Data Analysis Report

Authors

Rakshit Sareen (rs5606)
Saurabh Mahajan (sm6921)
Sneha Ghosh (sg3533)

Abstract

Data cleaning plays an important role in data analysis applications. Data cleaning was done by finding missing values, incorrect values using type checks and other validations and filtering using reference dataset. We used these techniques to clean data for all columns and then merge the cleaned and corrected columns to obtain the final dataset. This dataset was used for data analysis and data exploration.
Data analysis helped us visualize the crimes that happened in New York for past 10 years. We aggregated data by year, month, day, boroughs, offense level, precincts, jurisdictions to visualize crimes trend.

Introduction

NYC is the most populated city in US. People from diverse background live here and it is crucial to analyse crimes and try to take actions for safety. Data analysis will allow us to visualize data trends by aggregating data using various attributes. We could analyze amount of crimes happening yearly, monthly, in different boroughs and also the types of crimes which happen.

Since the dataset consists of around 5 million rows, we have utilized the big data Hadoop framework for data analysis. Before data analysis, data cleaning is done to remove invalid data from the dataset so that our analysis does not get affected by invalid data. Some of the data was also corrected to reduce the incorrect data wherever possible. We used pyspark for data cleaning and data analysis. Also used Matplotlib and pandas to generate plots.

GitHub Repository -

https://github.com/snehaghosh91/BigDataProject

Part I - Data quality issues

Results and discussion

- **Column 0**
  This column is a unique identifier for each row.
  Script : col0.py
  The property we checked for this column was basically that every value in the column is unique. One output file generated : **col1_statistics.out** which contains the count for the valid and invalid keys.
  INVALID: 0
  VALID : 5580035

- **Column 1 and Column 3 and Column 5**
  These two columns represent the From and To Date of the crime.
  Script : col1_3.py
  The script validates all the dates in the column 1,3 and 5 of the data.
  These were tricky columns as the combination of both to and from columns determine whether it is invalid data or not.
  I decided to mark it invalid only if both the values are empty.
  If one of the values is not empty, then we mark the dates as VALID. We follow the below semantic and mark the dates as EXACT, RANGE, ENDPOINT.

EXACT : If only FROM Date is provided
ENDPOINT : If only TO Date is provided
RANGE : If both dates are provided.

OUTPUT Files:
Col3_valid_data.out : Valid To Dates
Col3_invalid_data.out : Invalid To Dates
Col1_invalid_data.out : Invalid From Dates
Col1_valid_data.out : Valid From Dates
Col1_3_valid_data.out : Corrected data
exactDates.out : Columns which have only from date
rangeDates.out : Columns which have both from and to date
endPointDates.out : Columns which have only to date

- **Column 2 and Column 4**
  These two columns represent the from and to time of the crime represented by the row.
  Script : col2-4.py
  The script cleans the data related to time. It also corrects the data in which the time was incorrect, such as 24:00:00, as there is no such time. It has been changed to 00:00:00.
  The tagging for the data is INVALID and VALID. INVALID can be any reasons such as time wrongly recorded, no proper formatting of the time, empty string.

  OUTPUT Files:
  Col2_invalid_data.out : contains invalid from time data
  Col2_valid_data.out : contains valid from time data
  Col2_corrected.out : corrected from time
  Col4_invalid_data.out : contains invalid to time data
  Col4_valid_data.out : contains valid to time data
  Col4_corrected.out : corrected to time
  Col2_statistics.out : statistics for INVALID and VALID from time
  Col4_statistics.out : statistics for INVALID and VALID to time

- **Column 6 - Offense Classification Code**
  Script col6_7.py - Code checks if the classification code is not empty and is a three digit integer.
  3 output files are generated -
  1. **col6_invalid_data.out** - Contains all invalid values in the column.
     Result - There are no invalid values in this column.

2. **col6_valid_data.out** - Contains all valid values in the column. CMPLNT_NUM - unique value to identify rows, KY_CD - column value
   Result -
   Command - head -5 col6_valid_data.out
   ```
   CMPLNT_NUM  KY_CD
   101109527     113
   153401121     101
   569369778     117
   968417082     344
   ```

3. **col6_statistics.out** - Contains statistics about the column like invalid elements count, valid elements count and invalid value which occurs maximum number of times.
   Result -
   Command - head -5 col6_statistics.out
   ```
   INVALID COUNT:  0
   VALID COUNT:    5580035
   ```

- **Column 7 - Description of offense**
  Script col6_7.py - Code checks if the description of offense is empty. Checks if the description is valid for its key by finding the description for each key which has max count and comparing it to the description.
  4 output files are generated -
  1. **col7_invalid_data.out** - Contains all invalid values in the column. CMPLNT_NUM - unique value to identify rows, OFNS_DESC - column value, REASON - tells why value is invalid.
     Result -
     Command - head -5 col7_invalid_data.out
     ```
     CMPLNT_NUM  OFNS_DESC  REASON
     932125924              EMPTY VALUE
     327111538              EMPTY VALUE
     651408610              EMPTY VALUE
     737618153              EMPTY VALUE
     ```

  2. **col7_valid_data.out** - Contains all valid values in the column. CMPLNT_NUM - unique value to identify rows, OFNS_DESC - column value
     Result -
     Command - head -5 col7_valid_data.out
     ```
     CMPLNT_NUM  OFNS_DESC
     710792599     ASSAULT 3 & RELATED OFFENSES
     423573333     SEX CRIMES
     302356516     ROBBERY
     349433504     CRIMINAL MISCHIEF & RELATED OF
     ```

3. **col7_statistics.out** - Contains statistics about the column like invalid elements count, valid elements count and invalid value which occurs maximum number of times.
   Result -
   Command - head -5 col7_statistics.out
   ```
   INVALID COUNT:  22593
   VALID COUNT:    5557442
   MAX INVALID OCCURRENCE          EMPTY VALUE     18892
   ```

4. **col7_corrected.out** - Obtained by merging the valid data with data obtained by correcting invalid data. CMPLNT_NUM - unique value to identify rows, OFNS_DESC - valid/corrected column value
   Result -
   Command - head -5 col7_corrected.out
   ```
   CMPLNT_NUM  OFNS_DESC
   710792599     ASSAULT 3 & RELATED OFFENSES
   423573333     SEX CRIMES
   354521636     GRAND LARCENY OF MOTOR VEHICLE
   827898780     DANGEROUS DRUGS
   ```

- **Column 8 - Internal Classification Code**
  Script col8_9.py - Code checks if the classification code is not empty and is a three digit integer.
  3 output files are generated -
  1. **col8_invalid_data.out** - Contains all invalid values in the column. CMPLNT_NUM - unique value to identify rows, PD_CD - column value, REASON - tells why value is invalid.
     Result -
     Command - head -5 col8_invalid_data.out
     ```
     CMPLNT_NUM  PD_CD  REASON
     153401121              EMPTY VALUE
     940141475              EMPTY VALUE
     586976434              EMPTY VALUE
     388875685              EMPTY VALUE
     ```

  2. **col8_valid_data.out** - Contains all valid values in the column. CMPLNT_NUM - unique value to identify rows, PD_CD - column value
     Result -
     Command - head -5 col8_valid_data.out
     ```
     CMPLNT_NUM  PD_CD
     101109527     729
     569369778     503
     ```

```
968417082	101
641637920	101
```

3. **col8_statistics.out** - Contains statistics about the column like invalid elements count, valid elements count and invalid value which occurs maximum number of times.
   Result -
   Command - head -5 col8_statistics.out
   ```
   INVALID COUNT:        4909
   VALID COUNT: 5575126
   MAX INVALID OCCURRENCE              EMPTY VALUE  4909
   ```

- **Column 9 - Description of internal classification**
  Script col8_9.py - Code checks if the description of internal classification is empty. Checks if the description is valid for its key by finding the description for each key which has max count and comparing it to the description.
  3 output files are generated -
  1. **col9_invalid_data.out** - Contains all invalid values in the column. CMPLNT_NUM - unique value to identify rows, PD_DESC - column value, REASON - tells why value is invalid.
     Result -
     Command - head -5 col9_invalid_data.out
     ```
     CMPLNT_NUM  PD_DESC  REASON
     735957494             EMPTY VALUE
     566621021             EMPTY VALUE
     843046086             EMPTY VALUE
     872932532             EMPTY VALUE
     ```

  2. **col9_valid_data.out** - Contains all valid values in the column. CMPLNT_NUM - unique value to identify rows, PD_DESC - column value
     Result -
     Command - head -5 col9_valid_data.out
     ```
     CMPLNT_NUM  PD_DESC
     423573333     SODOMY 1
     710792599     ASSAULT 3
     827898780     CONTROLLED SUBSTANCE, POSSESSI
     138050170     LARCENY,GRAND FROM BUILDING (NON-RESIDENCE) UNATTENDED
     ```

  3. **col9_statistics.out** - Contains statistics about the column like invalid elements count, valid elements count and invalid value which occurs maximum number of times.
     Result -

Command - head -5 col9_statistics.out

```
INVALID COUNT:        4909
VALID COUNT: 5575126
MAX INVALID OCCURRENCE              EMPTY VALUE  4909
```

4. **col9_corrected.out -** Obtained by merging the valid data with data obtained by correcting invalid data. CMPLNT_NUM - unique value to identify rows, PD_DESC - valid/corrected column value
   Result -
   Command - head -5 col9_corrected.out
   ```
   CMPLNT_NUM   PD_DESC
   710792599    ASSAULT 3
   423573333    SODOMY 1
   827898780    CONTROLLED SUBSTANCE, POSSESSI
   349433504    CRIMINAL MISCHIEF,UNCLASSIFIED 4
   ```

- **Column 10 - Crime was successfully completed or attempted**
  Script col10.py - Code checks if the values in column are not empty and are valid values - ATTEMPTED , COMPLETED.
  3 output files are generated -
  1. **col10_invalid_data.out -** Contains all invalid values in the column. CMPLNT_NUM - unique value to identify rows, CRM_ATPT_CPTD_CD - column value, REASON - tells why value is invalid.
     Result -
     Command - head -5 col10_invalid_data.out
     ```
     CMPLNT_NUM   CRM_ATPT_CPTD_CD  REASON
     448788620              EMPTY VALUE
     363472497              EMPTY VALUE
     559717358              EMPTY VALUE
     181835873              EMPTY VALUE
     ```

  2. **col10_valid_data.out -** Contains all valid values in the column. CMPLNT_NUM - unique value to identify rows, CRM_ATPT_CPTD_CD - column value.
     Result -
     Command - head -5 col10_valid_data.out
     ```
     CMPLNT_NUM   CRM_ATPT_CPTD_CD
     101109527    COMPLETED
     153401121    COMPLETED
     569369778    COMPLETED
     968417082    COMPLETED
     ```

  3. **col10_statistics.out -** Contains statistics about the column like invalid elements count, valid elements count and invalid value which occurs maximum

number of times.
Result -
Command - head -5 col10_statistics.out

```
INVALID COUNT:        7
VALID COUNT: 5580028
MAX INVALID OCCURRENCE              EMPTY VALUE  7
```

- **Column 11 - Level of Offense**
  Script col11.py - Code checks if the values in column are not empty and are valid values - FELONY, MISDEMEANOR, VIOLATION.
  3 output files are generated -
  1. **col11_invalid_data.out** - Contains all invalid values in the column. CMPLNT_NUM - unique value to identify rows, LAW_CAT_CD - column value, REASON - tells why value is invalid.
     Result - There are no invalid values in this column.
  2. **col11_valid_data.out** - Contains all valid values in the column. CMPLNT_NUM - unique value to identify rows, LAW_CAT_CD - column value.
     Result -
     Command - head -5 col11_valid_data.out
     ```
     CMPLNT_NUM  LAW_CAT_CD
     101109527      FELONY
     153401121      FELONY
     569369778      FELONY
     968417082      MISDEMEANOR
     ```
  3. **col11_statistics.out** - Contains statistics about the column like invalid elements count, valid elements count and invalid value which occurs maximum number of times.
     Result -
     Command - head -5 col11_statistics.out
     ```
     INVALID COUNT:        0
     VALID COUNT: 5580035
     ```

- **Column 12 - Jurisdiction responsible for incident.**
  Script col12.py - Code checks if the values in column are not empty and are not invalid values - OTHER (identified from manual analysis).
  3 output files are generated -
  1. **col12_invalid_data.out** - Contains all invalid values in the column. CMPLNT_NUM - unique value to identify rows, JURIS_DESC - shows column value, REASON - tells why value is invalid.
     Result -
     Command - head -5 col12_invalid_data.out
     ```
     CMPLNT_NUM   JURIS_DESC    REASON
     ```

```
675941712      OTHER          INVALID
277565054      OTHER          INVALID
389301033      OTHER          INVALID
514698127      OTHER          INVALID
```

2. **col12_valid_data.out** - Contains all valid values in the column. CMPLNT_NUM - unique value to identify rows, JURIS_DESC - column value.
   Result -
   Command - head -5 col12_valid_data.out
   ```
   CMPLNT_NUM   JURIS_DESC
   101109527      N.Y. POLICE DEPT
   153401121      N.Y. POLICE DEPT
   569369778      N.Y. POLICE DEPT
   968417082      N.Y. POLICE DEPT
   ```

3. **col12_statistics.out** - Contains statistics about the column like invalid elements count, valid elements count and invalid value which occurs maximum number of times.
   Result -
   Command - head -5 col12_statistics.out
   ```
   INVALID COUNT:        14964
   VALID COUNT: 5565071
   MAX INVALID OCCURRENCE    OTHER INVALID          14964
   ```

- **Column 13- Borough name**
  Script col13.py - Code checks if the values in column are not empty and are valid values - QUEENS, MANHATTAN, BRONX, STATEN ISLAND, BROOKLYN.
  3 output files are generated -
  1. **col13_invalid_data.out** - Contains all invalid values in the column. CMPLNT_NUM - unique value to identify rows, BORO_NM - shows column value, REASON - tells why value is invalid.
     Result -
     Command - head -5 col13_invalid_data.out
     ```
     CMPLNT_NUM   BORO_NM     REASON
     187370390              EMPTY VALUE
     284743219              EMPTY VALUE
     219649982              EMPTY VALUE
     699763801              EMPTY VALUE
     ```

  2. **col13_valid_data.out** - Contains all valid values in the column. CMPLNT_NUM - unique value to identify rows, BORO_NM - column value.
     Result -
     Command - head -5 col13_valid_data.out

```
CMPLNT_NUM  BORO_NM
101109527   BRONX
153401121   QUEENS
569369778   MANHATTAN
968417082   QUEENS
```

3. **col13_statistics.out** - Contains statistics about the column like invalid elements count, valid elements count and invalid value which occurs maximum number of times.
   Result -
   Command - head -5 col13_statistics.out

   ```
   INVALID COUNT:      463
   VALID COUNT: 5579572
   MAX INVALID OCCURRENCE          EMPTY VALUE  463
   ```

- **Column 14 - Precinct**
  Script col14.py - Code checks if the values in column are not empty, are integers and are valid values.
  Valid values for Precincts are obtained from this link -
  https://www1.nyc.gov/site/nypd/bureaus/patrol/precincts-landing.page
  Manual Analysis after data cleaning - Analysing the data reference three values were not integers - Midtown South Precinct, Midtown North Precinct and Central Park Precinct. These need to be converted into integers as the values in crime dataset were all integers. Before converting these values the data was wrongly marked as invalid so adding the mapping of String precincts to integers helped fix the issue. **Mapping String to integers**- Midtown South Precinct - 14, Midtown North Precinct - 18, Central Park Precinct - 22

  3 output files are generated -
  1. **col14_invalid_data.out** - Contains all invalid values in the column. CMPLNT_NUM - unique value to identify rows, ADDR_PCT_CD - shows column value, REASON - tells why value is invalid.
     Result -
     Command - head -5 col14_invalid_data.out

     ```
     CMPLNT_NUM  ADDR_PCT_CD   REASON
     594173303           EMPTY VALUE
     713215539           EMPTY VALUE
     602049379           EMPTY VALUE
     989678893           EMPTY VALUE
     ```

  2. **col14_valid_data.out** - Contains all valid values in the column. CMPLNT_NUM - unique value to identify rows, ADDR_PCT_CD - column value.

Result -
Command - head -5 col14_valid_data.out

```
CMPLNT_NUM  ADDR_PCT_CD
101109527     44
153401121     103
569369778     28
968417082     105
```

3. **col14_statistics.out** - Contains statistics about the column like invalid elements count, valid elements count and invalid value which occurs maximum number of times.
Result -
Command - head -5 col14_statistics.out

```
INVALID COUNT:        390
VALID COUNT: 5579645
MAX INVALID OCCURRENCE             EMPTY VALUE  390
```

- **Column 15 - Specific location of occurrence**
Script col15.py - Code checks if the values in column are not empty and are valid values - FRONT OF, INSIDE, OPPOSITE OF, REAR OF, OUTSIDE.
3 output files are generated -
    1. **col15_invalid_data.out** - Contains all invalid values in the column. CMPLNT_NUM - unique value to identify rows, LOC_OF_OCCUR_DESC - shows column value, REASON - tells why value is invalid.
    Result -
    Command - head -5 col15_invalid_data.out

```
CMPLNT_NUM  LOC_OF_OCCUR_DESC        REASON
569369778             EMPTY VALUE
898496564             EMPTY VALUE
566081066             EMPTY VALUE
584555879             EMPTY VALUE
```

    2. **col15_valid_data.out -** Contains all valid values in the column. CMPLNT_NUM - unique value to identify rows, LOC_OF_OCCUR_DESC - column value.
    Result -
    Command - head -5 col15_valid_data.out

```
CMPLNT_NUM  LOC_OF_OCCUR_DESC
101109527     INSIDE
153401121     OUTSIDE
968417082     INSIDE
641637920     FRONT OF
```

3. **col15_statistics.out** - Contains statistics about the column like invalid elements count, valid elements count and invalid value which occurs maximum number of times.
   Result -
   Command - head -5 col15_statistics.out
   ```
   INVALID COUNT:        1223605
   VALID COUNT: 4356430
   MAX INVALID OCCURRENCE              EMPTY VALUE  1223392
   ```

- **Column 16 - Specific description of premises**
  Script col16.py -Code checks if the values in column are not empty and are not invalid values - OTHER (identified from manual analysis).
  3 output files are generated -
  1. **col16_invalid_data.out** - Contains all invalid values in the column. CMPLNT_NUM - unique value to identify rows, PREM_TYP_DESC - shows column value, REASON - tells why value is invalid.
     Result -
     Command - head -5 col16_invalid_data.out
     ```
     CMPLNT_NUM   PREM_TYP_DESC   REASON
     153401121                    EMPTY VALUE
     569369778      OTHER         INVALID
     641637920      OTHER         INVALID
     340513307      OTHER         INVALID
     ```

  2. **col16_valid_data.out** - Contains all valid values in the column. CMPLNT_NUM - unique value to identify rows, PREM_TYP_DESC - column value.
     Result -
     Command - head -5 col16_valid_data.out
     ```
     CMPLNT_NUM   PREM_TYP_DESC
     101109527      BAR/NIGHT CLUB
     968417082      RESIDENCE-HOUSE
     365661343      DRUG STORE
     608231454      STREET
     ```

  3. **col16_statistics.out** - Contains statistics about the column like invalid elements count, valid elements count and invalid value which occurs maximum number of times.
     Result -
     Command - head -5 col16_statistics.out
     ```
     INVALID COUNT:        183608
     VALID COUNT: 5396427
     MAX INVALID OCCURRENCE    OTHER          INVALID         148410
     ```

- **Column 17 - Name of NYC park**
  Script col17.py - Code checks if the values in column are not empty.
  3 output files are generated -
  1. **col17_invalid_data.out** - Contains all invalid values in the column. CMPLNT_NUM - unique value to identify rows, PARKS_NM - shows column value, REASON - tells why value is invalid.
     Result -
     Command - head -5 col17_invalid_data.out
     ```
     CMPLNT_NUM  PARKS_NM  REASON
     101109527              EMPTY VALUE
     153401121              EMPTY VALUE
     569369778              EMPTY VALUE
     968417082              EMPTY VALUE
     ```

  2. **col17_valid_data.out** - Contains all valid values in the column. CMPLNT_NUM - unique value to identify rows, PARKS_NM - column value.
     Result -
     Command - head -5 col17_valid_data.out
     ```
     CMPLNT_NUM  PARKS_NM
     590638275     MADISON SQUARE PARK
     557672328     COLUMBUS PARK AT MANHATTAN
     253843712     ARCILLA PLAYGROUND
     189160748     CENTRAL PARK
     ```

  3. **col17_statistics.out** - Contains statistics about the column like invalid elements count, valid elements count and invalid value which occurs maximum number of times.
     Result -
     Command - head -5 col17_statistics.out
     ```
     INVALID COUNT:        5567497
     VALID COUNT: 12538
     MAX INVALID OCCURRENCE              EMPTY VALUE  5567497
     ```

- **Column 18 - Name of NYCHA housing development**
  Script col18.py - Code checks if the values in column are not empty, and are valid values.
  Valid values for **Housing development** are obtained from this link - http://www1.nyc.gov/site/nycha/about/developments.page (Heading - Development Maps)
  3 output files are generated -
  1. **col18_invalid_data.out** - Contains all invalid values in the column. CMPLNT_NUM - unique value to identify rows, HADEVELOPT - shows column value, REASON - tells why value is invalid.

Result -
Command - head -5 col18_invalid_data.out

```
CMPLNT_NUM   HADEVELOPT   REASON
101109527               EMPTY VALUE
153401121               EMPTY VALUE
569369778               EMPTY VALUE
968417082               EMPTY VALUE
```

2. **col18_valid_data.out** - Contains all valid values in the column. CMPLNT_NUM - unique value to identify rows, HADEVELOPT - column value.
Result -
Command - head -5 col18_valid_data.out

```
CMPLNT_NUM   HADEVELOPT
251546004      MARCY
824663386      MORRIS I
978579954      FARRAGUT
609719707      BORINQUEN PLAZA I
```

3. **col18_statistics.out** - Contains statistics about the column like invalid elements count, valid elements count and invalid value which occurs maximum number of times.
Result -
Command - head -5 col18_statistics.out

```
INVALID COUNT:        5359618
VALID COUNT: 220417
MAX INVALID OCCURRENCE             EMPTY VALUE  5302218
```

- **Column 19 - X-Coordinate of the location of crime**
Script col19-22.py - Code checks if the values in column are not empty and integer values with the New York City limits.
3 output files are generated -
    1. **col19_invalid_data.out** - Contains all invalid values in the column. CMPLNT_NUM - unique value to identify rows, X_COORD_CD - shows column value.
    2. **col19_valid_data.out** - Contains all valid values in the column. CMPLNT_NUM - unique value to identify rows, X_COORD_CD - column value.
    3. **col19_statistics.out** - Contains statistics about the column like invalid elements count, valid elements count and invalid value which occurs maximum number of times.

- **Column 20 - Y-Coordinate of the location of crime**
Script col19-22.py - Code checks if the values in column are not empty and integer values with the New York City limits.

3 output files are generated -

2. **col20_invalid_data.out** - Contains all invalid values in the column. CMPLNT_NUM - unique value to identify rows, Y_COORD_CD - shows column value.
2. **col20_valid_data.out** - Contains all valid values in the column. CMPLNT_NUM - unique value to identify rows, Y_COORD_CD - column value.
3. **col20_statistics.out** - Contains statistics about the column like invalid elements count, valid elements count and invalid value which occurs maximum number of times.

- **Column 21 - Latitude of the location of crime**
  Script col19-22.py - Code checks if the values in column are not empty and decimal values with the New York City limits.
  3 output files are generated -
  3. **col21_invalid_data.out** - Contains all invalid values in the column. CMPLNT_NUM - unique value to identify rows, Latitude - shows column value.
  2. **col21_valid_data.out** - Contains all valid values in the column. CMPLNT_NUM - unique value to identify rows, Latitude - column value.
  3. **col21_statistics.out** - Contains statistics about the column like invalid elements count, valid elements count and invalid value which occurs maximum number of times.

- **Column 22 - Longitude of the location of crime**
  Script col19-22.py - Code checks if the values in column are not empty and decimal values with the New York City limits.
  3 output files are generated -
  4. **col22_invalid_data.out** - Contains all invalid values in the column. CMPLNT_NUM - unique value to identify rows, Longitude - shows column value.
  2. **col22_valid_data.out** - Contains all valid values in the column. CMPLNT_NUM - unique value to identify rows, Longitude - column value.
  3. **col22_statistics.out** - Contains statistics about the column like invalid elements count, valid elements count and invalid value which occurs maximum number of times.

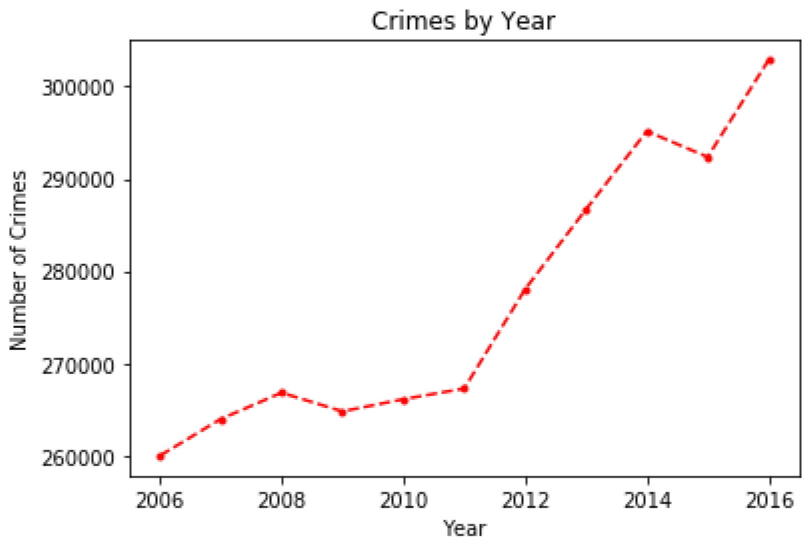
- **Column 23 - Latitude, Longitude of the location of crime**
  Script col23.py - Code checks if the values in column are not empty and decimal values match the values that are printed the column 21 and 22.
  3 output files are generated -
  5. **col23_invalid_data.out** - Contains all invalid values in the column. CMPLNT_NUM - unique value to identify rows, Lat_Lon - shows column value.
  2. **col23_valid_data.out** - Contains all valid values in the column. CMPLNT_NUM - unique value to identify rows, Lat_Lon - column value.

3. **col23_statistics.out** - Contains statistics about the column like invalid elements count, valid elements count and invalid value which occurs maximum number of times.

Part 2 - Data Analysis

Plots obtained from data analysis

1) **Crimes by Year**



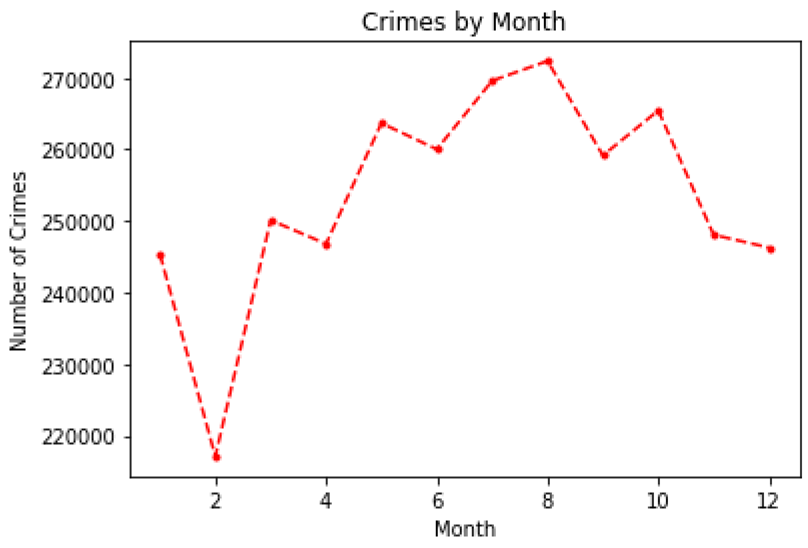
Observations -

a) Plot shows crimes in general increased over the years.
b) There was a slight dip in the crime rate during the years 2009 and 2015.

Scripts used for data aggregation and plotting-

crimes_by_year_month.py , crimes_by_year.ipynb

2) Crimes by Month

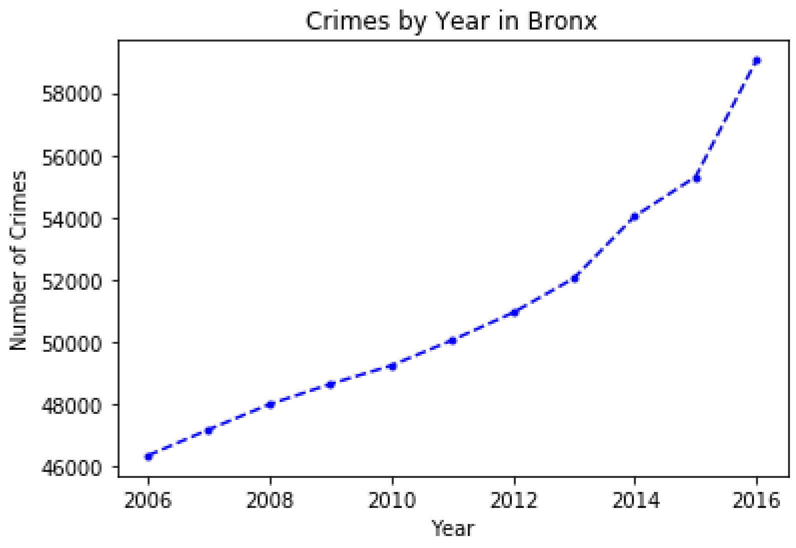


Observations -

a) Plot shows months of July and August had most crimes, while February had the least number of crimes.

Scripts used for data aggregation and plotting-
crimes_by_year_month.py , crimes_by_month.ipynb

3) Crimes by Year in Bronx

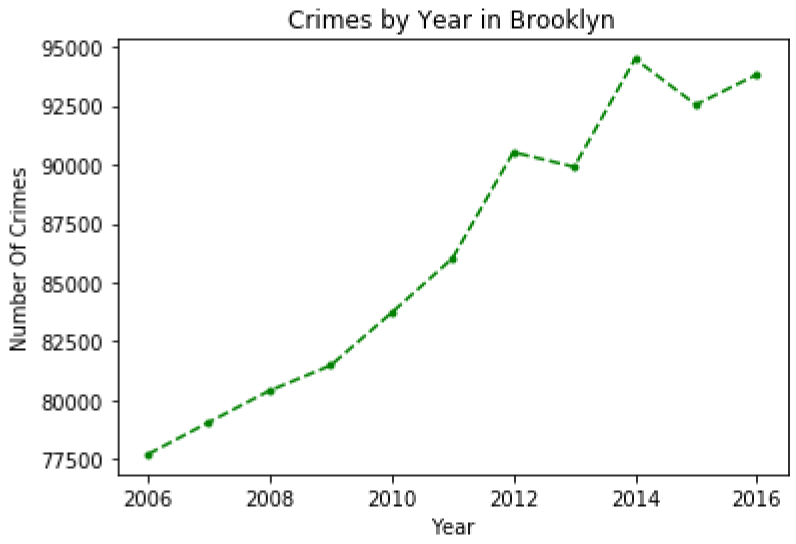


Observations -

a) The Plots show that crimes have increased in Bronx over the years.
b) Least crimes happened in 2006
c) Crimes seem to increase exponentially

Scripts used for data aggregation and plotting-
crimes_by_borough.py, crime_per_borough_per_year.ipynb

4) **Crimes by Year in Brooklyn**



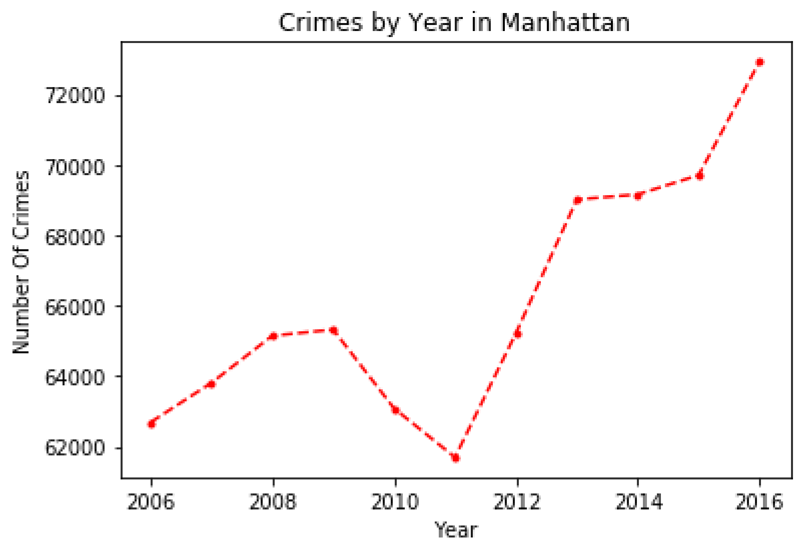
Observations -

a) Here again crimes seem to increase over the years. We can see a dip in 2013 and 2015 but overall, it's an increase in the number of crimes.
b) Least crimes happened in 2006
c) Initially the curve same as crimes in Bronx but later differs. In fact fact greater than those in Bronx.

Scripts used for data aggregation and plotting-

crimes_by_borough.py, crime_per_borough_per_year.ipynb

5) **Crimes by Year in Manhattan**



Observations-

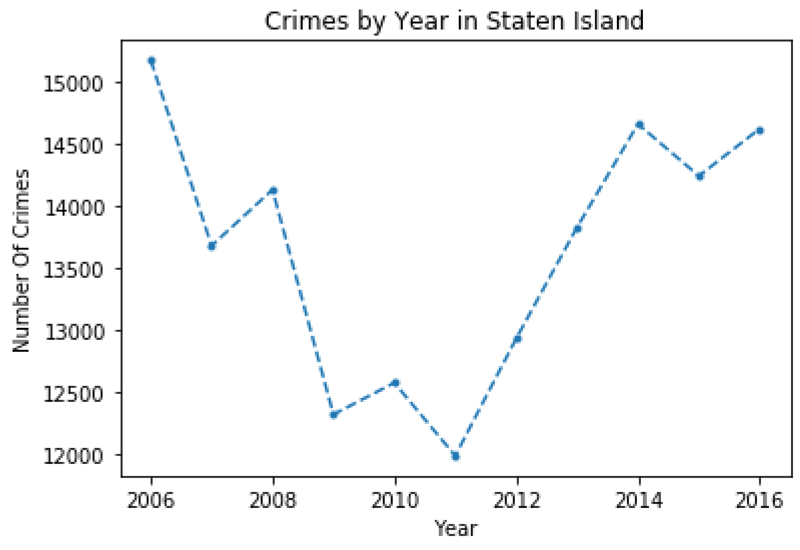
a) Crimes increase overall. There seem to be a great dip in 2011

b) The overall crimes in 2016 are low than that of brooklyn.

Scripts used for data aggregation and plotting-

crimes_by_borough.py, crime_per_borough_per_year.ipynb

6) Crimes by Year in Staten Island



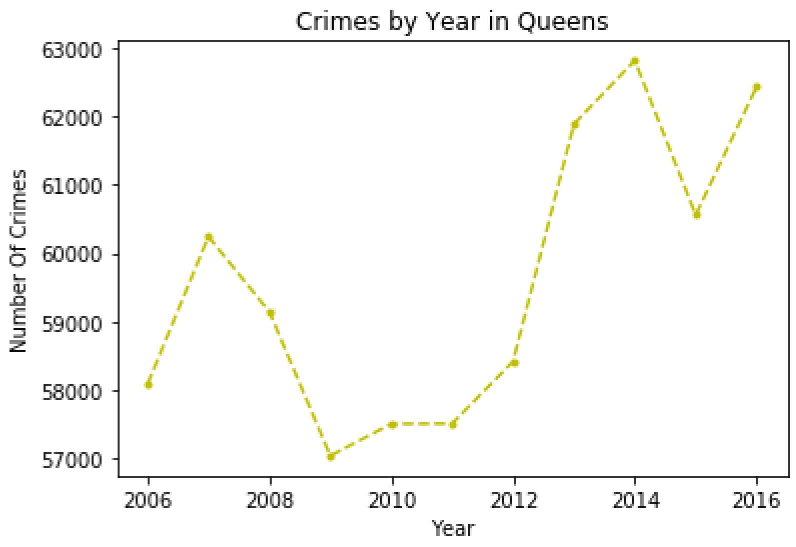
Observations -

a) Crimes in Staten Island follow a very different pattern than all other graphs.

b) The crime are low than many boroughs but it is maybe because the population less compared to other boroughs.

Scripts used for data aggregation and plotting-
crimes_by_borough.py, crime_per_borough_per_year.ipynb

7) **Crimes by Year in Queens**

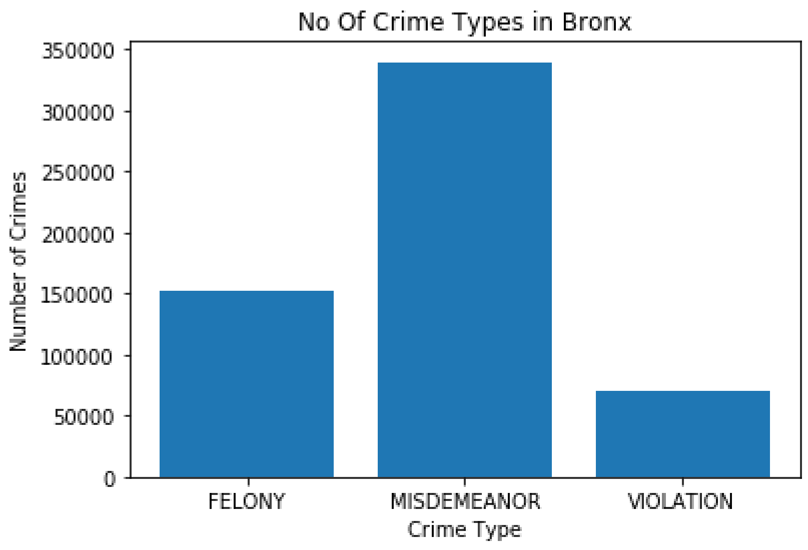


Observations -

a) This graph follows an erratic pattern than other graphs.
b) The crimes increase overall but the is a dip in 2008-2009 and then they increase rapidly till 2016

Scripts used for data aggregation and plotting-
crimes_by_borough.py, crime_per_borough_per_year.ipynb

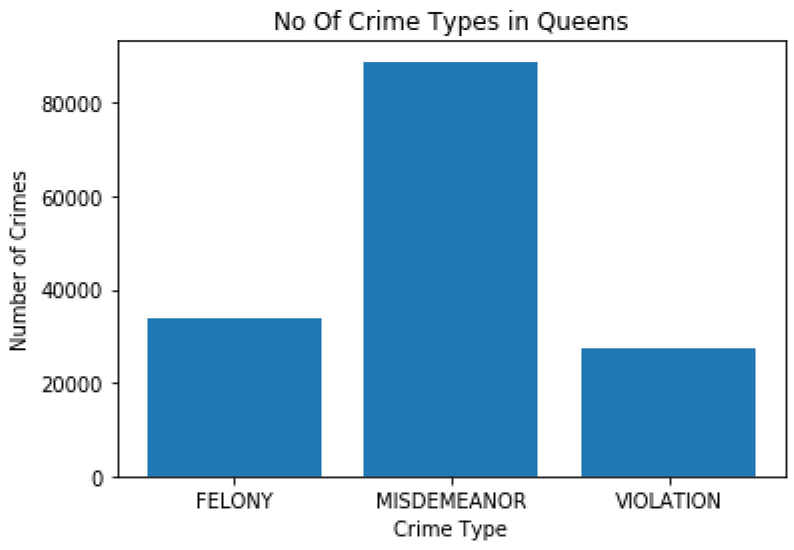
8) **Crimes types and their counts in Bronx**



Observation -

a) Major types of crime seem to be Misdemeanor, and then felony and then Violations
b) This pattern in observed in all the boroughs but with different figures.
Scripts used for data aggregation and plotting-
crimes_by_borough_and_law_cd.ipynb, crimes_by_borough_and_law_cd.py

9) **Crimes types and their counts in Queens**

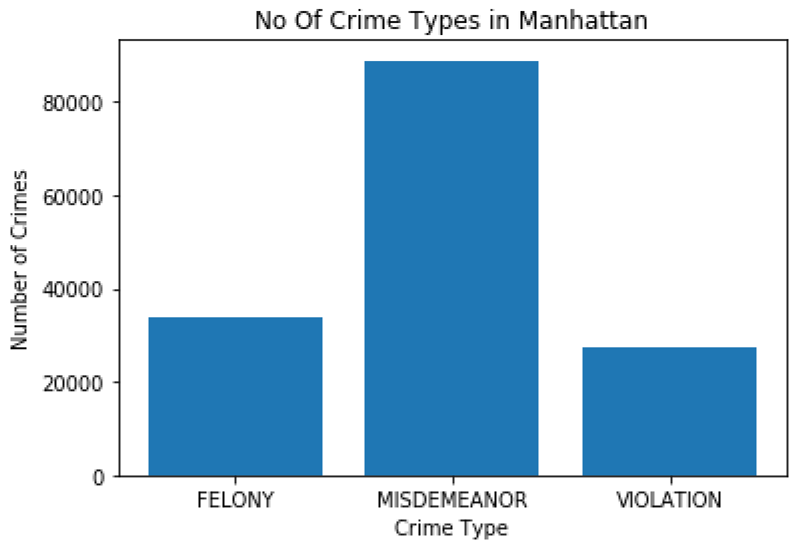


Observations -

a) Again the same pattern if followed, but violations are less than those in Bronx
b) Misdemeanor is very low as compared to bronx, which has the highest over all boroughs.

Scripts used for data aggregation and plotting-
crimes_by_borough_and_law_cd.ipynb, crimes_by_borough_and_law_cd.py

10) Crimes types and their counts in Manhattan

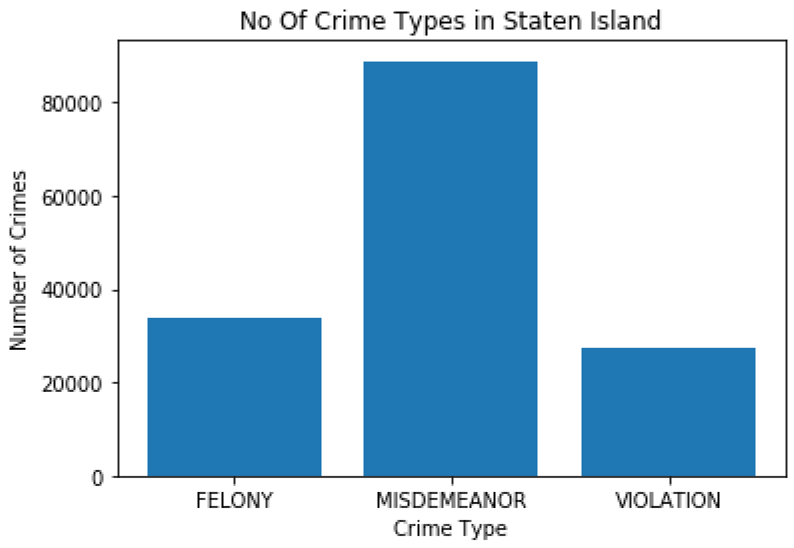


Observations -

a) Manhattan is no different than other boroughs.
b) Violations seem to catch up with Felony type.

Scripts used for data aggregation and plotting-
crimes_by_borough_and_law_cd.ipynb, crimes_by_borough_and_law_cd.py

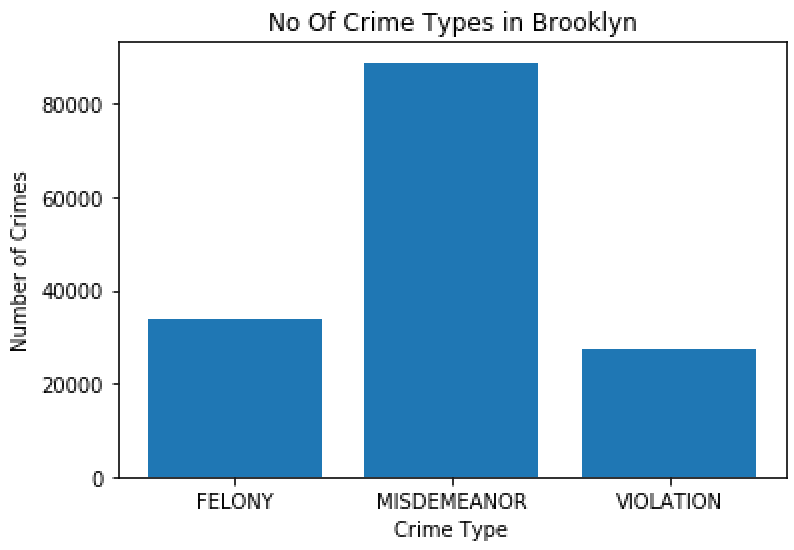
11) Crimes types and their counts in Staten Island



Observations -

a) Misdemeanor is the maximum type of crime in Staten Island since 2006-2016
b) Again, there are misdemeanor reported staten island as well. It is interesting to see that all the boroughs follow the same pattern.
Scripts used for data aggregation and plotting-
crimes_by_borough_and_law_cd.ipynb, crimes_by_borough_and_law_cd.py

12) Crimes types and their counts in Brooklyn

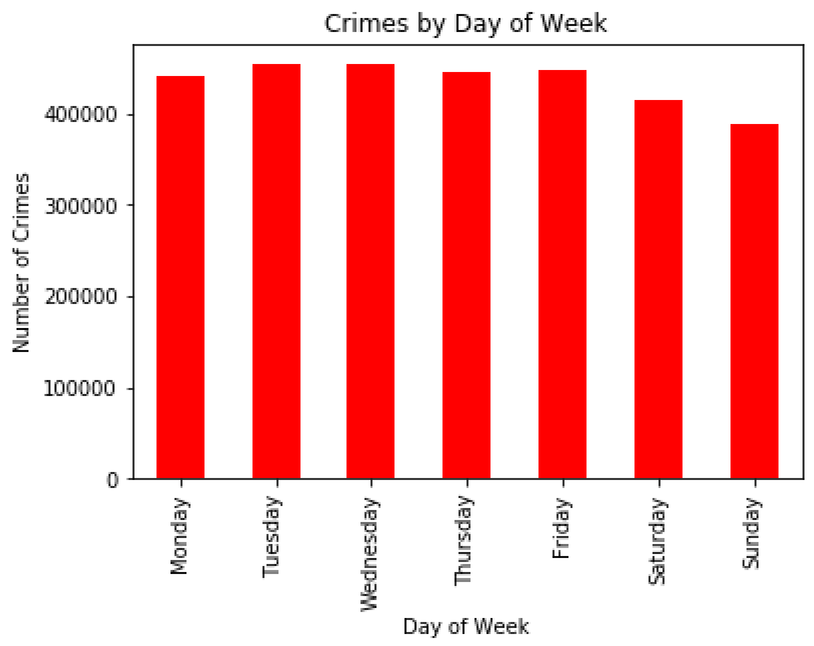


Observations -

a) Again the same pattern if followed, but violations are less than those in Bronx
b) Misdemeanor is very low as compared to bronx, which has the highest over all boroughs.

Scripts used for data aggregation and plotting-
crimes_by_borough_and_law_cd.ipynb, crimes_by_borough_and_law_cd.py

13) **Crimes by Day**



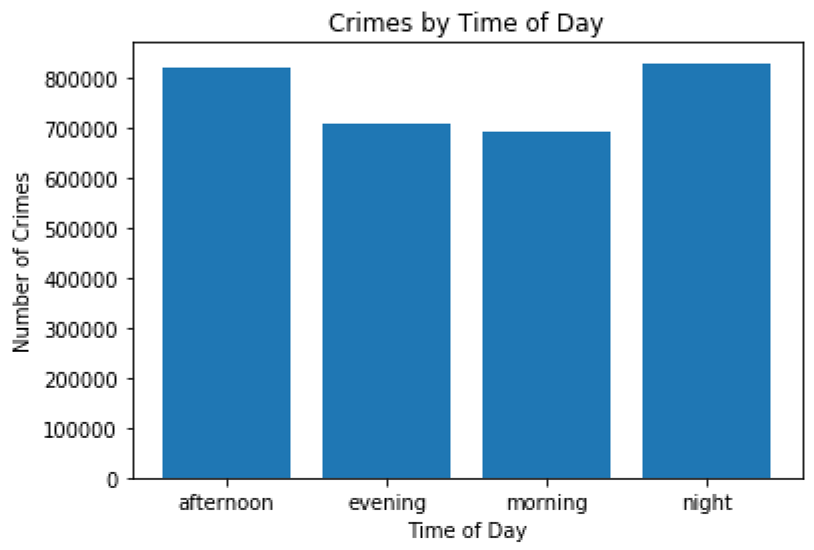
Observations -

a) Plot shows more crimes happened on weekdays as compared to weekends.
b) Least crimes happened on Sunday.

Scripts used for data aggregation and plotting-

crimes_by_day.py , crimes_by_day.ipynb

14) **Crimes by Time of Day**



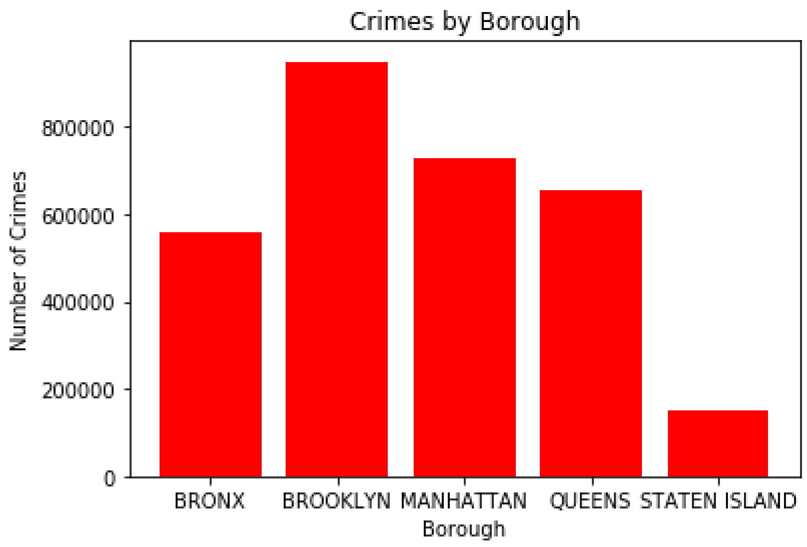
Observations -

a) Plot shows more crimes took place during afternoon and night.
b) Least crimes happened in the morning.

Scripts used for data aggregation and plotting-

crimes_by_time.py , crimes_by_time_of_day.ipynb

15) Crimes by Borough



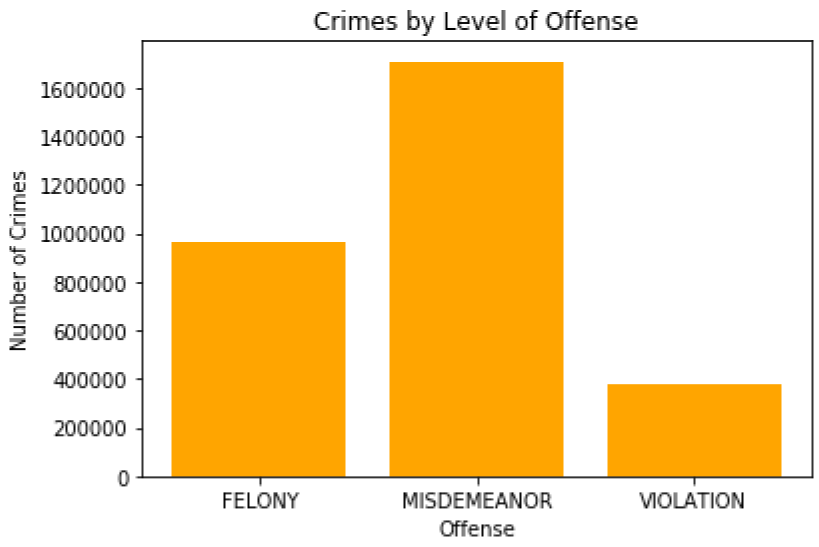
Observations -

a) Plot shows most crimes happened in Brooklyn as compared to other boroughs.
b) Staten Island observed least crimes.

Scripts used for data aggregation and plotting-

crimes_by_all.py , crimes_by_all.ipynb

16) Crimes by Level Of Offense

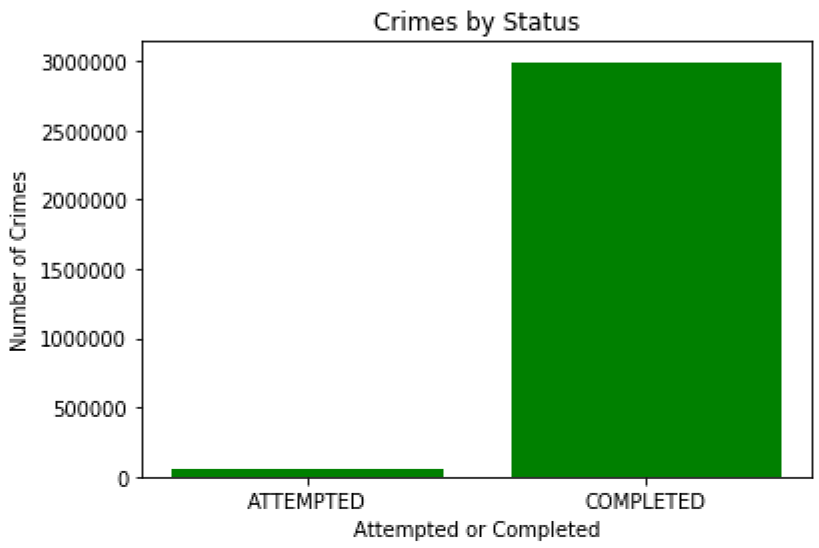


Observations -

a) Plot shows the level of offense for crimes was maximum in case of Misdemeanor followed by Felony(approx 50% of Misdemeanor type).
b) Least offense level observed was Violation.

Scripts used for data aggregation and plotting-
crimes_by_all.py , crimes_by_all.ipynb

17) Crimes by Status

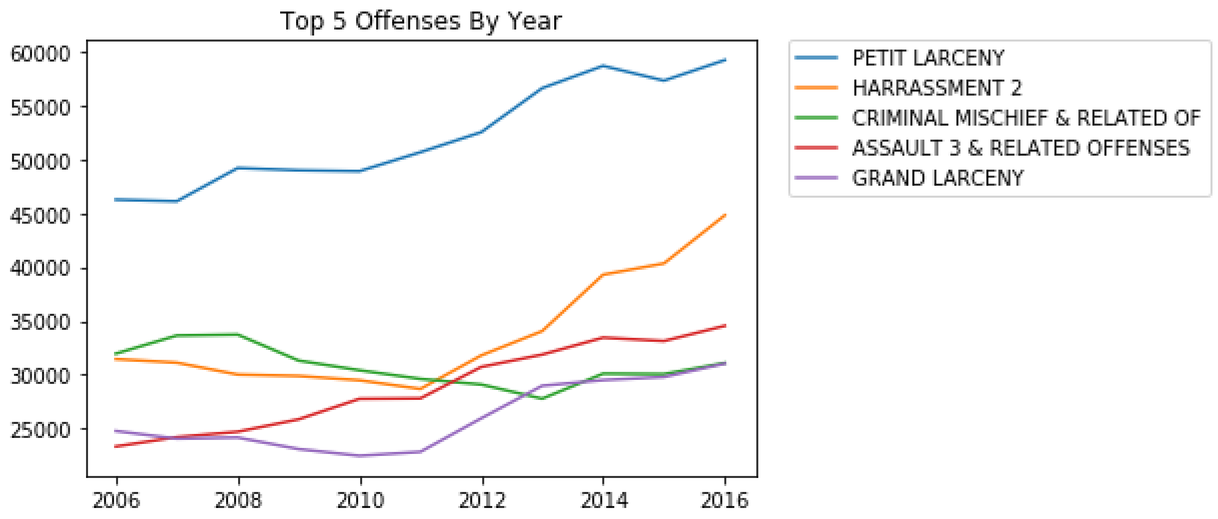


Observations -

a) Plot shows most crimes were Completed.
b) Very less crimes were Attempted and failed.

Scripts used for data aggregation and plotting-
crimes_by_all.py , crimes_by_all.ipynb

18) Crimes by Offenses over the year

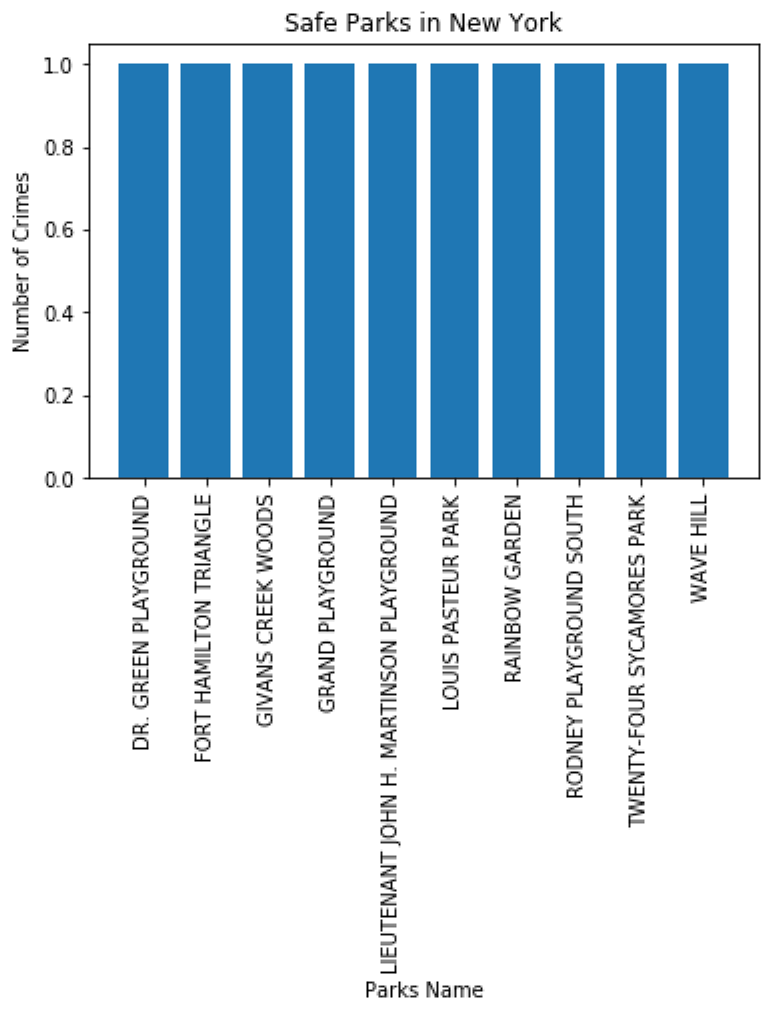


Observations -

a) Plot shows “Petit Larceny” type of crimes happened most.
b) General trend of crimes increased over the years in all kinds of offenses.

Scripts used for data aggregation and plotting-
crimes_by_ofns.py , offences_by_year.ipynb

19) Safe Parks in New York

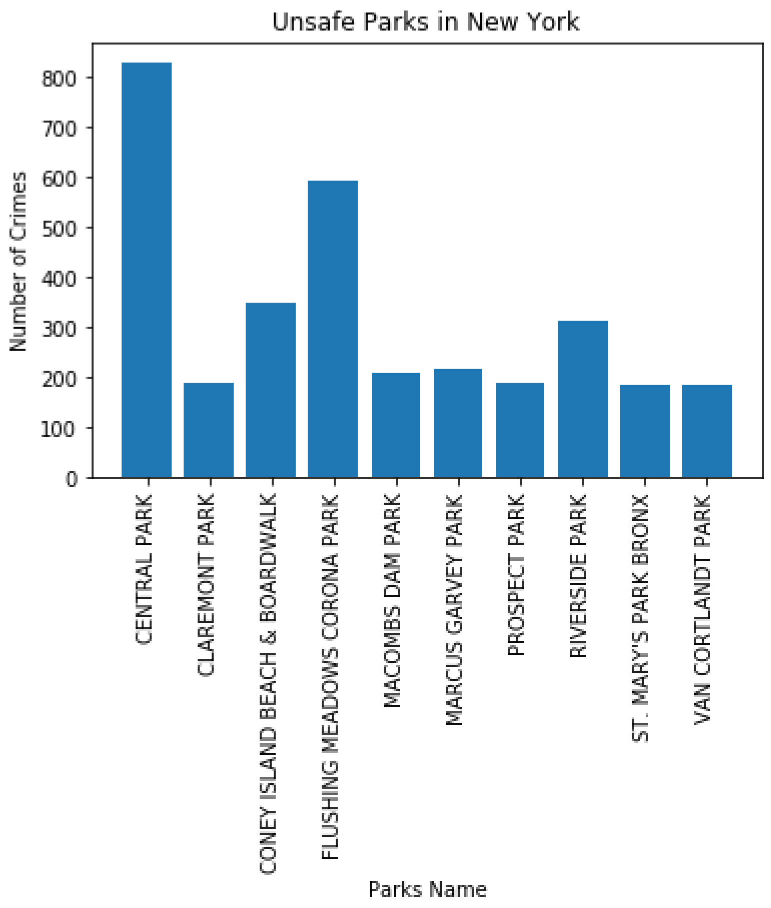


Observations -

a) Plot shows 10 parks in New York with least crimes reported.

Scripts used for data aggregation and plotting-
crimes_by_park.py, safe_parks.ipynb

20) Unsafe Parks in New York



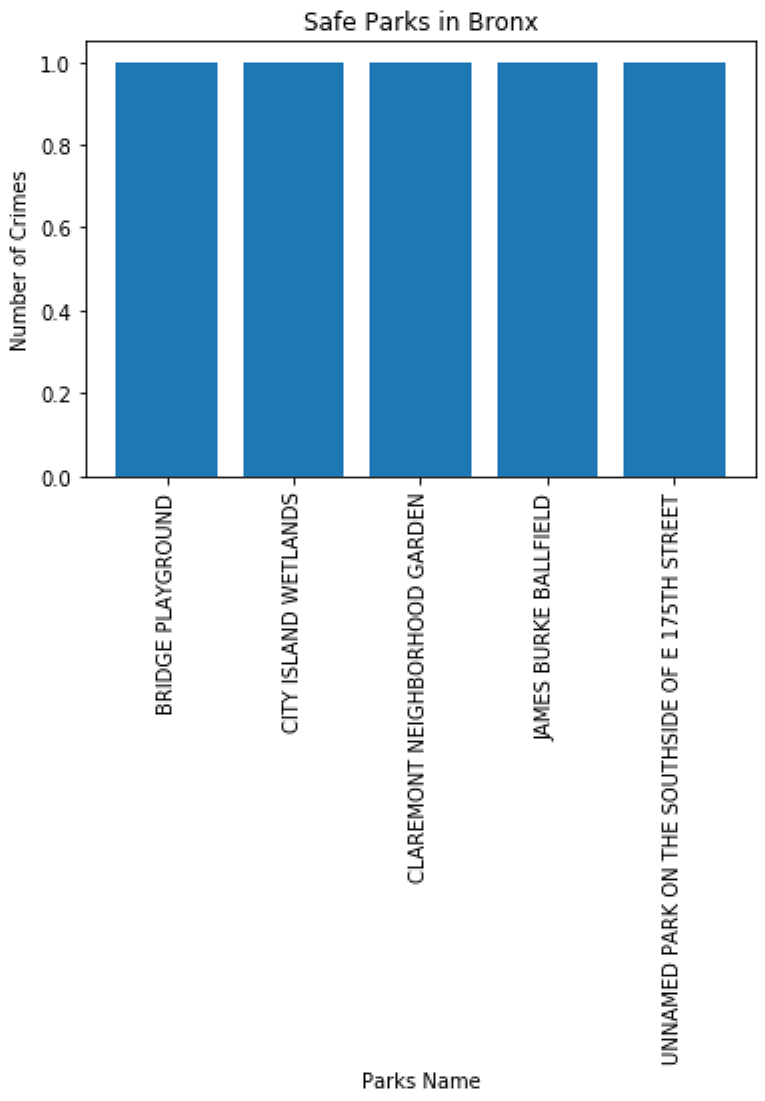
Observations -

a) Plot shows 10 parks in New York with most crimes reported.
b) Central Park and Flushing Meadows Corona Park have more crimes than other parks in New York.

Scripts used for data aggregation and plotting-

crimes_by_park.py, unsafe_parks.ipynb

21) Safe Parks in Bronx



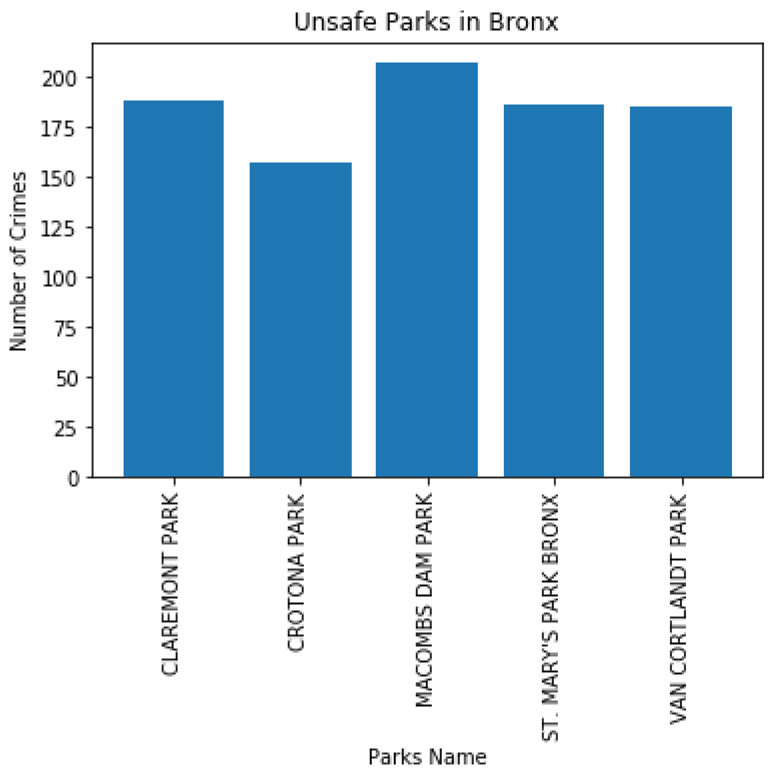
Observations -

a) Plot shows 5 parks in Bronx with least crimes reported.

Scripts used for data aggregation and plotting-

park_boro.py, safe_parks.ipynb

22) Unsafe Parks in Bronx



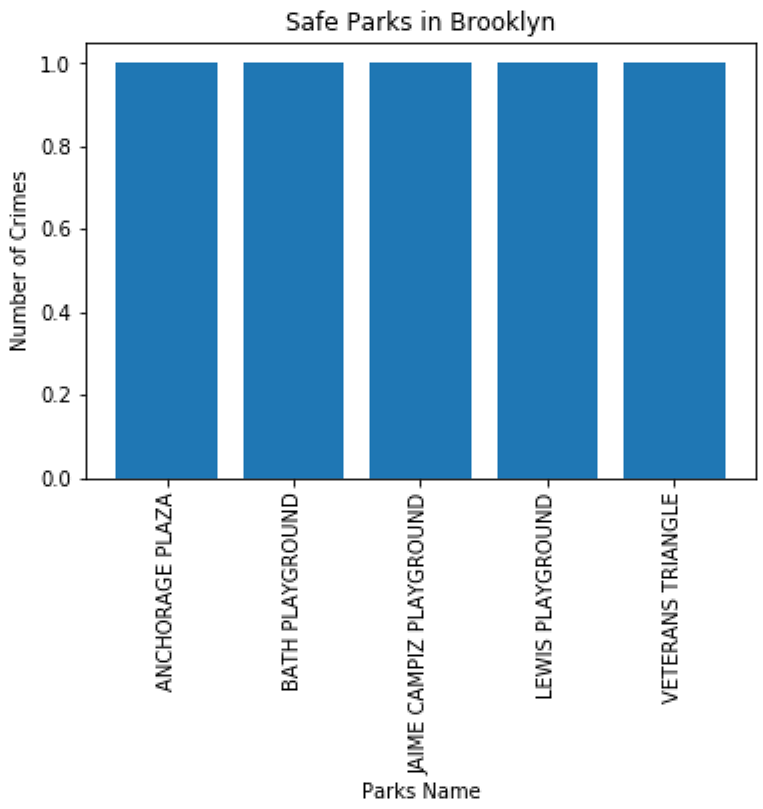
Observations -

a) Plot shows 5 parks in Bronx with most crimes reported.

Scripts used for data aggregation and plotting-

park_boro.py, unsafe_parks.ipynb

23) Safe Parks in Brooklyn



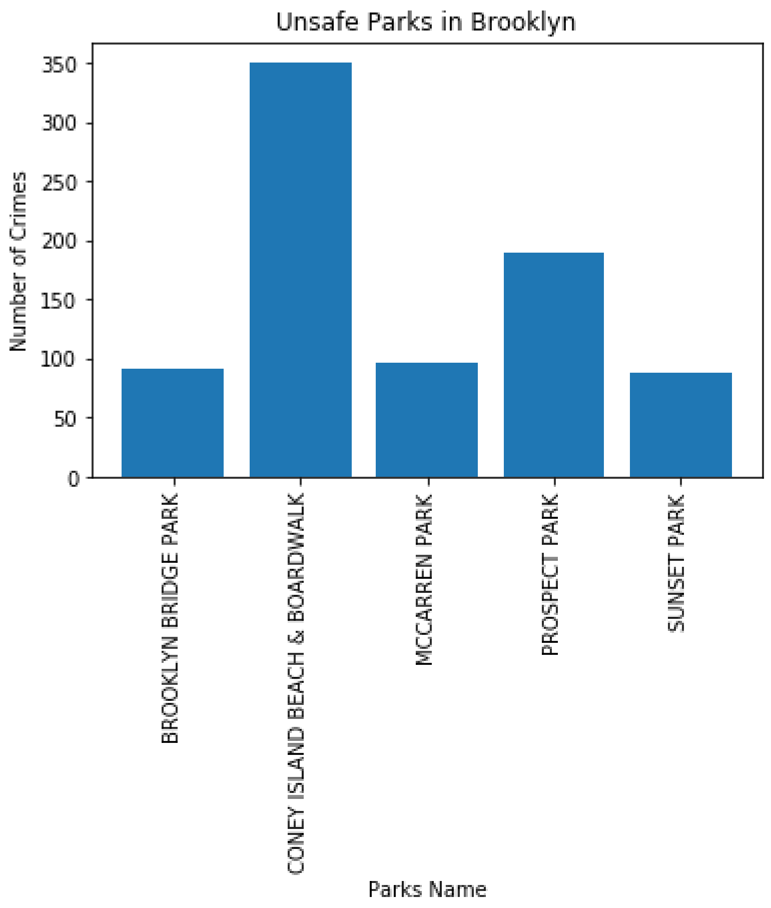
Observations -

a) Plot shows 5 parks in Brooklyn with least crimes reported.

Scripts used for data aggregation and plotting-

park_boro.py, safe_parks.ipynb

24) Unsafe Parks in Brooklyn



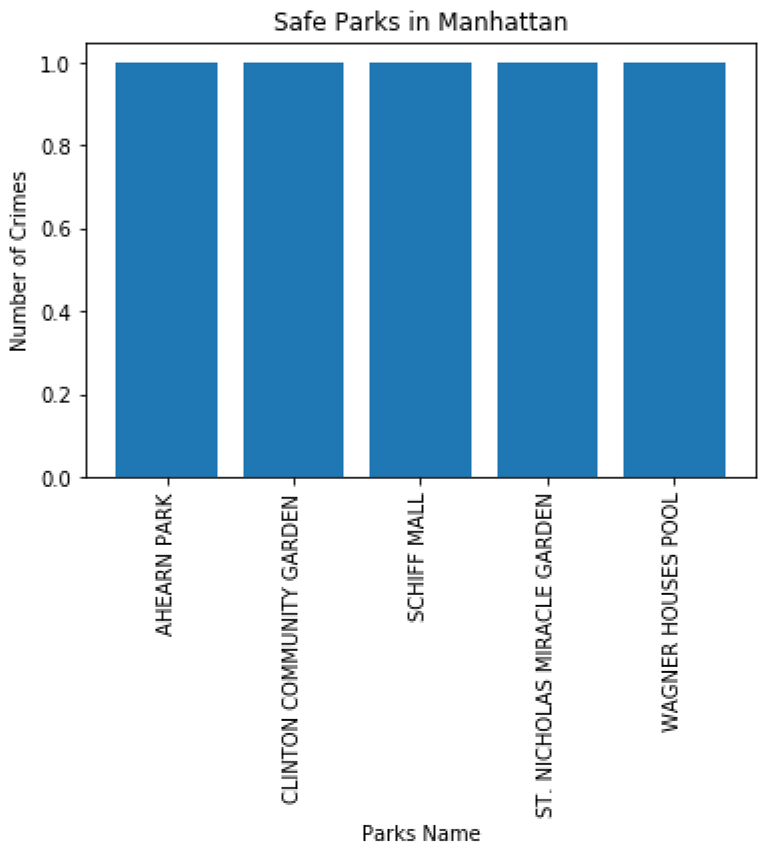
Observations -

a) Plot shows 5 parks in Brooklyn with most crimes reported.

Scripts used for data aggregation and plotting-

park_boro.py, unsafe_parks.ipynb

25) Safe Parks in Manhattan



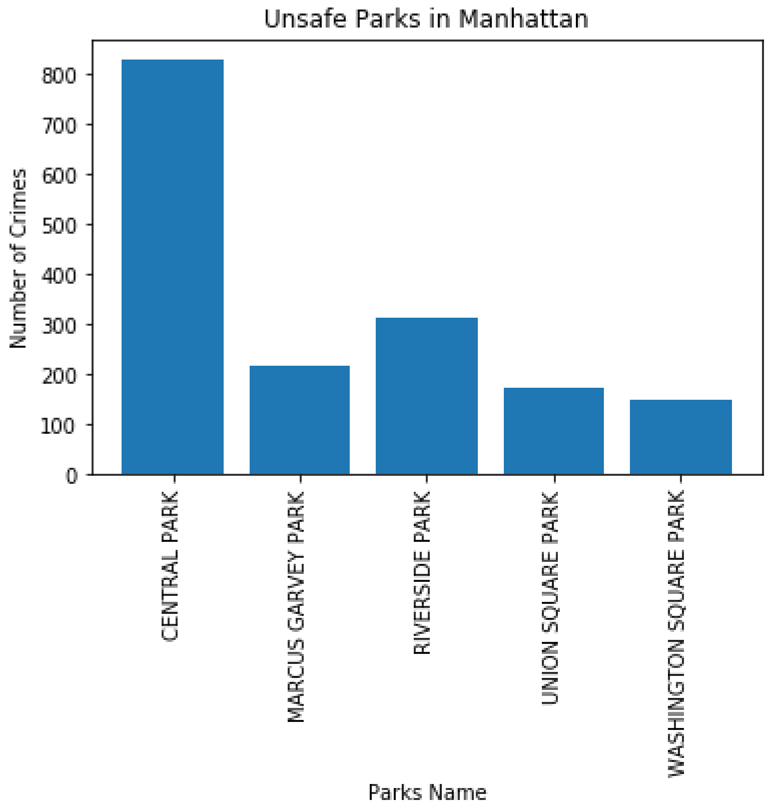
Observations -

a) Plot shows 5 parks in Manhattan with least crimes reported.

Scripts used for data aggregation and plotting-

park_boro.py, safe_parks.ipynb

26) Unsafe Parks in Manhattan



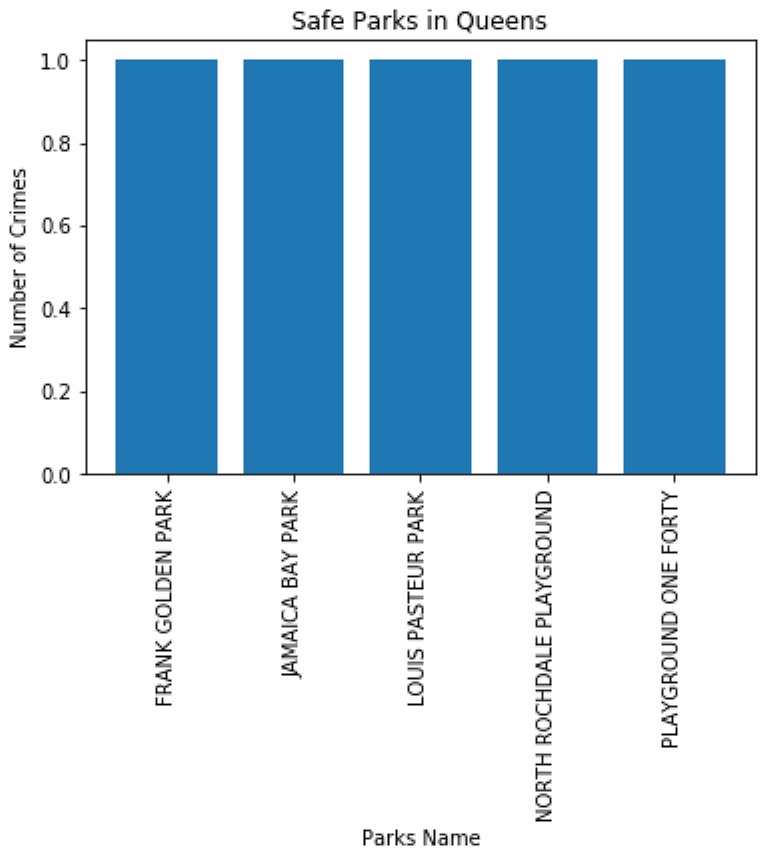
Observations -

a) Plot shows 5 parks in Manhattan with most crimes reported.

Scripts used for data aggregation and plotting-

park_boro.py, unsafe_parks.ipynb

27) Safe Parks in Queens



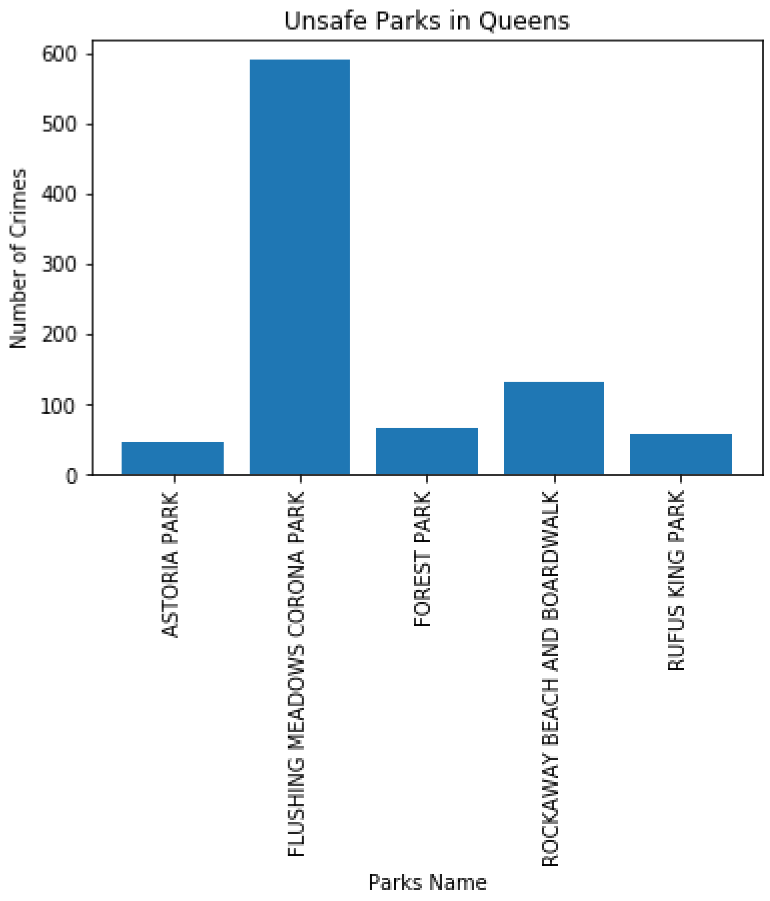
Observations -

a) Plot shows 5 parks in Queens with least crimes reported.

Scripts used for data aggregation and plotting-

park_boro.py, safe_parks.ipynb

28) Unsafe Parks in Queens



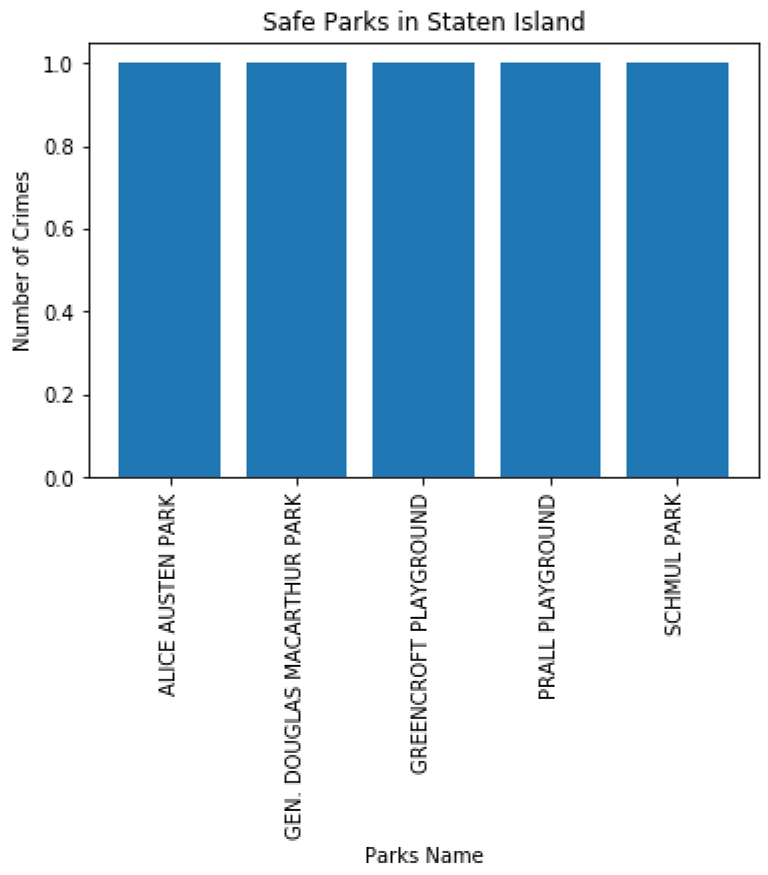
Observations -

a) Plot shows 5 parks in Queens with most crimes reported.

Scripts used for data aggregation and plotting-

park_boro.py, unsafe_parks.ipynb

29) Safe Parks in Staten Island



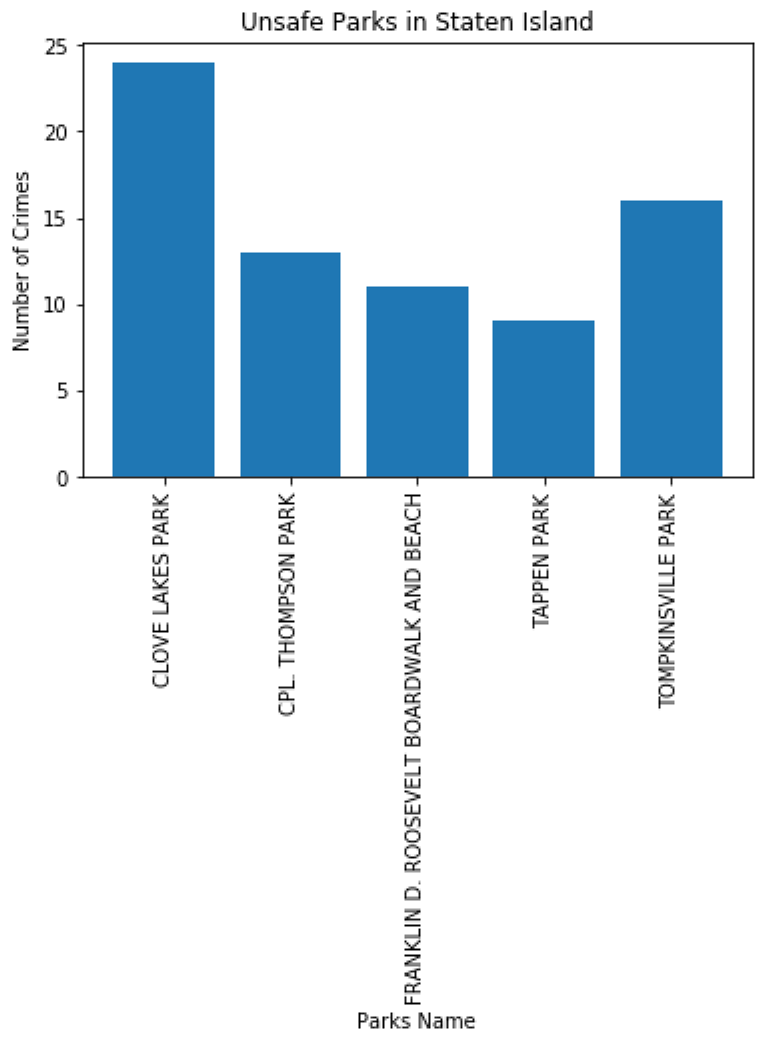
Observations -

a) Plot shows 5 parks in Staten Island with least crimes reported.

Scripts used for data aggregation and plotting-

park_boro.py, safe_parks.ipynb

30) Unsafe Parks in Staten Island



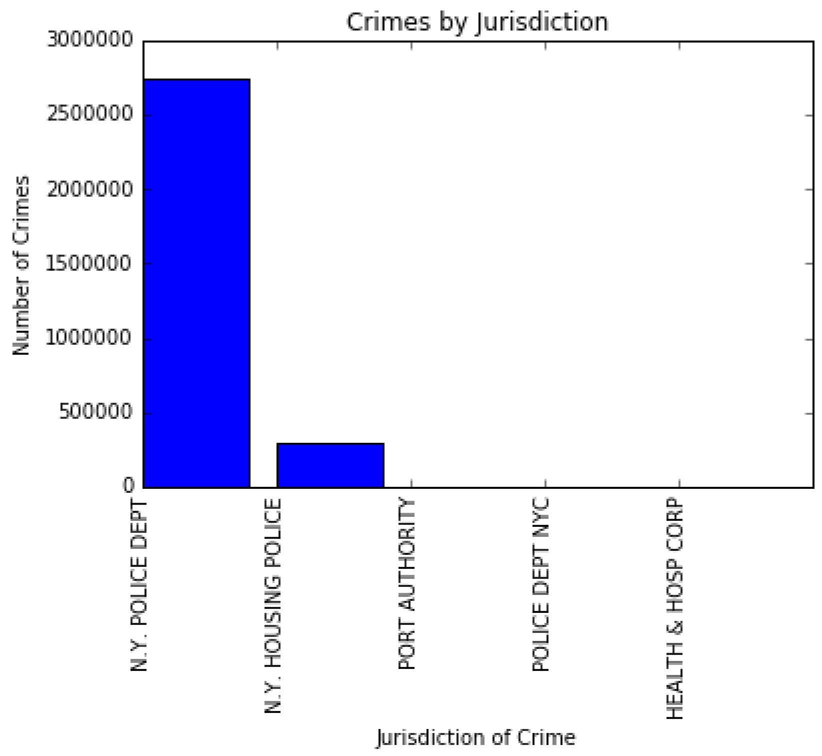
Observations -

a) Plot shows 5 parks in Staten Island with most crimes reported.

Scripts used for data aggregation and plotting-

park_boro.py, unsafe_parks.ipynb

31) Crimes by Jurisdiction



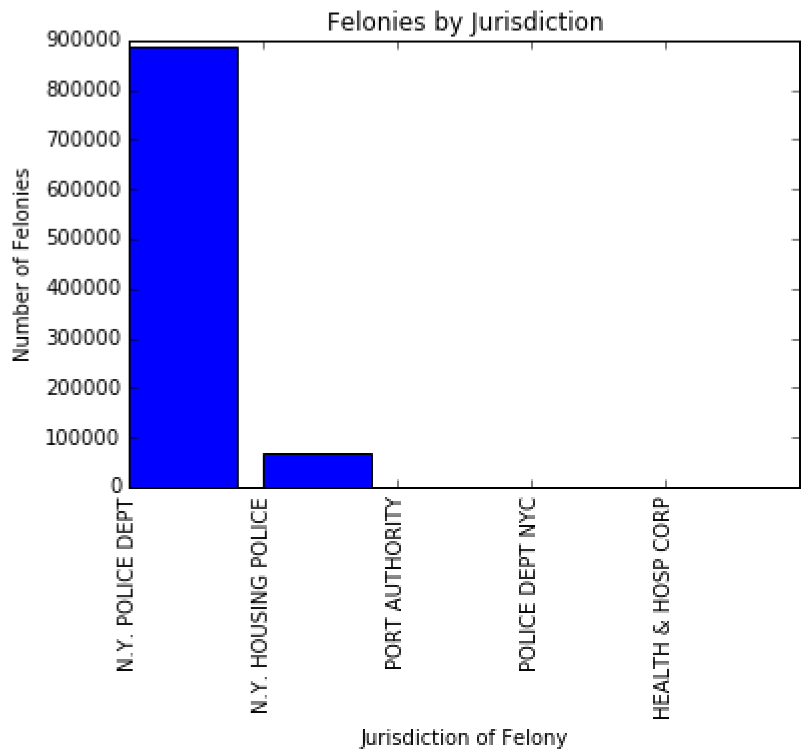
Observations -

a) Plot shows 5 jurisdictions with most crimes reported.

Scripts used for data aggregation and plotting-

juris.py, crime_juris.ipynb

32) Felonies by Jurisdiction



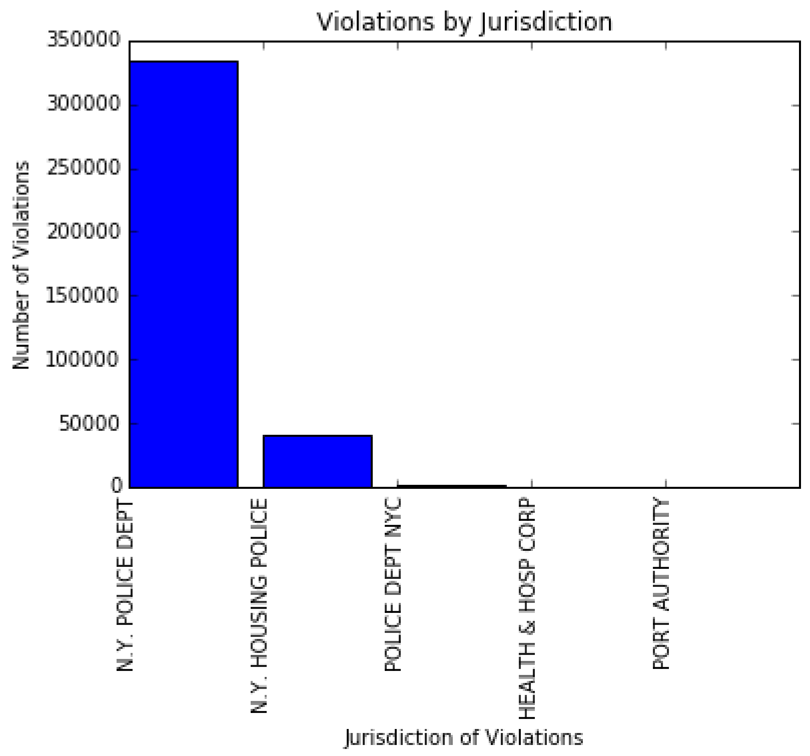
Observations -

a) Plot shows 5 jurisdictions with most felonies reported.

Scripts used for data aggregation and plotting-

crime_juris.py, felony_juris.ipynb

33) Violations by Jurisdiction



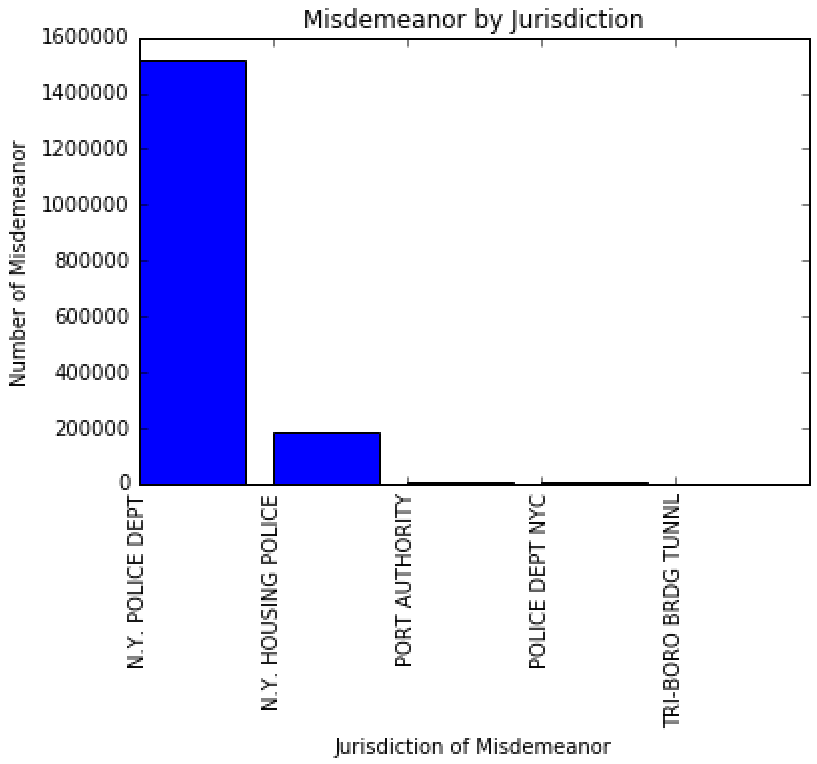
Observations -

a) Plot shows 5 jurisdictions with most violations reported.

Scripts used for data aggregation and plotting-

crime_juris.py, violation_juris.ipynb

34) Misdemeanor by Jurisdiction



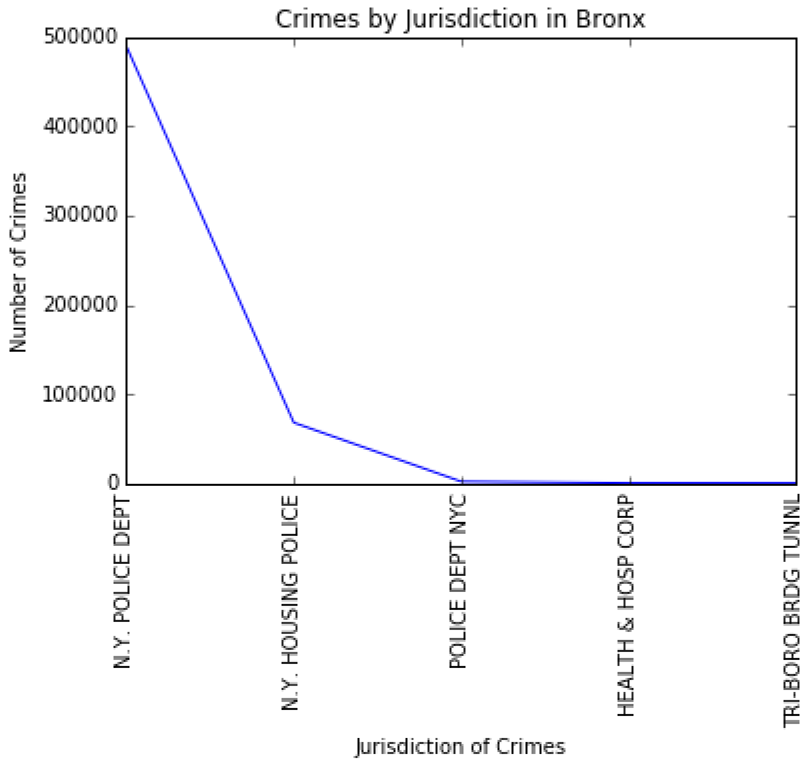
Observations -

a) Plot shows 5 jurisdictions with most misdemeanors reported.

Scripts used for data aggregation and plotting-

juris.py, misdemeanor_juris.ipynb

35) Crimes by Jurisdiction in Bronx



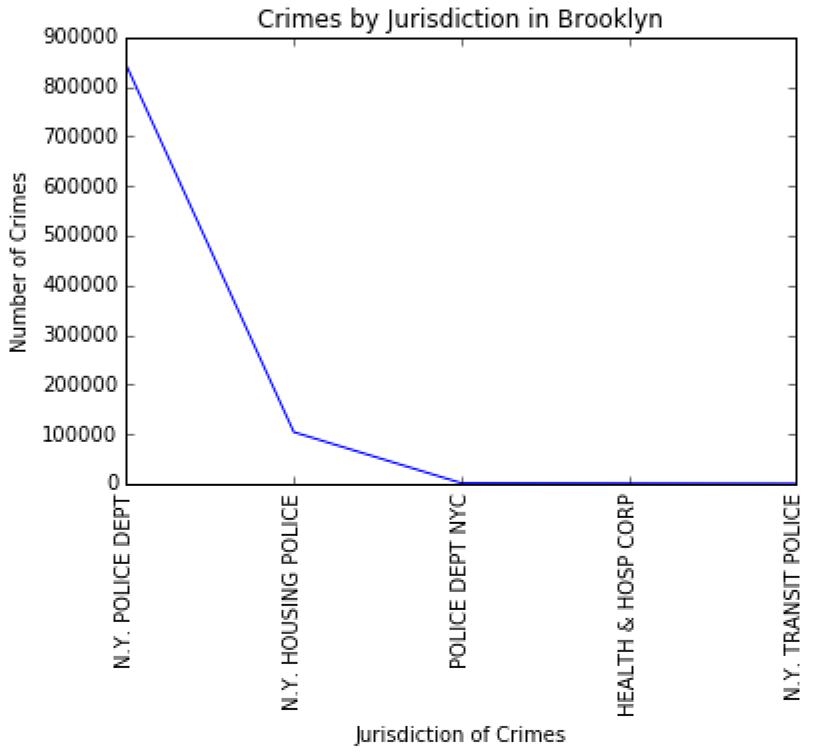
Observations -

a) Plot shows 5 jurisdictions with most crimes reported in Bronx.

Scripts used for data aggregation and plotting-

juris_boro.py, juris_boro.ipynb

36) Crimes by Jurisdiction in Brooklyn



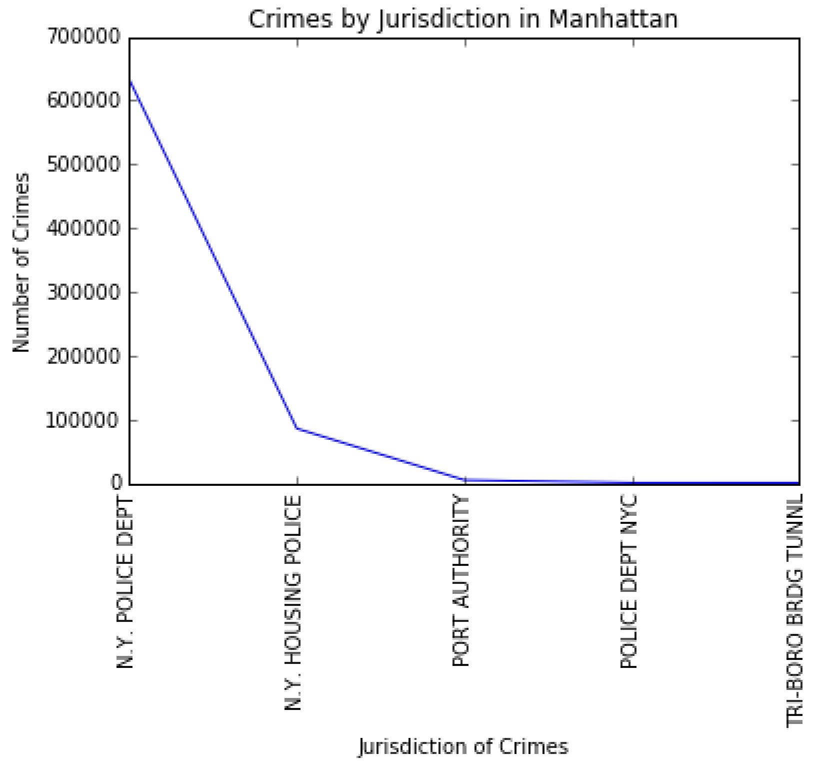
Observations -

a) Plot shows 5 jurisdictions with most crimes reported in Brooklyn.

Scripts used for data aggregation and plotting-

juris_boro.py, juris_boro.ipynb

37) Crimes by Jurisdiction in Manhattan



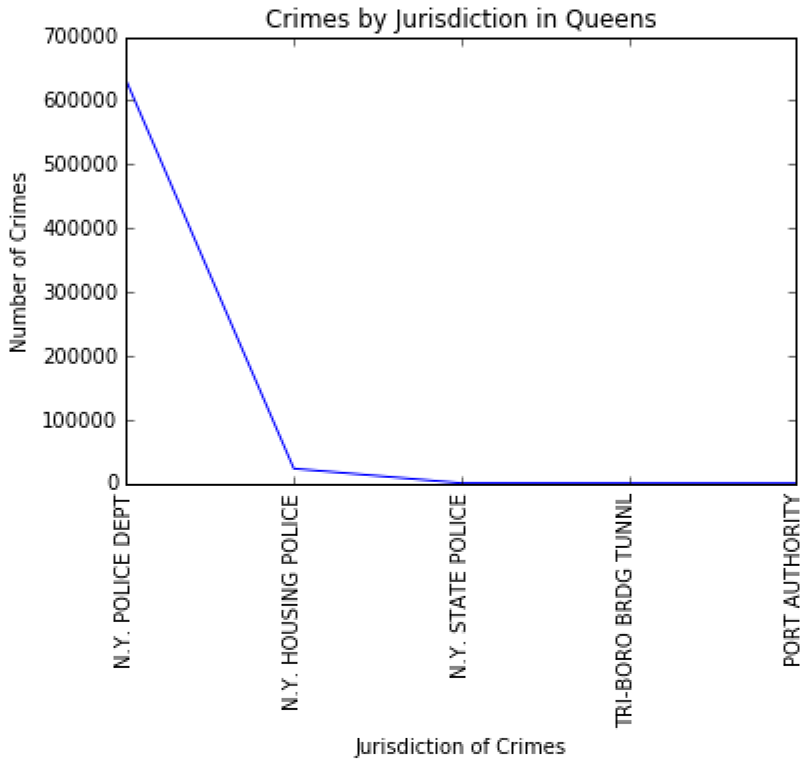
Observations -

a) Plot shows 5 jurisdictions with most crimes reported in Manhattan.

Scripts used for data aggregation and plotting-

juris_boro.py, juris_boro.ipynb

38) Crimes by Jurisdiction in Queens



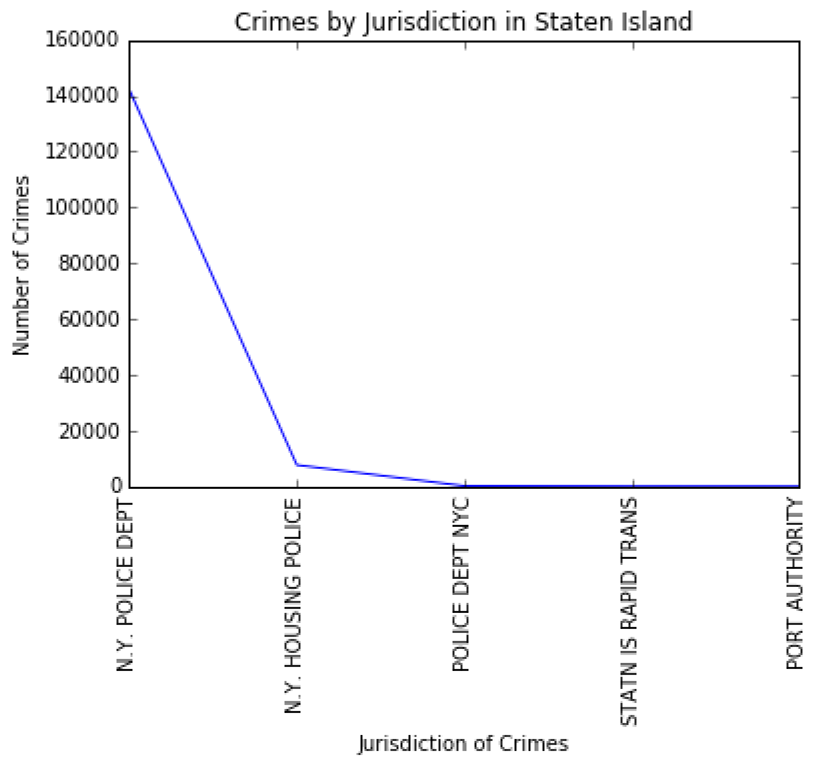
Observations -

a) Plot shows 5 jurisdictions with most crimes reported in Queens.

Scripts used for data aggregation and plotting-

juris_boro.py, juris_boro.ipynb

39) Crimes by Jurisdiction in Staten Island



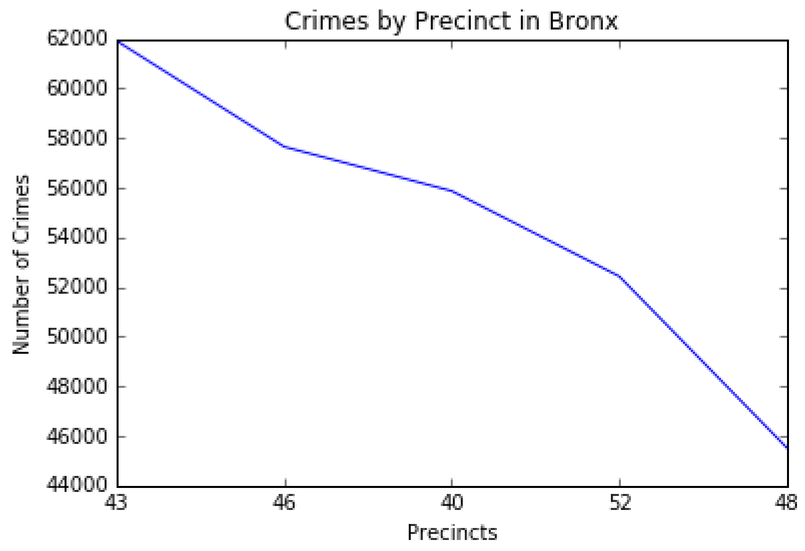
Observations -

a) Plot shows 5 jurisdictions with most crimes reported in Staten Island.

Scripts used for data aggregation and plotting-

juris_boro.py, juris_boro.ipynb

40) Crimes by Precincts in Bronx

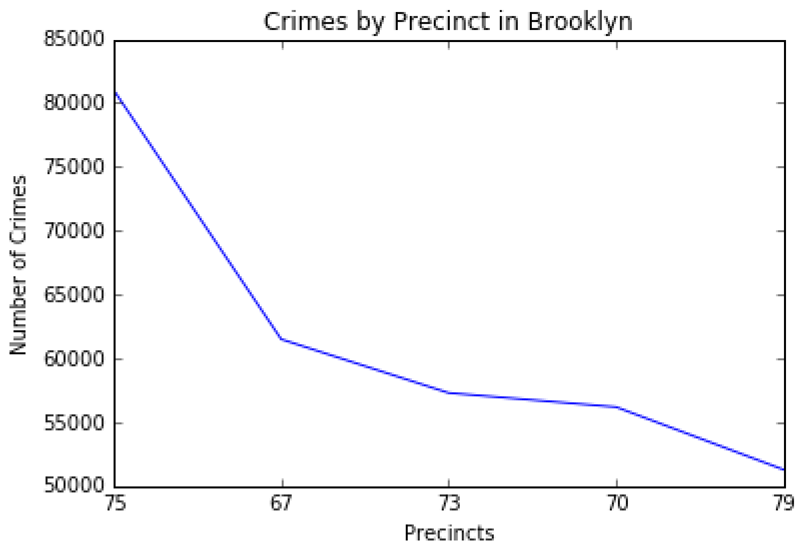


Observations -

a) Plot shows 5 precincts with most crimes reported in Bronx.

Scripts used for data aggregation and plotting-
boro_precincts.py, prec_boro.ipynb

41) Crimes by Precincts in Brooklyn

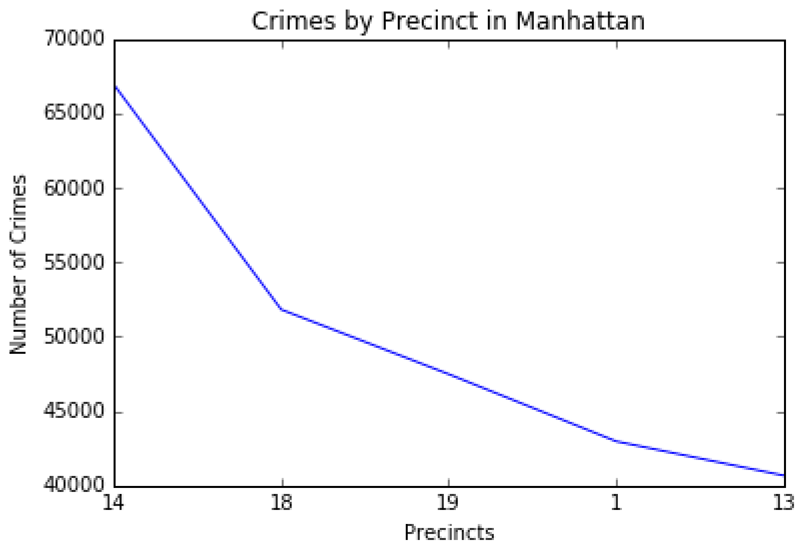


Observations -

a) Plot shows 5 precincts with most crimes reported in Brooklyn.

Scripts used for data aggregation and plotting-
boro_precincts.py, prec_boro.ipynb

42) Crimes by Precincts in Manhattan



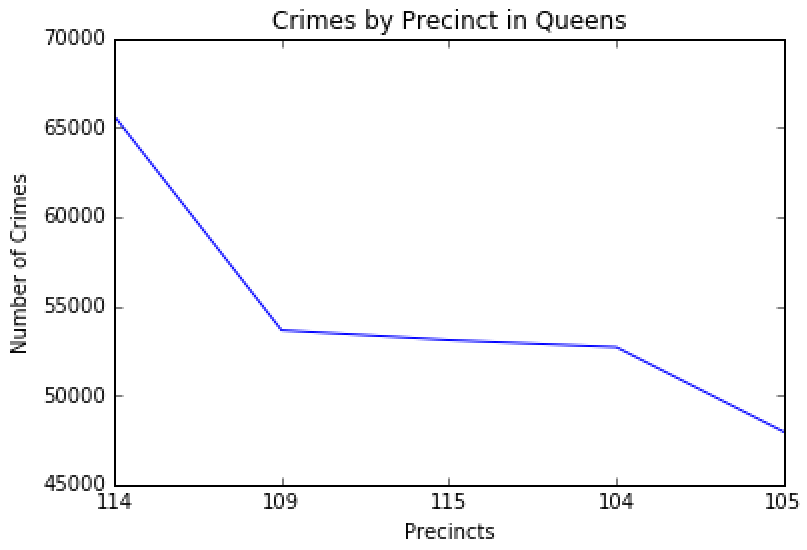
Observations -

a) Plot shows 5 precincts with most crimes reported in Manhattan.

Scripts used for data aggregation and plotting-

boro_precincts.py, prec_boro.ipynb

43) **Crimes by Precincts in Queens**



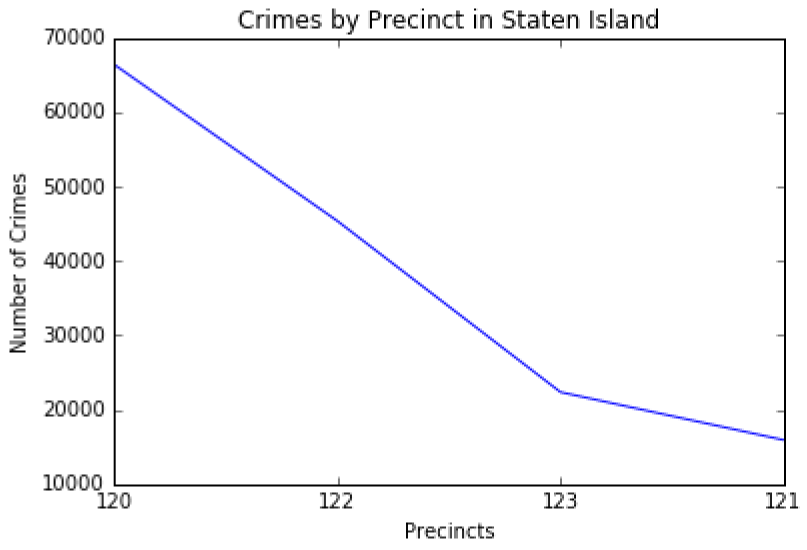
Observations -

a) Plot shows 5 precincts with most crimes reported in Queens.

Scripts used for data aggregation and plotting-

boro_precincts.py, prec_boro.ipynb

44) **Crimes by Precincts in Staten Island**



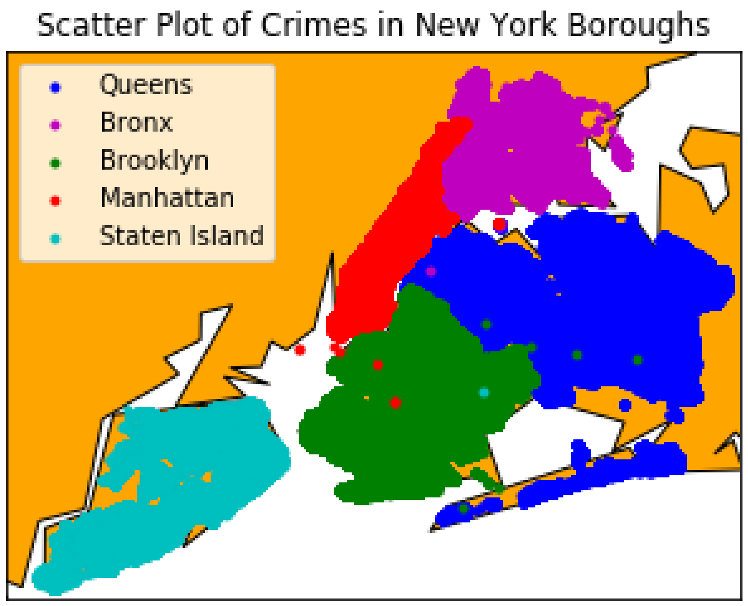
Observations -

a) Plot shows 5 precincts with most crimes reported in Staten Island.

Scripts used for data aggregation and plotting-

boro_precincts.py, prec_boro.ipynb

45) Scatter plot of Crimes in New York Boroughs



Observations -

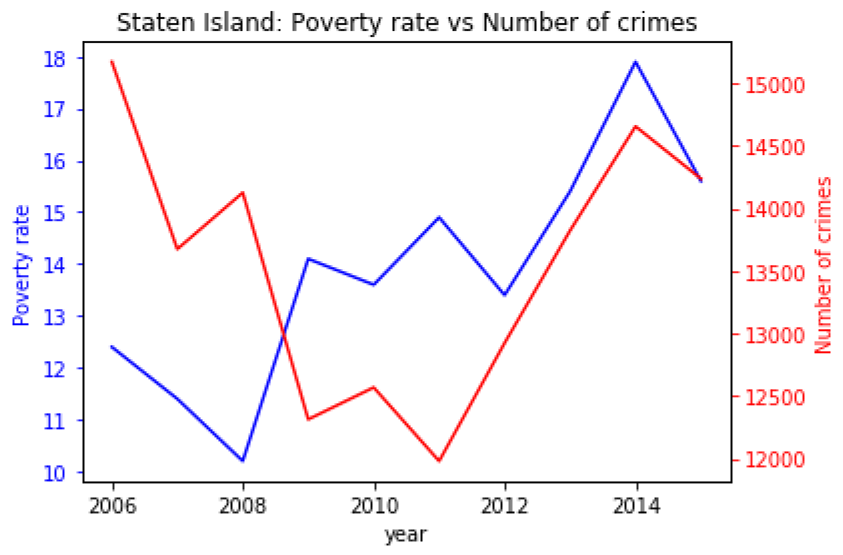
a) Scatter plot of locations of crimes with various boroughs highlighted in different colors.

Scripts used for data aggregation and plotting-
latlong_boro.py, scatterPlot.ipynb

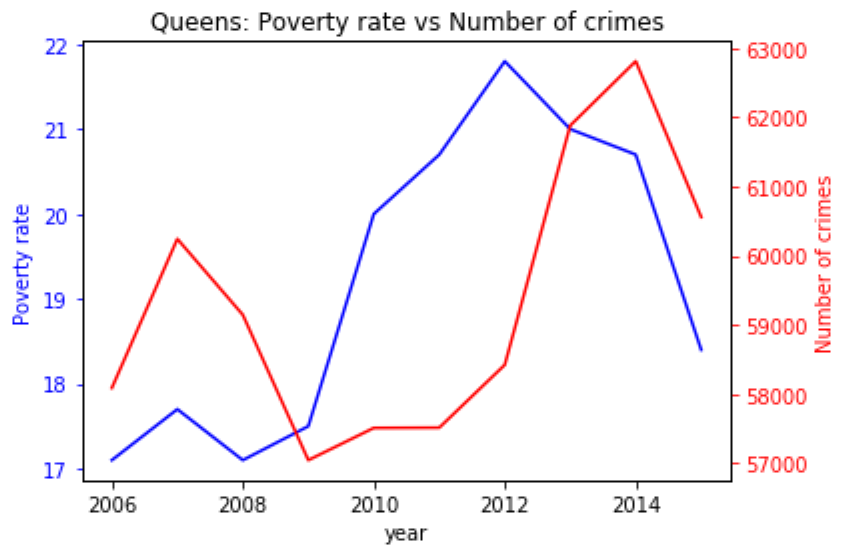
Part 3 - Data Exploration

1) **Hypothesis: Crime rate increases with increase in poverty**
To prove this hypothesis, I explored the poverty data over the years for different boroughs in New York and compared it with the total number of crimes in the corresponding borough and year. Following are the plots for various boroughs:

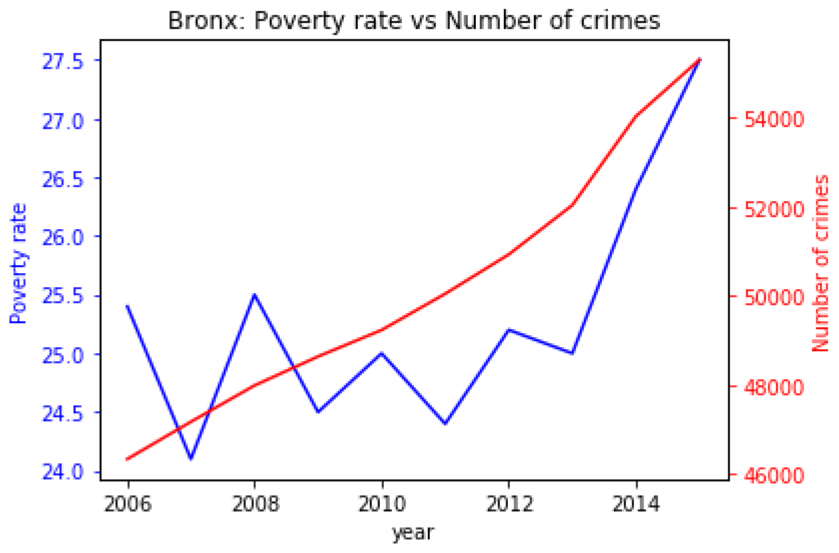
Staten Island



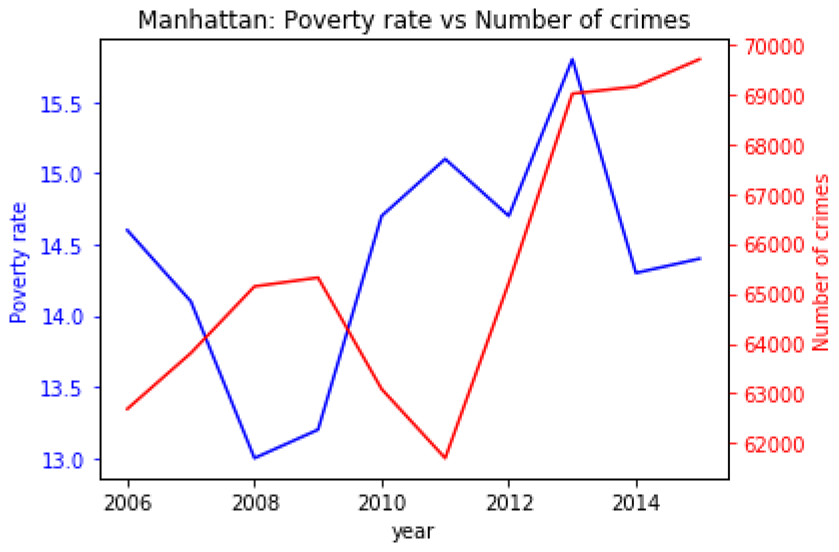
Queens



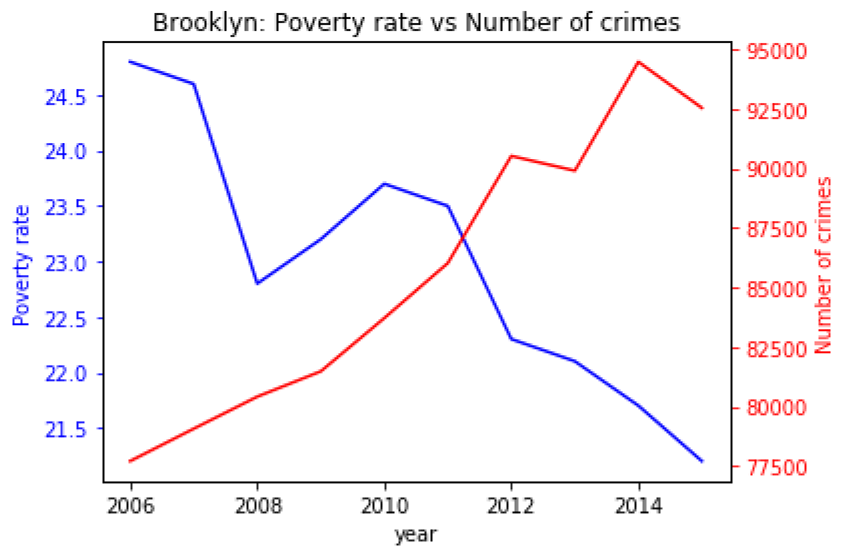
Bronx



Manhattan

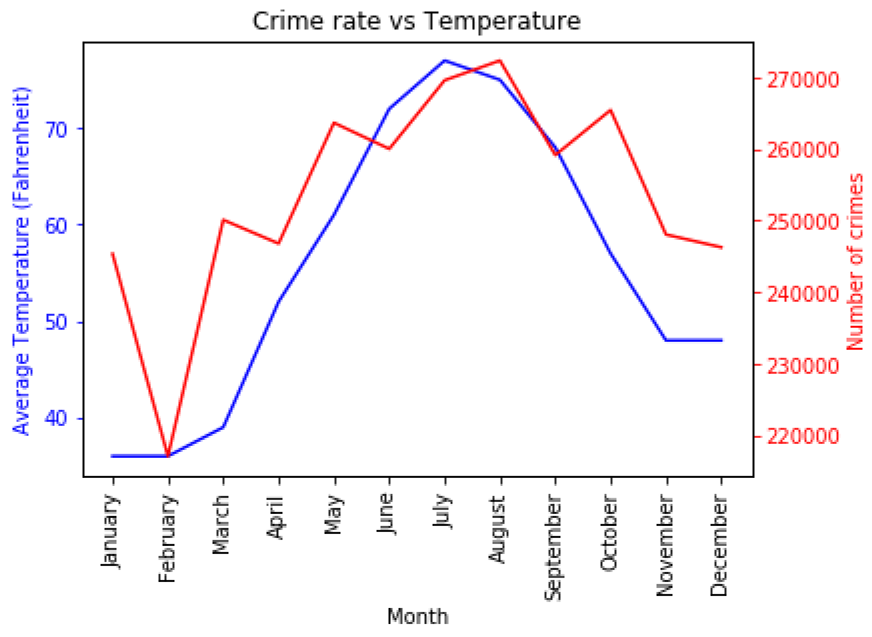


Brooklyn



Observation: From the above plots, it can be seen that crime rate definitely strongly correlates with poverty in Bronx, Queens and Staten Island. There is a weak correlation in Manhattan. In Brooklyn the crime rate increased in spite of decreasing poverty, indicating that there were other factors contributing to crime rate in Brooklyn.
Scripts used-
poverty_by_year.ipynb

2) **Hypothesis: Crime rate is high during summer as compared to winter**
To prove this hypothesis, I found the average temperature in New York in different months and compared that with the total number of crimes in that month.



Observation: From the above plot, the hypothesis is clearly proven, as it can be seen that crime rate is high during summer months (reaching a peak during July/August) and low during winter months (the lowest point being in February).
Scripts used-
weather_by_year.ipynb

Individual Contributions

Rakshit - Worked on columns 0 to 5 for data cleaning and data correction. The data contained from and to dates and time of the crime.

- Observation was that some entries had wrong data in terms of time and some was empty. I have corrected the data and produced relevant data files.
- The tricky part of this data was that To date of the crime had many null/empty values, hence we cannot just throw that tuple because it is not present.
- We categorised the dates into 4 major categories, INVALID, EXACT, RANGE and ENDPOINT. Everything apart from INVALID is VALID data and is used to produce valid/corrected data.
- Throwing away any empty value did not make sense because that would have reduced data, hence I performed cross column validations.
- All the data is categorized and tagged in separate files which are mentioned above in the column details.
- Regular expression were used to validate the formatting of data and has been corrected wherever possible.
- The dates and time we also checked for correct semantic values and range.
- The output files contain detailed summary for statistics for the data and tagging.
- The merging is possible because the unique identifier is maintained in the data set of the valid and the invalid files. This unique identifier is used to merge the data at a later stage.
- Scripts were not written for individual columns but rather based on semantics, such as cleaning for dates data went into single script and for time, a different single script.
- Generated the plots for analysis for crime per year and per law_cd_type with per borough.

Saurabh - Worked on columns 19 to 23 for data cleaning and data correction.

- The columns contained data for the locations of the crime. All the rows with empty columns were marked invalid while the other values were checked if the were valid locations within the city.
- Performed cross column validation to check if the locations values are accurate.
- Merged the valid and corrected columns to obtain final cleaned dataset.
- Analyzed crime using various columns like borough and jurisdiction, borough and precincts to determine which borough had the most cases for reported crimes.

- Generated plots after analysis for the above columns using Pandas.

Sneha - **Worked on columns 6 to 18 for data cleaning and data correction**.

Data Cleaning Part -

- Manual analysis of data to look for invalid values. Eg - “Other” value was present in column 12 and column 16. This was identified by manually analyzing the distinct values for every column and identifying the invalid values.
- Cross column validations done for key and description(columns 6, 7 and columns 8, 9). Idea used - For a particular key, calculate the count of each description and find the description with max count. The other description values for the key are considered invalid. For data correction, the other description values are replaced with the description having max count.
- Created common code in helper.py which is used for validating all columns 6 to 18.
- Research for Reference data - Created reference data file for Precinct[2] and Housing Development[3]. Modified reference dataset for “Precinct” by mapping String values to integers to prevent incorrect validation of precinct data. Used this dataset to validate those column values.
- Data type validation was performed for integer values. Values which were enum are also validated using the list of valid values. This list is obtained by manual analysis.
- For every column separate files are generated - valid, invalid, statistics, corrected. Corrected file is obtained by correcting values in the column whenever possible. Statistics file contains the summary about count of valid data, invalid data and invalid data which has max frequency.

Data Analysis

- Crime rate over the years
- Total crimes in different months, showing which months have most/least crimes.
- Total crimes on different weekdays, showing which weekdays have frequent crimes.
- Total crimes at different times of day, showing what time crimes are more likely to happen.
- Total crimes by borough, showing which boroughs have highest/lowest number of crimes.
- Total crimes classified by top 5 offense descriptions.
- Yearly crimes classified by top 5 offense descriptions.
- Total crimes classified by level of offense.
- Yearly crimes classified by level of offense.
- Top 5 safe and unsafe parks in each borough.
- Top 10 safe and unsafe parks over all boroughs.
- Total number of failed and successful crime attempts.
- A scatter plot of the location of crimes in different boroughs.

Data Exploration

- Proposed a hypothesis that the crime rate increases with poverty and demonstrated that the hypothesis holds for most boroughs.

- Proposed a hypothesis that the crime rate is higher in summer than in winter and proved it using average temperature data.

Summary

Data Cleaning

- Among the invalid data, count of empty values was maximum.
- There were instances where same key had different descriptions. This was fixed by replacing such descriptions with the description having the highest count for a particular key.
- For the column "Housing Development"[3] and "Precinct"[2], some invalid values were obtained by using a different data set as reference.
- By manual data analysis, some invalid values (like "Other") were obtained for some columns.
- The original NYPD crime file 1396 MB in size is reduced to 903MB file after Data Cleaning.

Data Analysis

- Crimes have increased over the past ten years.
- More crimes were reported on weekdays than weekends.
- N.Y. Police Department had jurisdiction for about 95 percent of the crimes.
- Felonies are the most common crime reported to N.Y.P.D with numbers almost double than violations and misdemeanors combined.
- Brooklyn has a higher crime rate reported to the N.Y.P.D than other boroughs.
- Staten Island borough has the least reported crime.
- Brooklyn has crime concentrated in certain regions as evident by the crimes reported in a precinct.
- Central Park and Flushing Meadows Corona Park have more crimes than other parks in New York.

Data Exploration

- Crime rate strongly correlates with poverty in Bronx, Queens and Staten Island.
- Crime rate is high during summer months as compared to winter months.

References

1. https://data.cityofnewyork.us/Public-Safety/NYPD-Complaint-Data-Historic/qgea-i56i
2. https://www1.nyc.gov/site/nypd/bureaus/patrol/precincts-landing.page

3. http://www1.nyc.gov/site/nycha/about/developments.page
4. http://www1.nyc.gov/site/opportunity/poverty-in-nyc/data-tool.page
5. http://www.holiday-weather.com/new_york_city/averages/