Exploratory Analysis of Parking Violation

Rishi Bamb Sanjana Balagar Shrivatson Ramaratnam Giridharan Sonali Gupta

Abstract* — Big data is the result of technological advancements that have resulted in the advent of massive amounts of data. Big data refers to datasets that are not only large in size but also contain a lot of different types of data. When properly analyzed, these data can assist industries in making important decisions in a variety of ways. Parking violations are a daily problem in today's fast-paced environment. Parking a vehicle illegally may result in an offense, resulting in a large number of traffic citations being issued. 'Parking Violation' data is one such data set for our exploratory investigation. Every day, millions of automobiles are parked in cities, and New York, as a major metropolis, is no exception, with most residents having parking issues.New York City itself collected approx $957 million in fine revenues In them more than 59% that is approx $565 million of the $957 million, come from parking tickets. The analytics and visualizations are performed using various AWS Services.

I. Introduction

[1] In this project we have used the parking violation dataset from the open nyc and kaggle which contained more than 50 million records and performed various analytics to generate meaningful insights.The NYC Department of Finance collects data on every parking ticket issued in NYC and is responsible for collecting and processing payments of all tickets. Because of the huge number of cars and the limited geography, there are a lot of parking tickets. This prompted us to conduct an exploratory analysis on such data in order to gain insights such as when and where tickets are more likely to be issued, if there is a specific season for it, what types of vehicles are receiving tickets, comparing state data to determine which state has the most tickets issued on a monthly basis, and which vehicle body type receives the most tickets. This analysis is carried out utilizing different AWS services like the S3 for data storage, Redshift for performing queries and analytics , and QuickSight to perform dynamic visualizations with the goal of developing a graphical solution for real-time analytics using the parking dataset.

II. Motivation

The project's inspiration came from academic coursework.We decided to use the parking violation dataset for our project which would help in analysing the number of violations issued based on multiple factors.

The project will help us in exploring the big data concepts and tools to understand the parking ticketing system which will moreover help us to gain analytical insights from the available data and help the ticketing department and issuing agencies to analyze the regions where vehicles are getting the highest number of summons and the reasons. We have also seen the rise in the violations according to each fiscal year.By doing this project we will analyse the highest number of tickets raised in each state, the number of summons in each month, and the highest number of violation code committed by the citizens.

III. Literature Survey

A. Automatic Detection of Parking Violation and Capture of License Plate

Author Z. Liu, W. Chen and C. K. Yeo stated about the automatic detection of parking violations.Illegal parking is a common problem in urban areas, causing traffic congestion and posing a safety concern to other road users. Surveillance video is saved for post-event forensics, and detecting offending vehicles often necessitates manual review. Foreground and background segmentation, which is less resistant to environmental influences, as well as the more robust Single Shot MultiBox Detector (SSD) and its derivatives, are examples of automated detection systems. This study presents a fully automated pipeline for detecting illegal parking from start to finish. You Only Look Once Version 3 is a deep learning-based object detection system that provides reliable and rapid car detection (YOLOv3). The stationary time of the vehicle-in-violation is tracked using template matching and Intersection over Union (IoU) calculations, with built-in error tolerance measures. The license plate is extracted using OpenALPR. Under different conditions, empirical results reveal a high level of vehicle detection and movement tracking accuracy.

B. Applied research on Real estate price prediction by neural network

Authors N. Lin, E. Liu, F. Tenorio, X. Yang and D. Woodbridge wrote this work in which they highlighted using networked systems and unsupervised machine learning methods on a huge dataset, they investigated the similarities in parking ticket records in this study. They used an algorithm to cluster existing tickets and dug deeper to identify the distribution of precincts within various clusters using 37 million ticket data (9 GB) gathered by the New York City Department of Finance. This project made use of Amazon Web Services, such as S3, EC2, and EMR, as well as technologies like MongoDB and Apache Spark. The computational time and cost for various EMR settings were studied in this study. They found that when implementing unsupervised learning on a big dataset, as well as storing and managing data, distributed systems offer significant computing benefits. They also discovered that a cluster with more workers is faster than a cluster with fewer workers but more memory space for the data set in use. outcomes based on human behavior, decision-making, and a variety of other elements. Based on the pricing and controlling components, we can analyze and seek to evaluate the property categories that are a great match to the users in our project.

IV. *Proposed work*

In this project, we have obtained the dataset from Kaggle and NYC Open data platform. These data are preprocessed using Talend, an etl tool and passed into AWS services like S3 for storage and Redshift for effective analysis of it. And finally, the data is visualized using AWS QuickSight and matplotlib library in python, the resultant showcase the exploratory analysis of the parking violations.

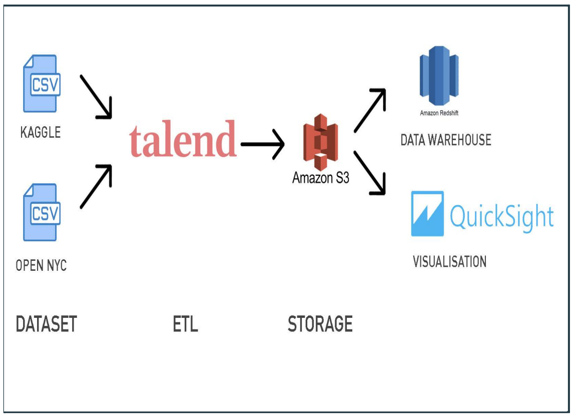
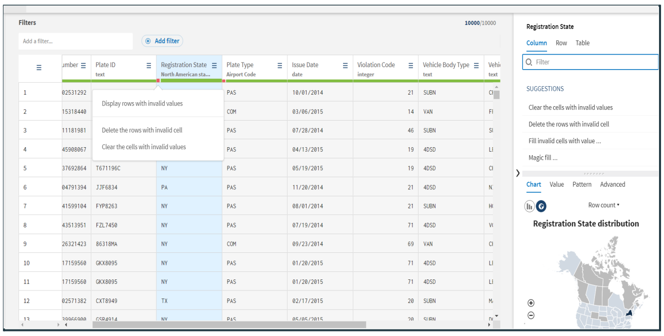


Figure.1

1. *Data Cleaning - Talend*

As we were having four fiscal years data from 2013 to 2017, we cleaned and normalized individual years files using Talend which provides feasible services to process and prepare data. Below diagram is a brief look of one of the normalized files.



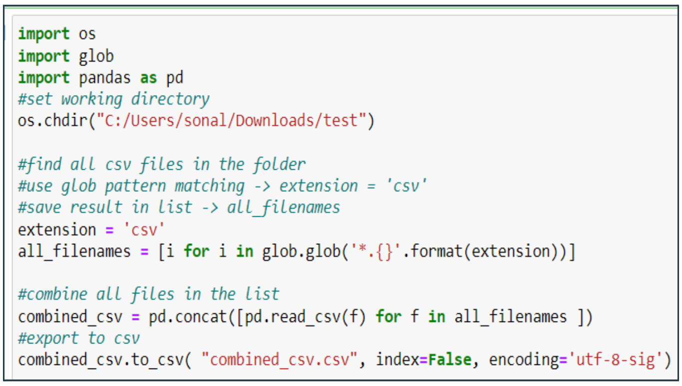
Filters
10000/10000
Add a filter...
Add filter
Display rows with invalid values
Delete the rows with invalid cell
Clear the cells with invalid values
Registration State
Column
Row
Table
Filter
SUGGESTIONS
Clear the cells with invalid values
Delete the rows with invalid cell
Fill invalid cells with value ...
Magic fill ...
Chart
Value
Pattern
Advanced
Row count
Registration State distribution

	umber	Plate ID text	Registration State North American sta...	Plate Type Airport Code	Issue Date date	Violation Code integer	Vehicle Body Type text
1	02531292			PAS	10/01/2014	21	SUBN
2	15318440			COM	03/06/2015	14	VAN
3	11181981			PAS	07/28/2014	46	SUBN
4	45908067			PAS	04/13/2015	19	4DSD
5	37692864	T671196C	NY	PAS	05/19/2015	19	4DSD
6	04791394	JJF6834	PA	PAS	11/20/2014	21	4DSD
7	41599104	FYP8263	NY	PAS	08/01/2014	21	SUBN
8	43513951	FZL7450	NY	PAS	07/19/2014	71	4DSD
9	26321423	86318MA	NY	COM	09/23/2014	69	VAN
10	17159560	GKX8095	NY	PAS	01/20/2015	71	4DSD
11	17159560	GKX8095	NY	PAS	01/20/2015	71	4DSD
12	02571382	CXT8949	TX	PAS	02/17/2015	20	SUBN
13	39966900	GSR4914	NY	PAS	05/05/2015	20	SUBN

Figure.2

2. *Merging of files*

To have data in a single file makes an analysis and working on it easy hence we merged all four files into one using python code.



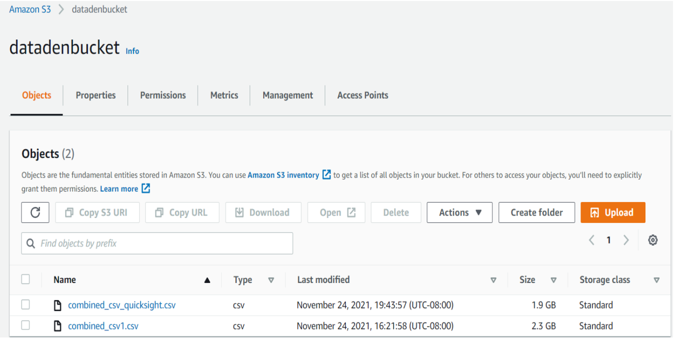
```
import os
import glob
import pandas as pd
#set working directory
os.chdir("C:/Users/sonal/Downloads/test")

#find all csv files in the folder
#use glob pattern matching -> extension = 'csv'
#save result in list -> all_filenames
extension = 'csv'
all_filenames = [i for i in glob.glob('*.{}'.format(extension))]

#combine all files in the list
combined_csv = pd.concat([pd.read_csv(f) for f in all_filenames ])
#export to csv
combined_csv.to_csv( "combined_csv.csv", index=False, encoding='utf-8-sig')
```

Figure 3

3. *Loading transformed data into AWS S3 bucket*



Objects (2)

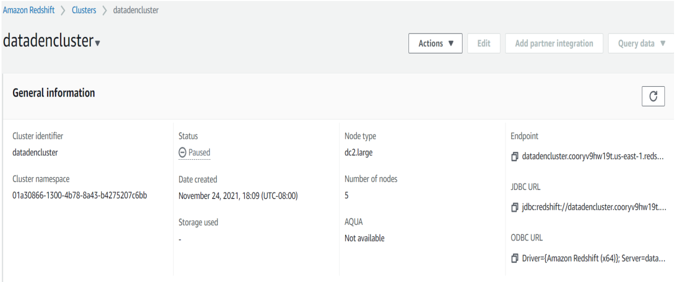
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

□	Name	Type	Last modified	Size	Storage class
□	combined_csv_quicksight.csv	csv	November 24, 2021, 19:43:57 (UTC-08:00)	1.9 GB	Standard
□	combined_csv1.csv	csv	November 24, 2021, 16:21:58 (UTC-08:00)	2.3 GB	Standard

Figure. 4

3. *AWS Redshift*

Created cluster on redshift service by providing access to user through an associated IAM role to perform analysis using its query editor.



datadencluster

General information

Cluster identifier
datadencluster

Cluster namespace
01a30866-1300-4b78-8a43-b4275207c6bb

Status
Paused

Date created
November 24, 2021, 18:09 (UTC-08:00)

Storage used
-

Node type
dc2.large

Number of nodes
5

AQUA
Not available

Endpoint
datadencluster.cooryv9hw19t.us-east-1.reds...

JDBC URL
jdbc:redshift://datadencluster.cooryv9hw19t....

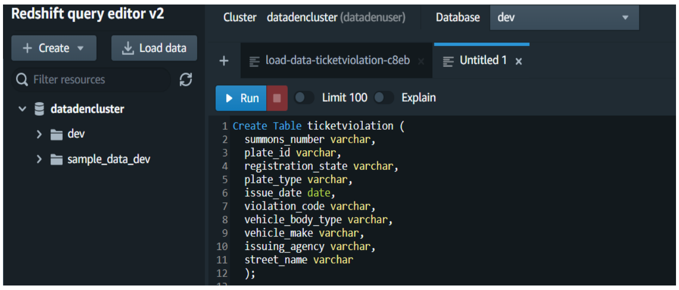
ODBC URL
Driver={Amazon Redshift (x64)}; Server=data...

Figure. 5

4. Analysis - Redshift Query editor

First we have created a ticket violation table including all required attributes with metadata in a database. Then, loaded data into it from a stored csv file in s3 bucket to perform queries and analysis. The following analysis helps in giving the insights on multiple factors to explore trend and number of summons being issued in four consecutive years.

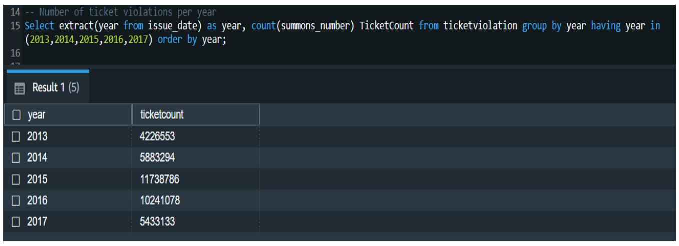
Creation of parking violations table:



```
Create Table ticketviolation (
  summons_number varchar,
  plate_id varchar,
  registration_state varchar,
  plate_type varchar,
  issue_date date,
  violation_code varchar,
  vehicle_body_type varchar,
  vehicle_make varchar,
  issuing_agency varchar,
  street_name varchar
  );
```

Figure. 6

Number of ticket violations per year:



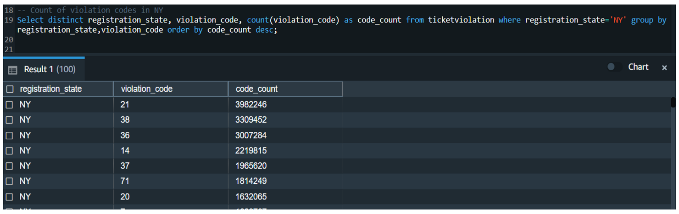
```
-- Number of ticket violations per year
Select extract(year from issue_date) as year, count(summons_number) TicketCount from ticketviolation group by year having year in
(2013,2014,2015,2016,2017) order by year;
```

Result 1 (5)

year	ticketcount
2013	4226553
2014	5883294
2015	11738786
2016	10241078
2017	5433133

Figure. 7

Count of violation in NY:



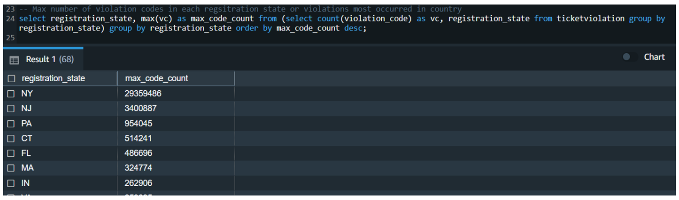
```
-- Count of violation codes in NY
Select distinct registration_state, violation_code, count(violation_code) as code_count from ticketviolation where registration_state='NY' group by
registration_state,violation_code order by code_count desc;
```

Result 1 (100)

registration_state	violation_code	code_count
NY	21	3982246
NY	38	3309452
NY	36	3007284
NY	14	2219815
NY	37	1965620
NY	71	1814249
NY	20	1632065

Figure.8

Max number of violation codes in each registration state or violations most occurred in country:



```
-- Max number of violation codes in each regsitration state or violations most occurred in country
select registration_state, max(vc) as max_code_count from (select count(violation_code) as vc, registration_state from ticketviolation group by
registration_state) group by registration_state order by max_code_count desc;
```

Result 1 (68)

registration_state	max_code_count
NY	29359486
NJ	3400887
PA	954045
CT	514241
FL	486696
MA	324774
IN	262906

Figure.9

V. Data Visualisation

In our project we will be showcasing visualizations on AWS Quicksight and python. Quicksight's analytic platform empowers any skill level target audience to work with data through actionable and insightful visualizations. Below are the screenshots of visual analysis done on Quicksight.We have done visualisations using quicksight where we loaded a JSON manifest file via S3 bucket into quicksight for performing visualisations.

The first visualization is a pie chart which shows the top agencies which are issuing the tickets.· As we can see, the majority of the tickets are issued by T which is the traffic department and it Is obvious since New York is such a populated city and Traffic jams are normal on a day to day basis.The next agency issuing the maximum violation is Transportation department (V).

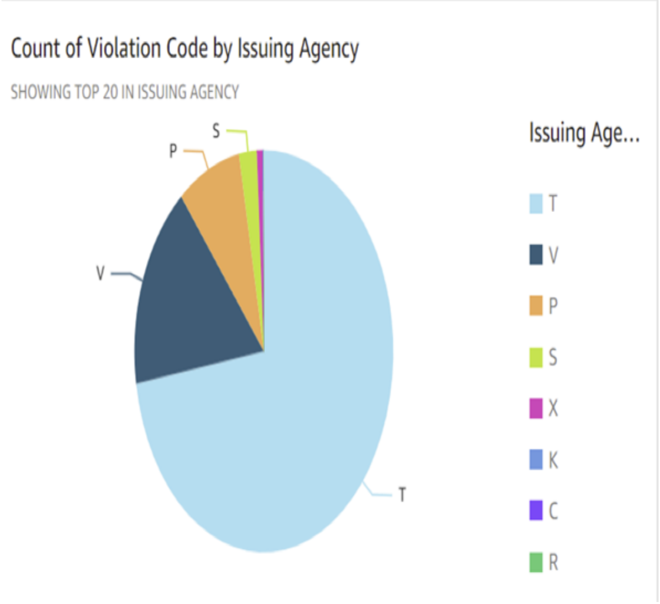


Figure.10

The next bar chart depicts the different violations done by a particular vehicle. From the visualization we can see that ford is the highest and then mg and so on.

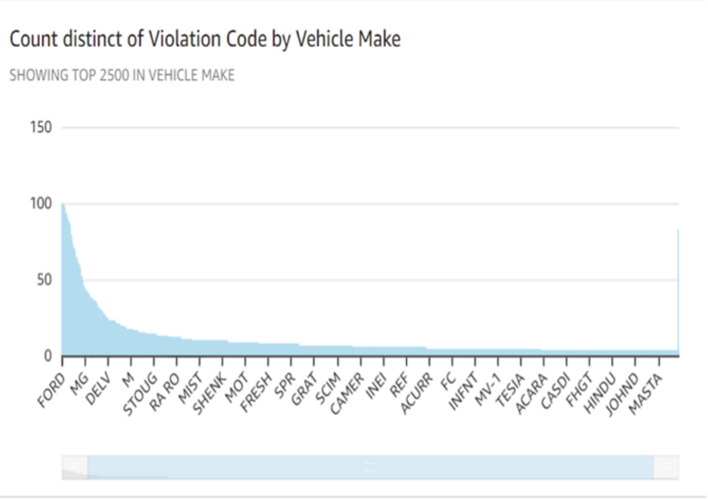


Figure.11

The pie chart shows the plate type which has the maximum violations and from this we can see that plate type PAS has the most number of tickets.

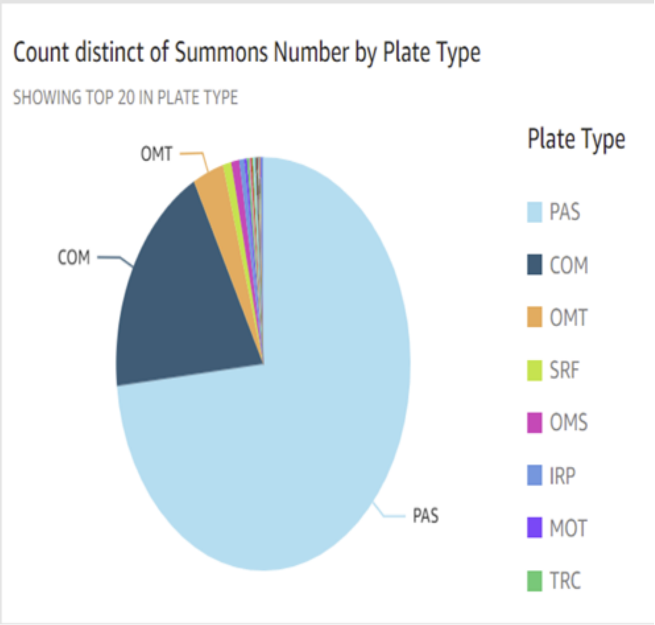


Figure.12

The next geospatial chart shows the count of summons numbers according to the registration state. Here too we can infer that out of North America most of the violations are done in New York and New Jersey being the second.

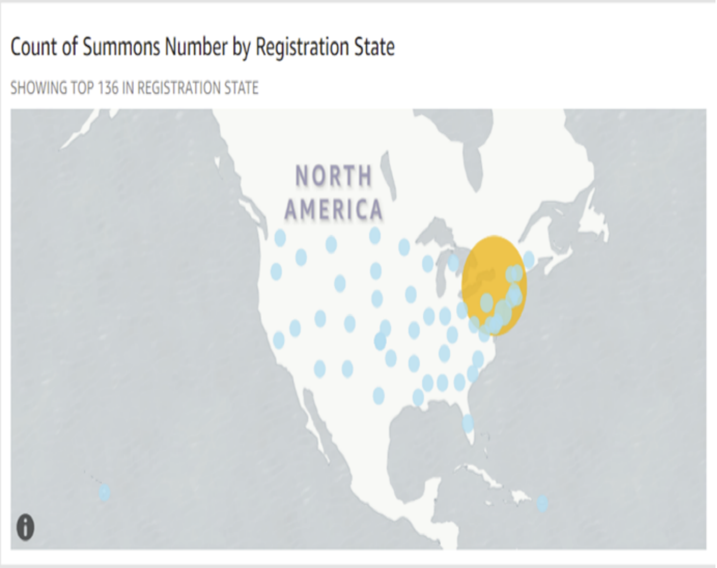

Count of Summons Number by Registration State
SHOWING TOP 136 IN REGISTRATION STATE
NORTH
AMERICA

Figure.13

The pie chart shows that manufacturer ford has got the maximum number of summons with Toyota being the second.

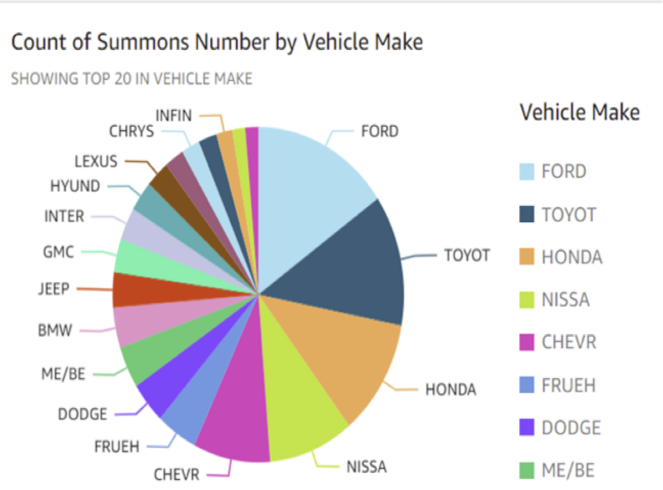


Figure.14

The word chart shows that New York has got the most number of violation code per registration state by looking at the size of NY when compared with other word sizes.

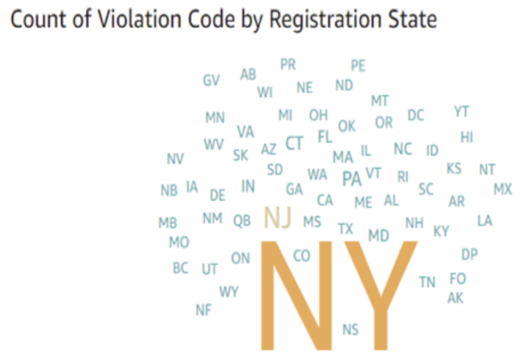


Figure.15

The next donor chart shows the number of summons that have been issued according to the vehicle type or the body type . Here we can infer that suburbans contribute to a major part about 35% . We analysed that since suburbans are large in size and they are difficult to park in a street setting it has the highest number of summons issued.

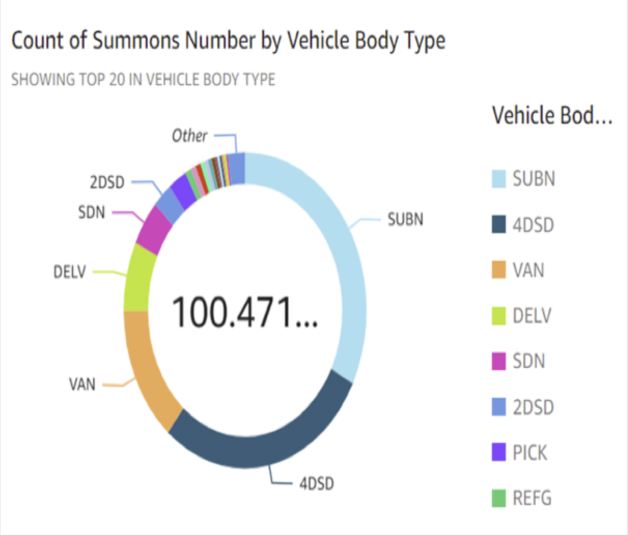


Figure.16

The next bar chart shows the street which has the maximum number of summons by street name . Thus broadway street in New York city has the maximum number of violations due to the fact that it has large number of theatres , pubs , restaurants etc which aid to the violations.

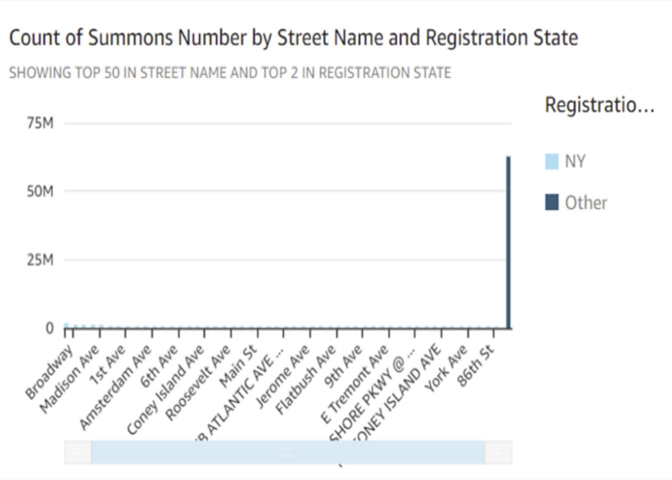
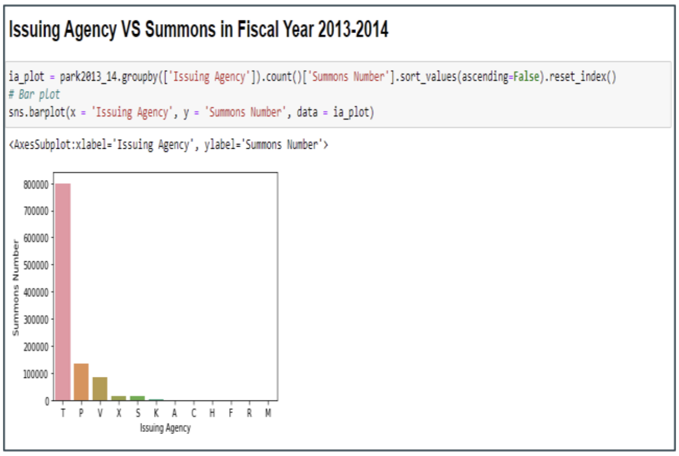


Figure.17

Understanding of data is easy when we place it in a visual context. Python offers great libraries and features to represent data graphically. In our project we are generating visualizations using matplotlib. Below are the screenshots of visualizations generated in python.

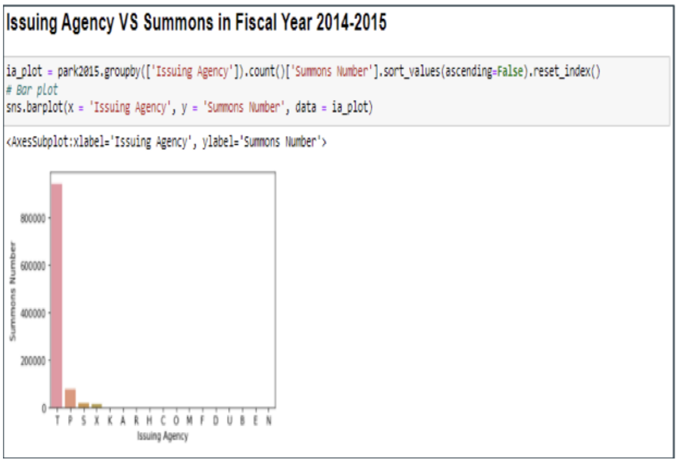


Issuing Agency VS Summons in Fiscal Year 2013-2014

```
ia_plot = park2013_14.groupby(['Issuing Agency']).count()['Summons Number'].sort_values(ascending=False).reset_index()
# Bar plot
sns.barplot(x = 'Issuing Agency', y = 'Summons Number', data = ia_plot)
```

```
<AxesSubplot:xlabel='Issuing Agency', ylabel='Summons Number'>
```

Figure.18

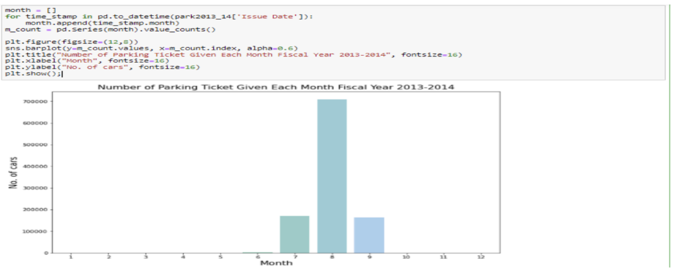


Issuing Agency VS Summons in Fiscal Year 2014-2015

```
ia_plot = park2015.groupby(['Issuing Agency']).count()['Summons Number'].sort_values(ascending=False).reset_index()
# Bar plot
sns.barplot(x = 'Issuing Agency', y = 'Summons Number', data = ia_plot)
```

```
<AxesSubplot:xlabel='Issuing Agency', ylabel='Summons Number'>
```

The above bar chart obtained from python shows the number of summons issued by each issuing agency from 2013 - 2015.

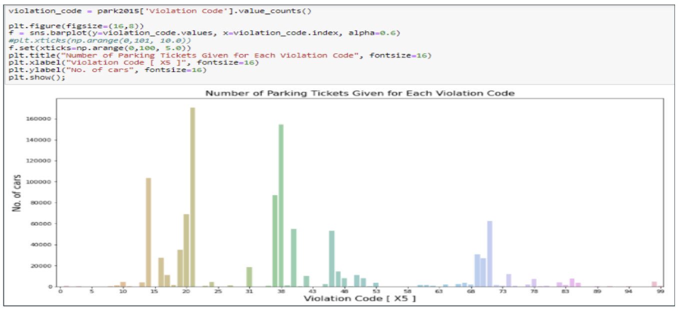


```
month = []
for time_stamp in pd.to_datetime(park2013_14['Issue Date']):
    month.append(time_stamp.month)
m_count = pd.Series(month).value_counts()

plt.figure(figsize=(12,8))
sns.barplot(y=m_count.values, x=m_count.index, alpha=0.6)
plt.title("Number of Parking Ticket Given Each Month Fiscal Year 2013-2014", fontsize=16)
plt.xlabel("Month", fontsize=16)
plt.ylabel("No. of cars", fontsize=16)
plt.show();
```

Figure.19

The above bar chart shows the number of tickets raised in each month in the year 2013 - 2014 here we can see that the peak number of tickets starts from the month of june and goes all the way till september.



```
violation_code = park2015['Violation Code'].value_counts()

plt.figure(figsize=(16,8))
f = sns.barplot(y=violation_code.values, x=violation_code.index, alpha=0.6)
#plt.xticks(np.arange(0,101, 10.0))
f.set(xticks=np.arange(0,100, 5.0))
plt.title("Number of Parking Tickets Given for Each Violation Code", fontsize=16)
plt.xlabel("Violation Code [ X5 ]", fontsize=16)
plt.ylabel("No. of cars", fontsize=16)
plt.show();
```

Figure.20

The above chart shows the number of tickets raised for each violation code.

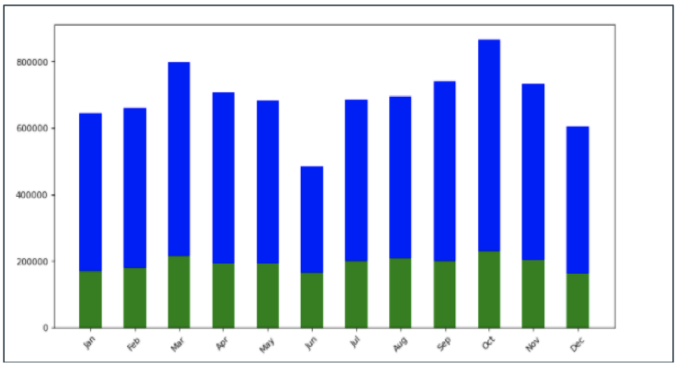


Figure.21

The above stacked bar chart shows the number of tickets raised in New York versus the other states in each month for the year 2015 - 2016. Here Blue represents New York and Green represents other states.

IV. Conclusion

We were able to study this large volume of parking dataset released between 2013 and 2017 which consisted of more than 50M records.We performed analytics using quicksight and the visualizations shows that New York gets the highest number of tickets among all the states in United States and in new york on the broadwalk street there are highest number of summons. From our analysis it can also be concluded that in the month of august there are a high number of violations issued as in summers there are many events. It can also be observed that in each month more tickets are raised in New York than in other states.

References

[1] Z. Liu, W. Chen and C. K. Yeo, "Automatic Detection of Parking Violation and Capture of License Plate," 2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON), 2019, pp. 0495-0500, doi: 10.1109/IEMCON.2019.8936164.

[2] Lin, N., Liu, E., Tenorio, F., Yang, X. and Woodbridge, D., 2019, August. Distributed Data Analytics Framework for Cluster Analysis of Parking Violation. In 2019 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and SmartCityInnovation(SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI) (pp. 1958-1963). IEEE.