METHODOLOGY and ANALYSIS

Analysis Million of NYC Parking Violations

Iris Yeung

Baruch College

STA 9760 Big Data Technologies

Professor Mottaqui Karim

23th March, 2021

Background

This project aims to analyze the Open Parking and Camera Violations dataset on the website NYC Open Data by using big data technologies including Python, AWS EC2, Elasticsearch, Kibana and Docker, to build a data pipeline, analyze data and visualize the data to a Dashboard. This article will discuss the methodologies used in this project and analyze the data to provide insight.

Methodology

Virtualization

When dealing with a high volume of data in the project, it is essential to have excellent hardware like powerful hard drives and RAM storage to support the workload.[1] However, not everyone, especially a college student like me, can afford a futuristic data centre for a single project. In this case, moving to cloud service platforms could be a solution. In this project, I used the Amazon Elastic Compute Cloud (AWS EC2), which provides a web service that allows the user to launch and manage UNIX/Linux or Window instances in its data centre. So users could enjoy the experience of using compelling hardware even with a light-weight laptop.

---

[1]: Big Data requires bigger hardware https://tdan.com/big-data-requires-bigger-hardware/24339

Containerization

Regarding the purpose of one single project, I do not need to install everything into that instance. For example, I don't need to run a whole operating system with a User interface. All I needed is to run the program and to install the required library packages. Therefore, this project used a containerization technique - Docker. It provides a blueprint for the environment to build.

In practice, I write a docker image to install Python 3.9 for writing the data ETL program, create a new folder to include all the required files, copy the files into that new folder, install libraries and set up the environment for running the program. The docker file allows users to run the program without installing them into an operating system on a computer or any data centre. It only takes a few command lines in the terminal to run the program.

Cloud-based Database and Data Visualization

A large amount of data required a large capacity of storage. The same as the Virtualization part, we could use cloud service platforms to benefit from their powerful Data centre. In this project, I used AWS Elasticsearch to run a database for data storage and AWS Kibana as the data visualization tool. To host a domain on Elasticsearch, users need to pass through a few parameters like MasterName, Password, decide the number of nodes and the type of license.

Data ELT Program

At the first step, I used PyPI's sodapy [2] to get the raw data. It is a Python library that provides a feature to collect the Socrata Open Data API [3] with a private token and transit data by API call.

Secondly, to present the data correctly, I needed to map the data format the same as the one on AWS Elasticsearch. This part is relatively simple, such as changing a string into an integer with a few lines of code in Python.

Finally, I used PyPI's elasticsearch [4] to upload the transformed data to the Elasticsearch database. It will automatically run the HTML request to upload the data into elasticsearch and create indexes for the NoSql database. Users only need to provide the host URL and login information.

Transfer Process

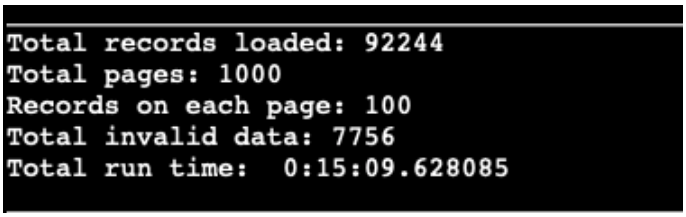
With all the environment set up, the project was moving on to the actual running process. The docker file, which contains the program, was built and run on the AWS EC2 terminal. In this phase, I was required to decide how many nodes to use, how many rows of data to include in each transaction, and how many containers to run simultaneously.

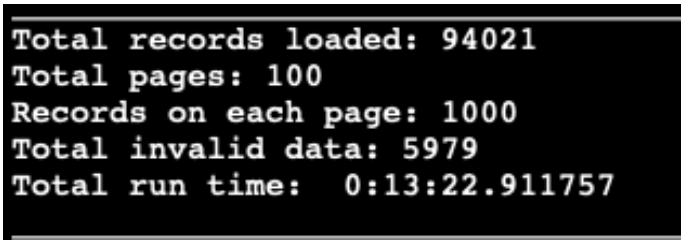
The nodes are related to the quantity of traffic and the capacity for the datasets. I decided to use ten nodes in this project, and it runs pretty well for around 1MM of data. It takes around three and a half hours to load 1,700,000 rows of data and use up for approximately 80% of capacity.

Volume

As for the volume for each transaction, I set up two arguments for the program, respectively num_pages for how many rows to include in one page, and page_size for how many pages to load in one transaction. The total data in one transaction would be num_pages * page_size. I tested different combinations in order to find the most efficient ways to load the data. The following are the reports for two of the times I tested:

I tried 100 num_pages and 1000 page_size in the first test, and the total loading time was around 15 minutes. Conversely, I tried 1000 num_pages and 100 page_size in the second test, and the total loading time reduced to approximately 13 and a half minutes.





In addition to these two tests, I loaded 1 million data in another transaction. It takes around two and a half hours to complete. However, the terminal has timed out, and I was unable to capture the report. Since 100k data takes around 15 minutes and 1MM data takes approximately 2.5 hours to complete, I discovered that the volume of data in one transaction did not significantly affect the loading time.

Maximize efficiency

To increase the effectiveness, I tested running multiple docker files at the same time. As such, I tried to run three docker files in different sections simultaneously, and it performed very well. I was able to upload 300K data (100K per section) within 15 minutes. Unfortunately, when I increased to running ten sections simultaneously, the whole thing crashed, and I could not continue the process. Luckily at that time, I have already received enough data for analysis purposes, and could move on to the next step.

Summary

In sum, I would conclude that the most efficient way of using my program is to upload 100K rows of data per transaction, with 1000 records per page and load 100 pages, and run up to three sections simultaneously to receive the maximum effectiveness. I will continue to test and update the methodology with more research in the future.

---

[2]: Sodapy Python Library https://pypi.org/project/sodapy/

[3]: Socrate Open Data API https://dev.socrata.com

[4]: elasticsearch Python Library https://pypi.org/project/elasticsearch/

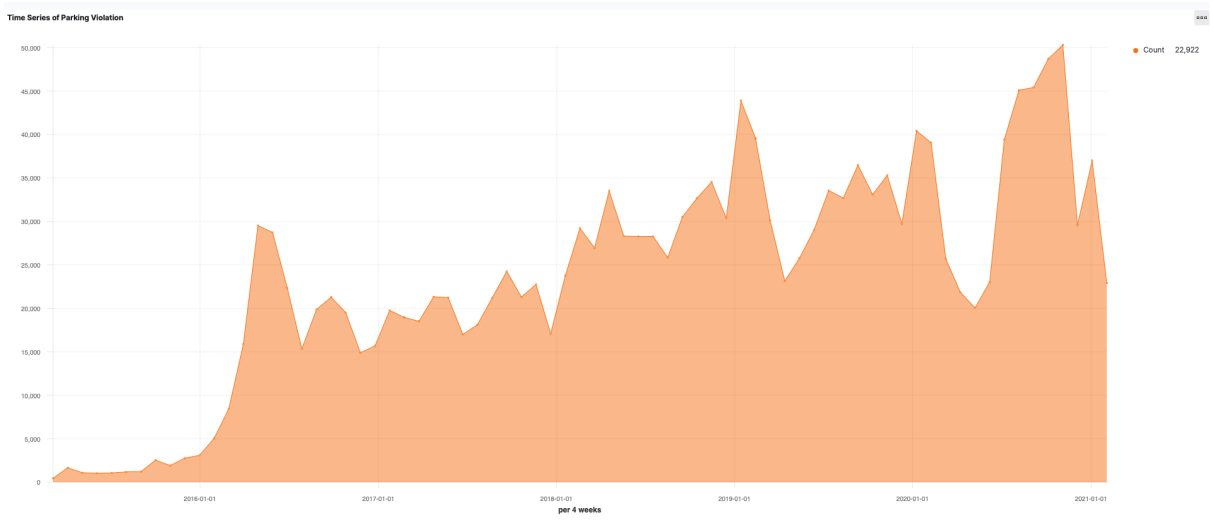
Analysis

The Data

The Open Parking and Camera Violations (OPCV) dataset was initially loaded with parking violations in the mainframe database from May 2016 [5]. It contained information about the violation's Plate, State, License Type, Summons Number, Issue Date, Violation Time, and description of the violation. The total rows of records from May 2016 to March 2021 was 62.3M. In this project, I randomly sampled 1.7M valid data for analysis.

Time series

The below figure shows the time series of OPCV from 1th Jan 2015 to 13th March 2021. As the data description mentioned, the dataset started actively recording in 2016. Therefore, I will not provide any analysis before that time.

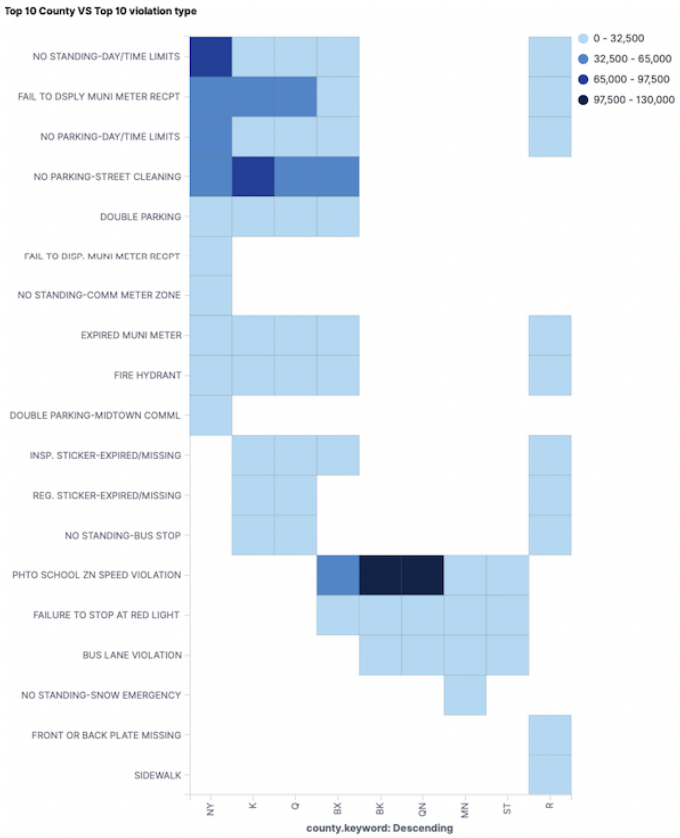


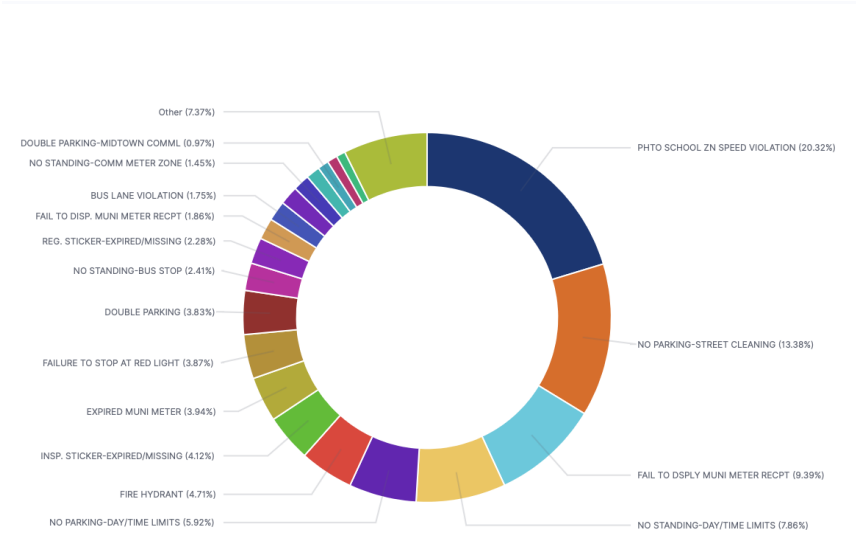
We can see that the overall trend is increasing from 2016 to 2021. However, it significantly decreased in May 2020. One of the reasons for this phenomenon could be the Covid-19 issue in 2020, which caused a new lockdown policy at New York State that month. We may speculate that when lesser people were going out, less parking violations happened.

Violation Type by County

This figure shows the relation between County and Violation Type. I only included the top 10 Counties and top 10 violation types in this dataset to receive more insight. It has the county presented on X-axis and the Violation type shown on the Y-axis.

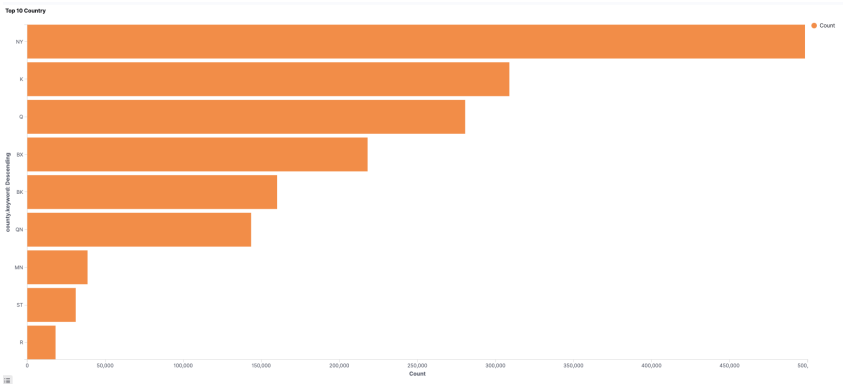
In New York (NY) county, the top violation type is No-standing Day/Time limit, Followed by failing to display parking receipt and no-parking day/time limit. We can observe that in NY county, the majority of violations violated the day/time limit. The reason would be when it is not easy to find a parking lot in a city/business district, people are more likely to park wherever possible. Based on this finding, maybe more parking lots are needed in this county.





The second interesting observation is the top violation type was the school speed violation in county BK and QN, in which both have recorded around 100k violations. One can say that most schools are located in these two counties. Coincidentally, this violation happens the most in this data set, as shown in the above pie chart. I would suggest that the related department could build more signs in these two areas to notice the speed limit.

This dataset's second high violation type is No-parking for street cleaning. When we look at the above heatmap, this violation type mainly happened in county K, Q, BK and NY. Similarly, I believe the related department could give more notices when cleaning the street to reduce violation.



Top 10 county

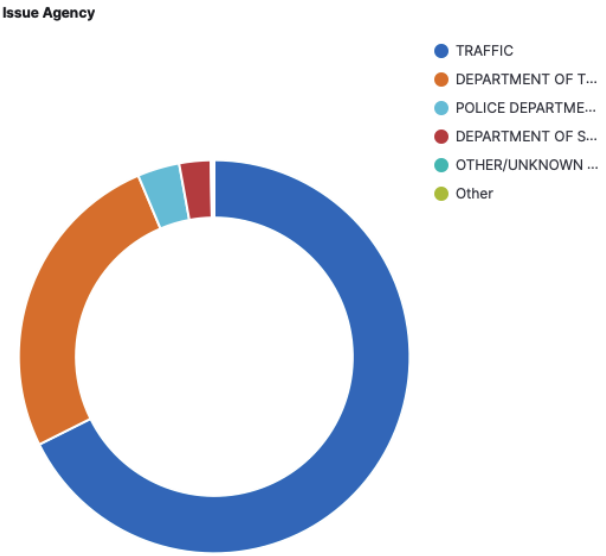
Most of the parking violations recorded in the OPVC dataset is in NY county. Therefore, this analysis was mainly focused in this county.

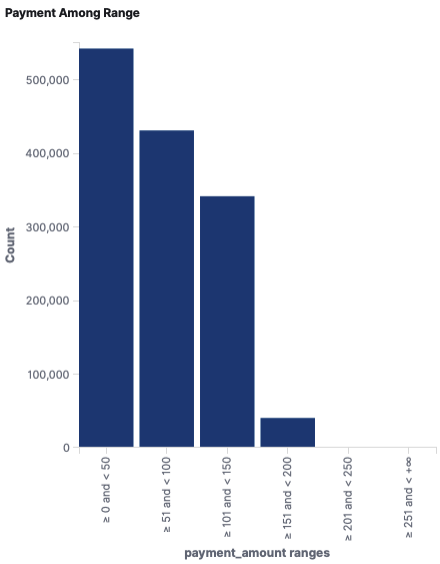
Time-Frequency

In this figure, we can see that most violations happened around 8 am - 12 pm. One possible reason for this phenomenon is that the traffic was heavier at this period, and drivers were possibly rushing to work and easy to ignore parking restrictions.

Issue Agency

This chart shows the ratio of the agency whose issue the parking violation ticket to drivers. We can see that most of the tickets are issued by the Traffic department.



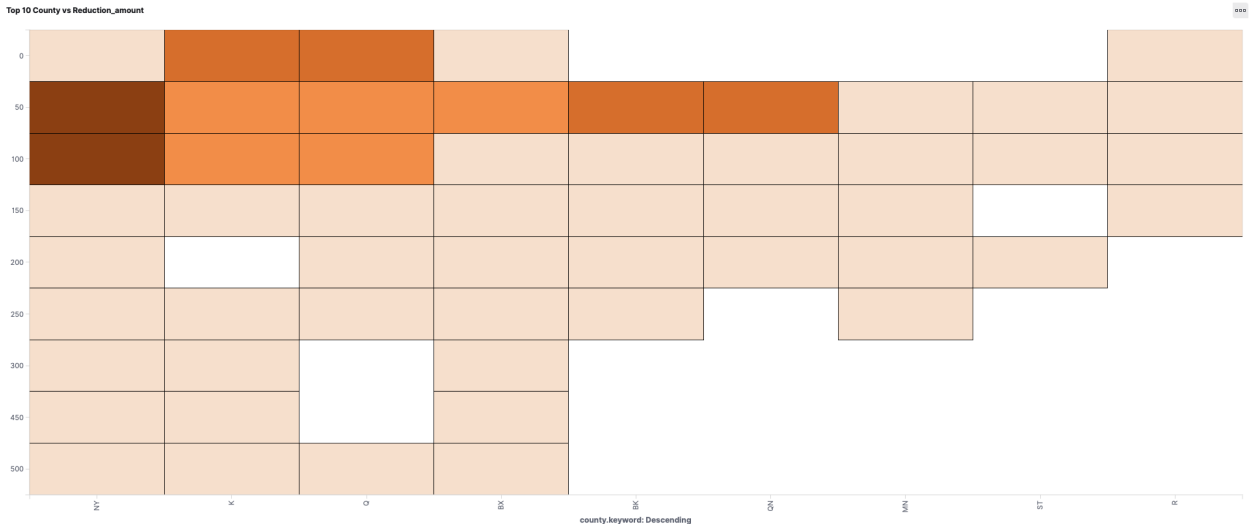


Penalty payment Amount

This chart is aimed to examine the distribution of penalty payment amount. We can see a high right-skewed distribution in the chart, the majority of violation penalties are under $50, with almost no data falling above $200.

Reduction amount by County

In this chart, I was exploring the reduction amount by county. Usually, the main reason for a reduction in parking violation is the parking ticket containing incorrect information, including lack of appropriate signage and wrong plate number etc.[6]



As the chart shows, in the NY county, it is ubiquitous to have a reduction with an amount between $50-$100. It raised another question, is it common to have an incorrect parking ticket in this county? However, I cannot conclude the causation between the deduction and the incorrect parking ticket. I can only say that if next time you have received a violation ticket in this area, try to figure out a reduction with the related department because you are not the only one.

---

[5]: Open Parking and camera violations Data Set
https://dev.socrata.com/foundry/data.cityofnewyork.us/nc67-uf89
[6]: How can I get A Parking Ticket dismissed or reduced?
https://www.ezticketsweb.com/articles.html

Conclusion

In sum, these findings could benefit drivers in NYC to improve decision-making when finding a spot to park their car and understand NYC parking violations more. I would recommend drivers be aware of the Day/time limit and be extra cautious when parking a vehicle in the morning.

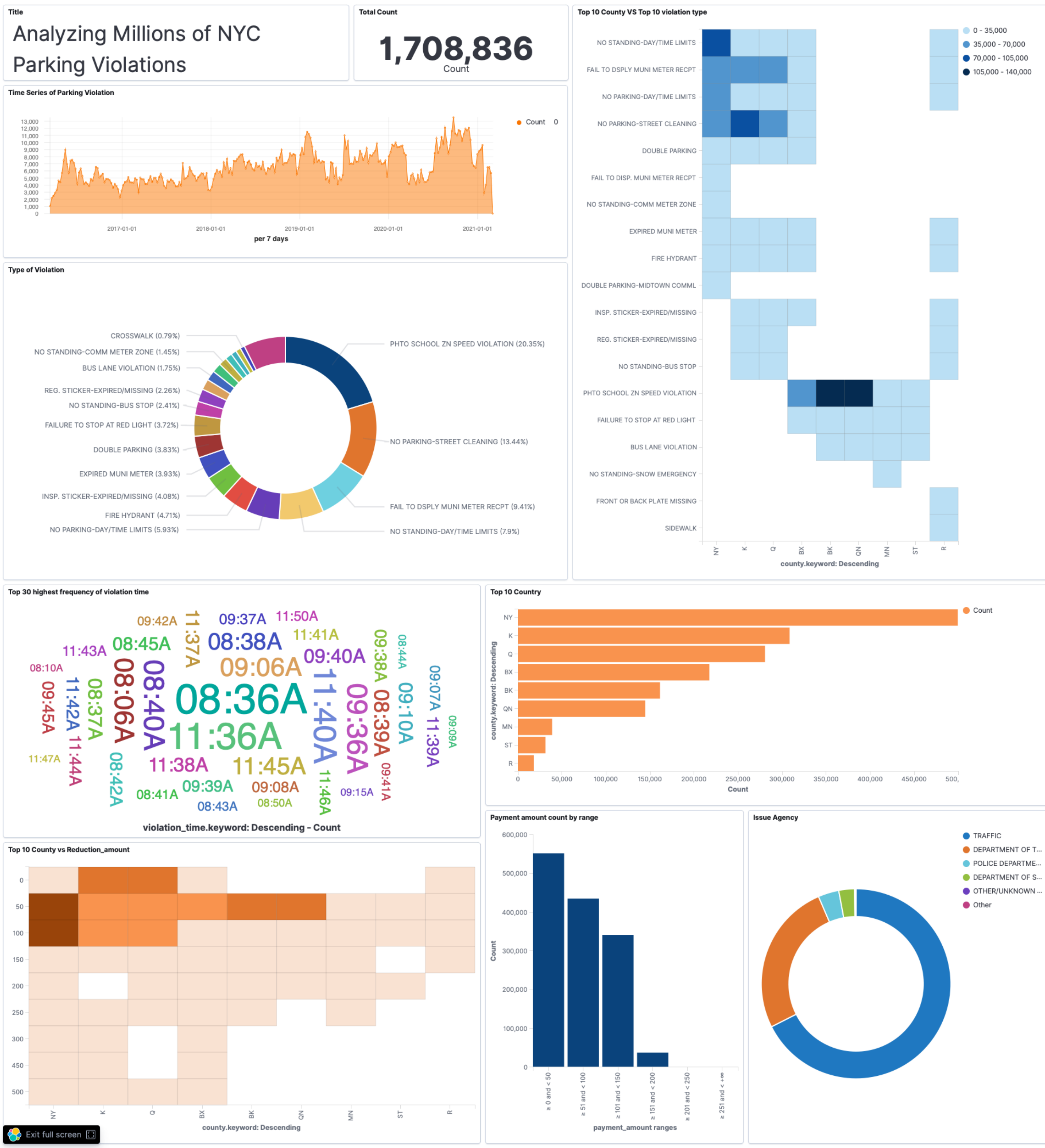
It would also aid law enforcement in analyzing the current issue in parking violations and how to reduce its number. For instance, as I mentioned in the previous section, they could add more signs to regard people if there is a No-parking restriction. And notice people for not exceeding the school district's speed limit. It is not only about the parking violation and penalty, but also about the safety of people in New York City.

However, this analysis is far from utilized in the real world. The study should include more research to improve quality, like joining different datasets to widen views on this issue, visualize the data on a map to gain more insight, and build a statistical model for predictive and prescriptive analysis.

Appendix

This article includes two appendices. Respectively, a Dashboard including the total number of data used in this analysis and all related charts. And a sample report of extracted, loaded and transformed data from the OPCV to AWS Elasticsearch. It is useful for examining the loading time and success rate of processing data. It will auto-generate each time the program runs.

Dashboard


Title
Analyzing Millions of NYC Parking Violations
Total Count
1,708,836
Count
Top 10 County VS Top 10 violation type
NO STANDING-DAY/TIME LIMITS
FAIL TO DSPLY MUNI METER RECPT
NO PARKING-DAY/TIME LIMITS
NO PARKING-STREET CLEANING
DOUBLE PARKING
FAIL TO DISP. MUNI METER RECPT
NO STANDING-COMM METER ZONE
EXPIRED MUNI METER
FIRE HYDRANT
DOUBLE PARKING-MIDTOWN COMML
INSP. STICKER-EXPIRED/MISSING
REG. STICKER-EXPIRED/MISSING
NO STANDING-BUS STOP
PHTO SCHOOL ZN SPEED VIOLATION
FAILURE TO STOP AT RED LIGHT
BUS LANE VIOLATION
NO STANDING-SNOW EMERGENCY
FRONT OR BACK PLATE MISSING
SIDEWALK
0 - 35,000
35,000 - 70,000
70,000 - 105,000
105,000 - 140,000
NY K Q BX BK QN MN ST R
county.keyword: Descending
Time Series of Parking Violation
Count 0
2017-01-01 2018-01-01 2019-01-01 2020-01-01 2021-01-01
per 7 days
Type of Violation
CROSSWALK (0.79%)
NO STANDING-COMM METER ZONE (1.45%)
BUS LANE VIOLATION (1.75%)
REG. STICKER-EXPIRED/MISSING (2.26%)
NO STANDING-BUS STOP (2.41%)
FAILURE TO STOP AT RED LIGHT (3.72%)
DOUBLE PARKING (3.83%)
EXPIRED MUNI METER (3.93%)
INSP. STICKER-EXPIRED/MISSING (4.08%)
FIRE HYDRANT (4.71%)
NO PARKING-DAY/TIME LIMITS (5.93%)
PHTO SCHOOL ZN SPEED VIOLATION (20.35%)
NO PARKING-STREET CLEANING (13.44%)
FAIL TO DSPLY MUNI METER RECPT (9.41%)
NO STANDING-DAY/TIME LIMITS (7.9%)
Top 30 highest frequency of violation time
08:36A 11:36A 11:40A 09:36A 08:40A 08:06A 08:38A 09:06A 09:40A 08:45A 11:45A 11:38A
violation_time.keyword: Descending - Count
Top 10 Country
Count
NY K Q BX BK QN MN ST R
county.keyword: Descending
0 50,000 100,000 150,000 200,000 250,000 300,000 350,000 400,000 450,000 500,
Count
Top 10 County vs Reduction_amount
0 50 100 150 200 250 300 450 500
NY K Q BX BK QN MN ST R
county.keyword: Descending
Payment amount count by range
600,000 500,000 400,000 300,000 200,000 100,000 0
Count
≥ 0 and < 50
≥ 51 and < 100
≥ 101 and < 150
≥ 151 and < 200
≥ 201 and < 250
≥ 251 and < +∞
payment_amount ranges
Issue Agency
TRAFFIC
DEPARTMENT OF T...
POLICE DEPARTME...
DEPARTMENT OF S...
OTHER/UNKNOWN ...
Other
Exit full screen

Sample report

```
Page 59      Succes: 972   Skipped: 28    Run time: 0:00:08.336871
Page 60      Succes: 995   Skipped: 5     Run time: 0:00:07.679750
Page 61      Succes: 959   Skipped: 41    Run time: 0:00:07.607481
Page 62      Succes: 976   Skipped: 24    Run time: 0:00:07.736089
Page 63      Succes: 982   Skipped: 18    Run time: 0:00:07.836457
Page 64      Succes: 976   Skipped: 24    Run time: 0:00:07.299682
Page 65      Succes: 989   Skipped: 11    Run time: 0:00:07.675472
Page 66      Succes: 1000  Skipped: 0     Run time: 0:00:07.214222
Page 67      Succes: 999   Skipped: 1     Run time: 0:00:07.625333
Page 68      Succes: 998   Skipped: 2     Run time: 0:00:07.396912
Page 69      Succes: 982   Skipped: 18    Run time: 0:00:07.675889
Page 70      Succes: 1000  Skipped: 0     Run time: 0:00:07.233879
Page 71      Succes: 991   Skipped: 9     Run time: 0:00:07.317144
Page 72      Succes: 975   Skipped: 25    Run time: 0:00:07.120961
Page 73      Succes: 999   Skipped: 1     Run time: 0:00:07.297628
Page 74      Succes: 986   Skipped: 14    Run time: 0:00:07.092129
Page 75      Succes: 927   Skipped: 73    Run time: 0:00:06.841894
Page 76      Succes: 950   Skipped: 50    Run time: 0:00:07.295339
Page 77      Succes: 959   Skipped: 41    Run time: 0:00:06.976603
Page 78      Succes: 934   Skipped: 66    Run time: 0:00:06.855175
Page 79      Succes: 949   Skipped: 51    Run time: 0:00:06.827688
Page 80      Succes: 958   Skipped: 42    Run time: 0:00:07.947465
Page 81      Succes: 982   Skipped: 18    Run time: 0:00:07.580391
Page 82      Succes: 996   Skipped: 4     Run time: 0:00:07.389641
Page 83      Succes: 944   Skipped: 56    Run time: 0:00:06.816563
Page 84      Succes: 963   Skipped: 37    Run time: 0:00:07.590178
Page 85      Succes: 908   Skipped: 92    Run time: 0:00:06.991816
Page 86      Succes: 1000  Skipped: 0     Run time: 0:00:07.315112
Page 87      Succes: 992   Skipped: 8     Run time: 0:00:07.241224
Page 88      Succes: 881   Skipped: 119   Run time: 0:00:07.347650
Page 89      Succes: 929   Skipped: 71    Run time: 0:00:07.130730
Page 90      Succes: 993   Skipped: 7     Run time: 0:00:07.630386
Page 91      Succes: 964   Skipped: 36    Run time: 0:00:07.921570
Page 92      Succes: 941   Skipped: 59    Run time: 0:00:08.039679
Page 93      Succes: 948   Skipped: 52    Run time: 0:00:07.320418
Page 94      Succes: 956   Skipped: 44    Run time: 0:00:07.178900
Page 95      Succes: 1000  Skipped: 0     Run time: 0:00:07.858547
Page 96      Succes: 988   Skipped: 12    Run time: 0:00:08.079419
Page 97      Succes: 973   Skipped: 27    Run time: 0:00:07.666446
Page 98      Succes: 997   Skipped: 3     Run time: 0:00:08.342067
Page 99      Succes: 998   Skipped: 2     Run time: 0:00:10.410212
Page 100     Succes: 981   Skipped: 19    Run time: 0:00:08.191311

Total records loaded: 96411
Total pages: 100
Records on each page: 1000
Total invalid data: 3589
Total run time:  0:12:37.715161
```