.size().sort\_values(ascending=False)

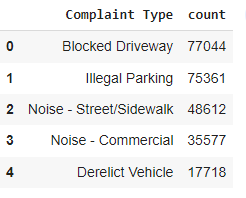
#Pandas reset\_index() is a method to reset index of a Data Frame. reset\_index()

#method sets a list of integer ranging from 0 to length of data as index. ... level

sortedComplaintType = sortedComplaintType.to\_frame('count').reset\_index()

* 1. #how many complaints are there, that list i have find out

sortedComplaintType.head()



* 1. #create a pie chart of this complaint type

sortedComplaintType = sortedComplaintType.head()

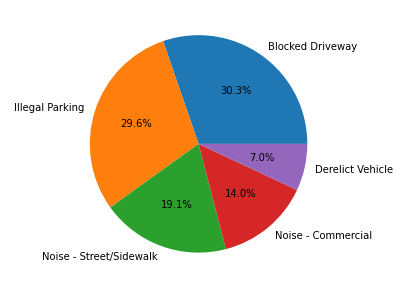
plt.figure(figsize=(5,5))

#We use autopct to display the percent value using Python string formatting.

#For example, autopct='%1.1f%%' means that for each pie wedge, the format string is '1.1f%'.

plt.pie(sortedComplaintType['count'], labels=sortedComplaintType['Complaint Type'], autopct="%1.1f%%")

plt.show()



* 1. #group dataset by complaint type to display plot against city

grouped\_by\_complaint\_type = Service\_Request\_csv.groupby('Complaint Type')

* 1. #groupeddata with Blocked Driverway column type

#get how many groups are present on Blocked Driveway

grp\_data = grouped\_by\_complaint\_type.get\_group("Blocked Driveway")

* 1. #get all column list

grp\_data.columns.to\_list()

['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']

* 1. #how many rows & columns are present for this Blocked Driveway dataset

grp\_data.shape



* 1. #to get the null values for this dataset

Service\_Request\_csv.isnull().sum()

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

* 1. #drop blank values for City column

Service\_Request\_csv["City"].dropna(inplace=True)

* 1. #check shape after dropping null values

Service\_Request\_csv["City"].shape



* 1. #count of null values in grouped city column data

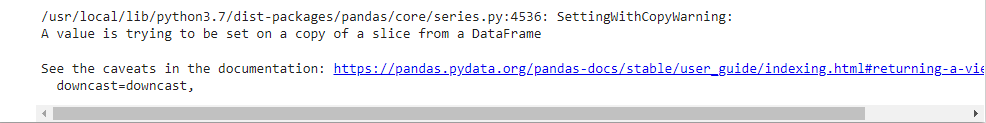
grp\_data["City"].isnull().sum()



* 1. #fix those nan values with "Unknown city" value instead

#The fillna() function is used to fill NA/NaN values using the specified method.

grp\_data["City"].fillna("Unknown City", inplace=True)



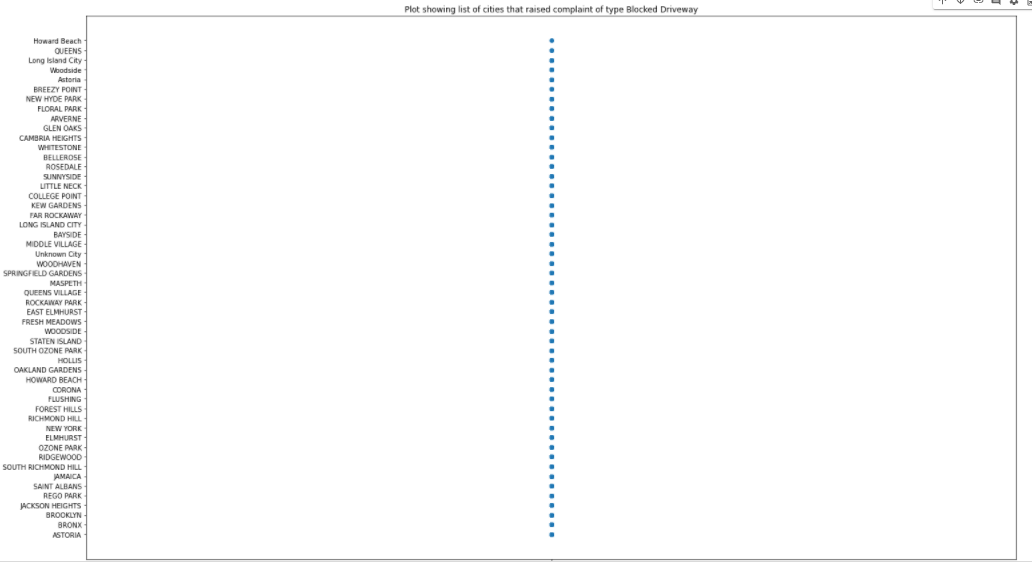
* 1. #Scatter plot displaying all the cities that raised complaint of type 'Blocked Driveway'

plt.figure(figsize=(25,15))

plt.scatter(grp\_data["Complaint Type"], grp\_data["City"])

plt.title("Plot showing list of cities that raised complaint of type Blocked Driveway")

plt.show()



* 1. #fix Location type those NAN with "unknown Location" value instead

Service\_Request\_csv["Location Type"].fillna("Unknown Loc",inplace=True)

* 1. #how many values are present for the column name Location Type

Service\_Request\_csv["Location Type"].values

array(['Street/Sidewalk', 'Street/Sidewalk', 'Street/Sidewalk', ...,

'Club/Bar/Restaurant', 'Club/Bar/Restaurant', 'Store/Commercial'],

dtype=object)

* 1. #Find top 10 major complaint types and their counts

grouped\_by\_complaint\_type["Complaint Type"].value\_counts().nlargest(10)

Text

Description automatically generated

* 1. #fix Location type those NAN with "unknown Location" value instead

Service\_Request\_csv["Location Type"].fillna("Unknown Loc", inplace=True)

Service\_Request\_csv['Location Type'].values

array(['Street/Sidewalk', 'Street/Sidewalk', 'Street/Sidewalk', ...,

'Club/Bar/Restaurant', 'Club/Bar/Restaurant', 'Store/Commercial'],

dtype=object)

* 1. #count of null values in grouped location type column data

grp\_data['Location Type'].isnull().sum()

