

NYPD Crime Complaint Analysis

[Course code : INFO I-535]

Raksha Rank

I. INTRODUCTION

In this project, I am working on the NYPD public dataset. It includes all valid felony, misdemeanor, and violation crimes reported to the New York City Police Department (NYPD) from 2006 to 2021. Each record represents an arrest affected in NYC by the NYPD and includes information about the type of crime, the location and time of enforcement. In addition, information related to suspect and victim demographics is also included. It would help us explore the nature of police enforcement activity. Furthermore, from the past historical data trends, we can gain insights on the most susceptible to crime locality, types of crimes most likely to take place around which location.

II. BACKGROUND

The NYPD criminal dataset is a public dataset which is annually updated. It consists of 7.38M records and 35 features where each row represents a column.

It becomes important to address and analyse this dataset to flag out places that are more crime prone, and take appropriate actions and precautions for the safety of the citizens.

Data Schema:

root

```
|-- :@computed_region_92fq_4b7q: string (nullable = true)
|-- :@computed_region_efsh_h5xi: string (nullable = true)
|-- :@computed_region_f5dn_yrer: string (nullable = true)
|-- :@computed_region_sbqj_enih: string (nullable = true)
|-- :@computed_region_yeji_bk3q: string (nullable = true)
|-- addr_pct_cd: string (nullable = true)
|-- boro_nm: string (nullable = true)
|-- cmplt_fr_dt: string (nullable = true)
|-- cmplt_fr_tm: string (nullable = true)
|-- cmplt_num: string (nullable = true)
|-- cmplt_to_dt: string (nullable = true)
|-- cmplt_to_tm: string (nullable = true)
```

```
|-- crm_atpt_cptd_cd: string (nullable = true)
|-- hadevelop: string (nullable = true)
|-- housing_psa: string (nullable = true)
|-- juris_desc: string (nullable = true)
|-- jurisdiction_code: string (nullable = true)
|-- ky_cd: string (nullable = true)
|-- lat_lon: struct (nullable = true)
|  |-- latitude: string (nullable = true)
|  |-- longitude: string (nullable = true)
|-- latitude: string (nullable = true)
|-- law_cat_cd: string (nullable = true)
|-- loc_of_occur_desc: string (nullable = true)
|-- longitude: string (nullable = true)
|-- ofns_desc: string (nullable = true)
|-- parks_nm: string (nullable = true)
|-- patrol_boro: string (nullable = true)
|-- pd_cd: string (nullable = true)
|-- pd_desc: string (nullable = true)
|-- prem_typ_desc: string (nullable = true)
|-- rpt_dt: string (nullable = true)
|-- station_name: string (nullable = true)
|-- susp_age_group: string (nullable = true)
|-- susp_race: string (nullable = true)
|-- susp_sex: string (nullable = true)
|-- transit_district: string (nullable = true)
|-- vic_age_group: string (nullable = true)
|-- vic_race: string (nullable = true)
|-- vic_sex: string (nullable = true)
|-- x_coord_cd: string (nullable = true)
|-- y_coord_cd: string (nullable = true)
```

III. METHODOLOGY

In this project, I create a Virtual Machine instance of “Pyspark-Ubuntu 18.04-instance” in Jetstream.

➤ Why Jetstream?

Jetstream is a cloud computing environment designed to provide Extreme Science and Engineering Discovery Environment (XSEDE)

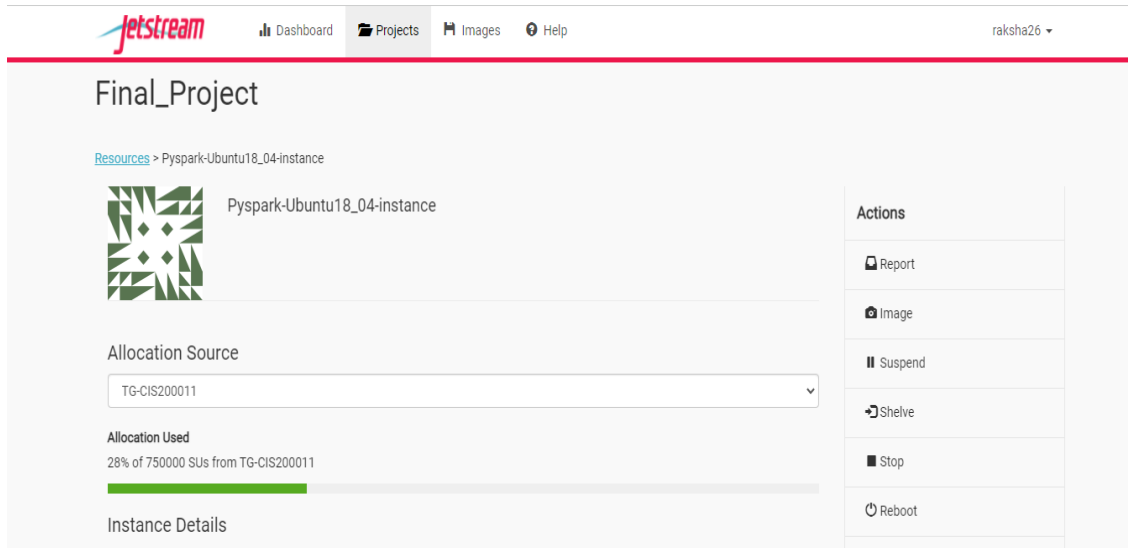
researchers and students user-friendly, on-demand access to interactive computing and data analysis resources. It is very similar to Google Cloud Platform as it provides a platform for launching Virtual Machine. Jetstream is more focused towards interactive research and small scale and has a very intuitive GUI which makes complex tasks easier to access. It also allows prototyping on a smaller scale and ships it to larger scale HCP post validation.

➤ **Why PySpark?**

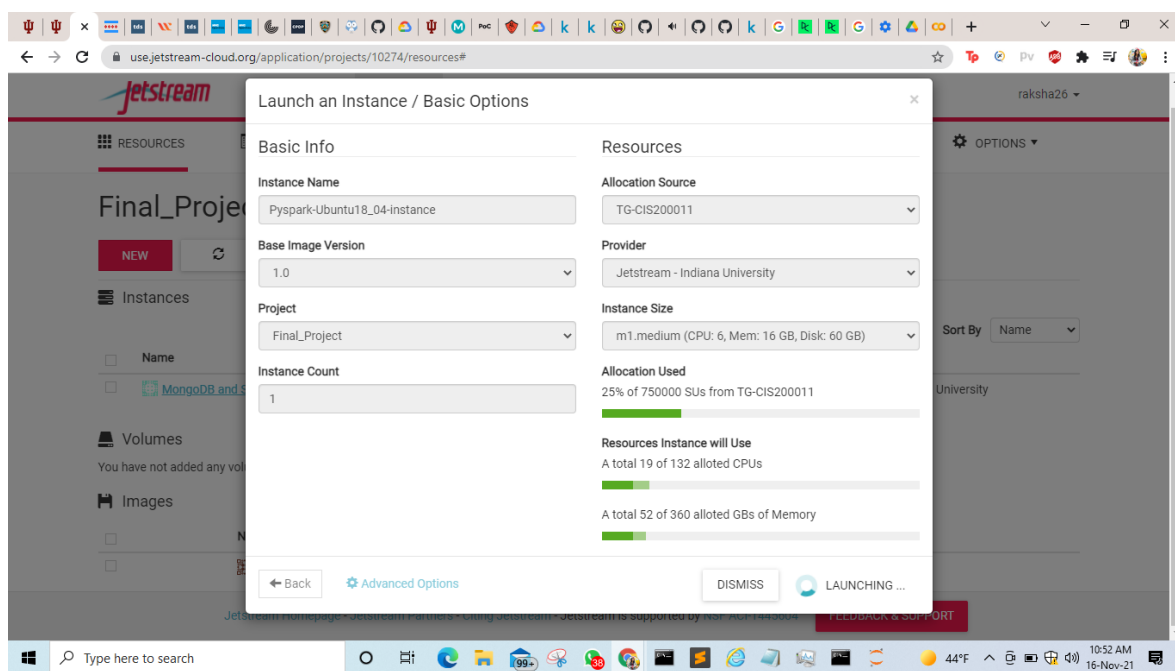
Spark is a general-purpose, in-memory, distributed processing engine that allows you to process your data efficiently in a distributed fashion. It is the most sought after big data processing platform, providing capability to process data on the petabyte scale. PySpark is one of the supported languages by Spark. PySpark allows us to write spark applications using python to process data and run it on Spark platform. We can use various cloud services to run spark jobs, AWS provides managed EMR spark platform, we can run Spark jobs in Google Cloud Platform by submitting spark jobs in DataProc or creating a spark instance in Jetstream. PySpark gives us the flexibility to read data in various formats like csv, parquet, json or from databases.

➤ **Setting up PySpark instance in Jetstream**

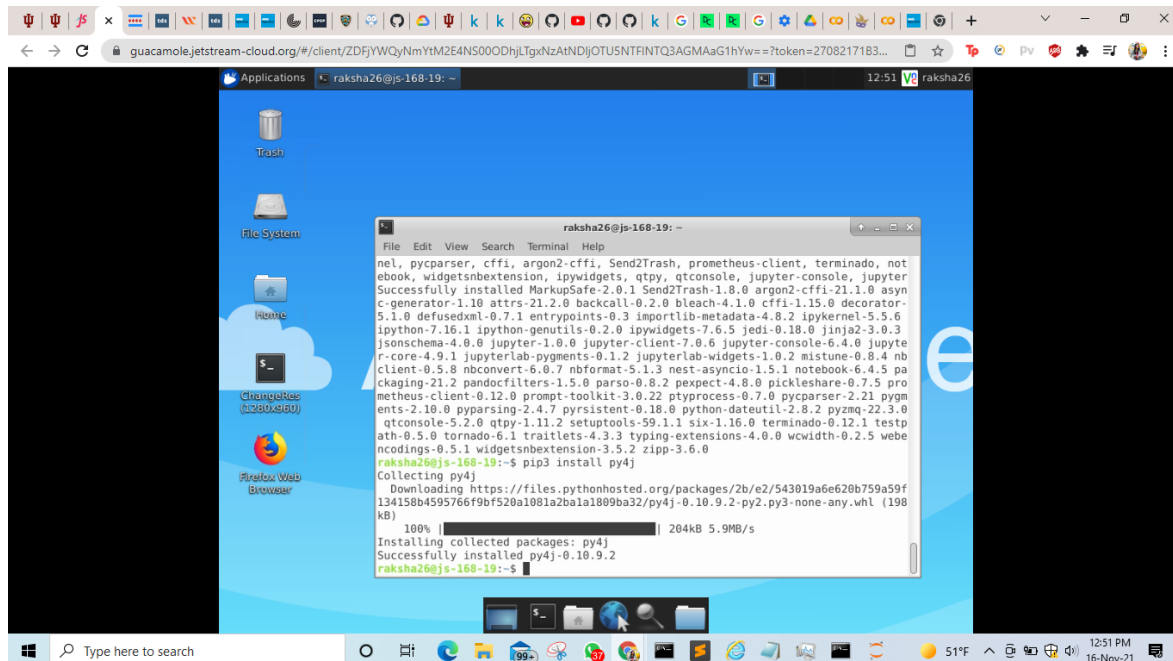
- To create a PySpark instance, we first create our new Project named FinalProject and find an image named “Pyspark-Ubuntu 18.04-instance”. This image has an up-to-date PySpark setup.
- Further, I launched an instance and created a medium sized VM. As soon as the instance is up and running, the instance status becomes Active.
- To work on a Jupyter notebook and run PySpark and Python code, launch the instance in Web Desktop mode. To verify if Spark installation is there, navigate to /opt/spark directory where Spark would be listed. After all the setup, open jupyter notebook in the running instance.



[Fig. Instance generation inside our project of interest]



[Fig. Instance configurations]



[Fig. Web Desktop version of Jetstream]

➤ PySpark SQL

- I fetch the NYPD data using the public API (<https://data.cityofnewyork.us/resource/qgea-i56i.json>) in JSON format. I load the data using RDD in Spark.

➔ Why RDD?

We use RDD which means Resilient Distributed Dataset, where Resilient means that it is fault tolerant, Distributed means it is distributed across many systems and Dataset means it contains some information in it. RDD helps in handling huge **Volumes** of data as it uses the model of distributed computing. RDD can be distributed across multiple systems and thus can handle huge volumes of data processing owing to parallel computing rather than performing it on a single system.

Data is generated with tremendous speed right now in every domain. Thanks to RDDs for being resilient in handling fault tolerance and keeping the data consistent. It would regenerate itself in runtime and never break down. With the high **Velocity** of data, owing to such resilience keeps the data available which is a huge perk.

Spark functions with the help of RDDs allow us to process different **Variety** of data and handle different dependencies. Distributed systems make it easier to process a vast variety of data. Map reduce returns key value pairs by breaking down the data. With summarization and aggregate operations, we can save a lot of processing time along with keeping the consistency intact.

- For data cleaning and further data preprocessing, we use MapReduce techniques of PySpark. Further, we use the SQL functionality of PySpark(pyspark.sql.functions) to understand the database schema in a user-friendly manner.

Processed Schema:

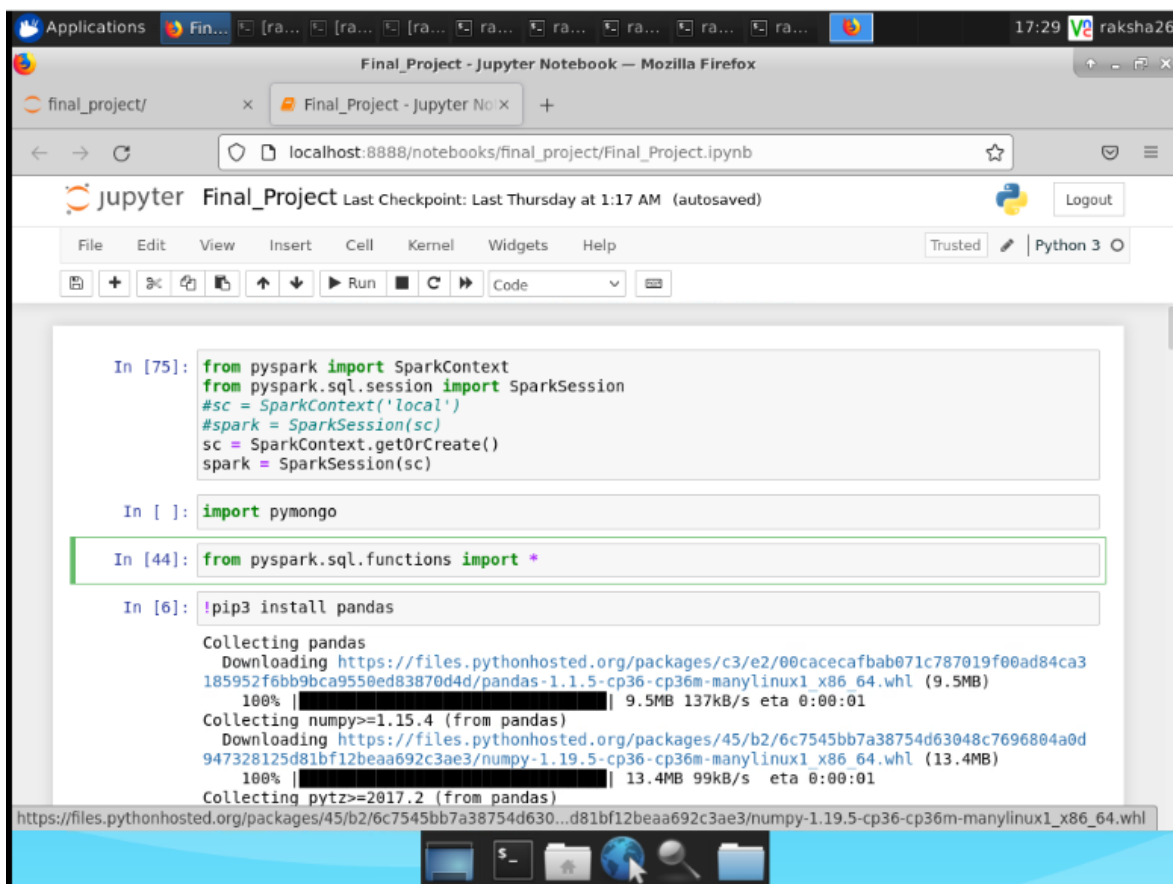
root

```
|-- :@computed_region_92fq_4b7q: string (nullable = true)
|-- :@computed_region_efsh_h5xi: string (nullable = true)
|-- :@computed_region_f5dn_yrer: string (nullable = true)
|-- :@computed_region_sbqj_enih: string (nullable = true)
|-- :@computed_region_yeji_bk3q: string (nullable = true)
|-- addr_pct_cd: string (nullable = true)
|-- boro_nm: string (nullable = true)
|-- crm_atpt_cptd_cd: string (nullable = true)
|-- jurisdiction_code: string (nullable = true)
|-- ky_cd: string (nullable = true)
|-- law_cat_cd: string (nullable = true)
|-- parks_nm: string (nullable = true)
|-- patrol_boro: string (nullable = true)
|-- station_name: string (nullable = true)
|-- susp_age_group: string (nullable = true)
|-- susp_race: string (nullable = true)
|-- susp_sex: string (nullable = true)
|-- transit_district: string (nullable = true)
|-- vic_age_group: string (nullable = true)
|-- vic_race: string (nullable = true)
|-- vic_sex: string (nullable = true)
```

➤ Pymongo as MongoDB

- Post Preprocessing, we use MongoDB NOSQL datastore as MongoDB allows storing of data in the document format and retrieving all of the information as a single record. It allows schemaless storage of data and accommodates more than 100 million documents per financial year making it extremely scalable. It even supports compressing data and information for the accumulation of huge amounts of data and information. It provides with the massive horizontal scaling of data and information that guarantee a high level of security.

We create a pymongo client and store the processed data in mongodb.



```
In [75]: from pyspark import SparkContext
from pyspark.sql.session import SparkSession
#sc = SparkContext('local')
#spark = SparkSession(sc)
sc = SparkContext.getOrCreate()
spark = SparkSession(sc)

In [ ]: import pymongo

In [44]: from pyspark.sql.functions import *

In [6]: !pip install pandas

Collecting pandas
  Downloading https://files.pythonhosted.org/packages/c3/e2/00cacafbab071c787019f00ad84ca3185952f6bb9bca9550ed83870d4d/pandas-1.1.5-cp36-cp36m-manylinux1_x86_64.whl (9.5MB)
    100% |#####| 9.5MB 137kB/s eta 0:00:01
Collecting numpy>=1.15.4 (from pandas)
  Downloading https://files.pythonhosted.org/packages/45/b2/6c7545bb7a38754d63048c7696804a0d947328125d81bf12beaa692c3ae3/numpy-1.19.5-cp36-cp36m-manylinux1_x86_64.whl (13.4MB)
    100% |#####| 13.4MB 99kB/s eta 0:00:01
Collecting pytz>=2017.2 (from pandas)
  Downloading https://files.pythonhosted.org/packages/45/b2/6c7545bb7a38754d630...d81bf12beaa692c3ae3/numpy-1.19.5-cp36-cp36m-manylinux1_x86_64.whl
```

[Fig. Using PySpark in Jupyter notebook in the Virtual Instance on Jetstream]

➤ Data Preservation

- For storage and preservation purposes, and ease of accessibility, the final data would be pushed in a Github repository.

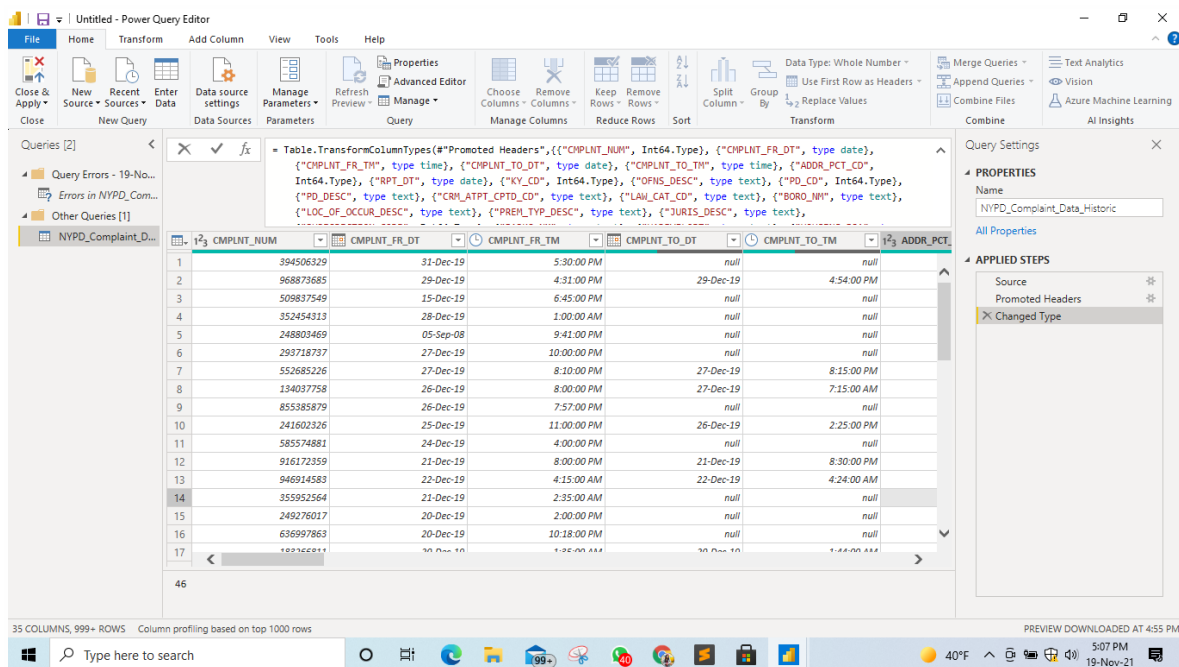
IV. RESULTS

Visualization is an important part of any data pipeline. It gives us a clear idea of the data by giving it visual context through graphs and maps. It makes human comprehension far simpler and hence, makes it easier to observe trends, patterns and outliers in a large dataset. There are many tools available that would help us in visualising data. Here, for our use case, I am using PowerBI.

➤ Why PowerBI?

Power BI service is a secure Microsoft hosted cloud service that lets users view dashboards, reports, and Power BI apps, a type of content that combines related dashboards and reports using a web browser or via mobile apps.

PowerBI users can connect to multiple formats of data, transform and model the data and create visually appealing dashboards, charts, reports which would help us gain relevant insights from the data at hand.

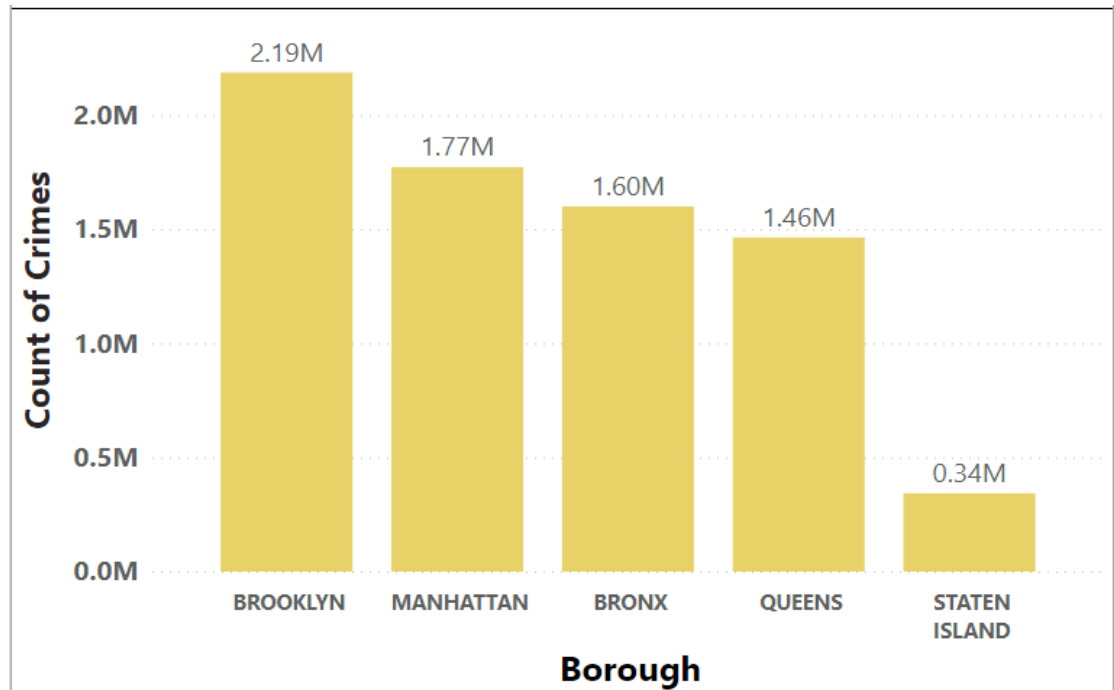


[Fig. PowerBI query editor for cleaning data and preprocessing for visualization]

➤ Visualisations

1. Count of Crimes for every Borough

As we can observe from the graph, most crimes have occurred in Brooklyn followed by Manhattan and the Bronx. Hence, we can state Brooklyn as the most unsafe borough in the city.

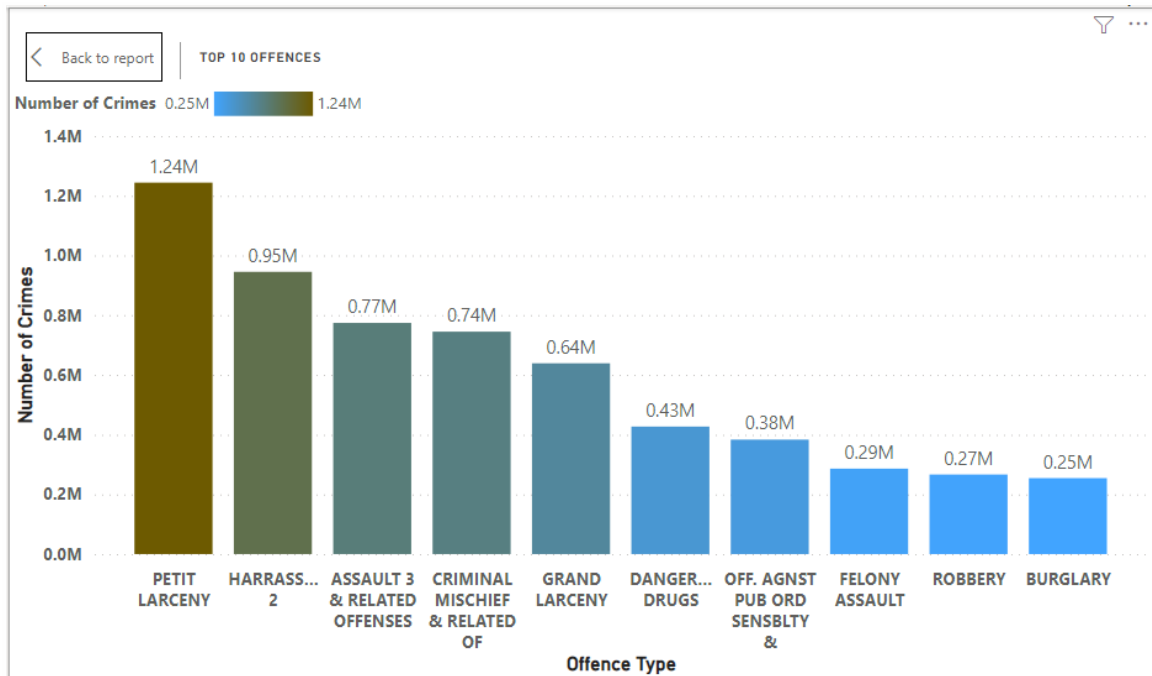


Below is the tabular form for the same which depicts the borough and its corresponding crime count.

Borough	Count of Crimes
BROOKLYN	2186681
MANHATTAN	1771637
BRONX	1599801
QUEENS	1463554
STATEN ISLAND	342991

2. Top 10 offences

We can observe petit larceny is the most common crime committed.

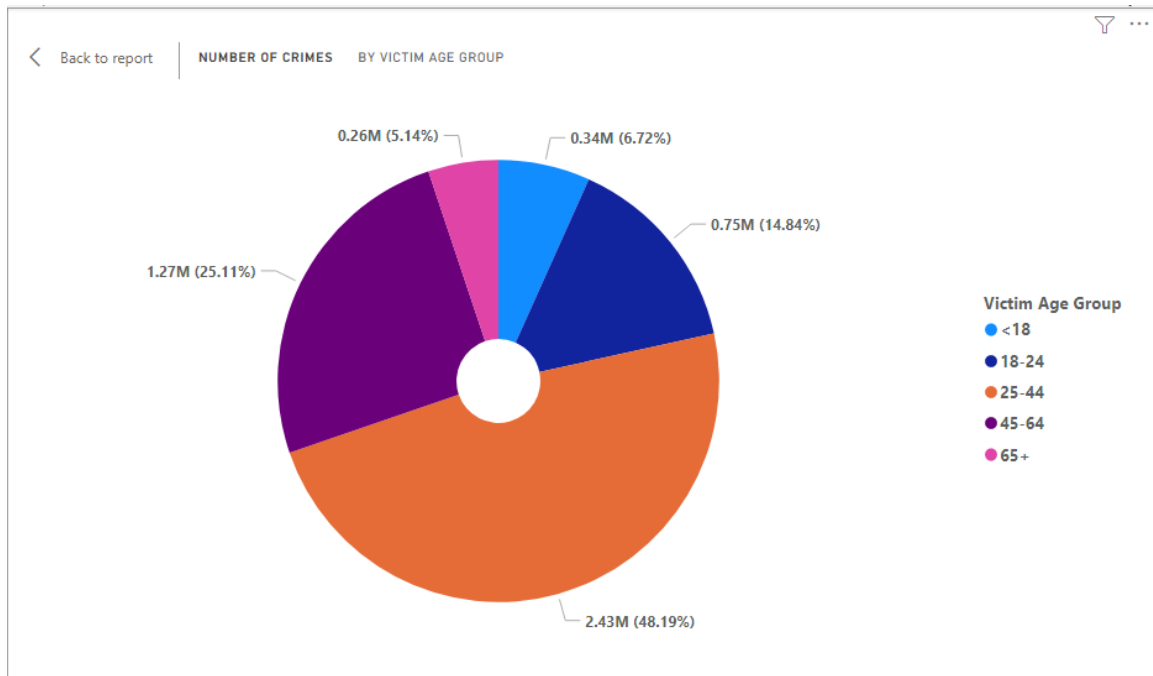


The data table shows in percentage the crime types where petit larceny leads with 20.86%.

Offence Type	Number of Crimes
PETIT LARCENY	20.86%
HARRASSMENT 2	15.85%
ASSAULT 3 & RELATED OFFENSES	12.98%
CRIMINAL MISCHIEF & RELATED OF	12.49%
GRAND LARCENY	10.71%
DANGEROUS DRUGS	7.16%
OFF. AGNST PUB ORD SENSBLTY &	6.43%
FELONY ASSAULT	4.80%
ROBBERY	4.47%
BURGLARY	4.26%

3. Number of Crimes on Victims by age group

We can observe that most crime victims were in the age group of 25-44



Below is the table shown for the same.

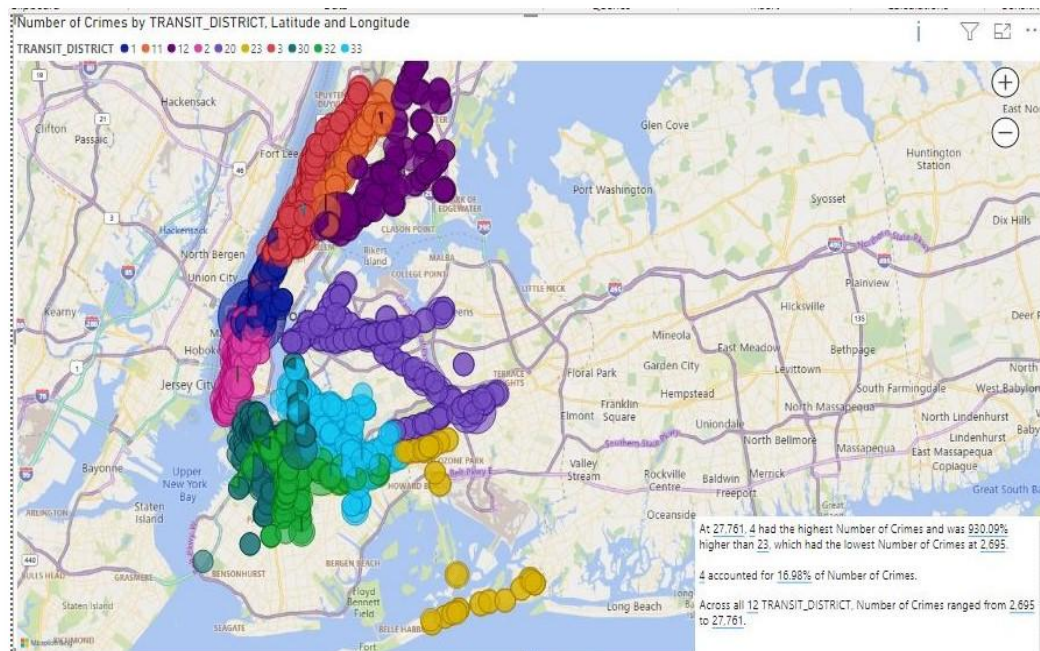
Victim Age Group	Number of Crimes
<18	339270
18-24	749451
25-44	2433923
45-64	1268365
65+	259662

4. Which Transit district is the most unsafe

From the below table, we can see that transit district number 4 has the highest number of crimes taking place.

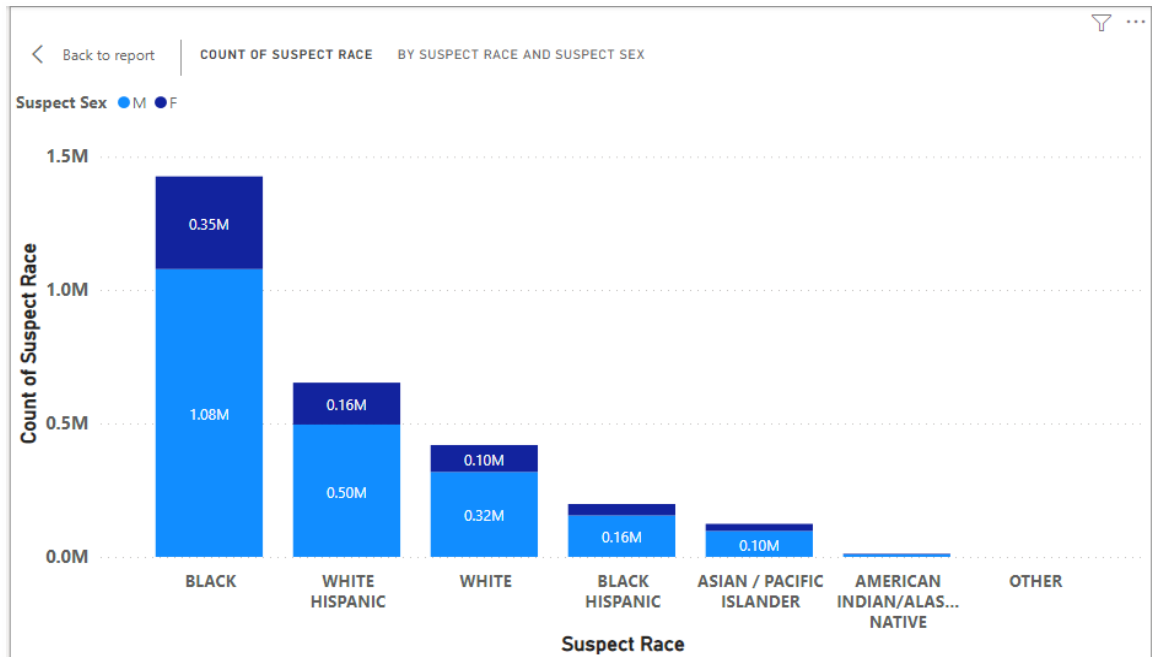
TRANSIT_DISTRICT	Number of Crimes
1	9.66%
11	6.90%
12	7.49%
2	12.37%
20	8.85%
23	1.65%
3	9.69%
30	6.43%
32	6.78%
33	8.79%
34	4.41%
4	16.98%

Below is the graphical representation for the same.



5. Suspect race and sex

We can infer from the graph that the majority of the crime suspects are Black and that too Male dominating.

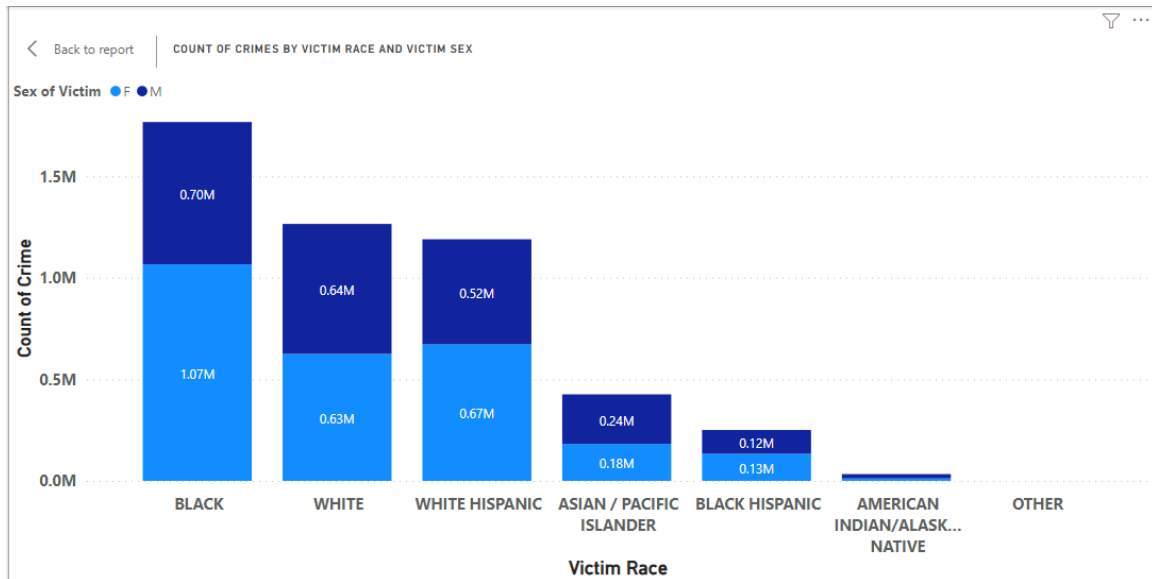


Below is the statistical data for the same.

Suspect Race	M	F
AMERICAN INDIAN/ALASKAN NATIVE	0.32%	0.08%
ASIAN / PACIFIC ISLANDER	3.43%	0.94%
BLACK	38.09%	12.27%
BLACK HISPANIC	5.49%	1.51%
OTHER	0.00%	0.00%
WHITE	11.24%	3.56%
WHITE HISPANIC	17.51%	5.55%

6. Victim Race and Sex

We can infer from the graph that most crime victims are Black females.



The data table shows the distribution of victim's race along with the sex.

Victim Race	F	M
BLACK	21.63%	14.20%
WHITE	12.68%	12.98%
WHITE HISPANIC	13.63%	10.49%
ASIAN / PACIFIC ISLANDER	3.67%	4.96%
BLACK HISPANIC	2.71%	2.37%
AMERICAN INDIAN/ALASKAN NATIVE	0.24%	0.43%
OTHER	0.00%	0.00%

V. DISCUSSION

Through this project, I was able to connect to the learnings during the course and implement them in various stages of the data pipeline. From creation of data pipeline, what components to choose, what can be done without, selection and importance of data store, how visualisation plays an important role in the presentation of the pipeline, importance of data preservation and at the end how AI fairness slides into picture without intentional focus.

From the visualisation results, we can gain many insights about the crime distribution in New York. As per our results, Brooklyn has the most number of crimes, with petit larceny being the most committed offence. The highest probability of suspect is a person of Black ethnicity that too Male Gender and for victim again nit is

Black ethnicity with female leading here. Also, transit district number 4 is the most unsafe.

VI. CONCLUSION

_____ In the project, we implemented an end to end big data pipeline. We used various technologies and there were tradeoffs to choose between them too which I have discussed at high level why I chose what tool or technology. We got many insights which can be used to work on in future to create AI based solutions.

➤ **Future Work**

Currently, we analysed the data through a big data pipeline. We can in future implement a prediction model that would predict crime rate based on time series analysis.

As we can observe from our analysis above, a random suspect is more likely to be someone from Black ethnicity compared to any other ethnicities. We can model the suspect ratio by ethnicity to that of the suspect actually found guilty. This would help us in judging the fairness of the AI model we would create. This is an important aspect to take into consideration as bias in AI models is very common. Bias can enter through the data itself while training and as the data is highly unbalanced, we are likely to run into that problem.

VII. REFERENCES

1. <https://data.cityofnewyork.us/Public-Safety/NYPD-Complaint-Data-Historic/qgea-i56i>
2. <https://towardsdatascience.com/scalable-efficient-big-data-analytics-machine-learning-pipeline-architecture-on-cloud-4d59efc092b5>
3. <https://nypdonline.org/link/1026>
4. <https://www.whizlabs.com/blog/real-time-big-data-pipeline/>
5. <https://towardsdatascience.com/a-beginners-guide-to-the-data-science-pipeline-a4904b2d8ad3>
6. <https://cloud.google.com/bigtable/docs/hbase-dataflow-java>
7. <https://iujetstream.atlassian.net/wiki/spaces/JWT/pages/75142187/Transferring+files+with+SCP+or+SFTP>
8. https://datanoon.com/blog/loading_data_rest_api_to_spark/