MSCI 6110: Big Data Management and Analytics
Midpoint Project Report
Group 2
Aarron Lebow, Jinghui Li, Tom Maples, Santhosh Raj Murugesan

Summary

Every year, millions of parking citations are issued in New York City. The data collected from each ticket could contribute to valuable information and knowledge gained. Trends in years and months, clusters of vehicle types, locations, issuing precincts, and violation descriptions are just a few of the many facets. Potential stakeholders include the offices of New York City—budgeting, finance, department of transportation, police—auto manufacturers, rideshare services, and potentially a searchable public database educating people on parking trends.

Data

The dataset that we have used for NYC parking violation analysis consists of approximately 42 million observations from New York City from August 2013 - June 2017. The data has been collected from New York City Department of Finance and made publicly available on opendata.cityofnewyork.us. Each observation comprises of 51 attributes pertaining to each individual ticket. All attributes are listed in the appendix

Methods

A temporary table was defined, and data from fiscal years 2014 – 2017 was loaded from the hive data file system. In order to reduce computing costs that stem from the size of the data, the analysis was refined to 17 features: summons number, plater ID, violation code, violation location, violation precinct, issuer precinct, issuer command, issuer squad, street name, vehicle color, vehicle make, vehicle body type, vehicle year, violation description, year, month, day, and hour. These features were then used to populate a more manageable pivot table by query. This table was dynamically partitioned by year and month.

Challenges

Some of the challenges that we faced were converting the date and time to required format and storing it in our hive tables. For example, the time was in the format of 0212p, 0124a. We had to strip off ‘a’ or ‘p’ and covert it to a 24-hour format and use it for analytics. And the date was in the format of mm/dd/yyyy but hive requires the date be in the format yyyy-mm-dd. So, we had run some scripts on the data to convert the date into required format. We are also planning to find some fine details such as month number, day of week, if it’s a weekend or not and stored it in the tables so we will be able to predict how many violations happen during the weekends and weekdays. We have also which month of the year and the years the parking violations are at its peak. The violation description was missing in many observations, but we populated it using information obtained from the Department of Finance Website

https://www1.nyc.gov/site/finance/vehicles/services-violation-codes.page

Analysis

Registration State

The data was grouped by Registration State and the number of tickets issued per state was counted and divided by the total number of tickets. The majority of tickets were issued to the vehicles registered in New York, followed by New Jersey and Pennsylvania.

NY	73%
NJ	8%
PA	3%

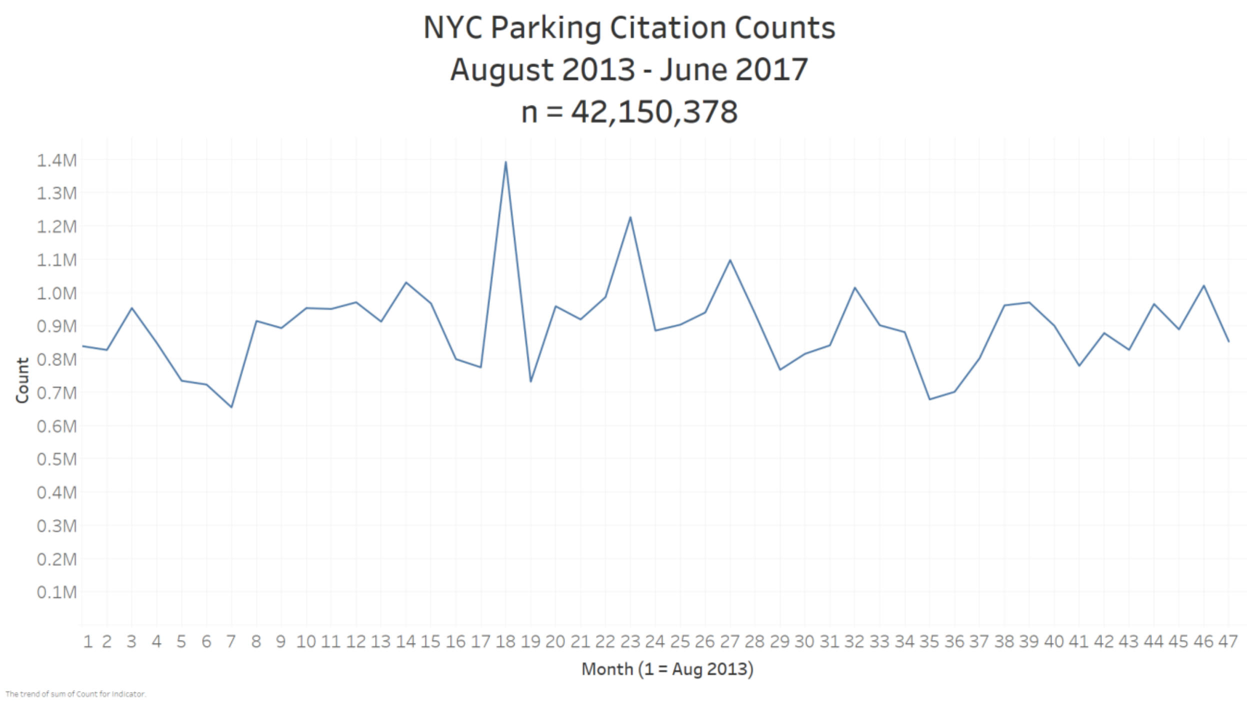
Plate Type

The data was grouped by plate type and the number of tickets issued was counted for each plate type and divided by the total number of tickets. The majority of tickets were issued to the passenger plate type, followed by commercial vehicles.

PAS	70%
COM	21%

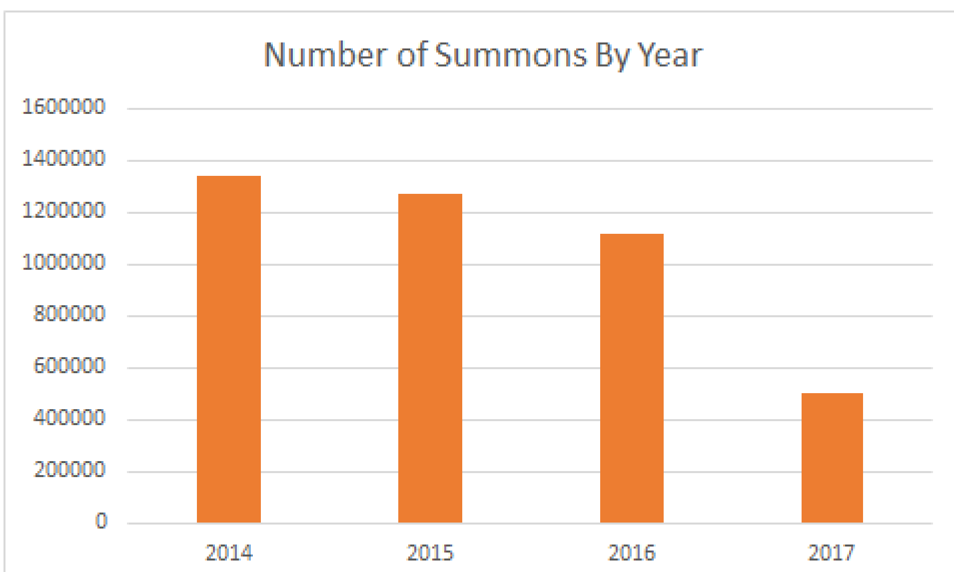
Citation Counts by Month

Below is a timeline of counts between August 2013 and June 2017. The sharp spike in January 2015 (18) is most striking, especially considering the low counts in December 2014 and February 2015. There doesn’t appear to be any true cyclical trend in the timeline, except that September and October appear to have consistently higher counts relative to local trends (2-3, 14-15, 26-27, 38-39). January 2014 appears to have approximately 725,000 citations compared to January 2015 with almost 1.4M citations—nearly double. The count for January 2016 (30) is approximately 820,000. There is a visible downward trend from January 2015. Further exploration of this time period centered around January 2015 is needed. Examining patterns and clusters of citations in the months of September and October would be prudent.



Summons per Year

The total number of summons was counted by year. Excluding the sharp decline in 2017, which only contains 6 months of data, the number of summons slightly decreased from 2014 to 2016

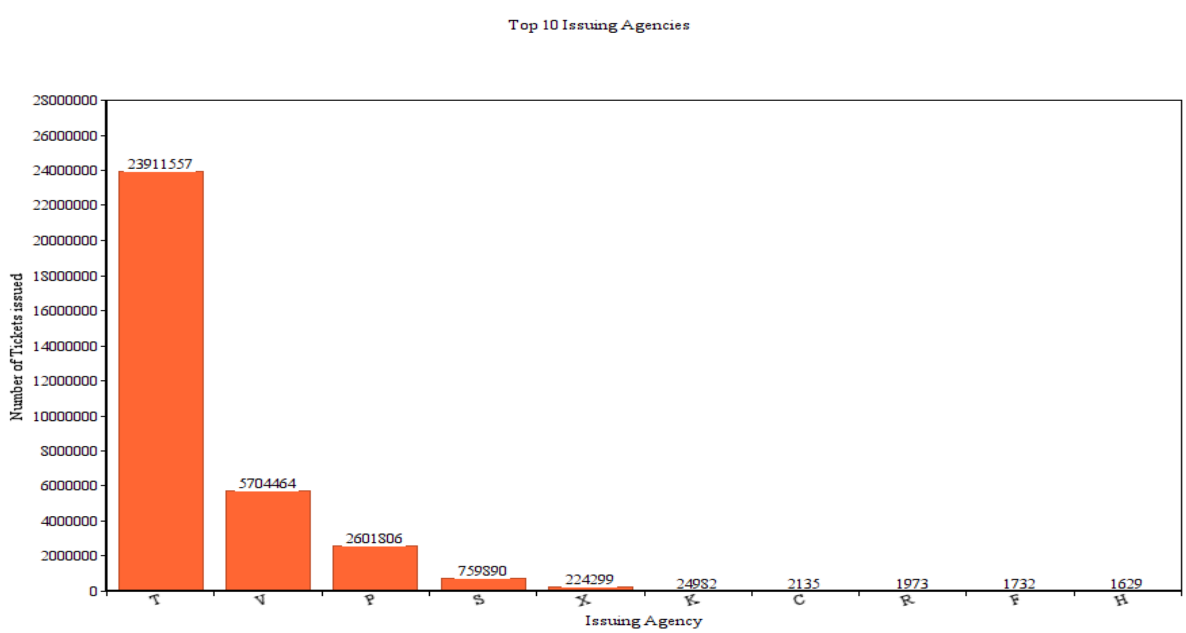


Street

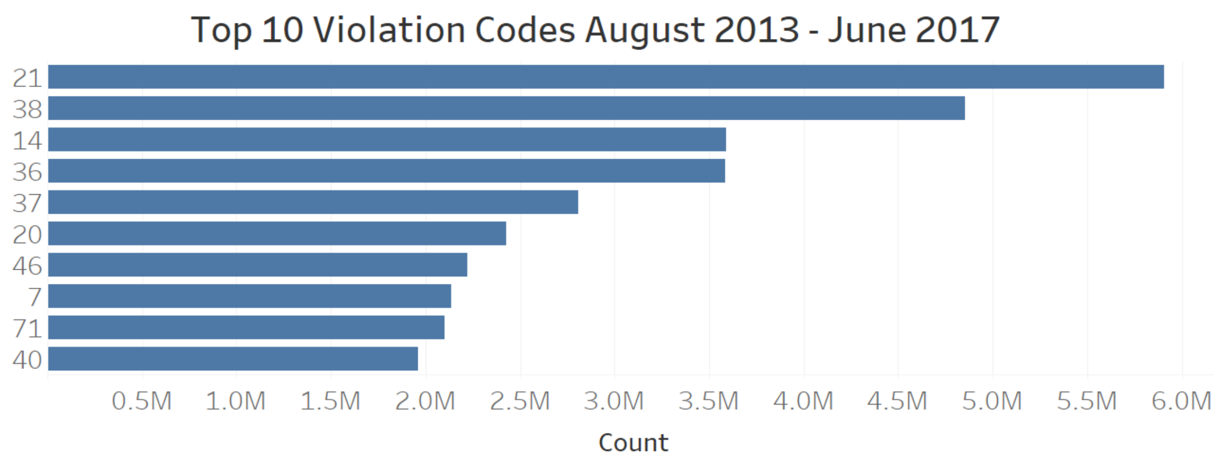
Manhattan has the most number of issuances. And Broadway has the maximum number of tickets. Since Broadway is the longest street in the city, we will study the violation location to locate where on Broadway the tickets were issued.

Issuing Agency

From the result, Agent T issued majority of tickets. It is about 80% of total issued tickets.



Type of violations



There are about 100 type codes, so the chart shows the top 10. They account for about 70% of all tickets.

Rank	Violation Code	DESC
1	21	Street Cleaning
2	38	Parking Meter: Parking in excess of the allowed time
3	14	General No Standing
4	36	Exceeding the posted speed limit in or near a designated school zone.
5	37	Parking Meter: Failing to show a receipt or tag in the windshield. Drivers get a 5-minute grace period past the expired time on parking meter receipts.
6	20	General No Parking
7	46	Standing or parking on the roadway side of a vehicle stopped, standing or parked at the curb
8	7	Vehicles photographed going through a red light at an intersection
9	71	Standing or parking a vehicle without showing a current New York inspection sticker.
10	40	Stopping, standing or parking closer than 15 feet of a fire hydrant.

Analysis Plan

The goal of the analysis will be to look for groupings among the parking violations. Features from the parking violations dataset will be clustered in order to look for these groupings of violations. The groups will then be characterized. Some initial features of interest are location, car make, car model, car color, date, time, registration state, and issuing precinct. Location of violations over time will be visualized by creating choropleth maps. Similarly, choropleths can potentially be used to visually explore the clusters.

Final Project Report

Group 2

Aarron Lebow, Jinghui Li, Tom Maples, Santhosh Raj Murugesan

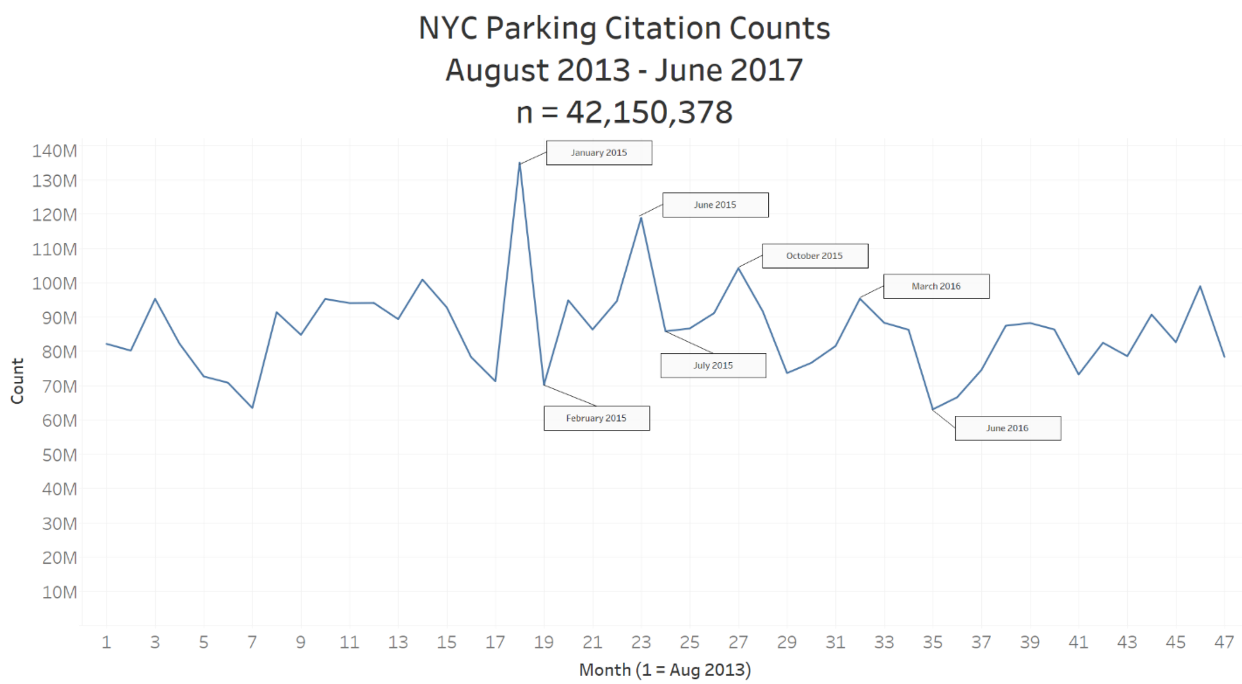
Analysis Phase II

Summary

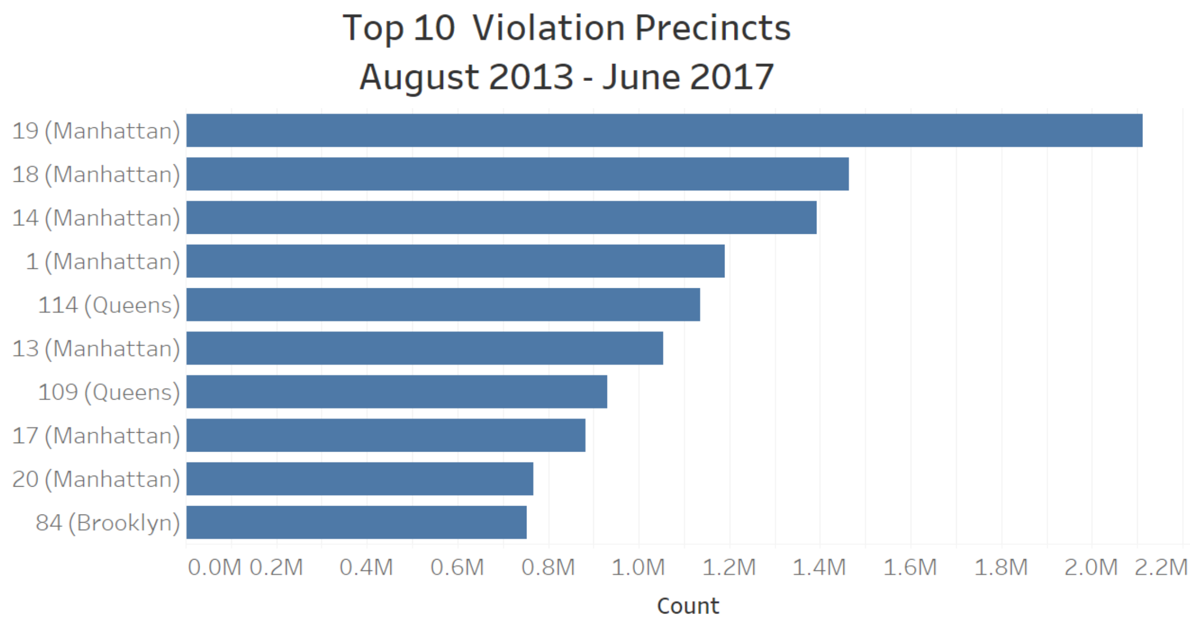
Based on the Phase I data analysis work, we continued to work on the data trend analysis, association Rule mining and prediction by using random forest algorithm.

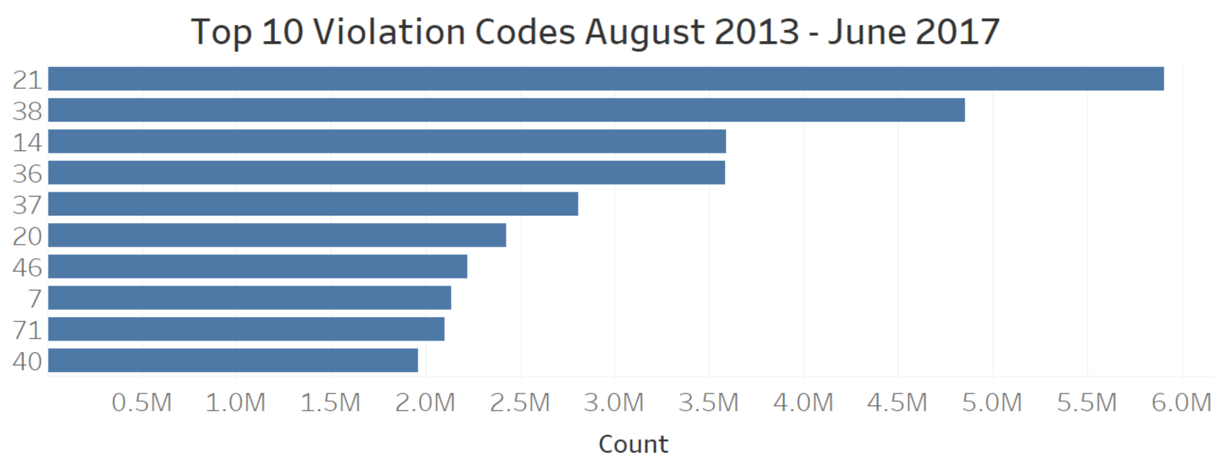
Trends

Below, the timeline from August 2013 – June 2017 shows a sharp spike in the count of citations in January 2015. This is preceded and followed by periods of low citation counts. Beginning in June 2015 there is a noticeable decline in the count of citations before the trend appears to level off in the second half of 2016.



In order to gain more insight into this time period, two variables were considered: violation codes and violation precincts. In other words, what violation codes were being issued, and where. The following bar plots display the top 10 violation precincts and violation codes respectively.





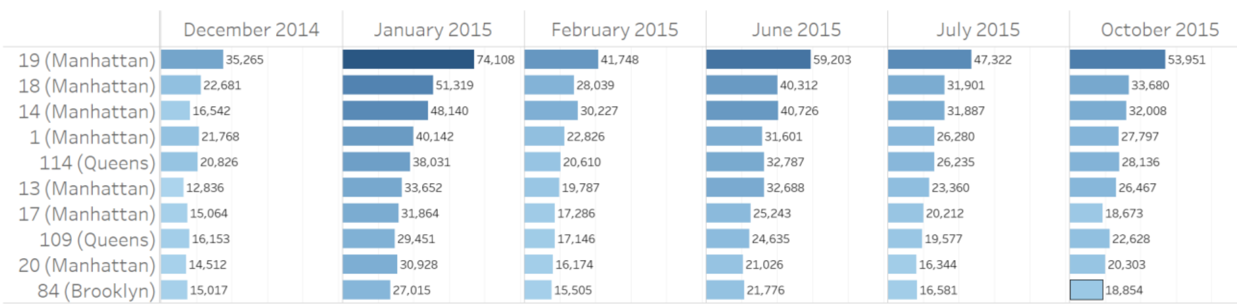
21: Street Cleaning	20: General No Parking
38: Parking Meter: Parking in excess of the allowed time	46: Standing/parking roadway stopped vehicle/curb
14: General No Standing	7: Vehicles photo'd going through red light
36: Speeding School Zone	71: Standing/parking w/out showing NY inspection sticker
37: Parking Meter: Fail to show receipt or tag in windshield	40: Stopping/standing/parking < 15 ft. fire hydrant

There are 6,379,081 precinct 0 citations, which were treated as null values and excluded from the top 10 list. The top 10 violation precincts account for 28% of the data, of which 76% are in Manhattan. The top 10 violation codes account for 75% of the data. Below, bar plots display citation counts by top 10 precincts and violation codes for the months December 2014, and January, February, June, July, October

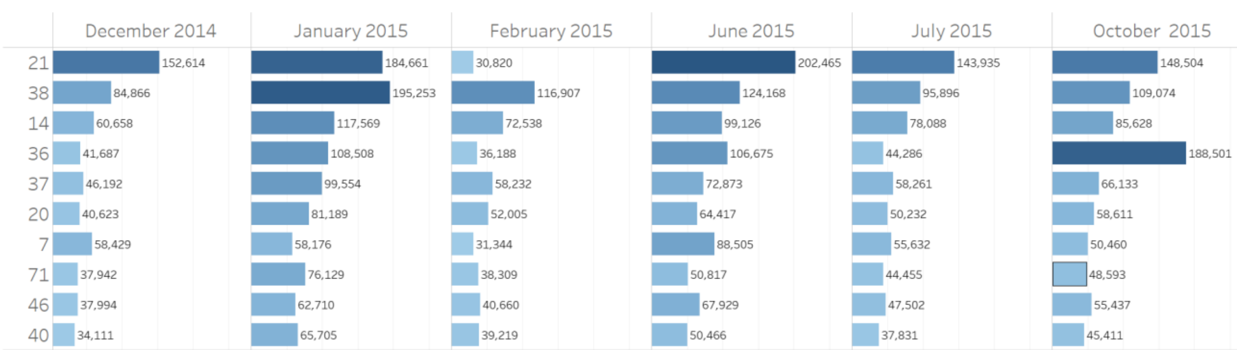
2015. Looking at the precincts, there does not appear to be an unusual change in which precincts are receiving more or less citations. The spike in January is noticeable. The violation code counts do show some unusual activity. In January 2015 there was a substantial increase in the number of citations for violation code 38, parking meter expired. In February 2015 there was a large drop in citation counts for violation code 21, street cleaning. And, in October 2015 there was a substantial increase in citations for violation code 36, speeding in a school zone.

The decrease in citations for violation code 21 in February could possibly be explained by inclement weather. If snowfall accumulation or excess bad Winter weather were present, street cleaning may not have been done on a regular schedule. The increase in counts for violation code 36 in October 2015 might be associated with back-to-school time in the Fall; law enforcement may have paid more attention to driver's speed in school zones. Towards the end of January 2015, there was a record blizzard that swept the North Eastern United States. The spike of citation counts in January 2015 may have been the result of unusual driving and parking activity within the city due to inclement weather.

Citation Counts by Precinct, Dec 2014, Jan, Feb, June, July, Oct 2015



Citation Counts by Violation Code, Dec 2014, Jan, Feb, June, July, Oct 2015



Association Rule Mining

For this project, association rule mining was used to perform an exploratory analysis. The goal was to identify interesting patterns between the type of vehicle and its color among vehicles with parking violations.

The Frequent-Pattern Growth (FP-Growth) algorithm was used to build a set of frequent item sets and corresponding association rules. A minimum support of .001 and a minimum confidence of .01 were used when building the itemset. Support is the fraction of instances of the observed pattern over the total number of observations. Confidence is the fraction of the overserved pattern over the total number of observations of the antecedent. The minimum support and minimum confidence are the minimum acceptable levels for the respective parameter, for a rule to be included in the output. Due to the high volume of colors and vehicle types, low parameter values were needed to explore the resulting rules. It should be noted that vehicle_color contains many repetitive values, most likely due to inconsistent reporting, such as "BRN" and "BROWN". This most likely lowered the support and confidence of the redundant rules.

Association Rules of Vehicle Type and Vehicle Color

Antecedent	**Consequent**	**Confidence**	**Lift**
DELV	BRN	0.0557	5.4520
BRN	DELV	0.4835	5.4520
DELV	BROWN	0.1894	4.5994
BROWN	DELV	0.4079	4.5994
BR	DELV	0.4049	4.5648
DELV	BR	0.0754	4.5648
UTIL	WHITE	0.3273	1.9923
SUBN	BLACK	0.1382	1.3405
BLACK	SUBN	0.4056	1.3405

The results show the strongest association between commercial vehicles and relatively uncommon colors. The best example of this can be seen in the two rules with the highest lift, shown above. These rules show that delivery vehicles are associated with brown and vice-versa. Brown is not a common color for a vehicle, and one of the most prominent delivery services, UPS, uses brown trucks to make their deliveries. Similarly, utility vehicles are associated with white. In this case, this association is one way, most likely because white is common color for several different types of vehicles beyond utility vehicles. The strongest association for passenger vehicles is between black and suburban vehicles reflexively. No rules had an antecedent size greater than one.

Violation Code Prediction

We found the top 10 violation code count which count for the 75% of all tickets. Then we picked 3 violation code for data prediction: Street Cleaning, General No Standing and Fire Hydrant Violation. We have developed two prediction model by using random forest algorithm.

1. <u>Violation Code Prediction for street cleaning (21) and general no standing (38)</u>
   a. **Pre-process data:**

   We put top 10,000 rows from 2014 violation data to a new data table. There are 43 columns in the original dataset and then we found some columns having too many empty values and some columns having very wide data range. We pre-process the data based on the phase I work to get categorical data:

- Only include the data has violation code = 21 or 38;
- Remove the columns which are not useful in the prediction model: such as summons_number, Plate_ID, House_number, Street_Name, etc.
- Redefine the column “Registration_State”: Since 80% tickets were issued to the vehicles registered in New York or New Jersey, we set the value of registration state column as NY, NJ, Others;
- Redefine the column “Plate_Type”: Since 90% tickets were issued to the vehicles having passenger or commercial plate type, we set the value of plate type as PAT, COM, Others;
- Redefine the column “Issuing_ Agency”: P, S, O(others);
- Redefine the column “vehicle_ body_ type”: SDN, SUBN, VAN, DELV and others;

b. Random Forest Prediction Algorithm

After we pre-process the data, we split the dataframe to training data frame and testing data frame.

```
df_list <- randomSplit(violation_df, c(7,3), 2)

violation_training_df <- df_list[[1]]

violation_testing_df <- df_list[[2]]
```

Then used random forest algorithm to build the prediction model by using the following 4 features:

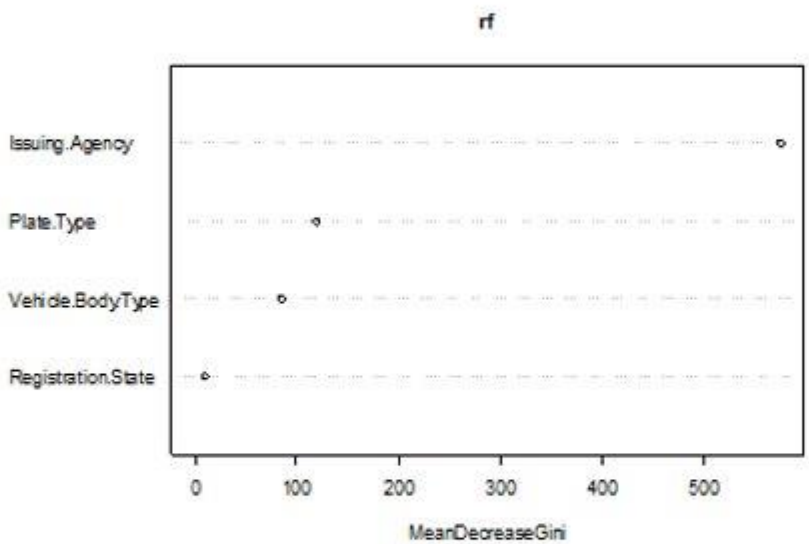
Registration State + Plate Type + Vehicle Body Type + Issuing Agency

Since we use the small data size and most of our data is categorical data, we use the default settings in Random Forest function so the trees can grow to the maximum possible.

We received the following evaluation result:

Accuracy = 0.9018405 ; Precision = 0;9673469; Recall = 0.8555956 (Code 21 as 1, code 38 as 0)

From the variables importance plot, the Issuing Agency makes the most contribution to this model and registration state has least contribution to this model.



2. <u>2017 Violation Code Prediction</u>

   The data was grouped on Violation Code and we have tried to predict the top two violations (Street-Sweeping Violation Prediction, Fire Hydrant Violation Prediction) that occurred in 2017 using a model which was trained on a variety of attributes. The model was trained on the first 1M (for street sweeping violation) and first 10M (for fire hydrant violation) data sets of 2016 violations.

   a. **Street-Sweeping Violation Prediction**

      The violation code for this type of violation is 21. Street-Sweeping Violation is the one that has been observed the most in both 2016 data set and 2017 data set. Street-Sweeping Violation ticket is issued for vehicles parked on a certain location during the time allotted for cleaning the street. Random Forest algorithm has been used to predict if this violation of type 21 will occur or not based on the other factors such as street_code1, street_code2, street_code3, violation location, issuer code and violation precinct attributes. The model was trained on the first 1M rows of 2016 data set and tested on the first 1M rows of 2017 data set.

      Occurrences of Street-Sweeping violation in the training dataset: 24336

      The following are the results of the algorithm on the testing data set

Precision	0.8
Recall	0.6

```
-----------+------------------+------------------+---------+
|summons_number|violation_location|issuer_code|violation_precinct|violation_code|violation_in_front_of_or_opposite|street_code1|stre
treet_code3|       rawPrediction|         probability|prediction|
+--------------+------------------+-----------+------------------+--------------+---------------------------------+------------+----
-----------+------------------+------------------+---------+
|    1413609545|                71|     960290|                71|           0.0|                                F|       54070|
   54930|[14.8406638026338...|[0.74203319013169...|       0.0|
|    1407740258|               106|     960979|               106|           0.0|                                 |           0|
   40404|[17.1788485274854...|[0.85894242637427...|       0.0|
|    1416492320|                44|     905733|                44|           1.0|                                F|       53620|
   74260|[18.8450299372093...|[0.94225149686046...|       0.0|
|    1413656420|                73|     960758|                73|           0.0|                                F|       59630|
   82230|[15.0471147697229...|[0.75235573848614...|       0.0|
|    1416638830|                17|     940179|                17|           0.0|                                O|       17650|
   10010|[19.2450291131398...|[0.96225145565699...|       0.0|
|    1419707358|                60|     589383|                60|           1.0|                                O|       28830|
   23030|[1.90856235401961...|[0.09542811770098...|       1.0|
|    1416527722|                45|          0|                45|           0.0|                                F|       11620|
   11130|[10.9411225180355...|[0.54705612590177...|       0.0|
|    1418809688|                32|     958723|                32|           0.0|                                O|       36770|
   13510|[18.7774345844962...|[0.93887172922481...|       0.0|
|    1416140300|                49|     535013|                49|           1.0|                                F|        9620|
   20520|[1.91308127018710...|[0.09565406350935...|       1.0|
|    1418609274|                18|     926685|                18|           0.0|                                O|           0|
       0|[19.2641563008996...|[0.96320781504498...|       0.0|
|    1418138575|                62|     981230|                62|           1.0|                                F|       10430|
   64730|[15.8701759957844...|[0.79350879978922...|       0.0|
|    1405823926|               100|     938358|               100|           0.0|                                F|       31640|
   15450|[18.3990373421482...|[0.91995186710741...|       0.0|
|    1400876217|               100|     539527|               100|           1.0|                                F|       33290|
   60820|[5.52580779542544...|[0.27629038977127...|       1.0|
|    1417599716|                24|     950441|                24|           0.0|                                F|       25690|
   35790|[18.8505526017758...|[0.94252763008879...|       0.0|
|    1414702700|                72|     534939|                72|           0.0|                                O|        5010|
    8980|[2.22385966479100...|[0.11119298323955...|       1.0|
|    1408575656|                90|     350318|                90|           0.0|                                 |           0|
       0|[16.6955940990578...|[0.83477970495289...|       0.0|
```

b. Fire Hydrant Violation Prediction

The violation code for this type of violation is 40. Fire Hydrant Violation is the second most observed violation in both 2016 data set and 2017 data set. Fire Hydrant Violation is issued for vehicles that are parked at a proximity of less that 15m from the fire hydrant. Random Forest algorithm has been used to predict if this violation of type 40 will occur based on the other factors such as street_code1, street_code2, street_code3, violation in front of or opposite of, violation location, issuer code and violation precinct attributes. The model was trained on the first 10M rows of 2016 data set and tested on the first 10M rows of 2017 data set.

Occurrences of Fire Hydrant violation in the training dataset: 83143

The following are the results of the algorithm on the testing data set

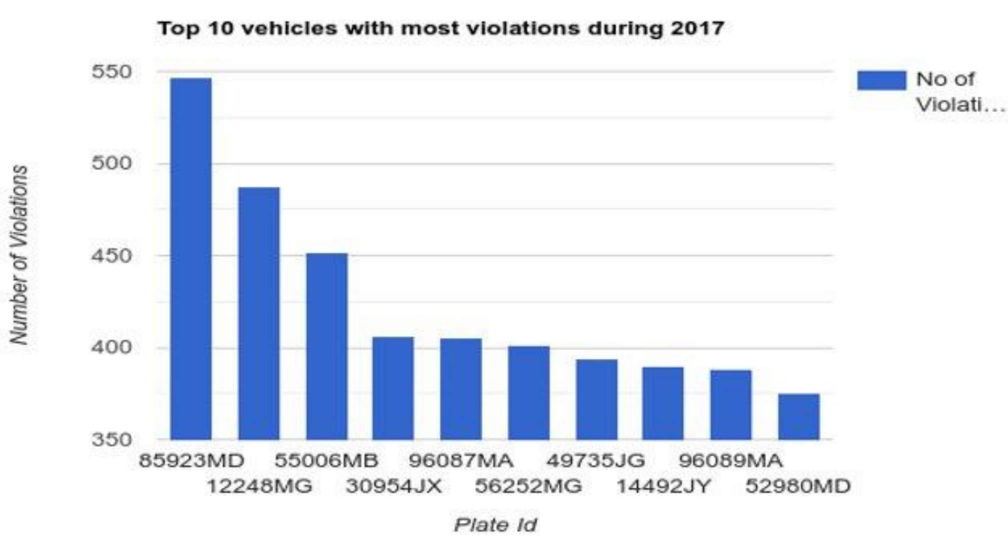
Precision	0.6
Recall	0.9

```
|summons_number|violation_location|issuer_code|violation_precinct|violation_code|violation_in_front_of_or_opposite|street_code1|street_code2|s
treet_code3|       rawPrediction|         probability|prediction|
+--------------+------------------+-----------+------------------+--------------+---------------------------------+------------+------------+-
-----------+--------------------+--------------------+----------+
|    1413609545|                71|     960290|                71|           1.0|                                F|       54070|       39430|
      54930|[18.4410514948202...|[0.92205257474101...|       0.0|
|    1407740258|               106|     960979|               106|           0.0|                                 |           0|       40404|
      40404|[18.4410514948202...|[0.92205257474101...|       0.0|
|    1416492320|                44|     905733|                44|           0.0|                                F|       53620|       74250|
      74260|[18.4410514948202...|[0.92205257474101...|       0.0|
|    1413656420|                73|     960758|                73|           1.0|                                F|       59630|       73470|
      82230|[18.4410514948202...|[0.92205257474101...|       0.0|
|    1416638830|                17|     940179|                17|           0.0|                                O|       17650|       10110|
      10010|[18.4410514948202...|[0.92205257474101...|       0.0|
|    1419707358|                60|     589383|                60|           0.0|                                O|       28830|       67780|
      23030|[18.4410514948202...|[0.92205257474101...|       0.0|
|    1416527722|                45|          0|                45|           0.0|                                F|       11620|       12550|
      11130|[18.4410514948202...|[0.92205257474101...|       0.0|
|    1418809688|                32|     958723|                32|           1.0|                                O|       36770|       10810|
      13510|[18.4410514948202...|[0.92205257474101...|       0.0|
|    1416140300|                49|     535013|                49|           0.0|                                F|        9620|       15620|
      20520|[18.4410514948202...|[0.92205257474101...|       0.0|
|    1418609274|                18|     926685|                18|           0.0|                                O|           0|           0|
          0|[18.4410514948202...|[0.92205257474101...|       0.0|
|    1418138575|                62|     981230|                62|           0.0|                                F|       10430|        6680|
      64730|[18.4410514948202...|[0.92205257474101...|       0.0|
|    1405823926|               100|     938358|               100|           0.0|                                F|       31640|       15425|
      15450|[18.4410514948202...|[0.92205257474101...|       0.0|
|    1400876217|               100|     539527|               100|           0.0|                                F|       33290|       60790|
      60820|[18.4410514948202...|[0.92205257474101...|       0.0|
|    1417599716|                24|     950441|                24|           0.0|                                F|       25690|       35770|
      35790|[18.4410514948202...|[0.92205257474101...|       0.0|
|    1414702700|                72|     534939|                72|           0.0|                                O|        5010|        8930|
       8980|[18.4410514948202...|[0.92205257474101...|       0.0|
|    1408575656|                90|     350318|                90|           0.0|                                 |           0|           0|
```

Vehicles with the greatest number of violations in 2017

The 2017 dataset was grouped by the plate id and counted for unique summons number. This led us to finding which was interesting. There were vehicles that received more than 1 parking ticket a day. The vehicle with the greatest number of violations is '85923MD'. When looked up for the type of violations, that were recorded the most, for this vehicle, it is seen that "Parking Standing or parking on the roadway side of a vehicle stopped" was the one that was most prevalent.

One assumption that we can infer from this is, this vehicle could belong to online food ordering and catering agencies like Dominos, Pizza hut who strive to deliver food on time. Delivering food on time to keep the delivery promise might mean more to them than getting a few parking tickets a day.



Appendix

Original variables from CSV files

1. Summons Number 2. Plate ID 3. Registration State 4. Plate Type 5. Issue Date 6. Violation Code 7. Vehicle Body Type 8. Vehicle Make 9. Issuing Agency 10. Street Code1 11. Street Code2 12. Street Code3 13. Vehicle Expiration Date 14. Violation Location 15. Violation Precinct 16. Issuer Precinct 17. Issuer Code 18. Issuer Command 19. Issuer Squad 20. Violation Time 21. Time First Observed 22. Violation County 23. Violation In Front Of Or Opposite 24. House Number 25. Street Name 26. Intersecting Street	27. Date First Observed 28. Law Section 29. Sub Division 30. Violation Legal Code 31. Days Parking In Effect 32. From Hours In Effect 33. To Hours In Effect 34. Vehicle Color 35. Unregistered Vehicle? 36. Vehicle Year 37. Meter Number 38. Feet From Curb 39. Violation Post Code 40. Violation Description 41. No Standing or Stopping Violation 42. Hydrant Violation 43. Double Parking Violation 44. Latitude 45. Longitude 46. Community Board 47. Community Council 48. Census Tract 49. BIN 50. BBL 51. NTA

SQL Code

```
--- Load CSV files into Hadoop File System
hdfs dfs -put /pylon5/cc5phlp/ever930/data/project/NYC_Parking_Citations/Parking_Violations_Issued_-
_Fiscal_Year_2014__August_2013___June_2014_.csv
hdfs dfs -put /pylon5/cc5phlp/ever930/data/project/NYC_Parking_Citations/Parking_Violations_Issued_-_Fiscal_Year_2015.csv
hdfs dfs -put /pylon5/cc5phlp/ever930/data/project/NYC_Parking_Citations/Parking_Violations_Issued_-_Fiscal_Year_2016.csv
hdfs dfs -put /pylon5/cc5phlp/ever930/data/project/NYC_Parking_Citations/Parking_Violations_Issued_-_Fiscal_Year_2017.csv

--- Create temporary table
create table if not exists nyc_parking_violations_temp
(summons_number int,
plate_ID varchar(10),
```

```
registration_state char(2),
plate_type varchar(3),
issue_date string,
violation_code int,
vehicle_body_type varchar(10),
vehicle_make varchar(10),
issuing_agency char(1),
street_code1 int,
street_code2 int,
street_code3 int,
vehicle_expiration_date int,
violation_location int,
violation_precinct int,
issuer_precinct int,
issuer_code int,
issuer_command varchar(10),
issuer_squad varchar(10),
violation_time varchar(10),
time_first_observed varchar(10),
violation_county char(5),
violation_in_front_of_or_opposite char(1),
house_number varchar(10),
street_name varchar(50),
intersecting_street varchar(50),
date_first_observed int,
law_section int,
sub_division varchar(2),
violation_legal_code varchar(1),
days_parking_in_effect varchar(10),
from_hours_in_effect varchar(10),
to_hours_in_effect varchar(10),
vehicle_color char(5),
unregistered_vehicle int,
vehicle_year int,
meter_number varchar(10),
feet_from_curb int,
violation_post_code varchar(5),
violation_description varchar(50),
no_standing_or_stopping_violation boolean,
hydrant_violation boolean,
double_parking_violation boolean,
latitude boolean,
longitude boolean,
community_board boolean,
community_council boolean,
census_tract boolean,
BIN boolean,
BBL boolean,
NTA boolean)
partitioned by(
year int,
month int,
day int,
hour int,
violation_code int,
```

```
issuer_precinct int
)
row format delimited
fields terminated by ','
lines terminated by '\n'
stored as textfile
tblproperties ("skip.header.line.count"="1");

--- Load data into temporary table
load data inpath "Parking_Violations_Issued_-_Fiscal_Year_2014__August_2013___June_2014_.csv" into table
nyc_parking_violations_temp;
load data inpath "Parking_Violations_Issued_-_Fiscal_Year_2015.csv" into table nyc_parking_violations_temp;
load data inpath "Parking_Violations_Issued_-_Fiscal_Year_2016.csv" into table nyc_parking_violations_temp;
load data inpath "Parking_Violations_Issued_-_Fiscal_Year_2017.csv" into table nyc_parking_violations_temp;

--- Create smaller partitioned pivot table for analysis
create table if not exists nyc_parking_violations
(summons_number int,
plate_ID varchar(10),
violation_code int,
violation_location int,
violation_precinct int,
issuer_precinct int,
issuer_command varchar(10),
issuer_squad varchar(10),
street_name varchar(50),
vehicle_color char(5),
vehicle_make varchar(10),
vehicle_body_type varchar(10),
vehicle_year int,
violation_description varchar(50),
day int,
hour int
)
partitioned by(
year int,
month int
)
row format delimited
fields terminated by ','
lines terminated by '\n'
stored as textfile;

--- Code for dynamic partitioning
set hive.exec.dynamic.partition =  TRUE;
set hive.exec.dynamic.partition.mode = nonstrict;
set hive.exec.max.dynamic.partitions = 3000;
set hive.exec.max.dynamic.partitions.pernode = 3000;

--- Load data into smaller partitioned pivot table
insert overwrite table nyc_parking_violations
partition(
year,
month
)
```

```
select
summons_number,
plate_ID,
violation_code,
violation_location,
violation_precinct,
issuer_precinct,
issuer_command,
issuer_squad,
street_name,
vehicle_color,
vehicle_make,
vehicle_body_type,
vehicle_year,
violation_description,
day(to_date(from_unixtime(unix_timestamp(issue_date, 'MM/dd/yyyy')))),
(case
when (violation_time regexp '[0-1][0-9][0-9][0-9][A-Z]') and (substring(violation_time,5,5) == 'A') and
(substring(violation_time,1,2) == '12') then cast('0' as int)
when (violation_time regexp '[0-1][0-9][0-9][0-9][A-Z]') and (substring(violation_time,5,5) == 'P') and
(substring(violation_time,1,2) == '12') then cast(substring(violation_time,1,2) as int)
when (violation_time regexp '[0-1][0-9][0-9][0-9][A-Z]') and (substring(violation_time,5,5) == 'P') and
(substring(violation_time,1,2) != '12') then cast(substring(violation_time,1,2) as int) + 12
when (violation_time regexp '[0-1][0-9][0-9][0-9][A-Z]') and (substring(violation_time,5,5) == 'A') and
(substring(violation_time,1,2) != '12') then cast(substring(violation_time,1,2) as int)
end),
year(to_date(from_unixtime(unix_timestamp(issue_date, 'MM/dd/yyyy')))),
month(to_date(from_unixtime(unix_timestamp(issue_date, 'MM/dd/yyyy'))))
from nyc_parking_violations_temp
where to_date(from_unixtime(unix_timestamp(issue_date, 'MM/dd/yyyy'))) between '2013-08-01' and '2017-06-30';
```

R code:

Violation Code Prediction for street cleaning (21) and general no standing (38)

Use jli;

```
# Get categorical data
insert overwrite table parking_violations_14
select top 100000
(case when registration_state='NY' then 'NY' when registration_state='NJ' then 'NJ' else 'Others' end),
(case when plate_type = 'PAT' then 'PAT' when plate_type = 'COM' then 'COM' else 'Others' end),
issue_date, violation_code,
(case when vehicle_body_type in ('SDN', 'SUBN', 'VAN', 'DELV') then vehicle_body_type else 'others' end) , vehicle_make,
(case when issuing_agency = 'P' then 'P' when issuing_agency='S' then 'S' else 'O' end),
vehicle_expiration_date, violation_location, violation_precinct, issuer_precinct, issuer_code, issuer_command,
from nyc_parking_violations_temp
where (violation_code = ‘21’ or violation_code = ‘38’ ) and to_date(from_unixtime(unix_timestamp(issue_date,
'MM/dd/yyyy'))) between '2014-01-01' and '2014-12-31';

# Evaluation  code = 21 -> 1 code = 38 -> 0
TP <- nrow(where(Output, Output$violation_code == '21’ & Output$prediction == '21’ ))
FP <- nrow(where(Output, Output$violation_code == ‘38’ & Output$prediction == '21’))
```

FN <- nrow(where(Output, Output$violation_code == '21’ & Output$prediction == ‘38’))
Precision = TP/(TP+FP)
Recall = TP/(TP+FN)

2017 Violation code prediction

#Predicting the violation type 21(Street Sweeping) using Random Forest
sql("use smurugesan")
nyc_parking_data <- sql("SELECT summons_number, violation_location, issuer_code, violation_precinct, violation_code, violation_in_front_of_or_opposite, street_code1, street_code2, street_code3 FROM nyc_parking_violations_temp_new limit 1000000")

nyc_parking_data_2017 <- sql("SELECT summons_number, violation_location, issuer_code, violation_precinct, violation_code, violation_in_front_of_or_opposite, street_code1, street_code2, street_code3 FROM nyc_parking_violations_temp_new_2017 limit 1000000")

nyc_parking_data <- dropna(nyc_parking_data)
nyc_parking_data_2017 <- dropna(nyc_parking_data_2017)

nyc_parking_data$violation_code <- ifelse(nyc_parking_data$violation_code ==21, 1, 0)
nyc_parking_data_2017$violation_code <- ifelse(nyc_parking_data_2017$violation_code ==21, 1, 0)

model <- spark.randomForest(nyc_parking_data, violation_code ~violation_location +issuer_code +violation_precinct +violation_in_front_of_or_opposite +street_code1 +street_code2 +street_code3, "classification", numTrees = 20, maxDepth = 5)
Output <- predict(model, nyc_parking_data_2017)
showDF(Output)

TP <- nrow(where(Output, Output$violation_code == 1 & Output$prediction == 1))
FP <- nrow(where(Output, Output$violation_code == 0 & Output$prediction == 1))
Precision = TP/(TP+FP)
FN <- nrow(where(Output, Output$violation_code == 1 & Output$prediction == 0))
Recall = TP/(TP+FN)
cat("Precision of the model = ", Precision)
cat("Recall of the model = ", Recall)

#Predicting the violation type 40 (Fire Hydrant Violation) using Random Forest
nyc_parking_data <- sql("SELECT summons_number, violation_location, issuer_code, violation_precinct, violation_code, violation_in_front_of_or_opposite, street_code1, street_code2, street_code3 FROM nyc_parking_violations_temp_new limit 10000000")

nyc_parking_data_2017 <- sql("SELECT summons_number, violation_location, issuer_code, violation_precinct, violation_code, violation_in_front_of_or_opposite, street_code1, street_code2, street_code3 FROM nyc_parking_violations_temp_new_2017 limit 10000000")

nyc_parking_data <- dropna(nyc_parking_data)
nyc_parking_data_2017 <- dropna(nyc_parking_data_2017)

nyc_parking_data$violation_code <- ifelse(nyc_parking_data$violation_code ==40, 1, 0)
nyc_parking_data_2017$violation_code <- ifelse(nyc_parking_data_2017$violation_code ==40, 1, 0)

model <- spark.randomForest(nyc_parking_data, violation_code ~violation_location +issuer_code +violation_precinct +violation_in_front_of_or_opposite +street_code1 +street_code2 +street_code3, "classification", numTrees = 20, maxDepth = 5)

```
Output <- predict(model, nyc_parking_data_2017)
showDF(Output)

TP <- nrow(where(Output, Output$violation_code == 1 & Output$prediction == 1))
FP <- nrow(where(Output, Output$violation_code == 0 & Output$prediction == 1))
Precision = TP/(TP+FP)
FN <- nrow(where(Output, Output$violation_code == 1 & Output$prediction == 0))
Recall = TP/(TP+FN)
cat("Precision of the model = ", Precision)
cat("Recall of the model = ", Recall)


#Top 10 vehicles with most violations for 2017
showDF(agg(groupBy(nyc_parking_data_2017, nyc_parking_data_2017$violation_code), summons_number="count"))
```

Association Rule Mining

```
sql("use lebow")

#load dataframe
df <- sql("select * from nyc_parking_violations where year != 2014 and vehicle_year != 0")
df <- dropna(df)

#collect and split dataframe into single-column list format for fpGrowth
df_item <- agg(groupBy(df, df$summons_number), vehicle_body_type = "collect_set", vehicle_color = "collect_set")
colnames(df_item) <- c("id", 'type', 'color')
df_item <- filter(df_item, df_item$type != 1)
items <- concat_ws(sep = ",", df_item$type, df_item$color)
df_fpm <- selectExpr(createDataFrame(items), "split(items, ',') AS items")

#run fpGrowth and save frequent itemset
fpm <- spark.fpGrowth(df_fpm, itemsCol = "items", minSupport=.001, minConfidence=.01)
showDF(spark.freqItemsets(fpm))
freqItems <- spark.freqItemsets(fpm)
write.df(freqItems, "freqItems_ay")
freqItems_ay <- read.df("freqItems_ay")

#sort and save rules
rules <- spark.associationRules(fpm)
rules_sorted <- orderBy(rules, -rules$lift)
showDF(rules_sorted, 20)
write.df(rules_sorted, "rules_ay")
rules_ay <- read.df("rules_ay")

#filter for rules with antecdents sizes of at least two
#sort and save filtered rules
rules2 <- spark.associationRules(fpm)
rules2 <- where(rules2, size(rules2$antecedent)>1)
rules2_sorted <- orderBy(rules2, -rules2$lift)
showDF(rules2_sorted, 20)
write.df(rules2_sorted, "rules2_ay")
rules2_ay <- read.df("rules2_ay")
```