23049138 Aagya Dahal

Islington College,Nepal

Document Details

Submission ID
trn:oid:::3618:95827975

Submission Date
May 14, 2025, 2:30 PM GMT+5:45

Download Date
May 14, 2025, 2:32 PM GMT+5:45

File Name
23049138 Aagya Dahal

File Size
17.5 KB

23 Pages

2,818 Words

14,564 Characters

12% Overall Similarity

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

Match Groups

- **32** Not Cited or Quoted 12%
  Matches with neither in-text citation nor quotation marks
- **0** 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

6% Internet sources
2% Publications
9% 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 focus your attention there for further review.

Match Groups

- **32** Not Cited or Quoted 12%
  Matches with neither in-text citation nor quotation marks
- **0** 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

- 6% Internet sources
- 2% Publications
- 9% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	Submitted works	islingtoncollege on 2025-04-18 — 3%
2	Submitted works	islingtoncollege on 2025-04-18 — 2%
3	Publication	Tanushri Banerjee, Arindam Banerjee, Dhaval Maheta, Vivek Gupta. "Business An... — <1%
4	Internet	www.coursehero.com — <1%
5	Internet	pdfcoffee.com — <1%
6	Submitted works	Glasgow Caledonian University on 2023-08-10 — <1%
7	Internet	actuariesindia.org — <1%
8	Internet	rstudio-pubs-static.s3.amazonaws.com — <1%
9	Submitted works	Kingston University on 2024-12-04 — <1%
10	Submitted works	QA Learning on 2024-07-12 — <1%

11 Submitted works

Swinburne University of Technology on 2023-05-22 <1%

12 Submitted works

Trevecca Nazarene University (Nashville) on 2025-05-06 <1%

13 Submitted works

London Metropolitan University on 2023-05-12 <1%

14 Internet

www.tandfonline.com <1%

15 Publication

Erol Tutumluer, Imad L. Al-Qadi. "Bearing Capacity of Roads, Railways and Airfield... <1%

1. Data Understanding

The provided coursework is the data set record of “The NYC 311 service” which is a service request calls that provides an effective means for community people to report non-emergency issues and get access to several city services. It has a high volume of requests per day that it processes and sorts them into different categories and forwards them to the appropriate agencies for resolution. Some of these common requests for help include noise complaints, plumbing issues or parking violations. Once the agency has addressed the problem, the request is closed, and the person is notified of the result. NYC 311 is designed to be convenient for the user and deliver effective service to all New Yorkers in a timely manner. However, NYC 311 is an important service to the New Yorkers because they can readily report non-emergency issues and access the city services. It helps community engagement as well as a responsive government for living a better urban life.

The below mentioned table shows all the key columns in the data set.

S. No

2 Column Name

Description

Data Type

1

Unique Key

This is the unique identifier for every service request made.

Int64

2

Created Date

This tells us when the service request was initiated.

Object

3

Closed Date

This tells us when the service request was resolved or closed.

Object

4

Agency

This tells which city agency is responsible for handling the request.

Object

5

Complaint Type

It tells the general category of the issue reported in the service request.

Object

6

Descriptor

It provides a more detailed description of the complaint type.

Object

7

Location Type

It indicates the type of place where the incident occurred (e.g., sidewalk).

Object

8

Incident Zip

This tells us the ZIP code where the incident occurred.

Float64

9

City

It represents the city where the request was reported.

Object

10

Status

It shows the current status of the request (e.g., Open, Closed).

Object

11

Resolution Description

It explains the final action taken or resolution for the service request.

Object

12

Borough

This tells which NYC borough the complaint was made in (e.g., Bronx, Queens).

Object

13

9 Latitude

The latitude coordinate where the incident occurred.

Float64

14

Longitude

The longitude coordinate where the incident occurred.

Float64

2. Data Preparation

2.1. Import the dataset.

In data preparation phase, the first step is to import NYC 311 service request dataset to a Python environment with a data analysis library like Pandas. Usually, read_csv() function is used to load the data from a .csv file into a DataFrame that is necessary or any further analysis or preprocessing. Importing the dataset successively is essential for further data manipulations and analysis.

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

In this case, it probably shows a first look at the data using commands like df.head(), df.info(), and df.describe(), etc to know the structure of the dataset. It shows number of rows and columns, types of data, and has any missing values. This is an important step, because it allows analysts to see what data quality issues there are and how to preprocess the data to figure out how to deal with different types of data.

2.3. 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.

We convert the "Created Date", "Closed Date" columns from string format to datetime format here. This allows time-based calculation and creates a new feature called "Request_Closing_Time" that shows the time taken to resolve the service request.

2

This new column calculates the difference between the "Created Date" and "Closed Date" and gives us some insights about the efficiency of the service system as well as how well the different complaints are handled.

2.4. Drop irrelevant Columns which are listed below using python.

In this we are here doing the cleaning process of our dataset by removing unnecessary columns for analysis. It is a process of maintaining only the features which gives the information that can be relevant and get rid of the rest to make it clutter free and more efficient. Functions such as drop() can be used to drop columns with duplicate data, or with irrelevant metadata, or lots of missing values. This will simplify the dataset having only useful attributes to analyze which will improve performance and it will be clearer.

4 2.5. Program to remove the NaN missing values from updated dataframe using python.

Data cleaning is outlined within the text, as well as how to handle missing values with the dropna() function to drop rows with null values. There are more sophisticated

imputation methods that can be used, but this basic approach is often used first to have a complete dataset that is essential for accurate statistical analysis and machine learning.

2.6. Program to see the unique values from all the columns in the dataframe using python.

Here we see the uniqueness of values in each column of the dataset, as in the case of categorical columns such as 'Complaint Type', 'Agency', and 'Location'. For instance, if we use Python functions like ad[column] unique() we can identify such distinct entries that can be useful for data preparation tasks such as encoding and filtering; or to find out any inconsistencies or errors in the data.

3. Data Analysis.

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

Important statistical measures such as sum, mean, standard deviation, skewness and kurtosis for numerical fields in the dataset are calculated. These statistics provide information on how the data is distributed and how it behaves. The average value is described by the mean, and the variability by the standard deviation. Skewness

provides the asymmetric part of the distribution, and kurtosis identifies hiding outliers, or 'tailedness' of the distribution.

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

This visualizes a correlation matrix, most represented as a heatmap, which measures the relationships between numerical variables in the dataset with Pearson's correlation coefficient. If values near +1 or -1 then they are high positive or negative correlation, and values close to 0 indicate no correlation. It helps to identify variables related to one another that can predict each other. This could lead to, for example, a strong correlation between 'Request_Closing_Time' and certain complaint types to be exploited during optimization efforts.

4. Data Exploration

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

Insight 1: Time-of-Day Analysis for Different Complaint Types

Here, we see how complaints are distributed by type at different times in the day. Each hour is shown on the x-axis, and the y-axis tells you how many complaints were reported for that hour. A different colour or line style is used for each complaint type, which makes it easy to compare them. The graph makes it clear which complaint types

get reported most often, such as noise complaints in the evening and heating complaints in the morning. It is useful for arranging resources and setting up staffing according to demand. Fluctuations in the graph can point to how people behave and highlight whether services are easy to access or responsive. Also, the visualization's code could group complaints by hour and type, and aggregate them to see how often each one appears. Pandas and matplotlib/seaborn are Python libraries that are used to organize the data and show it as line or bar graphs. Time is typically taken from pd.to_datetime(), and grouping happens with .groupby(['hour', 'complaint_type']).size().unstack() so the visualization is easier to read. This understanding is useful for assigning resources to field response teams. Frequency for every complaint type will be calculated with value_counts() or groupby().size() when making this visualization. The results get sorted from highest to lowest and then shown in a bar graph, either horizontal or vertical. This kind of initial data analysis lets us spot where resources should be focused or where changes to the system may help.

Insight 2: Response Time Analysis by Day of Week

The chart shows the number of times each complaint is mentioned. The graph uses the x-axis for complaint types, for example Noise, Water Leak, and Illegal Parking, and the y-axis for how many complaints were logged for each category. Bars that are taller represent complaints that are more common. The visualization highlights where the most complaints come from. City authorities can use this information to focus first on solving the most frequent problems. The visualization could point out infrequent complaints, which could be symptoms of specific or local problems. Stakeholders can easily understand the information because the bar chart is simple. The code would take the timestamp for each request, change it to the weekday name using .dt.day_name(), group the data by those names, and find the mean or median response time. Seaborn’s boxplot or barplot could be used to visualize the data here. Knowing this helps the authorities organize their staff and maintenance crews for days with high response times.

Insight 3: Monthly Trend Analysis

The figure probably illustrates changes in response times during the week from Monday to Sunday. Each column or point shown on the chart represents a different weekday, and the y-axis gives the time, such as in hours or days. It is possible that the graph reflects that weekends have slower responses because of fewer staff and weekdays have faster responses. Outliers on the graph may be evidence of operational problems or special events, for example, holidays. The results help agencies make decisions about how to assign work or ensure better weekend support. The variation shown here might have an effect on public satisfaction. Complaint timestamps are first grouped by month with the help of datetime processing. A line chart or a bar plot is used to show the number of complaints each month. It makes it easier for staff to plan by seeing when demand is likely to be higher.

Insight 4: Response Time by City

This figure makes it possible to see how many complaints are raised in each month, from January through December. On the x-axis, you have the months, and along the y-axis is the total number of complaints. Seasonal trends are clear because complaints increase during the coldest or hottest months of the year. It is simpler to find rising or falling trends when the data is shown as a smooth line. If the chart covers several years, it becomes possible to see any differences in the trends. Policymakers can look at this information to be ready for regular problems before they become a big issue. For each 'City' or 'Borough', the computation of average 'Request_Closing_Time' takes place. First, groupby is used with aggregation functions like mean or median, and then the values are sorted and shown with matplotlib or seaborn. It supports authorities in finding out where there are problems with efficiency.

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

The chart shows how quickly cities or boroughs respond to complaints on average. The x-axis shows different cities, while the y-axis shows the average time it takes to respond. This chart makes it clear that response times vary between cities, with some always performing better than others. Such differences might come from staffing shortages, high population in some places, or poor infrastructure. It promotes responsibility and urges specific reviews or financial support in certain areas. Color can be used to show if the response times are good, fair, or bad. A pivot table or crosstab is made between 'Complaint Type' and 'Borough', and seaborn.heatmap is used to show where complaints are most common. This helps with Test 2 (Chi-square test of independence), showing if there is a correlation between complaint type and location.

5. Statistical Testing

1. Test 1: Whether the average response time across complaint types is similar or not.

· State the Null Hypothesis (H0) and Alternate Hypothesis (H1).

Test 1: Analysis of Response Time Across Complaint Types

Null Hypothesis (H0): The average response time is the same across all complaint types

Alternative Hypothesis (H1): The average response time differs significantly across complaint types

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

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

This test's Null Hypothesis (H_0) says that the average response time is the same for all complaint types, and the Alternative Hypothesis (H_1) proposes that the response

times differ meaningfully between complaint types. The aim of this test is to see if the city responds the same way to every complaint or if some issues get resolved quicker than others. It is necessary for finding out if there are problems with how services are provided. If we cannot reject H_0 , then we assume all complaint types are managed with the same sense of urgency. If H_0 is rejected, it means that specific improvements should be made so that service is equal and prompt for all complaints.

To check the hypothesis, it is common to use a statistical procedure such as ANOVA. ANOVA is used to compare the average response times for each type of complaint and decide if the variations are real or just by chance. If the p-value comes out to be less than 0.05, we reject the null hypothesis. There is enough proof here to say that response times for different complaint types are not the same. The process looks at the difference between response time variance within complaint types and variance among all complaint types. If there is a big gap between the groups, it means the service is not given equally.

The figure here generated is a bar plot which is showing the average response time per complaint type. There is a clear trend that the fastest responses are given to 'Water Leak' or 'Power Outage,' while 'Noise Complaint' is handled more slowly. When the visual differences are backed up by a low p-value, they make a strong argument for operational changes. Looking at the chart, one can easily see which complaint types need to be handled differently or deserve more resources based on the results of the hypothesis test. The information helps managers decide on response time targets

or find out why certain services are slower to address. In brief, it points out how some places may get less attention than others in cities.

2. Test 2: Whether the type of complaint or service requested, and location are related.

· State the Null Hypothesis (H0) and Alternate Hypothesis (H1).

Test 2: Relationship Between Complaint Type and Location (Borough)

Null Hypothesis (H0): Complaint type and Borough are independent

Alternative Hypothesis (H1): There is a significant association between complaint type and Borough

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

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

This test uses H_0 to state that complaint type and borough (or location) are not related, which means the type of complaint filed is expected to be the same everywhere. According to H_1 , there is a meaningful tie between the borough and the type of complaint reported. It is an important test since it reveals how complaints are distributed over different areas. If the null hypothesis is rejected, it means that some types of complaints are more frequent in certain places, suggesting possible local issues or repeating problems. This test lets municipalities use resources where they are needed most and learn more about local concerns.

3 Usually, the Chi-Square Test of Independence is used to check this hypothesis. It compares how the complaint types are distributed across boroughs to how they would be distributed if every borough had the same number of each complaint type. When the p-value is significant (most often below 0.05), it means the null hypothesis is rejected, which means borough and complaint type are related. Basically, the test sees if the pattern of complaints across boroughs matches the pattern we would expect if complaints were evenly distributed. The test statistic is largely driven by big differences between observed and expected values and this suggests dependence.

In a heatmap here, the rows show the boroughs in the complaint distribution heatmap, with the columns representing different complaint types and colours showing the frequency of each. Boroughs with darker or brighter zones on the map have a higher number of complaints of that type. As an example, the majority of 'Noise Complaints' could be in Manhattan, whereas 'Water Supply Issues' are mostly seen in the Bronx. Using the heatmap, you can easily see how the complaint types are related to each borough. The Chi-Square test results along with the figure, it shows the statistical findings confirmed and can easily see the hotspots. This information lets city authorities provide better services by focusing on the boroughs where they are needed most, making citizens happier.