Data Analytics and Visualization Cohort 5 - U of MN

Project 2: St Paul Police Stops

Team: The St. Paul 99

Purpose

Tools Used

- Mongo DB
- Python, Pandas, MatPlotLib, Numpy
- Jupyter Notebooks
- FLASK Server
- HTML, CSS, Bootstrap
- Leaflet
- Plotly
- D3

```
In [1]: # import Dependencies
        import pandas as pd
        import requests
        import pymongo
```

Extraction

Sources used for this project:

From Saint Paul Minnesota Public Safety

Web Site: ***https://information.stpaul.gov/Public-Safety/Saint-Paul-Police-Grid-Shapefile/ykwt-ie3e (https://information.stpaul.gov/Public-Safety/Saint-Paul-Police-Grid-Shapefile/ykwt-ie3e)***

GEOjSON Shapes: ***https://information.stpaul.gov/api/geospatial/ykwt-ie3e?method=export&format=GeoJSON (https://information.stpaul.gov/api/geospatial/ykwt-ie3e?method=export&format=GeoJSON)***

Traffic Stop Dataset: ***https://information.stpaul.gov/api/views/kkd6-vvns/rows.csv?accessType=DOWNLOAD (https://information.stpaul.gov/api/views/kkd6-vvns/rows.csv?accessType=DOWNLOAD)***

In [2]:
```
# read St Paul Traffic Stop csv into dataframe and display head
traffic_stops_csv = "../server/data/Traffic_Stop_Dataset.csv"
traffic_stops_df = pd.read_csv(traffic_stops_csv)
traffic_stops_df.head()
```

Out[2]:

	YEAR OF STOP	DATE OF STOP	RACE OF DRIVER	GENDER OF DRIVER	DRIVER SEARCHED?	VEHICLE SEARCHED?	CITATION ISSUED?	AGE OF DRIVER	REASO FO STO
0	2003	04/26/2003 06:36:00 PM	White	Male	Yes	Yes	No	NaN	No Da
1	2003	05/02/2003 12:05:00 PM	Black	Female	No	No	No	NaN	No Da
2	2003	04/30/2003 12:36:00 AM	Black	Male	No	No	No	NaN	No Da
3	2003	04/25/2003 11:07:00 PM	White	Female	No	No	No	NaN	No Da
4	2003	05/01/2003 10:06:00 AM	White	Female	No	No	No	NaN	No Da

Transformation

Explore the data set, clean it and prepare it for use

In [3]:
```
# Start by evaluating numerical the data
traffic_stops_df.describe()
```

Out[3]:

	YEAR OF STOP	AGE OF DRIVER	POLICE GRID NUMBER	COUNT
count	741482.000000	113269.000000	741471.000000	741482.0
mean	2008.604871	33.729034	101.221829	1.0
std	5.220995	13.016151	65.868562	0.0
min	2001.000000	2.000000	1.000000	1.0
25%	2004.000000	23.000000	59.000000	1.0
50%	2008.000000	30.000000	90.000000	1.0
75%	2013.000000	42.000000	128.000000	1.0
max	2018.000000	96.000000	999.000000	1.0

In [4]:
```
# List the columns
traffic_stops_df.columns
```

Out[4]:
```
Index(['YEAR OF STOP', 'DATE OF STOP', 'RACE OF DRIVER', 'GENDER OF DRIVER',
       'DRIVER SEARCHED?', 'VEHICLE SEARCHED?', 'CITATION ISSUED?',
       'AGE OF DRIVER', 'REASON FOR STOP', 'POLICE GRID NUMBER',
       'LOCATION OF STOP BY POLICE GRID', 'COUNT'],
      dtype='object')
```

In [5]:
```
# Determine the number of NaN values in the dataset
traffic_stops_df.isnull().sum().sum()
```

Out[5]:
```
629438
```

In [6]:
```
# Check the data types
traffic_stops_df.dtypes
```

Out[6]:
```
YEAR OF STOP                         int64
DATE OF STOP                        object
RACE OF DRIVER                      object
GENDER OF DRIVER                    object
DRIVER SEARCHED?                    object
VEHICLE SEARCHED?                   object
CITATION ISSUED?                    object
AGE OF DRIVER                      float64
REASON FOR STOP                     object
POLICE GRID NUMBER                 float64
LOCATION OF STOP BY POLICE GRID     object
COUNT                                int64
dtype: object
```

In [7]:
```
# Look at counts
traffic_stops_df.count()
```

Out[7]:
```
YEAR OF STOP                       741482
DATE OF STOP                       741482
RACE OF DRIVER                     741482
GENDER OF DRIVER                   741482
DRIVER SEARCHED?                   741482
VEHICLE SEARCHED?                  741482
CITATION ISSUED?                   741482
AGE OF DRIVER                      113269
REASON FOR STOP                    741482
POLICE GRID NUMBER                 741471
LOCATION OF STOP BY POLICE GRID    740268
COUNT                              741482
dtype: int64
```

In [8]:
```
# From above it is clear that not all traffic stops include Age, a Police Grid
Number or a Location (Lat, Lon)
# From the web site documentation:
#    - Reason for stop was not collected prior to 2017
#    - Age is only collected for citations
#    - Race is based on perception of the officer
# Based on the above information and discussion with the team
# we will eliminate the Age and Count columns, data prior to 2017 will be excl
uded
#    and all "No Data" or NaN values will be filtered out

# Start by dropping years prior to 2017
traffic_stops_2017plus = traffic_stops_df.loc[(traffic_stops_df["YEAR OF STOP"
] > 2016)]
traffic_stops_2017plus.head()
```

Out[8]:

	YEAR OF STOP	DATE OF STOP	RACE OF DRIVER	GENDER OF DRIVER	DRIVER SEARCHED?	VEHICLE SEARCHED?	CITATION ISSUED?	AGE OF DRIVER	F
328823	2017	05/12/2017 08:35:00 PM	Other	Male	No	No	No	NaN	
329089	2017	01/20/2017 09:13:00 PM	White	Female	No	No	No	NaN	
330943	2017	04/13/2017 02:05:00 PM	White	Male	No	No	Yes	26.0	
331296	2017	05/12/2017 08:35:00 PM	White	Female	No	No	Yes	NaN	
332033	2017	04/04/2017 01:48:00 PM	Other	Male	No	No	Yes	42.0	

In [9]:
```
# Next, look at Age
traffic_stops_2017plus = traffic_stops_2017plus.drop(columns = ["AGE OF DRIVE
R", "COUNT"])
traffic_stops_2017plus.head()
```

Out[9]:

	YEAR OF STOP	DATE OF STOP	RACE OF DRIVER	GENDER OF DRIVER	DRIVER SEARCHED?	VEHICLE SEARCHED?	CITATION ISSUED?	REASON FOR STOP
328823	2017	05/12/2017 08:35:00 PM	Other	Male	No	No	No	Moving Violation
329089	2017	01/20/2017 09:13:00 PM	White	Female	No	No	No	Moving Violation
330943	2017	04/13/2017 02:05:00 PM	White	Male	No	No	Yes	Moving Violation
331296	2017	05/12/2017 08:35:00 PM	White	Female	No	No	Yes	Moving Violation
332033	2017	04/04/2017 01:48:00 PM	Other	Male	No	No	Yes	Moving Violation

In [10]:
```python
# Now drop Nan for the entire data set
traffic_stops_2017plus = traffic_stops_2017plus.dropna()
traffic_stops_2017plus
```

Out[10]:

	YEAR OF STOP	DATE OF STOP	RACE OF DRIVER	GENDER OF DRIVER	DRIVER SEARCHED?	VEHICLE SEARCHED?	CITATION ISSUED?	REASON FOR STOP
332093	2017	02/17/2017 06:45:00 PM	White	Female	No	No	No	Equipment Violation
332094	2017	03/07/2017 07:27:00 PM	Latino	Male	No	No	No	Moving Violation
332095	2017	02/06/2017 09:28:00 AM	White	Female	No	No	Yes	Moving Violation
332096	2017	02/22/2017 01:48:00 PM	White	Male	No	No	Yes	Moving Violation
332097	2017	03/02/2017 11:39:00 AM	White	Male	No	No	No	Moving Violation
...	...	...	...	...	...	...	...	...
433515	2017	11/29/2017 04:56:00 PM	Black	Male	Yes	Yes	No	Moving Violation
433516	2017	11/29/2017 04:03:00 PM	Black	Male	No	No	Yes	Moving Violation
433517	2017	11/29/2017 03:44:00 PM	White	Female	No	No	Yes	Moving Violation
433518	2017	11/29/2017 02:52:00 PM	White	Male	No	No	Yes	Moving Violation
433519	2017	11/29/2017 04:27:00 PM	White	Female	No	No	Yes	Moving Violation

62696 rows × 10 columns

```
In [11]: # Describe the resulting data to ensure all counts are the same
         traffic_stops_2017plus.describe()
```

Out[11]:

	YEAR OF STOP	POLICE GRID NUMBER
count	62696.000000	62696.000000
mean	2017.486522	100.172738
std	0.499822	60.999176
min	2017.000000	1.000000
25%	2017.000000	54.000000
50%	2017.000000	89.000000
75%	2018.000000	131.000000
max	2018.000000	280.000000

```
In [12]: # Validate that all columns contain the same count
         traffic_stops_2017plus.count()
```

```
Out[12]: YEAR OF STOP                        62696
         DATE OF STOP                        62696
         RACE OF DRIVER                      62696
         GENDER OF DRIVER                    62696
         DRIVER SEARCHED?                    62696
         VEHICLE SEARCHED?                   62696
         CITATION ISSUED?                    62696
         REASON FOR STOP                     62696
         POLICE GRID NUMBER                  62696
         LOCATION OF STOP BY POLICE GRID     62696
         dtype: int64
```

Check all columns and ensure no bad data

```
In [13]: traffic_stops_2017plus['YEAR OF STOP'].value_counts()
```

```
Out[13]: 2017    32193
         2018    30503
         Name: YEAR OF STOP, dtype: int64
```

```
In [14]: traffic_stops_2017plus['DATE OF STOP'].value_counts()
```

```
Out[14]: 08/08/2018 08:26:00 AM    6
         05/07/2018 02:55:00 PM    5
         06/04/2017 03:37:00 PM    4
         04/25/2017 09:02:00 AM    4
         09/26/2018 10:27:00 AM    4
                                  ..
         08/09/2017 09:27:00 PM    1
         06/01/2018 06:53:00 PM    1
         09/13/2017 01:57:00 PM    1
         03/21/2018 02:17:00 PM    1
         11/08/2017 10:53:00 AM    1
         Name: DATE OF STOP, Length: 58834, dtype: int64
```

```
In [15]: traffic_stops_2017plus['RACE OF DRIVER'].value_counts()
```

```
Out[15]: White              25951
         Black              21587
         Asian               7700
         Latino              3697
         Other               3363
         Native American      264
         No Data              134
         Name: RACE OF DRIVER, dtype: int64
```

```
In [16]: traffic_stops_2017plus['GENDER OF DRIVER'].value_counts()
```

```
Out[16]: Male       39841
         Female     22721
         No Data      134
         Name: GENDER OF DRIVER, dtype: int64
```

```
In [17]: traffic_stops_2017plus['DRIVER SEARCHED?'].value_counts()
```

```
Out[17]: No         58010
         Yes         4552
         No Data      134
         Name: DRIVER SEARCHED?, dtype: int64
```

```
In [18]: traffic_stops_2017plus['VEHICLE SEARCHED?'].value_counts()
```

```
Out[18]: No         58519
         Yes         4043
         No Data      134
         Name: VEHICLE SEARCHED?, dtype: int64
```

```
In [19]: traffic_stops_2017plus[ 'CITATION ISSUED?'].value_counts()
```

```
Out[19]: Yes    33043
         No     29653
         Name: CITATION ISSUED?, dtype: int64
```

```
In [20]: traffic_stops_2017plus['REASON FOR STOP'].value_counts()
```

```
Out[20]: Moving Violation               47990
         Equipment Violation            12206
         Investigative Stop              2229
         911 Call / Citizen Reported      137
         No Data                          134
         Name: REASON FOR STOP, dtype: int64
```

```
In [21]: traffic_stops_2017plus['POLICE GRID NUMBER'].value_counts()
```

```
Out[21]: 133.0    2334
         94.0     1911
         32.0     1573
         54.0     1478
         74.0     1314
                  ...
         189.0       9
         82.0        9
         200.0       3
         197.0       2
         175.0       1
         Name: POLICE GRID NUMBER, Length: 200, dtype: int64
```

```
In [22]: traffic_stops_2017plus['LOCATION OF STOP BY POLICE GRID'].value_counts()
```

```
Out[22]: (44.949881354, -93.083240019)    2334
         (44.959171113, -93.071815477)    1911
         (44.980704001, -93.092622034)    1573
         (44.973868211, -93.071035949)    1478
         (44.966643264, -93.071031663)    1314
                                          ...
         (44.924902368, -93.124896483)       9
         (44.960082673, -93.193955554)       9
         (44.922464278, -93.012029259)       3
         (44.926515742, -93.036180376)       2
         (44.934461449, -93.060250103)       1
         Name: LOCATION OF STOP BY POLICE GRID, Length: 200, dtype: int64
```

In [23]:
```
# Since so many columns have 134 rows set to "No Data", let's see if they alig
n
# first look at the those with "No Data"
traffic_stops_no_data = traffic_stops_2017plus.loc[(traffic_stops_2017plus['GE
NDER OF DRIVER'] == "No Data")]
traffic_stops_no_data
```

Out[23]:

	YEAR OF STOP	DATE OF STOP	RACE OF DRIVER	GENDER OF DRIVER	DRIVER SEARCHED?	VEHICLE SEARCHED?	CITATION ISSUED?	REASON FOR STOP
368548	2017	01/04/2017 11:58:00 PM	No Data	No Data	No Data	No Data	Yes	No Data
368810	2017	01/02/2017 11:52:00 PM	No Data	No Data	No Data	No Data	Yes	No Data
368869	2017	01/11/2017 10:23:00 AM	No Data	No Data	No Data	No Data	No	No Data
368996	2017	01/09/2017 02:26:00 AM	No Data	No Data	No Data	No Data	No	No Data
369168	2017	01/27/2017 11:12:00 PM	No Data	No Data	No Data	No Data	Yes	No Data
...	...	...	...	...	...	...	...	...
378486	2017	03/22/2017 05:52:00 AM	No Data	No Data	No Data	No Data	No	No Data
378511	2017	03/17/2017 11:27:00 PM	No Data	No Data	No Data	No Data	No	No Data
378512	2017	03/25/2017 07:39:00 AM	No Data	No Data	No Data	No Data	No	No Data
378542	2017	03/24/2017 05:27:00 PM	No Data	No Data	No Data	No Data	No	No Data
378898	2017	03/18/2017 08:55:00 PM	No Data	No Data	No Data	No Data	No	No Data

134 rows × 10 columns

In [24]:
```
# Even though it seems they are all the same, let's validate
traffic_stops_no_data['RACE OF DRIVER'].value_counts()
```

Out[24]:
```
No Data    134
Name: RACE OF DRIVER, dtype: int64
```

```
In [25]: traffic_stops_no_data['GENDER OF DRIVER'].value_counts()

Out[25]: No Data    134
         Name: GENDER OF DRIVER, dtype: int64

In [26]: traffic_stops_no_data['DRIVER SEARCHED?'].value_counts()

Out[26]: No Data    134
         Name: DRIVER SEARCHED?, dtype: int64

In [27]: traffic_stops_no_data['VEHICLE SEARCHED?'].value_counts()

Out[27]: No Data    134
         Name: VEHICLE SEARCHED?, dtype: int64

In [28]: traffic_stops_no_data['REASON FOR STOP'].value_counts()

Out[28]: No Data    134
         Name: REASON FOR STOP, dtype: int64

In [29]: # Confirmed - all align
         # Now let's check the dates of these stops and compare to the rest of the data
         to see if there is a cut off date
         traffic_stops_no_data['DATE OF STOP'].sort_values(ascending=True).value_counts
         ()
         print("min:  " + traffic_stops_no_data['DATE OF STOP'].min())
         print("max:  " + traffic_stops_no_data['DATE OF STOP'].max())

         min:  01/01/2017 01:29:00 AM
         max:  03/28/2017 11:43:00 PM
```

In [30]:
```
# Now check the rest of the data
traffic_stops_data = traffic_stops_2017plus.loc[(traffic_stops_2017plus['GENDE
R OF DRIVER'] != "No Data")]
traffic_stops_data.head()
```

Out[30]:

	YEAR OF STOP	DATE OF STOP	RACE OF DRIVER	GENDER OF DRIVER	DRIVER SEARCHED?	VEHICLE SEARCHED?	CITATION ISSUED?	REASON FOR STOP
332093	2017	02/17/2017 06:45:00 PM	White	Female	No	No	No	Equipment Violation
332094	2017	03/07/2017 07:27:00 PM	Latino	Male	No	No	No	Moving Violation
332095	2017	02/06/2017 09:28:00 AM	White	Female	No	No	Yes	Moving Violation
332096	2017	02/22/2017 01:48:00 PM	White	Male	No	No	Yes	Moving Violation
332097	2017	03/02/2017 11:39:00 AM	White	Male	No	No	No	Moving Violation

In [31]:
```
# peek at the values sorted
traffic_stops_data['DATE OF STOP'].sort_values(ascending=True).value_counts()
print("min:  " + traffic_stops_data['DATE OF STOP'].min())
print("max:  " + traffic_stops_data['DATE OF STOP'].max())
```

```
min:  01/01/2017 01:17:00 AM
max:  12/31/2018 12:57:00 PM
```

In [32]:
```
# Not confirmed!  Data missing is the first quarter of 2017,
# BUT there is valid data during that timeframe.
# The team decided to keep all data for 2017 and note these rows were eliminat
ed
```

In [33]:
```
# And now, ensure there are no Nan values let in the dataset
traffic_stops_data.isnull().sum().sum()
```

Out[33]:
```
0
```

In [34]:
```
# Finally, let's simplify the column names
# rankings_pd.rename(columns = {'test':'TEST'}, inplace = True)
columns = {'YEAR OF STOP':"Year", 'DATE OF STOP':"Date", 'RACE OF DRIVER':"Rac
e", 'GENDER OF DRIVER':"Gender",
       'DRIVER SEARCHED?':"DriverSearched", 'VEHICLE SEARCHED?':"VehicleSearch
ed", 'CITATION ISSUED?':"Citation",
       'REASON FOR STOP':"Reason", 'POLICE GRID NUMBER':"Grid",
       'LOCATION OF STOP BY POLICE GRID':"Location"}
traffic_stops_data.rename(columns = columns, inplace=True)
traffic_stops_data.columns
```

```
C:\Users\katro\Anaconda3\envs\PythonData\lib\site-packages\pandas\core\frame.
py:4223: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/st
able/user_guide/indexing.html#returning-a-view-versus-a-copy
  return super().rename(**kwargs)
```

Out[34]:
```
Index(['Year', 'Date', 'Race', 'Gender', 'DriverSearched', 'VehicleSearched',
       'Citation', 'Reason', 'Grid', 'Location'],
      dtype='object')
```

In [35]:
```
# Display the final data set
traffic_stops_data
```

Out[35]:

	Year	Date	Race	Gender	DriverSearched	VehicleSearched	Citation	Reason
332093	2017	02/17/2017 06:45:00 PM	White	Female	No	No	No	Equipment Violation
332094	2017	03/07/2017 07:27:00 PM	Latino	Male	No	No	No	Moving Violation
332095	2017	02/06/2017 09:28:00 AM	White	Female	No	No	Yes	Moving Violation
332096	2017	02/22/2017 01:48:00 PM	White	Male	No	No	Yes	Moving Violation
332097	2017	03/02/2017 11:39:00 AM	White	Male	No	No	No	Moving Violation
...	...	...	...	...	...	...	...	...
433515	2017	11/29/2017 04:56:00 PM	Black	Male	Yes	Yes	No	Moving Violation
433516	2017	11/29/2017 04:03:00 PM	Black	Male	No	No	Yes	Moving Violation
433517	2017	11/29/2017 03:44:00 PM	White	Female	No	No	Yes	Moving Violation
433518	2017	11/29/2017 02:52:00 PM	White	Male	No	No	Yes	Moving Violation
433519	2017	11/29/2017 04:27:00 PM	White	Female	No	No	Yes	Moving Violation

62562 rows × 10 columns

Load

Using MongoDB to store our data and writing out to a csv for ease of reference/evidence

<NOTE: Insert App.py code here if we want to tell the story of the server side>

Save out the final clean dataset in csv form to be loaded by our server side code into MongoDB

In [36]:
```
# Write the cleaned data set for St Paul Traffic Stop data into csv
traffic_stops_final_csv = "../server/data/Traffic_Stop_Cleaned_Data.csv"
traffic_stops_data.to_csv(traffic_stops_final_csv)
```