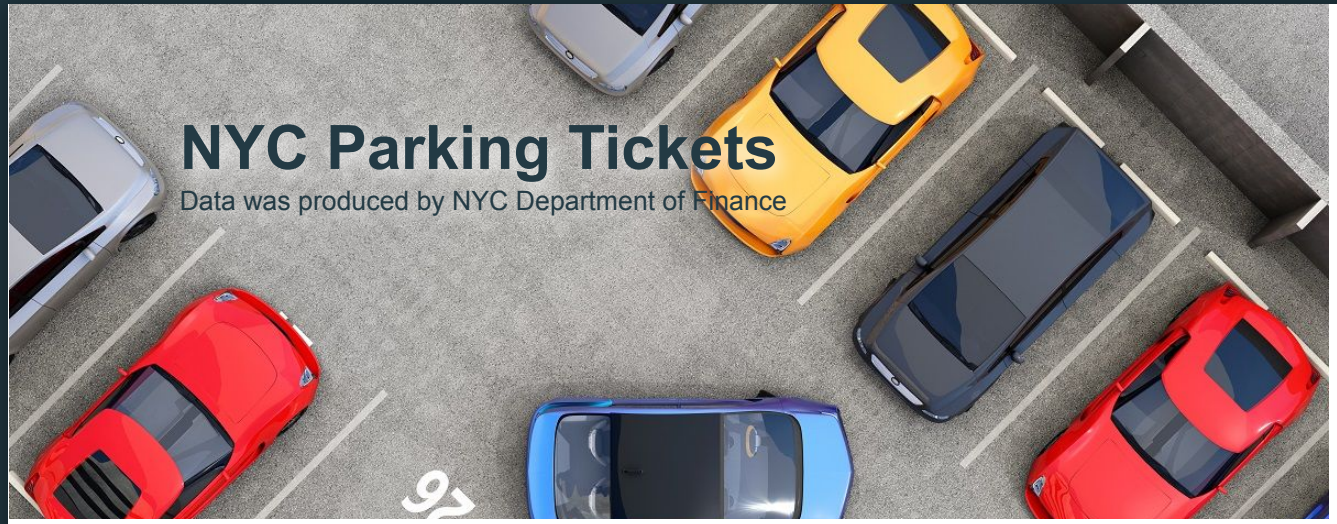
New York Parking Ticket Analytics

Data Processed with an AWS EMR Cluster

By Roger Ren, Zhiyi Ren, Karsten Cao, Xinyue Wang



The NYC Department of Finance collects data on every parking ticket issued in NYC. This dataset contains all the information of issued parking tickets in New York City from Aug 2013 to June 2017 with over 20 columns including street name, issue date and violation code.

Data Size: 4.66GB
Format: CSV
Time Window: August 2013 - September 2015
Source:https://www.kaggle.com/new-york-city/nyc-parking-tickets

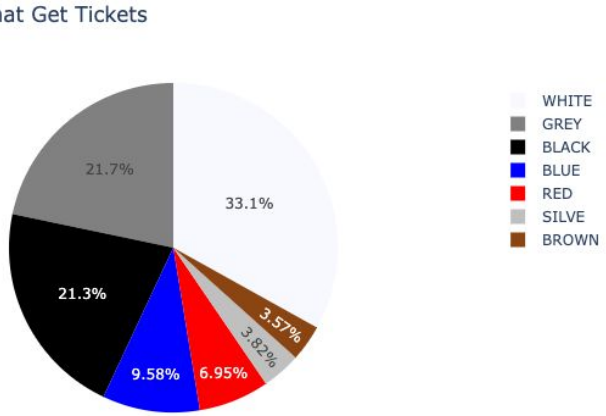
Data Analytics Goal

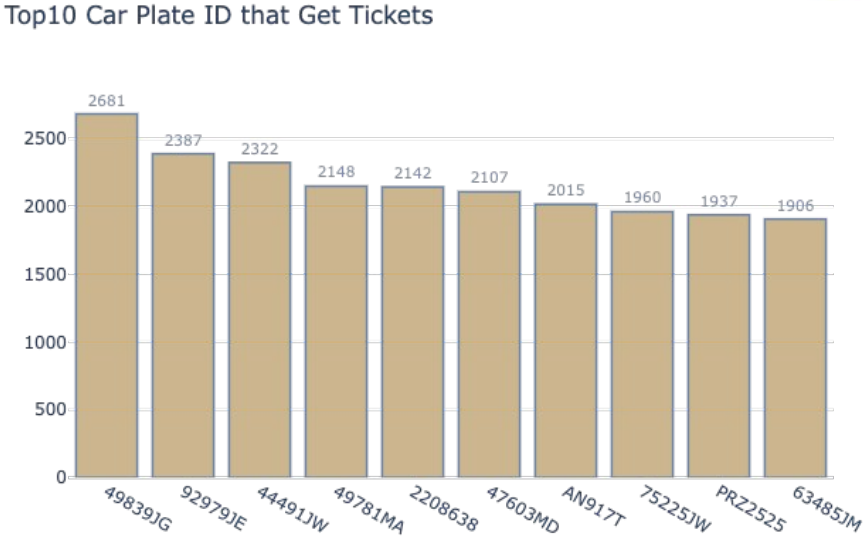
Post Hoc Analysis of Police Behavior

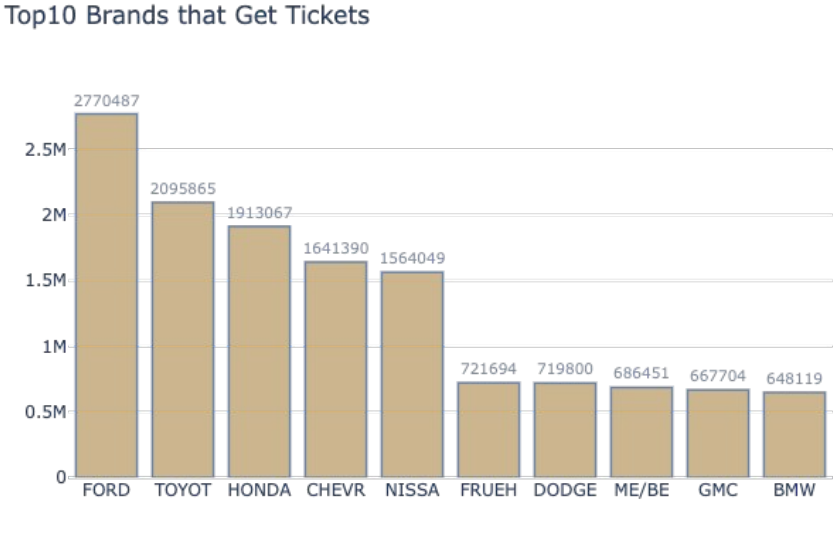
When, where, and which car the police are likely to issue tickets?
What are the major reasons for the cars to be stickered?

Outcomes

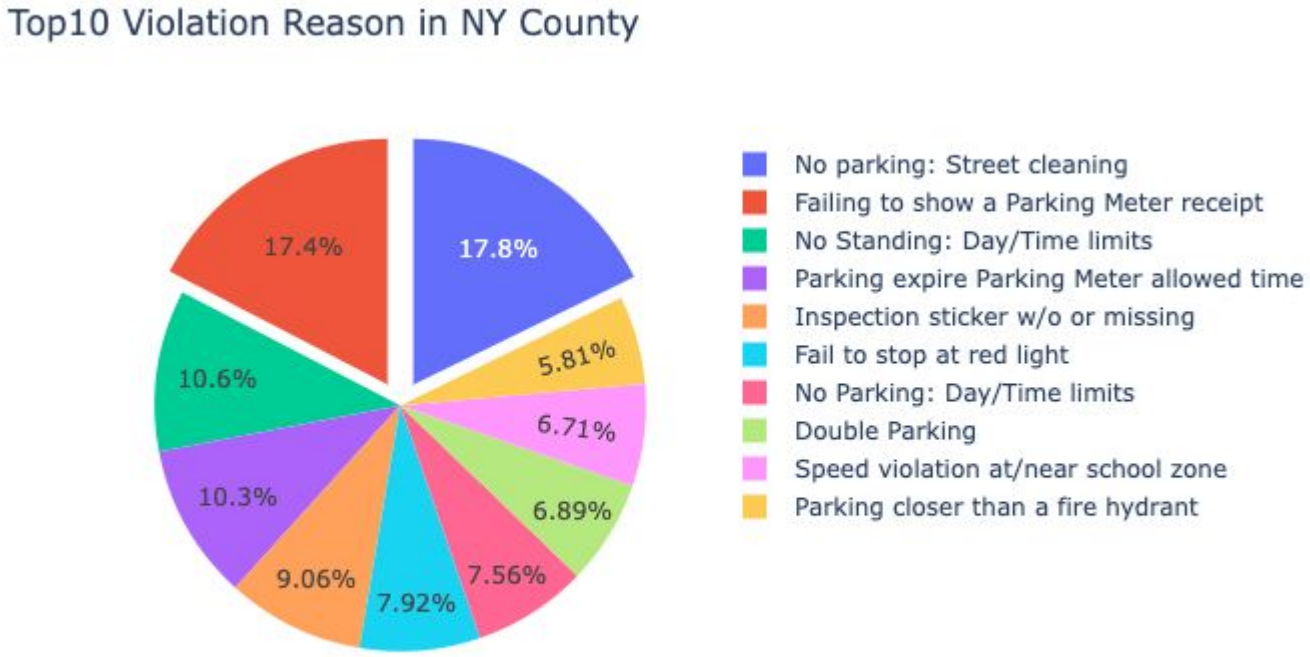
Which Car ?


Top7 Car Colors that Get Tickets
21.7%
33.1%
21.3%
3.57%
3.82%
9.58%
6.95%
WHITE
GREY
BLACK
BLUE
RED
SILVE
BROWN


Top10 Car Plate ID that Get Tickets
2681
2387
2322
2148
2142
2107
2015
1960
1937
1906
2500
2000
1500
1000
500
0
49839JG
92979JE
44491JW
49781MA
2208638
47603MD
AN917T
75225JW
PRZ2525
63485JM

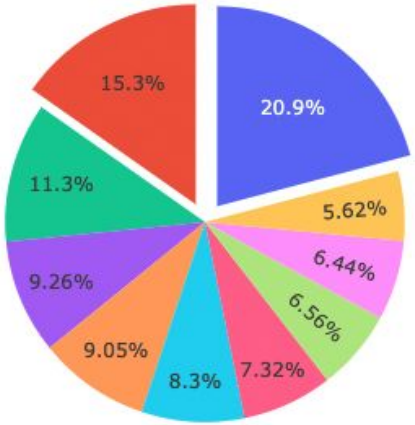

Top10 Brands that Get Tickets
2770487
2095865
1913067
1641390
1564049
721694
719800
686451
667704
648119
2.5M
2M
1.5M
1M
0.5M
0
FORD
TOYOT
HONDA
CHEVR
NISSA
FRUEH
DODGE
ME/BE
GMC
BMW

Why ?

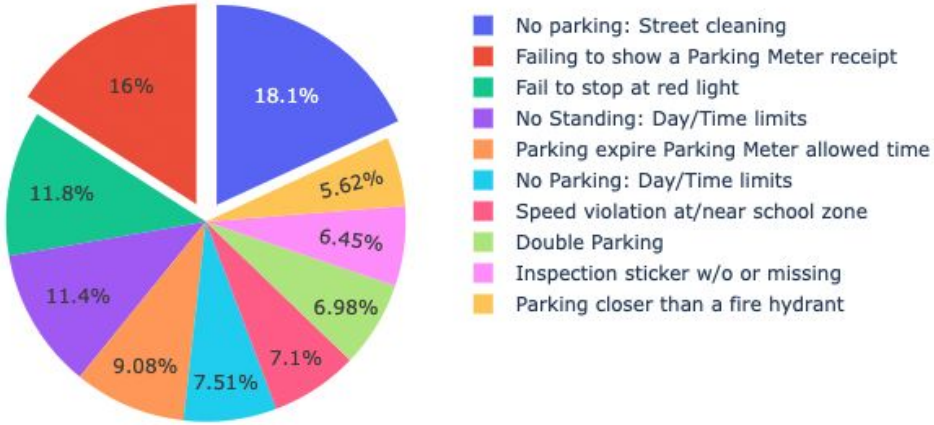

Top10 Violation Reason in NY County
17.8%
17.4%
10.6%
10.3%
9.06%
7.92%
7.56%
6.89%
6.71%
5.81%
No parking: Street cleaning
Failing to show a Parking Meter receipt
No Standing: Day/Time limits
Parking expire Parking Meter allowed time
Inspection sticker w/o or missing
Fail to stop at red light
No Parking: Day/Time limits
Double Parking
Speed violation at/near school zone
Parking closer than a fire hydrant

Why ?

Top10 Violation Reasons in NY County for June



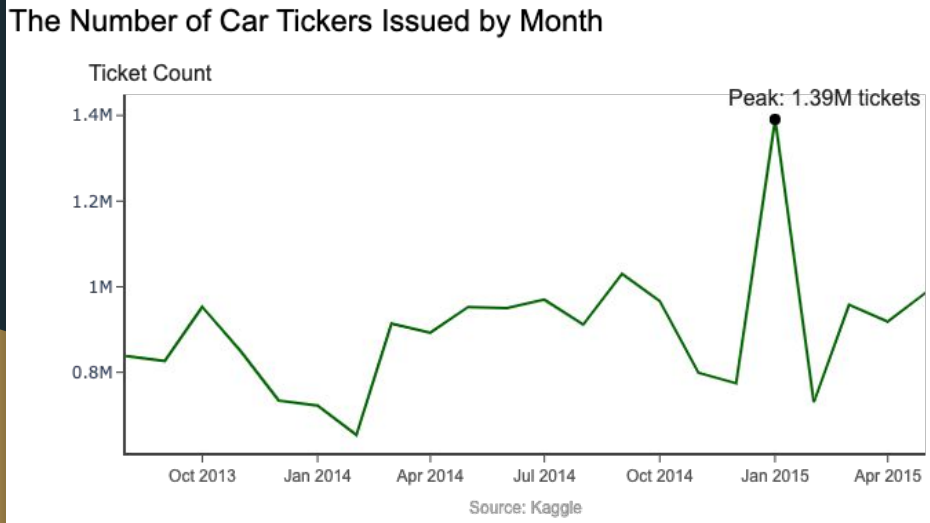
Top10 Violation Reasons in NY County for July

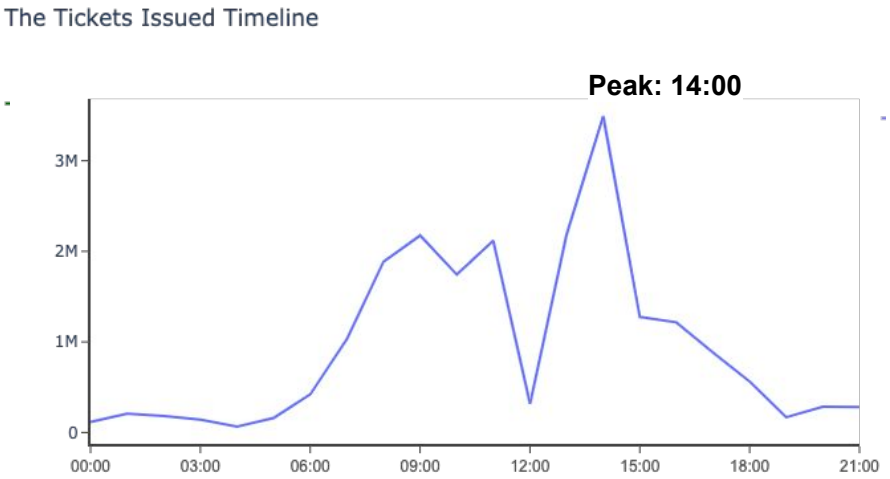


Time ?


Count of Issued Parking Tickets by Month
2M
1.5M
1M
0.5M
0
January
February
March
April
May
June
July
August
September
October
November
December
June 30th, End of Fiscal Year

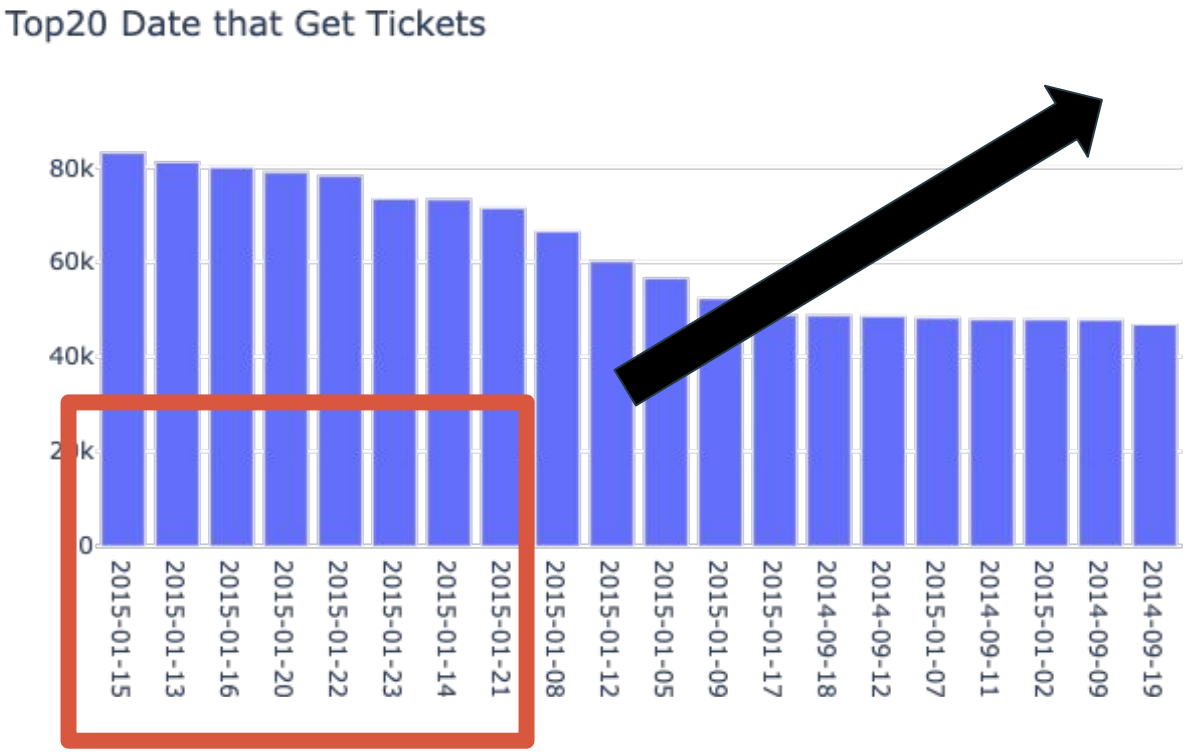
What Time?





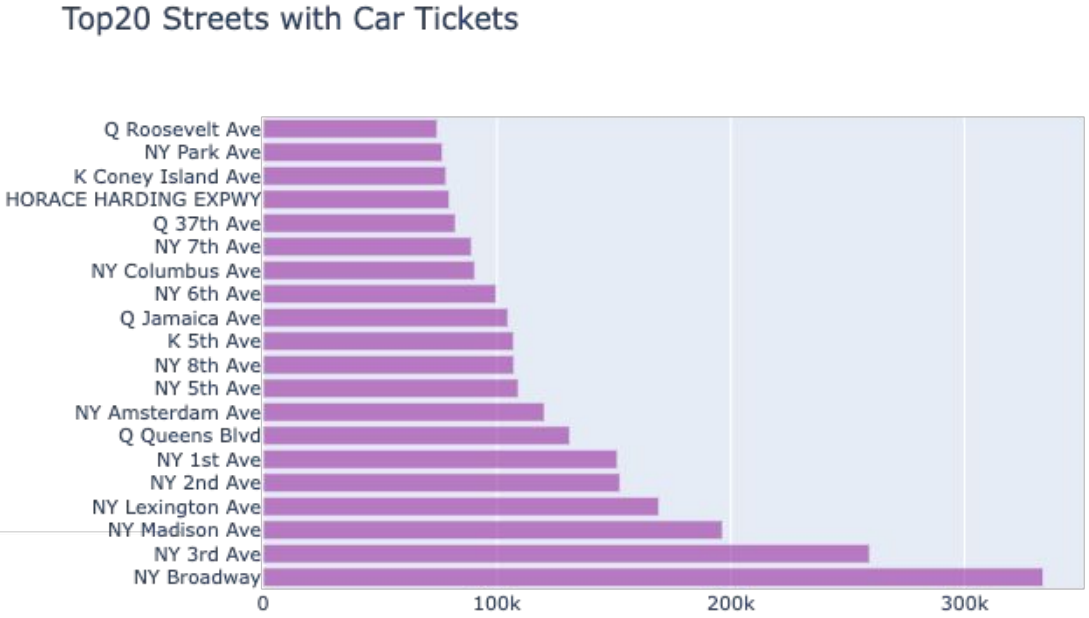
Working Time

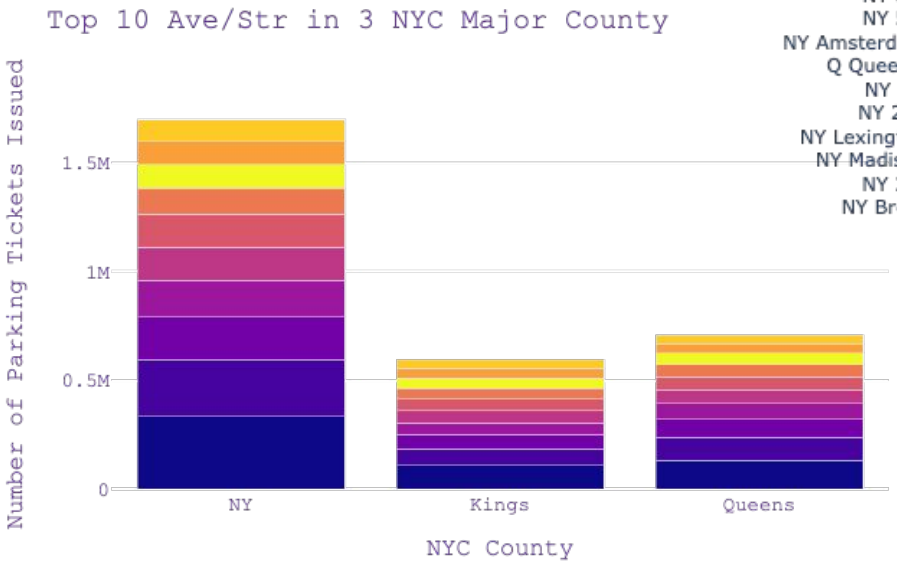
What Time?



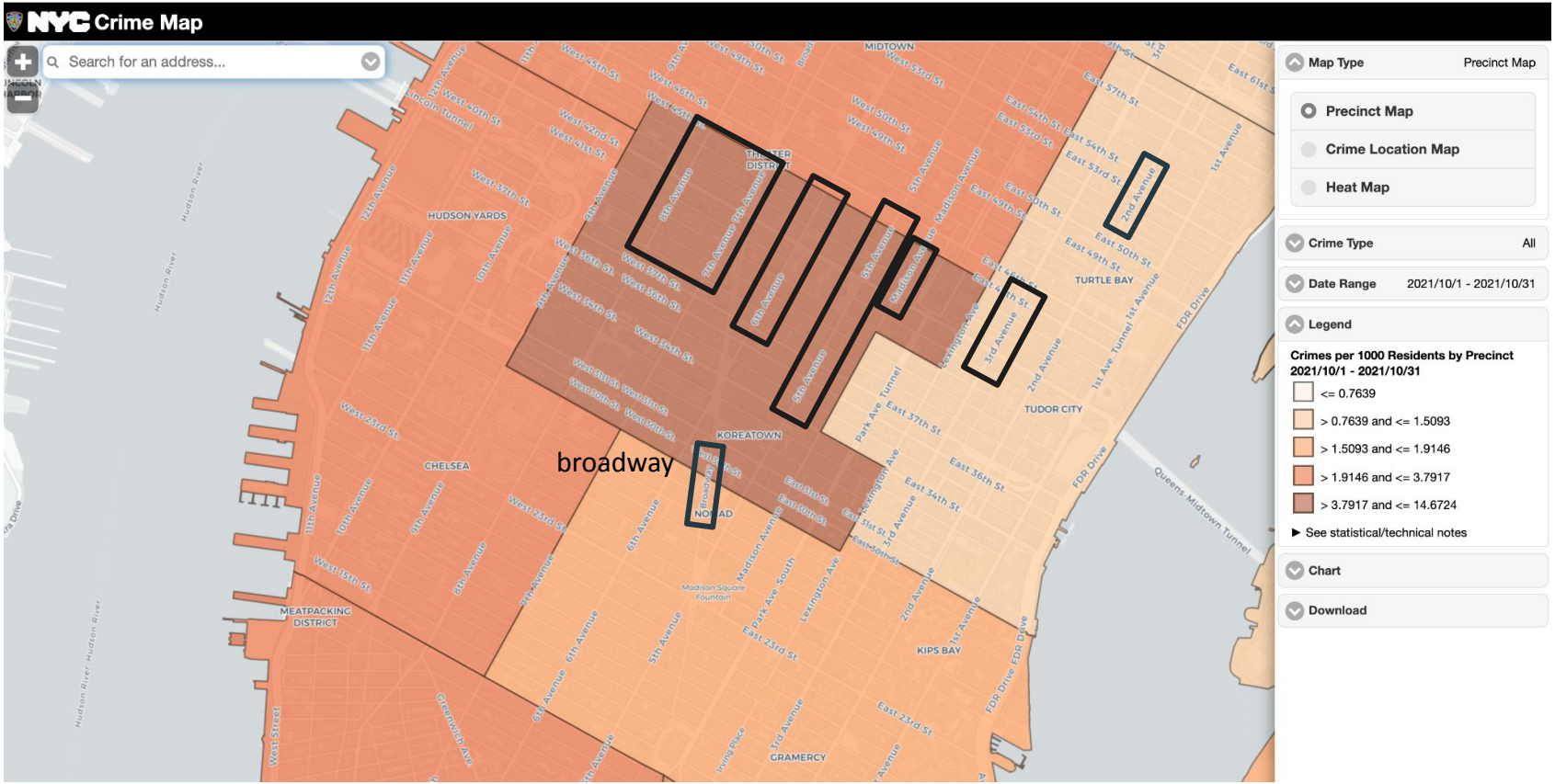
Subway Shutdown

Where?


Top20 Streets with Car Tickets
Q Roosevelt Ave
NY Park Ave
K Coney Island Ave
HORACE HARDING EXPWY
Q 37th Ave
NY 7th Ave
NY Columbus Ave
NY 6th Ave
Q Jamaica Ave
K 5th Ave
NY 8th Ave
NY 5th Ave
NY Amsterdam Ave
Q Queens Blvd
NY 1st Ave
NY 2nd Ave
NY Lexington Ave
NY Madison Ave
NY 3rd Ave
NY Broadway
0
100k
200k
300k


Top 10 Ave/Str in 3 NYC Major County
Number of Parking Tickets Issued
1.5M
1M
0.5M
0
NY
Kings
Queens
NYC County

Where?


NYC Crime Map
Search for an address...
broadway
HUDSON YARDS
CHELSEA
KOREATOWN
MIDTOWN
TURTLE BAY
TUDOR CITY
KIPS BAY
GRAMERCY
MEATPACKING DISTRICT
Map Type
Precinct Map
Precinct Map
Crime Location Map
Heat Map
Crime Type
All
Date Range
2021/10/1 - 2021/10/31
Legend
Crimes per 1000 Residents by Precinct
2021/10/1 - 2021/10/31
<= 0.7639
> 0.7639 and <= 1.5093
> 1.5093 and <= 1.9146
> 1.9146 and <= 3.7917
> 3.7917 and <= 14.6724
See statistical/technical notes
Chart
Download

Performance Results

Cluster Settings:

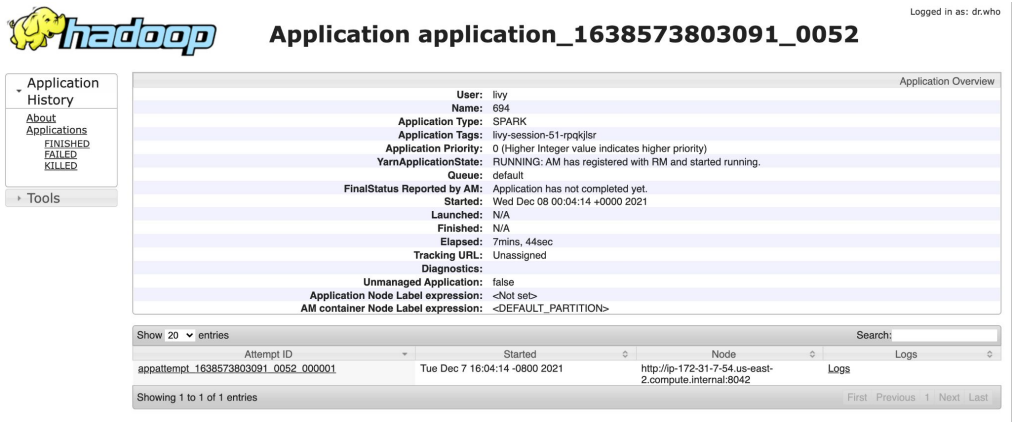
m5.xlarge

4 vCore, 16 GiB memory, EBS only storage

EBS Storage: 64 GiB

```
%%configure -f
{
"conf": {
    "spark.pyspark.python": "python3",
    "spark.pyspark.virtualenv.enabled": "true",
    "spark.pyspark.virtualenv.type": "native",
    "spark.pyspark.virtualenv.bin.path": "/usr/bin/virtualenv",
    "spark.app.name":"694"
    }
}
```

Execution Time: 7 Min 44S



Efficiency

- **Cache the whole data's rdd**
- For a single chunk of data operation
- Can imagine the actual power of how many minutes it saved for us

```
]: rdd.cache()

PythonRDD[12] at RDD at PythonRDD.scala:53
```

Without cache

```
#without cache
start = time.time()
def trans(data):
    try:
        if 'A' in data:
            data = data.replace('A','')
            data = data[:2]+':'+ '00'
        elif 'P' in data:
            data = data.replace('P','')
            d = int(data[1:2])+12
            data = str(d)+':'+ data[2:]
            data = data[:2]+':'+'00'
    except:
        data = 'K'
    return data
Q = rdd.map(lambda x: (x[5],x[1])).map(lambda x: (trans(x[0]),x[1])).countByKey()
Q = sorted(Q.items(),key=lambda x: x[0],reverse=False)
Q_rdd = sc.parallelize(Q)
Q_rdd_df = Q_rdd.filter(lambda x: x[1] > 21).filter(lambda x: x[0] != '').toDF(['Issue_Time','Ticket_Count'])
Q_rdd_df.createOrReplaceTempView("Q_rdd_view")
Q_rdd_df.show()
print("second", time.time() - start)
```

```
|      05:00|      159085|
|      06:00|      422918|
|      07:00|     1030195|
|      08:00|     1886260|
|      09:00|     2176025|
|      10:00|     1743116|
|      11:00|     2117534|
|      12:00|      316201|
|      13:00|     2182580|
|      14:00|     3492323|
|      15:00|     1276283|
|      16:00|     1216548|
|      17:00|      881265|
|      18:00|      561913|
|      19:00|      168937|
+----------+------------+
only showing top 20 rows

second 66.63361525535583
```

Before:66 seconds

```
#with cache
start = time.time()
def trans(data):
    try:
        if 'A' in data:
            data = data.replace('A','')
            data = data[:2]+':'+ '00'
        elif 'P' in data:
            data = data.replace('P','')
            d = int(data[1:2])+12
            data = str(d)+':'+ data[2:]
            data = data[:2]+':'+'00'
    except:
        data = 'K'
    return data
Q = rdd.map(lambda x: (x[5],x[1])).map(lambda x: (trans(x[0]),x[1])).countByKey()
Q = sorted(Q.items(),key=lambda x: x[0],reverse=False)
Q_rdd = sc.parallelize(Q)
Q_rdd_df = Q_rdd.filter(lambda x: x[1] > 21).filter(lambda x: x[0] != '').toDF(['Issue_Time','Ticket_Count'])
Q_rdd_df.createOrReplaceTempView("Q_rdd_view")
Q_rdd_df.show()
print("second", time.time() - start)
```

```
|      05:00|      159085|
|      06:00|      422918|
|      07:00|     1030195|
|      08:00|     1886260|
|      09:00|     2176025|
|      10:00|     1743116|
|      11:00|     2117534|
|      12:00|      316201|
|      13:00|     2182580|
|      14:00|     3492323|
|      15:00|     1276283|
|      16:00|     1216548|
|      17:00|      881265|
|      18:00|      561913|
|      19:00|      168937|
+----------+------------+
only showing top 20 rows

second 16.256643295288086
```

After: 16 seconds

Conclusion

Lesson Learned

- Large Dataset
  a. Samples do not represent the whole
  b. Data may not be as clean
- EMR Usage
- Coding Together - Logistics

Thank you !