LONDON METROPOLITAN UNIVERSITY

Islington college
(इस्लिङ्टन कलेज)

Module Code & Module Title

CC5067NI Smart Data Discovery

60% Individual Coursework

Submission : Final Submission

Academic Semester: Spring Semester 2025

Credit: 15 credit semester long module

Student Name: Shreya Shah

London Met ID: 23050326

College ID: NP01CP4A230346

Assignment Due Date: Thursday, May 15, 2025

Assignment Submission Date: Thursday, May 15, 2025

Submitted To: Mr. Roshan Shrestha

I confirm that I understand my coursework needs to be submitted online via MST Classroom under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded.

23050326 ShreyaShah.docx

Islington College,Nepal

Document Details

Submission ID
trn:oid:::3618:95910155

Submission Date
May 15, 2025, 12:30 AM GMT+5:45

Download Date
May 15, 2025, 12:31 AM GMT+5:45

File Name
23050326 ShreyaShah.docx

File Size
33.7 KB

33 Pages

5,423 Words

28,680 Characters

14% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Match Groups

- **59** Not Cited or Quoted 14%
  Matches with neither in-text citation nor quotation marks
- **2** Missing Quotations 0%
  Matches that are still very similar to source material
- **0** Missing Citation 0%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

9% Internet sources
4% Publications
13% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you

Table of Contents

Tables of Figures

Table of Tables

Introduction

The coursework on Customer Service Requests Analysis will apply Python programming using data from NYC 311 service requests. The dataset holds particulars about multiple factors affecting customer requests types, location, response times and many more. The main aim of this coursework will be to explore and understand the data and structure it for analysis as well as data mining. We will deal with missing values, remove unnecessary columns, create new variables for important information like time between request generation and request closure, etc.

You will also have to carry out some data wrangling techniques to get useful insights from the data. The specific data wrangling techniques that are necessary for the analysis are converting the date columns in the correct datetime format as well as creating new calculated columns. Further, the unimportant data points must be removed. These steps will ensure that the data is organized appropriately to allow effective analysis and visualization. Getting the data ready like this will help us expose patterns and trends that you might not notice right away. Also, we will do exploratory data analysis (EDA) in Python where we will create summary statistics and visualize trends, such as which complaint types are reported more often and what is the average response time based on location.

We will do statistical testing to test hypothesis and understand relationships between main variables. For example, we will check if any significant difference exists between average response times based on complaint type and if complaint type depends on request location using statistical tests. By doing these tests, we can find out what causes the service response times. Our findings will be supported by a robust technical report of the entire process. This will include the python code, visualizations and the insights drawn from the data. This will give a full picture of the customer service request patterns and help with possible improvements in the service management process. **(Elfman, 2025)**

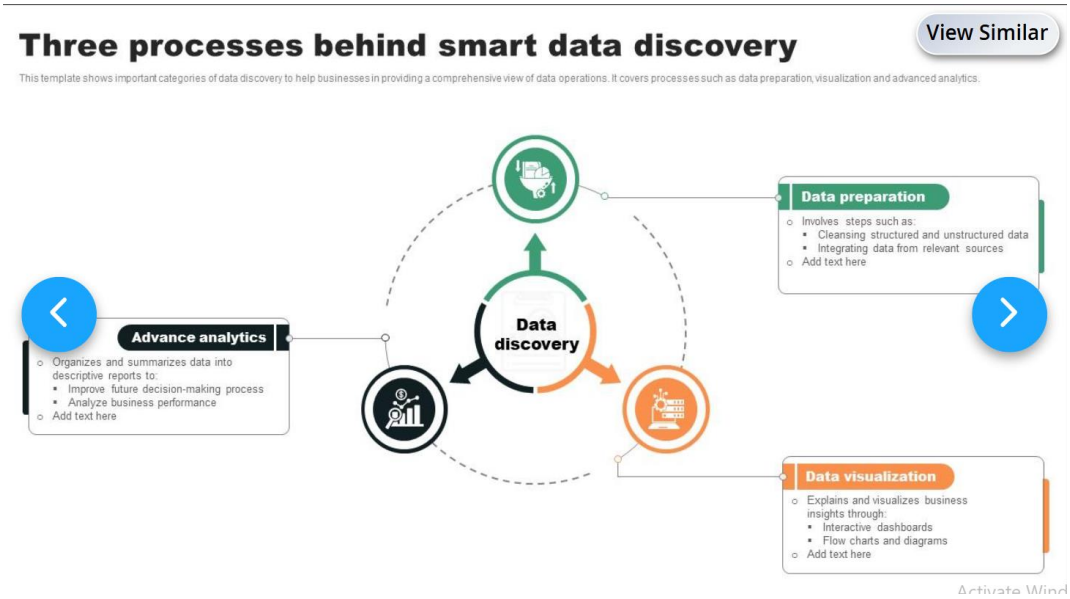

Figure 1 Smart Data Discovery Processes

i. Data Understanding

1. Overview of the Dataset.

This dataset has the customer service requests made to 311, a service of New York City. It keeps track of what kinds of complaints people made, where those complaints came from (by location) and when the issues were reported and resolved. The aim of this dataset is to look for common customer complaints and trust patterns from data that will help in fixing services of the city. (Morris, 2025)

2. Data Understanding Importance:

- It's the Starting Point of Everything

  Before cleaning or analyzing your data, it's important to be familiar with it first. What do the columns mean? Are there missing values? Is the data even useful? Without this step, everything else can go wrong
- You Find Out What Matters

  Not every piece of data is useful, When you study the data, you learn what is helpful and what is not This makes your work faster, smarter, and more focused
- You Catch Problems Early

  At other times, the data is dirty—it may have missing values or something looks off. When we understand our data. We can notice these problems early and not let them ruin our results later.
- It Keeps You From Making Wrong Assumptions

  What does score mean? Is it out of ten or a hundred? Is a low score good or bad? Being cognizant of the details will keep you from making wrong assumptions.
- It Makes Your Final Analysis Better

  When you are predicting or looking for patterns (for example with machine learning), understanding helps you select the right ones. This can make your results way more accurate

- It Connects Data to Real Life
  Data isn't just numbers on a screen—it tells a story Learning the data helps you translate those amounts into real life; consumer behavior, business performance, etc. (Morris, 2025)

3. **Structure of the Data**.

The data includes several types of information.

i. Text Data.

- One of the columns in the data set is "Complaint Type". This tells us what the complaint was about. For instance, there were complaints regarding sanitation, noise, etc. Apart from these, some people complained about broken streetlights.
- This column of the data tells us where the complaint was location wise which includes the boroughs and the neighborhoods.
- Resolution Description explains how that issue was resolved in brief.

ii. Datetime Data.

- Created Date: When the 311 complaint was reported.
- Closed Date: When the issues was fixed or closed.

iii. Numeric Data.

- This is a new field which we will calculate by measuring how long it took to resolve each complaint from the time it was created.

To get an idea of the structure and type of a table and what the columns store we will be using the functions .info() and .head().

4. Column Descriptions.

- All complaints relate to a particular type of complaint for example, a complaint regarding sanitation, streetlight complaint, noise complaint, etc.
- It helps us know the borough or neighborhood from where the complaint came from.

- The Created Date is when they made the complaint.
- The acknowledgment date and time of a complaint.
- The time period between complaint being registered and complaint being resolved will get captured in a new column called as Request closing time.

5. Checking Data Quality.

- We will format the Created Date and Closed Date columns to ensure the date and time formats are accurate. These columns should be in a datetime format, not text.
- We are going to check for null values by using .isnull(). If we find them, we'll find the best way to treat them, by filling those in or taking them away.
- We will also ensure that the data does not have or was not given any invalid values such as negative time or duplicate values. This will help ensure the data is clean and reliable.

6. Initial Observations.

- Analyzing the dataset quickly can help determine the nature of the complaints and the times when these complaints are usually filed or resolved. We might learn that specific complaint types happen during certain seasons or certain areas file complaints more often than others.

7. Methods for Exploring the Data.

To Understand the Data, We Will Use a Set of Functions

- info(): Let's take a look at the dataset using a few functions to get a better grasp of them. To get the data types and how much missing data there is in the dataset, we will use .info().
- head(): Using .head() we get to see the first few rows of the dataset to check for quick fixes.
- describe(): We can make use of .describe() to check out stats like average, maximum, minimum and so on of numeric columns.
- isnull(): This command will show the missing values in the dataset.
- Sample(): gives a single row selected randomly.

8. Preliminary Data Cleaning.

We will clean up the dataset before proceeding with more in-depth analyses. We might remove unneeded columns, say, Agency Name or Street Name and make sure the date fields are in the right format. We will also fix missing values in the dataset so it is ready for further analysis (Morris, 2025).

S.No	Column Name	Description	Data Type
1	Unique Key	Unique ID for each service request	Integer
2	Created Date	Date the request was submitted	DateTime
3	Closed Date	Date the request was resolved	DateTime
4	Agency	Responding agency, here always NYPD	Object
5	Agency	Name Full name of agency, here “New York Police Department”	Object
6	Complaint Type	General type of complaint	Object

7	Descriptor	Specific detail of complaint	Object
8	Location Type	Type of area where the issue occurred	Object
9	Incident Zip	ZIP Code of the incident	Integer
10	Incident Address	Street address of the complaint	Object
11	Street Name	Name of the street	Object
12	Cross Street 1	Closest cross street	Object
13	Cross Street 2	Second closest cross street	Object
14	Intersection Street 1	Primary intersecting street	Object
15	Intersection Street 2	Secondary intersecting street	Object
16	Address Type	Indicates “Address” or “Intersection”	Object

17	City	City in which the complaint was made	Object
18	Landmark	Nearby known landmark	Object
19	Facility Type	Type of responding unit (e.g., Precinct)	Object
20	Status	Current status of the case (Closed, Open)	Object
21	Due Date	Date by which action was due	DateTime
22	Resolution Action Updated Date	Date the resolution information was last updated	DateTime
23	Resolution Description	Who handled the case (e.g., The Police)	Object
24	Resolution Date	Final resolution date	DateTime

25	Community Board	Community board number	Integer
26	Borough	Borough in which complaint occurred	Object
27	X-Coordinate (State Plane)	NYC-specific X-coordinate for mapping	Float
28	Y-Coordinate (State Plane)	NYC-specific Y-coordinate for mapping	Float
29	Park Facility Name	If in a park, name of the park facility	Object
30	Park Borough	Borough of the park facility	Object
31	School Name	Name of the school involved	Object
32	School Number	Number of the school involved	Object
33	School Region	Education region of the school	Object

34	School Code	DOE-assigned code for the school	Object
35	School Phone	Number Contact number of the school	Object
36	School Address	Full address of the school	Object
37	School City	City of the school	Object
38	School State	State of the school	Object
39	School Zip	ZIP Code of the school	Integer
40	School Not Found	Indicator if school wasn't found in DOE system	Boolean
41	School or Citywide Complaint	Indicator for school-related or citywide issue	Object
42	Vehicle Type	Type of vehicle involved (if any)	Object

43	Taxi Company Borough	Borough of the involved taxi company	Object
44	Taxi Pick Up Location	Location where taxi picked up passenger	Object
45	Bridge Highway Name	Name of bridge/highway involved	Object
46	Bridge Highway Direction	Direction of bridge/highway	Object
47	Road Ramp	Name of ramp involved	Object
48	Bridge Highway Segment	Specific highway segment	Object
49	Garage Lot Name	Name of involved garage or lot	Object
50	Ferry Direction	Direction of ferry involved	Object

51	Ferry Terminal Name	Name of ferry terminal involved	Object
52	Location	Latitude and longitude in string format	Object
53	Latitude	Geographic latitude coordinate	Float

Table 1 Data Understanding Table

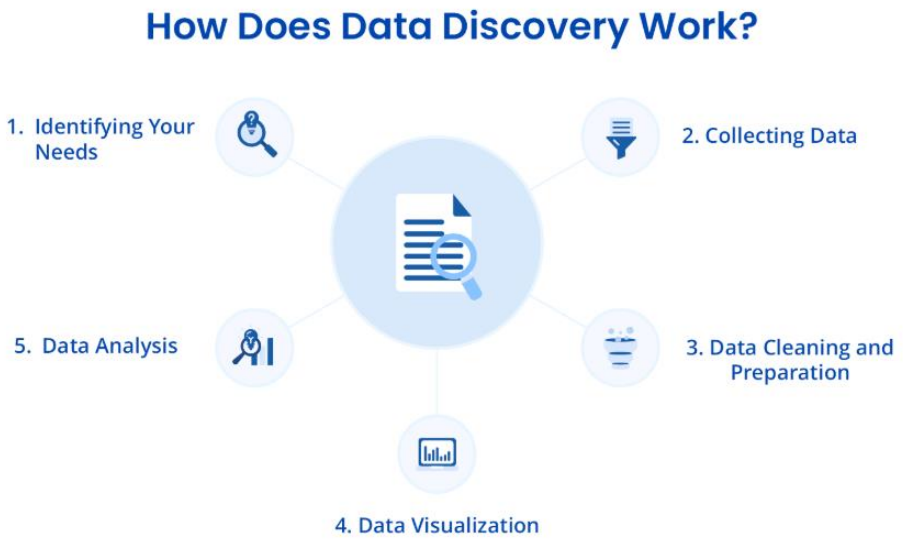

Figure 2 How Data Discovery Work?

ii. Data Preparation

Data preparation is the refinement of raw data to get it ready for analysis and processing. It is also known as data prep. When raw data has errors, duplicates, and missing values, it affects data quality and data-driven decisions. Data preparation is essential as it can use up to 80% of the time in a machine learning project. It is essential to use special tools for data preparation to do this.

According to findings released by Anaconda and Forbes, data scientists use 45 to 60 percent of their time to collect, organize and prepare data and data cleaning takes more than a quarter of the time of their day. When data scientists do this, they lose precious time that should go to core activity such as selecting, training, and deploying the model. So, such a practice raises doubts on whether we need our data scientists to do digital janitorial work. (Anon., 2025)

i. Import the dataset

The dataset gets loaded using the pandas library which is a python data manipulation library.

Pandas is a DataFrame and Series-focused library in Python utilized for data manipulation and analysis. It enables working with data in a more organized and clear form.

```
[ ]: import pandas as pd

30]: data = pd.read_csv('Customer Service_Requests_from_2010_to_Present.csv') # importing dataset
     data
```

```
C:\Users\Dell\AppData\Local\Temp\ipykernel_16020\2519893982.py:1: DtypeWarning: Columns (48,49) have mixed types. Specify dtype option on import or set l
ow_memory=False.
  data = pd.read_csv('Customer Service_Requests_from_2010_to_Present.csv') # importing dataset
```

30]:

	Unique Key	Created Date	Closed Date	Agency	Agency Name	Complaint Type	Descriptor	Location Type	Incident Zip	Incident Address	...	Bridge Highway Name	Bridge Highway Direction	Road Ramp	I Hig Se
0	32310363	12/31/2015 23:59	1/1/2016 0:55	NYPD	New York City Police Department	Noise - Street/Sidewalk	Loud Music/Party	Street/Sidewalk	10034.0	71 VERMILYEA AVENUE	...	NaN	NaN	NaN	
1	32309934	12/31/2015 23:59	1/1/2016 1:26	NYPD	New York City Police Department	Blocked Driveway	No Access	Street/Sidewalk	11105.0	27-07 23 AVENUE	...	NaN	NaN	NaN	
2	32309159	12/31/2015 23:59	1/1/2016 4:51	NYPD	New York City Police Department	Blocked Driveway	No Access	Street/Sidewalk	10458.0	2897 VALENTINE AVENUE	...	NaN	NaN	NaN	
3	32305098	12/31/2015 23:57	1/1/2016 7:43	NYPD	New York City Police Department	Illegal Parking	Commercial Overnight Parking	Street/Sidewalk	10461.0	2940 BAISLEY AVENUE	...	NaN	NaN	NaN	
4	32306529	12/31/2015 23:56	1/1/2016 3:24	NYPD	New York City Police Department	Illegal Parking	Blocked Sidewalk	Street/Sidewalk	11373.0	87-14 57 ROAD	...	NaN	NaN	NaN	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
		3/29/2015			New York	Noise -	Loud			CRESCENT					

Figure 3 Importing the dataset

Location Type	Incident Zip	Incident Address	...	Bridge Highway Name	Bridge Highway Direction	Road Ramp	Bridge Highway Segment	Garage Lot Name	Ferry Direction	Ferry Terminal Name	Latitude	Longitude	Location
Street/Sidewalk	10034.0	71 VERMILYEA AVENUE	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	40.865682	-73.923501	(40.86568153633767, -73.92350095571744)
Street/Sidewalk	11105.0	27-07 23 AVENUE	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	40.775945	-73.915094	(40.775945312321085, -73.91509393898605)
Street/Sidewalk	10458.0	2897 VALENTINE AVENUE	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	40.870325	-73.888525	(40.870324522111424, -73.88852464418646)
Street/Sidewalk	10461.0	2940 BAISLEY AVENUE	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	40.835994	-73.828379	(40.83599404683083, -73.82837939584206)
Street/Sidewalk	11373.0	87-14 57 ROAD	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	40.733060	-73.874170	(40.733059618956815, -73.87416975810375)
...	...	...	...	...	...	...	...	...	...	...	...	...	...
lub/Bar/Restaurant	NaN	CRESCENT AVENUE	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Street/Sidewalk	11418.0	100-17 87 AVENUE	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	40.694077	-73.846087	(40.69407728322387, -73.8460866160573)
lub/Bar/Restaurant	11206.0	162 THROOP AVENUE	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	40.699590	-73.944234	(40.69959035300927, -73.94423377144169)
		3151 EAST											

Figure 4 Output for Importing Data

The dataset was successfully imported using `pandas`. The dataset is about customer service request records which came from NYC 311. The data variable contains the full dataset in table format.

Pandas warned that the dataset had columns of mixed data types when it was read. This occurs as a result of Pandas attempting to infer the type of data—such as dates, text, or numbers that each column contains. However, there is a wide range of value kinds in a column, such as text in some rows and integers in others. It becomes unclear. Pandas reads the big file in segments to conserve memory, which may result in inconsistent type recognition. As a result, it is informing the user that columns (in this case, columns 48 and 49) contain mixed types, which could cause issues with analysis in the future. Although this is not an error, it is a warning that, in order to prevent problems, data may need to be cleansed or data types may need to be manually specified.

```
data = pd.read_csv('Customer Service_Requests_from_2010_to_Present.csv', low_memory=False) # importing dataset without warning
data
```

	Unique Key	Created Date	Closed Date	Agency	Agency Name	Complaint Type	Descriptor	Location Type	Incident Zip	Incident Address	...	Bridge Highway Name	Bridge Highway Direction	Road Ramp	I Hig Se
0	32310363	12/31/2015 23:59	1/1/2016 0:55	NYPD	New York City Police Department	Noise - Street/Sidewalk	Loud Music/Party	Street/Sidewalk	10034.0	71 VERMILYEA AVENUE	...	NaN	NaN	NaN	
1	32309934	12/31/2015 23:59	1/1/2016 1:26	NYPD	New York City Police Department	Blocked Driveway	No Access	Street/Sidewalk	11105.0	27-07 23 AVENUE	...	NaN	NaN	NaN	
2	32309159	12/31/2015 23:59	1/1/2016 4:51	NYPD	New York City Police Department	Blocked Driveway	No Access	Street/Sidewalk	10458.0	2897 VALENTINE AVENUE	...	NaN	NaN	NaN	
3	32305098	12/31/2015 23:57	1/1/2016 7:43	NYPD	New York City Police Department	Illegal Parking	Commercial Overnight Parking	Street/Sidewalk	10461.0	2940 BAISLEY AVENUE	...	NaN	NaN	NaN	
4	32306529	12/31/2015 23:56	1/1/2016 3:24	NYPD	New York City Police Department	Illegal Parking	Blocked Sidewalk	Street/Sidewalk	11373.0	87-14 57 ROAD	...	NaN	NaN	NaN	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
300693	30281872	3/29/2015 0:33	NaN	NYPD	New York City Police Department	Noise - Commercial	Loud Music/Party	Club/Bar/Restaurant	NaN	CRESCENT AVENUE	...	NaN	NaN	NaN	
300694	30281230	3/29/2015 0:33	3/29/2015 2:33	NYPD	New York City Police Department	Blocked Driveway	Partial Access	Street/Sidewalk	11418.0	100-17 87 AVENUE	...	NaN	NaN	NaN	

Figure 5 Removing Warning

300693	30281872	3/29/2015 0:33	NaN	NYPD	New York City Police Department	Noise - Commercial	Loud Music/Party	Club/Bar/Restaurant	NaN	CRESCENT AVENUE	...	NaN	NaN	NaN
300694	30281230	3/29/2015 0:33	3/29/2015 2:33	NYPD	New York City Police Department	Blocked Driveway	Partial Access	Street/Sidewalk	11418.0	100-17 87 AVENUE	...	NaN	NaN	NaN
300695	30283424	3/29/2015 0:33	3/29/2015 3:40	NYPD	New York City Police Department	Noise - Commercial	Loud Music/Party	Club/Bar/Restaurant	11206.0	162 THROOP AVENUE	...	NaN	NaN	NaN
300696	30280004	3/29/2015 0:33	3/29/2015 4:38	NYPD	New York City Police Department	Noise - Commercial	Loud Music/Party	Club/Bar/Restaurant	10461.0	3151 EAST TREMONT AVENUE	...	NaN	NaN	NaN
300697	30281825	3/29/2015 0:33	3/29/2015 4:41	NYPD	New York City Police Department	Noise - Commercial	Loud Music/Party	Store/Commercial	10036.0	251 WEST 48 STREET	...	NaN	NaN	NaN

300698 rows × 53 columns

I have set the low_memory = False parameter before opening the CSV file to prevent any problems caused by this message. This enables Pandas to make better decisions by first reading the full file before speculating on the data types of each column. Otherwise, I could have used the dtype parameter to manually specify the data types of the columns that were causing this problem if they were significant. This stage ensures that there is no unexpected activity and that the data is loaded accurately and prepared for analysis.

ii. Provide your insight on the information and details that the provided dataset carries.

The NYC 311 Service Request dataset contains detailed records of service complaints made by customers in New York City. After using python some insights were found that are as follows.

1. **info().**

This function provides a summary of the data including its column names, types and null values. It's useful to check for missing data and understand the structure of the dataset.

```
print("*** Dataset Info ***")
print(data.info())
```

```
*** 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  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
```

Figure 6 Using info().function

2. head().

Shows the first five rows of the dataset This lets us quickly get a sense of what the data is like and any major issues.

```
print("----------Give First five Rows ---------------")
print(data.head())
```

```
----------Give First five Rows ---------------
   Unique Key      Created Date    Closed Date Agency  \
0    32310363  12/31/2015 23:59  1/1/2016 0:55   NYPD
1    32309934  12/31/2015 23:59  1/1/2016 1:26   NYPD
2    32309159  12/31/2015 23:59  1/1/2016 4:51   NYPD
3    32305098  12/31/2015 23:57  1/1/2016 7:43   NYPD
4    32306529  12/31/2015 23:56  1/1/2016 3:24   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
```

Figure 7 Using head().function

3. describe().

Gives descriptions (mean, median, min, max, etc.) of numeric columns. It's helpful for identifying trends, ranges, and outliers.

```
print("-------Gives Statistical Summary ------------")
print(data.describe())
```

```
-------Gives Statistical Summary ------------
         Unique Key   Incident Zip  X Coordinate (State Plane)  \
count  3.006980e+05  298083.000000                2.971580e+05
mean   3.130054e+07   10848.888645                1.004854e+06
std    5.738547e+05     583.182081                2.175338e+04
min    3.027948e+07      83.000000                9.133570e+05
25%    3.080118e+07   10310.000000                9.919752e+05
50%    3.130436e+07   11208.000000                1.003158e+06
75%    3.178446e+07   11238.000000                1.018372e+06
max    3.231065e+07   11697.000000                1.067173e+06

       Y Coordinate (State Plane)  School or Citywide Complaint  Vehicle Type  \
count                297158.000000                           0.0           0.0
mean                 203754.534416                           NaN           NaN
std                   29880.183529                           NaN           NaN
min                  121219.000000                           NaN           NaN
25%                  183343.000000                           NaN           NaN
50%                  201110.500000                           NaN           NaN
75%                  224125.250000                           NaN           NaN
max                  271876.000000                           NaN           NaN

       Taxi Company Borough  Taxi Pick Up Location  Garage Lot Name  \
count                   0.0                    0.0              0.0
mean                    NaN                    NaN              NaN
std                     NaN                    NaN              NaN
min                     NaN                    NaN              NaN
25%                     NaN                    NaN              NaN
50%                     NaN                    NaN              NaN
75%                     NaN                    NaN              NaN
max                     NaN                    NaN              NaN

           Latitude      Longitude
count  297158.000000  297158.000000
mean       40.725885     -73.925630
```

Figure 8 Using describe().function

4. isnull().sum().

Shows the number of missing (NaN) values in each column It helps us decide which columns need cleaning or imputing.

```
print("=== Missing Values ===")
print(data.isnull().sum())
```

```
=== Missing Values ===
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
```

Figure 9 Using isnull().sum() function

5. sample().

Returns a random row from the dataset Helpful for manually verifying a single entry and understanding how the data appears in real-world cases.

```
print("=== Random Sample ===")
print(data.sample(1))
```

```
=== Random Sample ===
        Unique Key   Created Date    Closed Date Agency  \
196815    30990534  7/3/2015 1:34  7/3/2015 5:26   NYPD

                          Agency Name   Complaint Type  \
196815  New York City Police Department  Illegal Parking

                         Descriptor    Location Type  Incident Zip  \
196815  Posted Parking Sign Violation  Street/Sidewalk       10065.0

          Incident Address  ... Bridge Highway Name Bridge Highway Direction  \
196815  205 EAST 64 STREET  ...                 NaN                       NaN

       Road Ramp Bridge Highway Segment Garage Lot Name Ferry Direction  \
196815       NaN                   NaN             NaN             NaN

       Ferry Terminal Name  Latitude  Longitude  \
196815                 NaN  40.76452 -73.963893

                                     Location
196815  (40.76451988224334, -73.96389332238066)

[1 rows x 53 columns]
```

Figure 10 Using sample().function

iii. 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

```
data['Created Date'] = pd.to_datetime(data['Created Date'])
data['Closed Date'] = pd.to_datetime(data['Closed Date'])
data['Request_Closing_Time'] = data['Closed Date'] - data['Created Date']
data
```

	Unique Key	Created Date	Closed Date	Agency	Agency Name	Complaint Type	Descriptor	Location Type	Incident Zip	Incident Address	...
0	32310363	2015-12-31 23:59:00	2016-01-01 00:55:00	NYPD	New York City Police Department	Noise - Street/Sidewalk	Loud Music/Party	Street/Sidewalk	10034.0	71 VERMILYEA AVENUE	...
1	32309934	2015-12-31 23:59:00	2016-01-01 01:26:00	NYPD	New York City Police Department	Blocked Driveway	No Access	Street/Sidewalk	11105.0	27-07 23 AVENUE	...
2	32309159	2015-12-31 23:59:00	2016-01-01 04:51:00	NYPD	New York City Police Department	Blocked Driveway	No Access	Street/Sidewalk	10458.0	2897 VALENTINE AVENUE	...
3	32305098	2015-12-31 23:57:00	2016-01-01 07:43:00	NYPD	New York City Police Department	Illegal Parking	Commercial Overnight Parking	Street/Sidewalk	10461.0	2940 BAISLEY AVENUE	...
4	32306529	2015-12-31 23:56:00	2016-01-01 03:24:00	NYPD	New York City Police Department	Illegal Parking	Blocked Sidewalk	Street/Sidewalk	11373.0	87-14 57 ROAD	...
...	...	...	...	...	...	...	...	...	...	...	...
300693	30281872	2015-03-29 00:33:00	NaT	NYPD	New York City Police Department	Noise - Commercial	Loud Music/Party	Club/Bar/Restaurant	NaN	CRESCENT AVENUE	...
300694	30281230	2015-03-29	2015-03-29	NYPD	New York City Police	Blocked Driveway	Partial Access	Street/Sidewalk	11418.0	100-17 87 AVENUE	...

Figure 11 Converting the columns "Created Date" and "Closed Date" to datetime and creating new column

Location	Request_Closing_Time
(40.86568153633767, -73.92350095571744)	0 days 00:56:00
(40.775945312321085, -73.91509393898605)	0 days 01:27:00
(40.870324522111424, -73.88852464418646)	0 days 04:52:00
(40.83599404683083, -73.82837939584206)	0 days 07:46:00
(40.733059618956815, -73.87416975810375)	0 days 03:28:00
...	...
NaN	NaT

Figure 12 Output for Converting the columns "Created Date" and "Closed Date" to datetime and creating new column

We first convert the 'Created Date' and 'Closed Date' columns to proper datetime format using pd.to_datetime() to understand how much time was taken to resolve each customer request. This will ensure that Python will handle and calculate the difference in dates correctly, Once converted, we take the difference between the Created Date and Closed Date and create a new column Request_Closing_Time. This new column show the total time taken to close each request which helps in analyzing the efficiency and speed of service response.

iv. Write a python program to drop irrelevant Columns which are listed below.

['Agency Name','Incident Address','Street Name','Cross Street 1','Cross Street 2','Intersection Street 1', 'Intersection Street 2','Address Type','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','Landmark',

'X Coordinate (State Plane)','Y Coordinate (State Plane)','Due Date','Resolution Action Updated Date','Community Board','Facility Type',

'Location']

```
]: columns_to_drop = [
'Agency Name','Incident Address','Street Name','Cross Street 1','Cross Street 2',
'Intersection Street 1','Intersection Street 2','Address Type','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',
'Landmark','X Coordinate (State Plane)','Y Coordinate (State Plane)','Due Date',
'Resolution Action Updated Date','Community Board','Facility Type','Location'
]

data = data.drop(columns=[col for col in columns_to_drop if col in data.columns])

# Confirm what columns remain

print(data.columns.tolist())

['Unique Key', 'Created Date', 'Closed Date', 'Agency', 'Complaint Type', 'Descriptor', 'Location Type', 'Incident Zip', 'City', 'Status', 'Resolution De
scription', 'Borough', 'Taxi Pick Up Location', 'Latitude', 'Longitude', 'Request_Closing_Time', 'Request_Closing_Time_sec']
```

Figure 13 Droping irrelevant Columns

This method is used to clean the dataset by removing irrelevant columns that are unnecessary for the analysis. It defines a list of column names to be removed and filters it to include only those columns that exist in the actual dataset to prevent errors. The `drop()` method is then called to remove these columns, and the remaining columns are

printed for validation of cleanup. This technique is designed to streamline the dataset, making it easier to analyze and visualize the most important information.

v. Write a python program to remove the NaN missing values from updated dataframe.

At first, we check nan values in our dataset. Then, we will identify them and drop them.

```
# To remove the NaN missing values from updated dataframe.
#step 1:
print("Missing values in each column:\n", data.isnull().sum())
```

```
Missing values in each column:
 Unique Key                      0
Created Date                     0
Closed Date                      0
Agency                           0
Complaint Type                   0
Descriptor                    5909
Location Type                  128
Incident Zip                   507
City                           506
Status                           0
Resolution Description           0
Borough                          0
Taxi Pick Up Location       298534
Latitude                      1432
Longitude                     1432
Request_Closing_Time             0
Request_Closing_Time_sec         0
dtype: int64
```

Figure 14 Checking missing values

```
# To remove the NaN missing values from updated dataframe.
# Step 2: Drop rows with any missing (NaN) values
df_cleaned = data.dropna()
print("Missing values after cleaning:\n")
print(df_cleaned)
```

```
Missing values after cleaning:

Empty DataFrame
Columns: [Unique Key, Created Date, Closed Date, Agency, Complaint Type, Descriptor, Location Type, Incident Zip, City, Status, Resolution Description, B
orough, Taxi Pick Up Location, Latitude, Longitude, Request_Closing_Time, Request_Closing_Time_sec]
Index: []
```

Figure 15 Droping nan values

We will drop all the rows of the dataset that have missing values (NaN). We employ a method called dropna to get rid of all the rows with NaN. It is important to handle missing values as they can result in incorrect analysis or error in data processing. Cleaning the

dataset allows only complete records to be taken for analysis without inaccuracies or errors in the results. After applying this function, the dataset will become more consistent and will be ready for exploration and modeling.

vi. Write a python program to see the unique values from all the columns in the dataframe.

```
data.nunique()
```

```
Unique Key                 298534
Created Date               197454
Closed Date                169123
Agency                          1
Complaint Type                 23
Descriptor                     43
Location Type                  16
Incident Zip                  201
City                           53
Status                          4
Resolution Description         14
Borough                         6
Taxi Pick Up Location           0
Latitude                   125115
Longitude                  125209
Request_Closing_Time         3153
Request_Closing_Time_sec     3153
dtype: int64
```

Figure 16 program to see the unique values

The command data.nunique() in Python is used to determine unique values present in each column of our data frame data. A fast overview of diversity in different features in the dataset. If a column has one unique value, it may not provide much value and can be removed. Columns with many unique values on the contrary shows high variability of identifiers, date, location coordinates and so on. This function is useful in the data understanding and exploration phase. It helps the analysts identify categorical features. Further, it helps in data distribution and columns with low or high unique values.

iii. Data Analysis

With data analysis, you can use your existing data to easily see how decisions are influencing your business. IT offers you a tool to drive the ship in the right direction by using facts you have gathered and presenting them in a thoughtful way. Thus, this can lead to making a decision that is backed by facts and not a fuzzy feeling of yours or rolling a dice and crossing your fingers it was the right decision to make. (hightouch, 2025)

i. Write a Python program to show summary statistics of sum, mean, standard deviation, skewness, and kurtosis of the data frame.

```
summary = {
    'Sum': data.sum(numeric_only=True),
    'Mean': data.mean(numeric_only=True),
    'Standard Deviation': data.std(numeric_only=True),
    'Skewness': data.skew(numeric_only=True),
    'Kurtosis': data.kurt(numeric_only=True)
}

summary_data = pd.DataFrame(summary)
print(summary_data)
```

Figure 17 show summary statistics of sum, mean, standard deviation, skewness, and kurtosis of the data frame.

```
                                Sum          Mean  Standard Deviation  \
Unique Key                9.344373e+12  3.130087e+07       574083.694222
Incident Zip              3.233274e+09  1.084893e+04          583.173185
Taxi Pick Up Location     0.000000e+00           NaN                 NaN
Latitude                  1.209974e+07  4.072588e+01            0.082014
Longitude                -2.196345e+07 -7.392563e+01            0.078452
Request_Closing_Time_sec  4.636776e+09  1.553182e+04        21922.155511
Request_Closing_Hours     1.287993e+06  4.314394e+00            6.089488
Request_Closing_Time_hrs  1.287993e+06  4.314394e+00            6.089488

                           Skewness    Kurtosis
Unique Key                 0.019106   -1.170309
Incident Zip              -2.448807   36.002132
Taxi Pick Up Location           NaN         NaN
Latitude                   0.116765   -0.718927
Longitude                 -0.291152    1.441333
Request_Closing_Time_sec  14.449752  845.618760
Request_Closing_Hours     14.449752  845.618760
Request_Closing_Time_hrs  14.449752  845.618760
```

Figure 18 Output for showing summary statistics of sum, mean, standard deviation, skewness, and kurtosis of the data frame.

The Python code given reads a dataset of customer service requests and then calculates some key summary statistics of the columns. The statistics contain sum, mean, standard deviation, skewness, and more.

- The total of everything, a sum shows the whole picture of the column. For example, it can show the total service requests on a specific area.
- The mean is an average of the values which gives the centre of the values or a typical location of the incident when plotted geographically or based on postal code.
- The standard deviation measures the amount of variation of a set of values. It must be grasped how proud data is to be standard deviation, that can now help identify how much the observation has changed out of the mean.
- Skewness is the measure of the symmetry of a distribution. In layman's terms the skewness of a data set tells us whether the data is heavier or lighter on one side. A positive skew shows that many data points are around left but there are few outliers in the right direction. The opposite is true for a negative skew.
- Kurtosis measures how "tailed" the data is, which helps identify if the data has extreme values or outliers. When data have high kurtosis, it means there is more extreme outliers, while low kurtosis means more even data (Menon, 2025).

Summary statistics allow us to uncover several significant aspects of the dataset. The values in the Request_Closing_Hours and Request_Closing_Time_hrs columns are identical, indicating consistency in time calculations. Both exhibit a strong positive skewness of around 14.45 and an extremely high kurtosis exceeding 845. This suggests that while most service requests close relatively quickly, a few take an exceptionally long time—indicative of significant outliers. There is a negative skew and very high kurtosis in the Incident Zip column, which may indicate geographic clustering within specific ZIP codes. The standard deviation of the Latitude and Longitude columns is relatively low, and their skewness is close to zero—characteristics typical of geographically clustered data with few outliers. All values in the Taxi Pick Up Location column are either zeros or nulls, as shown by the NaN mean and standard deviation. This may suggest that the column is mostly empty or not pertinent for analysis. Overall, the output emphasizes that distributions of service request closing times are often highly skewed and peaked. This is an important factor to consider when conducting further statistical tests or modeling.

ii. Write a Python program to calculate and show correlation of all variables.

The correlation matrix shows how the numeric variables in the data correlate. Each value shows how strongly and in which way two values/variables are related. The diagonal values are all 1.0 as a variable is perfectly correlated with itself.

```
numeric_data = data.select_dtypes(include=['number'])

# Calculate and print the correlation matrix
correlation = numeric_data.corr()
print(correlation)
```

```
                          Unique Key  Incident Zip  Taxi Pick Up Location  \
Unique Key                  1.000000      0.024897                    NaN
Incident Zip                0.024897      1.000000                    NaN
Taxi Pick Up Location            NaN           NaN                    NaN
Latitude                   -0.032276     -0.498492                    NaN
Longitude                  -0.009158      0.391360                    NaN
Request_Closing_Time        0.052209      0.058359                    NaN
Request_Closing_Time_sec    0.052209      0.058359                    NaN

                          Latitude  Longitude  Request_Closing_Time  \
Unique Key               -0.032276  -0.009158              0.052209
Incident Zip             -0.498492   0.391360              0.058359
Taxi Pick Up Location          NaN        NaN                   NaN
Latitude                 1.000000   0.364962              0.023820
Longitude                0.364962   1.000000              0.109899
Request_Closing_Time     0.023820   0.109899              1.000000
Request_Closing_Time_sec 0.023820   0.109899              1.000000

                          Request_Closing_Time_sec
Unique Key                                0.052209
Incident Zip                              0.058359
Taxi Pick Up Location                          NaN
Latitude                                  0.023820
Longitude                                 0.109899
Request_Closing_Time                      1.000000
Request_Closing_Time_sec                  1.000000
```

Figure 19 Correlation of all variables

```
import seaborn as sns
import matplotlib.pyplot as plt
corr = df.corr(numeric_only=True)
sns.heatmap(corr, cmap='cool', fmt='.2f', linewidth=0.5)
plt.show()
```

Figure 20 Using HeatMap

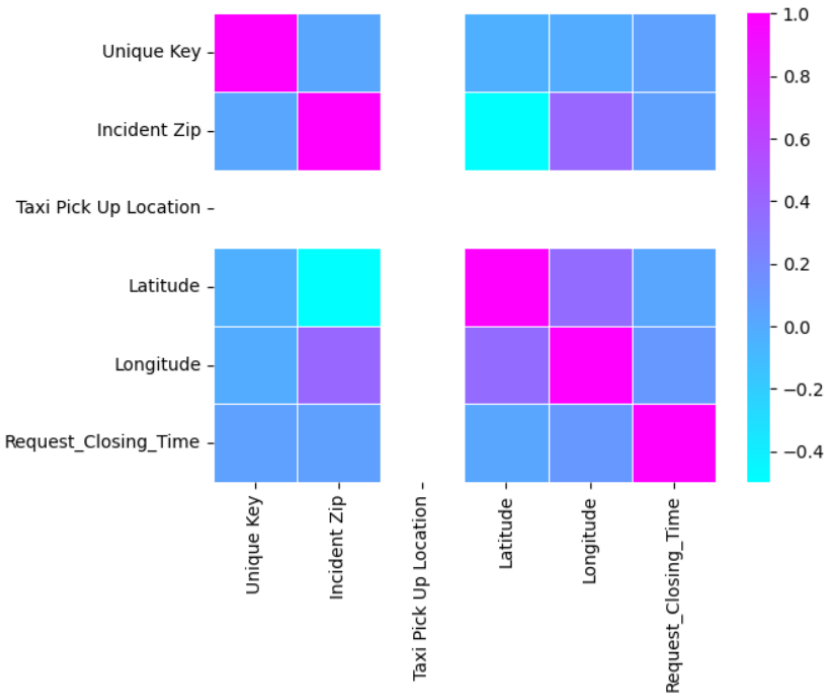

Figure 21 HeatMap of Correlation

A heat map is a data visualization in two dimensions where different values are mapped to color. A simple heat map provides an instant visual insight into information about two axes such that users may quickly grasp the most important or pertinent points of data. Sophisticated heat maps allow a viewer to see complex sets of data (Awati, 2025).

The heatmap illustrates the correlation matrix for every numerical variable within the data set. It indicates the strength and direction of variable relationships, with values ranging from -1 (indicating a perfect negative relationship) to +1 (indicating a perfect positive relationship). The color map transitions from cyan (indicating a negative relationship) to magenta (indicating a positive relationship), and the `'cool'` colormap is u The heatmap shows that most variables are not significantly correlated, as evidenced by the prevalence of blue and cyan near zero. Since `Request_Closing_Time` shows no strong correlation with any other variable, this suggests that response times are generally independent from factors like latitude, longitude, or zip code. In the absence of a `Taxi Pick Up Location`

column, which may be missing or nonexistent, the `Latitude` and `Longitude` columns are strongly correlated due to their spatial connection. However, most of the time there are **largely uncorrelated numeric variables** that should be kept in mind during feature selection for modeling.sed for contrast.

iv. Data Exploration

The first stage of data analysis is data exploration, which involves using statistical methods and data visualization tools to identify the properties of the data collection and preliminary trends (Robinson, 2025).

Provide four major insights through visualization that you come up after data mining.

1) Bar Graph

A bar chart is a visual representation that uses rectangular bars to show and compare discrete data categories. The height or length of each bar corresponds to the value or frequency of the related category (Anon., 2019).

```
# Chart 1: Top 10 Complaint Types (Bar Chart)
top_complaints = data['Complaint Type'].value_counts().head(10)
plt.figure(figsize=(10, 5))

# List of 10 colors for the bars
colors = ['lightblue', 'red', 'lightgreen', 'brown', 'pink', 'orange', 'black', 'lavender', 'purple', 'gray']

plt.bar(top_complaints.index, top_complaints.values, color=colors)
plt.title("Top 10 Complaint Types")
plt.xlabel("Complaint Type")
plt.ylabel("Number of Complaints")
plt.xticks(rotation=45)
plt.tight_layout()
```

Figure 22 Making Bar graph

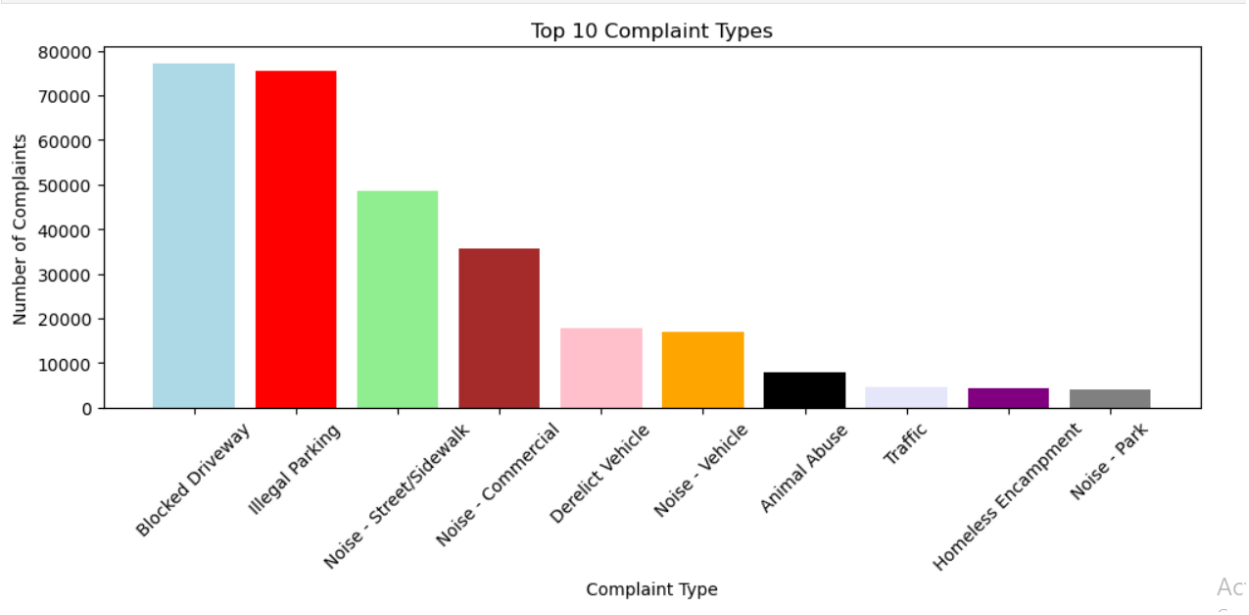

Figure 23 Bar graph

The bar chart illustrates the ten most common types of complaints recorded in the NYC 311 dataset. The complaints are organized from most to least frequent, with each bar having a different color for clarity and visual appeal. The chart illustrates the frequency of various types of complaints, especially those concerning parking and noise. When the figure is appropriately sized and the x-axis labels are angled, the chart maintains its readability even when category names are lengthy. This visualization offers an instant grasp of the areas of public discontent, potentially aiding city agencies in directing resource distribution and planning responses.

Key Findings:

- With over 70,000 complaints for each category, "Blocked Driveway" and "Illegal Parking" are the most commonly complained-about categories.
- The majority of the top 10 complaints are related to noise (i.e., noise-street/sidewalk, noise-commercial, noise-vehicle, and noise-park), indicating widespread urban issues.

- 'Derelict Vehicle' and 'Animal Abuse' reports are equally common, suggesting that the public is still upset about abandoned cars and safety.
- Although they are not as frequently complained about, "traffic" and "homeless encampment," which rank just below the bottom of the top 10, are nevertheless significant public issues.
- Public space infractions and quality-of-life interruptions related to streets and cars account for the majority of the top complaints, according to the chart.

2) Pie chart

Data can be arranged and displayed as a percentage of the total using a pie chart. As the name suggests, the entire is represented by a circle in this type of representation, while the individual categories that make up the whole are represented by slices of the circle, or "pie" (Anon., 2019).

```
borough_counts = data['Borough'].value_counts()

plt.figure(figsize=(7, 7))
plt.pie(borough_counts, labels=borough_counts.index, autopct='%.1f%%',
        colors=['pink', 'lavender', 'lightgreen', 'lightblue', 'orange'])
plt.title('Complaint Distribution by Borough')
plt.axis('equal')
plt.tight_layout()
```

Figure 24 Making Pie chart

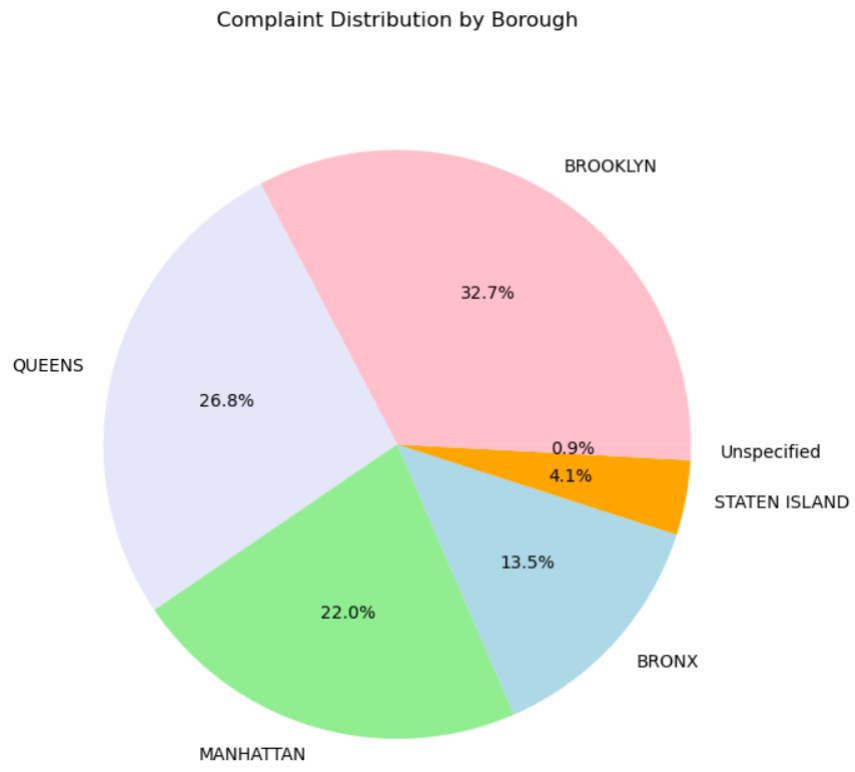

Figure 25 Pie chart

Key Findings

- With 32.7% of all complaints, Brooklyn is the most complained about.
- Queens comes in second with 26.8%, indicating a high volume of interactions with public services.
- Manhattan contributes 22.0%, presumably as a result of its dense population and business.
- Staten Island has a comparatively low complaint rate of 4.1%, whereas the Bronx adds 13.5%.
- Unspecified boroughs make up 0.9% of the total, suggesting that some geographical data is missing or not classified in the database.

3) Line Graph

```
import pandas as pd
import matplotlib.pyplot as plt

# Convert 'Created Date' to datetime
df['Created Date'] = pd.to_datetime(df['Created Date'])

# Extract Month from Created Date
df['Month'] = df['Created Date'].dt.to_period('M')

# Group by Month and count complaints
monthly_complaint = df.groupby('Month')['Complaint Type'].count()

# Plot
plt.figure(figsize=(10, 5))
monthly_complaint.plot(kind='line', marker='o', color='red')
plt.title('Monthly Complaint Volume')
plt.xlabel('Month')
plt.ylabel('Number of Complaints')
plt.xticks(rotation=45)
plt.grid(True)
plt.tight_layout()
```

Figure 26 Making Histogram

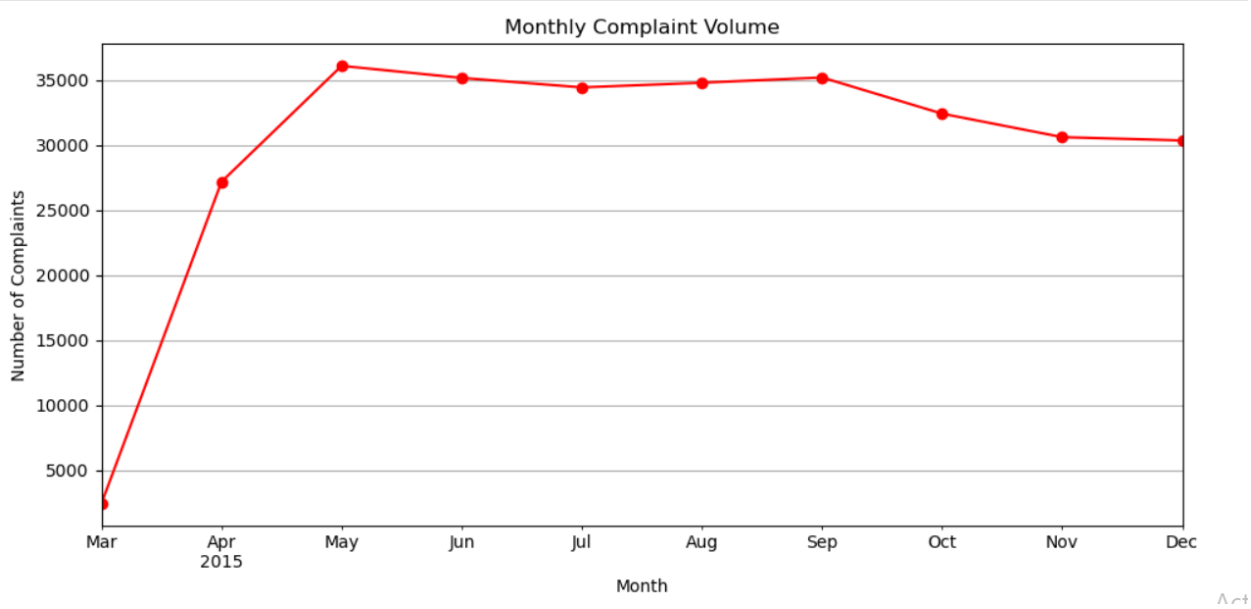

Figure 27 Histogram

The line graph charts the monthly sum of 311 complaints, grouped and counted by the Created Date field. Having extracted dates and pulled out the month part, data is summarized to show how complaint frequency changes over time. The red line with circle

markers provides a readable visualization of complaint trends over the course of the year. This plot is useful to identify seasonal patterns or outliers in public demand for city services. The plot shows that volumes of complaints increased exponentially in earlier months, followed by a plateau, with discernible oscillations towards the second half of the year. Such observations are valuable to visualize when city services peak in public demand.

Key Findings:

- A sharp increase in complaints from March to May suggests that demand for spring services is growing.
- May has the highest level of public activity or service controversy and the highest number of complaints.
- The fact that the number of complaints is high and largely consistent from June to September suggests that demand is still there.
- Between October and December, there is a slow decrease, which can be due to a decrease in public activity brought on by colder weather.
- Seasonal usage is reflected in the trend, which could help local agencies better plan and prepare for periods of high demand.

4) Heatmap

A heat map is a data visualization in two dimensions where different values are mapped to color. A simple heat map provides an instant visual insight into information about two axes such that users may quickly grasp the most important or pertinent points of data. Sophisticated heat maps allow a viewer to see complex sets of data (Awati, 2025).

```
import seaborn as sns
import matplotlib.pyplot as plt

top_complaints = df['Complaint Type'].value_counts().nlargest(10).index
df_top = df[df['Complaint Type'].isin(top_complaints)]
heatmap_data = df_top.pivot_table(index='Complaint Type', columns='Borough',
                                  values='Unique Key', aggfunc='count', fill_value=0)

plt.figure(figsize=(12, 6))
sns.heatmap(heatmap_data, annot=True, fmt='d', cmap='YlGnBu')
plt.title('Top 10 Complaint Types by Borough')
plt.tight_layout()
```

Figure 28 Making Heatmap

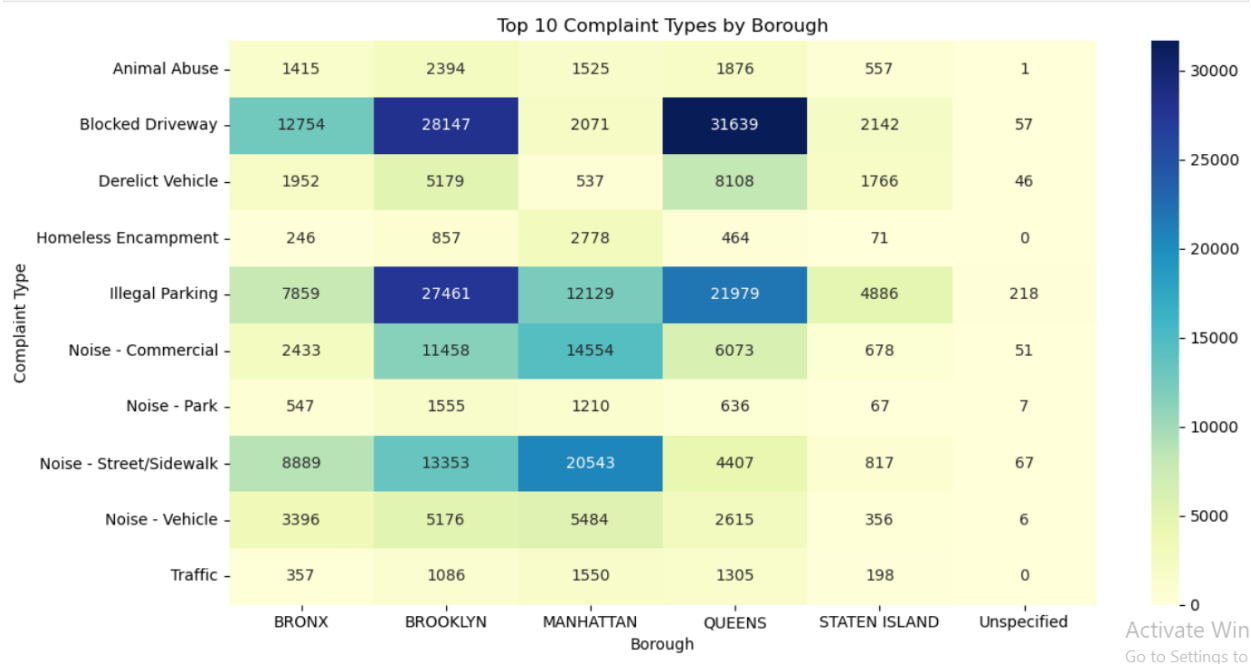

Figure 29 HeatMap

With a few unidentified cases, the heatmap shows a classification of the top 10 complaint categories in the five main boroughs of New York City. The number of complaints for a certain complaint type in a given borough is shown in each heatmap cell, and the intensity is color-coded, with darker values denoting higher volumes. To make it easy to compare the geographic distribution of each complaint type, this map employs a pivot table with

complaint kinds as rows and boroughs as columns. The chart is a useful tool for targeted public service planning and budget allocation since it gives a rapid visual picture of which boroughs are most impacted by particular concerns.

Key Findings:

- The most frequent complaint is "Blocked Driveway," particularly in Queens (31,639) and Brooklyn (28,147).
- Another major problem is "illegal parking," which is prevalent throughout the city and is particularly prevalent in Brooklyn, Manhattan, and Queens.
- In Manhattan, noise-related complaints are common, with street/sidewalk noise and commercial noise being the most common causes.
- Queens has the most complaints about derelict vehicles, which is indicative of a higher number of abandoned vehicles there.
- Perhaps because central neighborhoods have the highest concentration of visible homelessness, concerns about homeless encampments are most prevalent in Manhattan.
- Without any peaks or troughs, the Bronx continuously reports reasonable numbers for every category of complaints Because it is less densely inhabited and has the smallest population, Staten Island has the lowest numbers in every category.
- Small but there are unspecified borough listings, which could be the result of inadequate or missing location data..

● Arrange the complaint types according to their average 'Request_Closing_Time', categorized by various locations. Illustrate it through graph as well.

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

# Calculate Request_Closing_Time and convert to hours
data['Request_Closing_Time'] = data['Closed Date'] - data['Created Date']
data['Request_Closing_Hours'] = data['Request_Closing_Time'].dt.total_seconds() / 3600

# Drop missing values
data = data.dropna(subset=['Complaint Type', 'Borough', 'Request_Closing_Hours'])

# Group by Complaint Type and Borough and calculate average time
avg_times = data.groupby(['Complaint Type', 'Borough'])['Request_Closing_Hours'].mean().reset_index()

# Select top 5 most common complaint types for clear visualization
top_complaints = data['Complaint Type'].value_counts().head(5).index
filtered_data = avg_times[avg_times['Complaint Type'].isin(top_complaints)]

# Plot grouped bar chart
plt.figure(figsize=(12, 6))
sns.barplot(data=filtered_data, x='Complaint Type', y='Request_Closing_Hours', hue='Borough')
plt.title('Average Request Closing Time by Complaint Type and Borough')
plt.xlabel('Complaint Type')
plt.ylabel('Average Closing Time (in hours)')
plt.xticks(rotation=45)
plt.legend(title='Borough')
plt.tight_layout()
```

Figure 30 complaint types according to their average 'Request_Closing_Time'

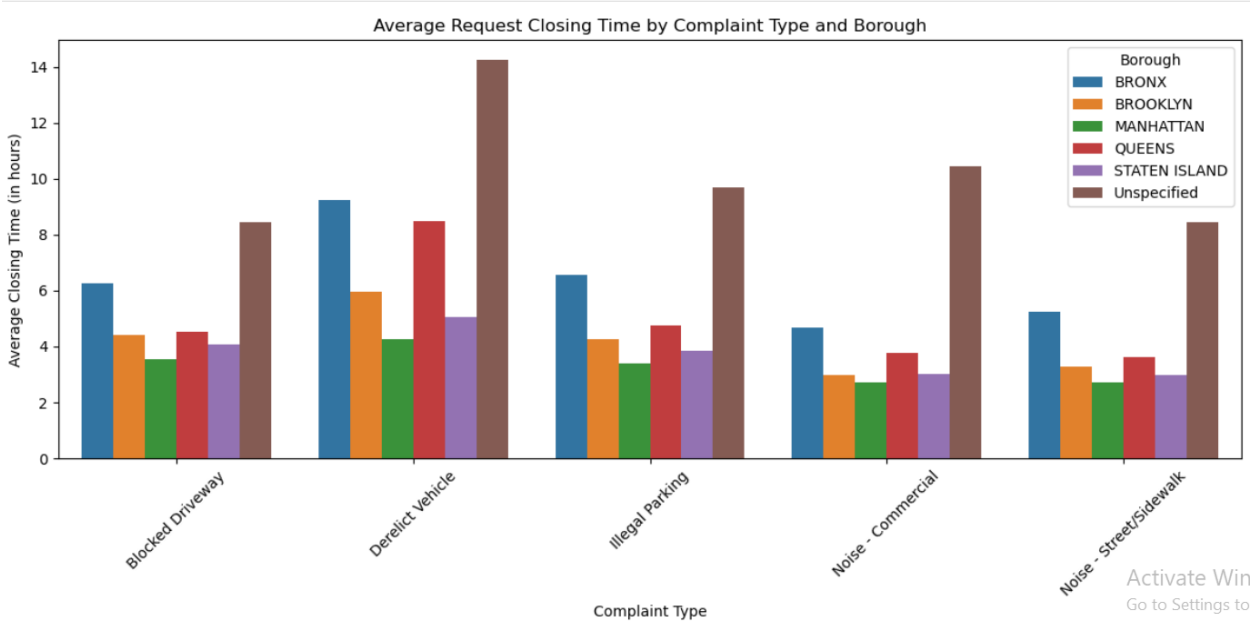

Figure 31 Output for complaint types according to their average 'Request_Closing_Time'

The average closing time (in hours) for the most common complaint kinds in New York City is displayed by borough in the group bar chart. The y-axis displays the appropriate average closing time, while the x-axis displays the various complaint kinds (such as "Blocked Driveway" and "Noise-Commercial"). A borough is indicated by each colored bar in a complaint category, making it possible to compare closing times within places directly. To determine the request closure hours, Request_Closing_Hours was calculated from the dataset and grouped to determine means per borough.

Key Findings:

- The mean closing time for all complaint kinds is longest in Unspecified Boroughs, often exceeding 14 hours.
- In general, derelict vehicle complaints take the longest to resolve, particularly in boroughs that are Unspecified and STATEN ISLAND.
- Commercial and street/sidewalk noise complaints close the quickest in the majority of boroughs, particularly in Queens and Brooklyn.
- Response times for blocked driveway reports are more uniform and comparatively shorter across all boroughs.

- There may be differences in the degree of service efficiency between Manhattan, Queens, and Brooklyn, as they often have quicker closing times than Staten Island or Unspecified.

v. Statistical Testing

- **Test 1**: Whether the average response time across complaint types is similar or not.
- State the Null Hypothesis (H0) and Alternate Hypothesis (H1).

Null Hypothesis (H0) - We use this as our starting point. The word, "null", means "nothing", or, "no change". In other words, this idea, indicates that there are no real differences between groups, or no connection between what you are studying. The null hypothesis indicates that everything remains the same with no changes (Anon., 2025).

Alternative Hypothesis (Ha)- This Hypothesis expresses what you think might be true. In other words, it describes what you expect to find once you collect and analyze the data for your research. The alternative hypothesis helps answer your particular research question (Anon., 2025).

Null hypothesis (H_0): The mean response time is equivalent for all complaint types.
Alternative hypothesis (H_1): The mean response time is different for at least one complaint type.

- Perform the statistical test and provide the p-value.
  We performed a one-way ANOVA test to compare the average request closing time among the top 5 most common complaint types.

  ANOVA (Analysis of Variance) is a statistical test used to compare the means of three or more groups. It is an extension of the t-test, which is used to compare the means of two groups.

```
# Test 1: ANOVA on top 5 complaint types
import pandas as pd
from scipy.stats import f_oneway

df['Request_Closing_Time'] = (df['Closed Date'] - df['Created Date']).dt.total_seconds()/3600
df = df.dropna(subset=['Complaint Type','Request_Closing_Time'])

top5 = df['Complaint Type'].value_counts().head(5).index
groups = [df[df['Complaint Type']==t]['Request_Closing_Time'] for t in top5]
f_stat, p_value = f_oneway(*groups)

# Display result
print("F-statistic:", round(f_stat, 2))
print("p-value:", p_value)

# Interpret result
if p_value < 0.05:
    print("Conclusion: Reject the null hypothesis. Average response time differs among complaint types.")
else:
    print("Conclusion: Fail to reject the null hypothesis. No significant difference in response time.")
```

Figure 32 Test Whether the average response time across complaint types is similar or not.

The conditional test establishes whether or not the null hypothesis—that the groups have equal means—should be rejected after printing the F-statistic and p-value from the same. The null hypothesis, which states that at least one category of complaint has a markedly different average response time, is rejected when the p-value < 0.05 'Created Date' is subtracted from 'Closed Date' in order to determine the response time, which is then divided by hours by the program. After that, it returns the top five complaint types by frequency and removes any missing values. The `scipy.stats` library's `f_oneway` method is then used to execute the one-way ANOVA test after retrieving the reaction times for each of them. The conditional test establishes whether

or not the null hypothesis—that the groups have equal means—should be rejected after printing the F-statistic and p-value from the same. The null hypothesis, which states that at least one category of complaint has a markedly different average response time, is rejected when the p-value < 0.05. The analysis is useful for determining how various complaint categories are handled differently, which can then be used to improve resource usage and service efficiency.

- Interpret the results to accept or reject the Null Hypothesis.

```
F-statistic: 1799.61
p-value: 0.0
Conclusion: Reject the null hypothesis. Average response time differs among complaint types.
```

Figure 33 Results to accept or reject the Null Hypothesis.

The findings indicate a p-value of 0.0 and a F-statistic value of 1799.61. We reject the null hypothesis because the p-value is significantly lower than the conventional cut off value of 0.05. The average response time varies for each complaint category, according to strong statistical evidence. In other words, some complaints take a lot longer than others to resolve. This type of variation may be due to a variety of complaint types that demand greater intricacy, gravity, or resource utilization. The results are important for urban governance because they show that complaints that take longer to settle need to be examined and possibly improved.

- **Test 2**: Whether the type of complaint or service requested and location are related.
- State the Null Hypothesis (H0) and Alternate Hypothesis (H1).

  Null Hypothesis (H0) - We use this as our starting point. The word, "null", means "nothing", or, "no change". In other words, this idea, indicates that there are no real differences between groups, or no connection between what you are studying. The null hypothesis indicates that everything remains the same with no changes (Anon., 2025).

  Alternative Hypothesis (Ha)- this Hypothesis expresses what you think might be true. In other words, it describes what you expect to find once you collect and analyze the data for your research. The alternative hypothesis helps answer your particular research question (Anon., 2025).

  Null hypothesis (H_0): Complaint type is independent of location.

  Alternative hypothesis (H_1): Complaint type is dependent on location.

- Perform the statistical test and provide the p-value.

Smart data analysis uses the chi-square test, a widely used statistical test, to determine whether two or more category variables have a statistically significant association.

```
# --- Test 2: Chi-square ---
from scipy.stats import chi2_contingency
# Build contingency table of complaint type vs borough
ct = pd.crosstab(data['Complaint Type'], data['Borough'])

# Perform Chi-square test
chi2, p_value_chi2, dof, expected = chi2_contingency(ct)

# Display result
print("\nChi-square statistic:", round(chi2, 2))
print("Chi-square p-value:", p_value_chi2)

# Interpret result
if p_value_chi2 < 0.05:
    print("Conclusion: Reject the null hypothesis. Complaint type and borough are related.")
else:
    print("Conclusion: Fail to reject the null hypothesis. Complaint type and borough are independent.")
```

Figure 34 Whether the type of complaint or service requested and location are related.

This code executes a Chi-square test of independence to assess whether there exists a notable association between complaint type and borough in the NYC 311 dataset. We run the test because the two variables we have here, complaint-type and borough, are categorical, and we are interested in testing whether complaint types occur with significantly different frequency distributions across boroughs. To find out how often each complaint occurs in each borough, a contingency table is created using `pd.crosstab()`. The test is then executed utilizing scipy.stats chi2 contingency. The result provides a Chi-square statistic and a p-value. Based on the p-value code, a decision is made: If the p-value is less than 0.05, the null hypothesis is rejected, and a correlation between complaint type and borough is established. This means that specific complaint types occur more frequently in particular boroughs. If the p-value is greater than 0.05, it finds that complaint type and borough are uncorrelated, i.e., there is no noticeable relationship. The survey

will help assess whether public views differ by location which is relevant to the resource goal-oriented allocation and policy design.

- Interpret the results to accept or reject the Null Hypothesis.

```
Chi-square statistic: 79641.56
Chi-square p-value: 0.0
Conclusion: Reject the null hypothesis. Complaint type and borough are related.
```

Figure 35 Results to accept or reject the Null Hypothesis.

The Chi-square test yields a p-value of 0.0 and a Chi-square statistic of 79,641.56. We reject the null hypothesis because the p-value is significantly below the standard level of significance of 0.05. In other words, complaint type and borough are statistically significantly associated. In other words, the relative distribution of complaint types is not borough-independent; rather, the frequency of various complaint types varies by borough.

vi. Conclusion:

In this session, we used data cleansing, visualization, and statistical testing methods to examine and analyze 311 service request data. We determined which complaint kinds were most common and where they occurred in different boroughs. We were given clear information about complaint trends and response times using bar charts, pie charts, histograms, and line plots. According to statistical testing, there is a considerable correlation between the type of complaint and the location, and the average response time varies significantly depending on the type of complaint. All things considered, the analysis assisted in identifying noteworthy trends in the city's use of public services.

vii. References

Anon., 2019. *Pie Chart Definition.* [Online]
Available at: https://www.tableau.com/chart/what-is-pie-chart
[Accessed 1 May 2025].

Anon., 2019. *What is a Bar Chart?.* [Online]
Available at: https://www.jaspersoft.com/articles/what-is-a-bar-chart
[Accessed 1 May 2025].

Anon., 2025. *Histogram - Definition, Types, Graph, and Examples.* [Online]
Available at: https://www.geeksforgeeks.org/histogram/
[Accessed 01 May 2025].

Anon., 2025. *Statistics Resources.* [Online]
Available at: https://resources.nu.edu/statsresources/hypothesis
[Accessed 01 May 2025].

Anon., 2025. *What Is Data Preparation? Comprehensive Guide + 9 Steps For Effective Data Prep.* [Online]
Available at: https://www.astera.com/type/blog/data-preparation/
[Accessed 4 04 2025].

Awati, R., 2025. *Tech Target.* [Online]
Available at: https://www.techtarget.com/searchbusinessanalytics/definition/heat-map
[Accessed 10 May 2025].

Elfman, L., 2025. *data.world.* [Online]
Available at: https://data.world/blog/smart-data-discovery/
[Accessed 31 03 2025].

Elizabeth Coleman, C. H., 2023. *Line Plot Definition & Examples.* [Online]
Available at: https://study.com/learn/lesson/line-plot-example-creation.html#:~:text=A%20line%20plot%2C%20also%20called,also%20called%20the%20x%2Daxis.
[Accessed 01 May 2025].

hightouch, 2025. *The five steps for data analysis.* [Online]
Available at: https://hightouch.com/blog/steps-for-data-analysis
[Accessed 5 04 2025].

Menon, k., 2025. *Simplilearn.* [Online]
Available at: https://www.simplilearn.com/tutorials/statistics-tutorial/skewness-and-kurtosis
[Accessed 05 04 2025].

Morris, A., 2025. *Oracle NetSuite.* [Online]
Available at: https://www.netsuite.com/portal/resource/articles/erp/data-discovery.shtml
[Accessed 31 03 2025].

Morris, S., 2025. *coresignal.* [Online]
Available at: https://coresignal.com/blog/data-discovery/
[Accessed 31 03 2025].

Robinson, S., 2025. *data exploration.* [Online]
Available at: https://www.techtarget.com/searchbusinessanalytics/definition/data-exploration#:~:text=Data%20exploration%20is%20the%20first,set%20characteristics%20and%20initial%20patterns.
[Accessed 01 May 2025].