An Analysis of Police Stops in Rhode Island

by Gursev Pirge

This study is an analysis of police stops in Rhode Island and covers more than 509,000 police stops conducted between the years 2005 and 2015. In this article, I analysed a dataset using Pandas in order to figure out whether race, age, gender, time of the day and other factors were effective in the decision made by the police to make the stop. After some iterations, I decided to analyze the effect of gender and leave the rest of the parameters for future studies.

Considering the effect of gender, there are quite a number of studies, with different results – one study (Racial and gender profiling can affect outcome of traffic stops) claims that gender is important in getting a traffic stop, whereas another study (Gender Bias in Power Relationships: Evidence from Police Traffic Stops) has consistently found gender-based disparities in traffic stops.

Specifically, I tried to examine whether police officers are less likely to issue traffic tickets to men or to women during traffic stops. In the conventional wisdom, it is widely accepted that women are less likely to receive tickets. Then I tried to analyze the results about police search and arrest rates.

The dataset is provided by data.world, home to the world's largest collaborative data community, which is free and open to the public. It's where people discover data, share analysis, and team up on everything from social bot detection to award-winning data journalism.

Data Cleaning

The first step is to import and clean the data using pandas before exploring the relationships between possible parameters and policing. There will be visual exploratory analysis of the police stops using Pandas and Seaborn.

```
In [3]: import pandas as pd
        import matplotlib.pyplot as plt
        import seaborn as sns

In [2]: ri = pd.read_csv('police.csv')

C:\Users\Alp\anaconda3\lib\site-packages\IPython\core\interactiveshell.py:3063: DtypeWarning: Columns (8,16) have mix
ed types.Specify dtype option on import or set low_memory=False.
  interactivity=interactivity, compiler=compiler, result=result)

In [5]: ri.head()
Out[5]:
```

	id	state	stop_date	stop_time	location_raw	county_name	county_fips	fine_grained_location	police_department	driver_gender	...	search_conduc
0	RI-2005-00001	RI	2005-01-02	01:55	Zone K1	NaN	NaN	NaN	600	M	...	Fa
1	RI-2005-00002	RI	2005-01-02	20:30	Zone X4	NaN	NaN	NaN	500	M	...	Fa
2	RI-2005-00003	RI	2005-01-04	11:30	Zone X1	NaN	NaN	NaN	0	NaN	...	Fa
3	RI-2005-00004	RI	2005-01-04	12:55	Zone X4	NaN	NaN	NaN	500	M	...	Fa
4	RI-2005-00005	RI	2005-01-06	01:30	Zone X4	NaN	NaN	NaN	500	M	...	Fa

5 rows × 26 columns

As always, there are missing values (NaNs), so the number of missing values should be determined before starting a detailed analysis.

```
In [7]: ri.shape

Out[7]: (509681, 26)
```

```
In [6]: ri.isnull().sum()

Out[6]: id                            0
        state                         0
        stop_date                    10
        stop_time                    10
        location_raw                  0
        county_name              509681
        county_fips              509681
        fine_grained_location    509681
        police_department            10
        driver_gender             29097
        driver_age_raw            29049
        driver_age                30695
        driver_race_raw           29073
        driver_race               29073
        violation_raw             29073
        violation                 29073
        search_conducted             10
        search_type_raw          491919
        search_type              491919
        contraband_found              0
        stop_outcome              29073
        is_arrested               29073
        stop_duration             29073
        out_of_state              29881
        drugs_related_stop            0
        district                      0
        dtype: int64
```

There are 509681 rows of data, but some attributes (columns) have a lot of missing values and this will cause problems during the analysis. So, deleting the columns with no useful information for the analysis will be more reasonable.

```
In [8]: drop_columns = ['state', 'county_name', 'county_fips', 'fine_grained_location']
        ri.drop(drop_columns, axis = 'columns', inplace = True)

In [9]: ri.head()

Out[9]:
```

	id	stop_date	stop_time	location_raw	police_department	driver_gender	driver_age_raw	driver_age	driver_race_raw	driver_race	...	search_conc
0	RI-2005-00001	2005-01-02	01:55	Zone K1	600	M	1985.0	20.0	W	White	...	
1	RI-2005-00002	2005-01-02	20:30	Zone X4	500	M	1987.0	18.0	W	White	...	
2	RI-2005-00003	2005-01-04	11:30	Zone X1	0	NaN	NaN	NaN	NaN	NaN	...	
3	RI-2005-00004	2005-01-04	12:55	Zone X4	500	M	1986.0	19.0	W	White	...	
4	RI-2005-00005	2005-01-06	01:30	Zone X4	500	M	1978.0	27.0	B	Black	...	

5 rows × 22 columns

So, four columns are deleted and will carry on with the remaining 22 rows of data.

When you know that a specific column will be critical to the analysis, and only a small fraction of rows is missing a value in that column, it often makes sense to remove those rows from the dataset. The driver gender column will most probably be critical to many of the analyses. Missing rows are only small proportion (around 5 %) of the total, so it will be a good idea to drop those rows from the dataset instead of filling them. Now 480584 solid rows of data are left instead of the starting value of 509681.

```
In [11]: ri['driver_gender'].isnull().sum()
Out[11]: 29097

In [12]: ri.dropna(subset = ['driver_gender'], inplace = True)

In [13]: ri.shape
Out[13]: (480584, 22)

In [ ]:
```

The next step will be to set the stop_datetime column as the DataFrame's index. By replacing the default index with a DatetimeIndex, it will be easier to analyze the dataset by date and time, if needed.

```
In [21]: ri.set_index('stop_datetime', inplace = True)

In [23]: ri.head()
Out[23]:
```

stop_datetime	id	location_raw	police_department	driver_gender	driver_age_raw	driver_age	driver_race_raw	driver_race	violation_raw	viol
2005-01-02 01:55:00	RI-2005-00001	Zone K1	600	M	1985.0	20.0	W	White	Speeding	Spee
2005-01-02 20:30:00	RI-2005-00002	Zone X4	500	M	1987.0	18.0	W	White	Speeding	Spee
2005-01-04 12:55:00	RI-2005-00004	Zone X4	500	M	1986.0	19.0	W	White	Equipment/Inspection Violation	Equip
2005-01-06 01:30:00	RI-2005-00005	Zone X4	500	M	1978.0	27.0	B	Black	Equipment/Inspection Violation	Equip
2005-01-12 08:05:00	RI-2005-00006	Zone X1	0	M	1973.0	32.0	B	Black	Call for Service	C

```
In [24]: ri.shape
Out[24]: (480584, 20)
```

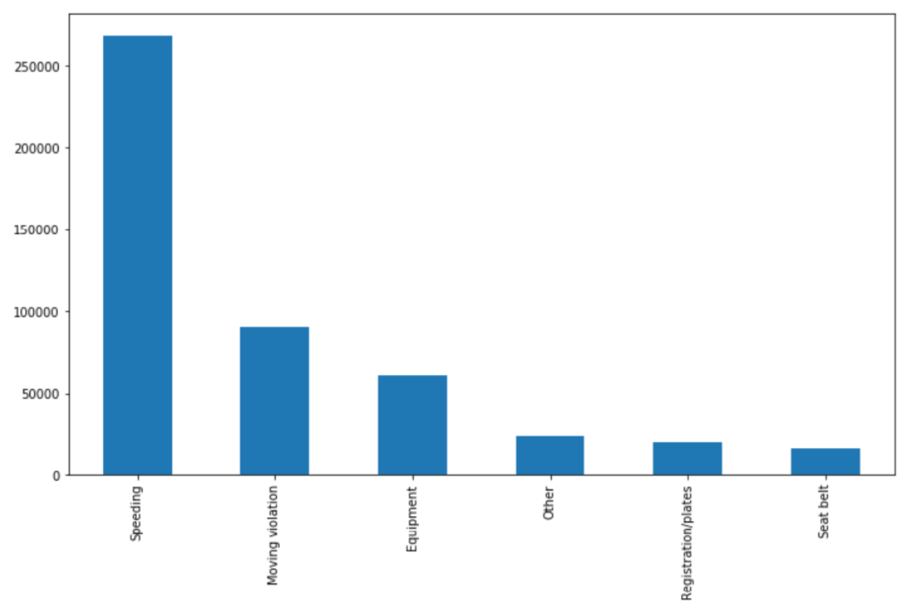
Now, consider the types of violations and try to find a pattern on this parameter.

```
In [26]: ri.violation.value_counts()

Out[26]: Speeding                268736
         Moving violation         90228
         Equipment                61250
         Other                    24216
         Registration/plates      19830
         Seat belt                16324
         Name: violation, dtype: int64
```

```
In [27]: ri.violation.value_counts(normalize = True)

Out[27]: Speeding                0.559186
         Moving violation        0.187747
         Equipment               0.127449
         Other                   0.050389
         Registration/plates     0.041262
         Seat belt               0.033967
         Name: violation, dtype: float64
```

Speeding is by far the most prevalent violation and its percentage among all the violations is 55.9 %. The plot below shows the distribution of violations.

```
In [30]: ri.violation.value_counts().plot(kind = 'bar', figsize = (12, 7));
```



Does Gender Have an Effect on the Police *Behavior*?

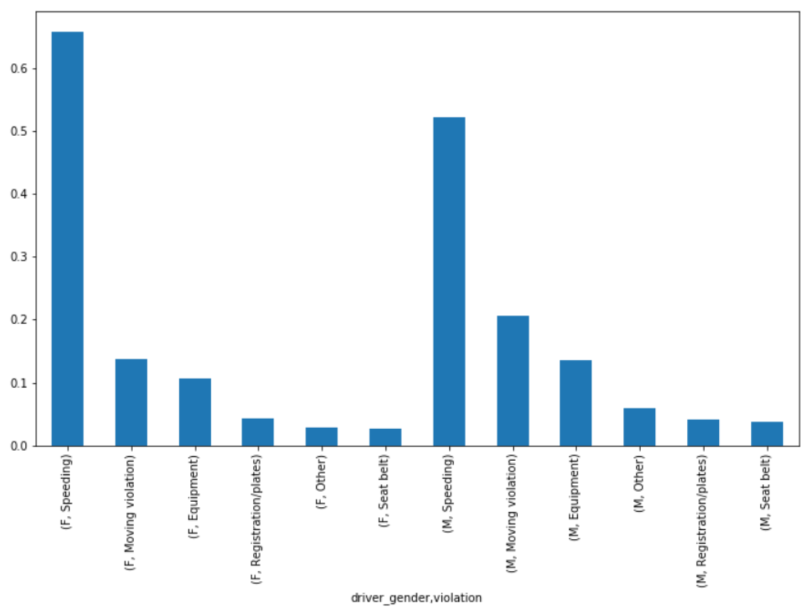
Now that the data is almost ready for gender-based analysis, two separate dataframes for female and male drivers are prepared and the results for the number of violations grouped by the driver gender are given as:

```
In [17]: female.violation.value_counts()

Out[17]: Speeding               86198
         Moving violation       17911
         Equipment              14039
         Registration/plates     5649
         Other                   3791
         Seat belt               3550
         Name: violation, dtype: int64
```

```
In [19]: male.violation.value_counts()

Out[19]: Speeding               182538
         Moving violation        72317
         Equipment               47211
         Other                   20425
         Registration/plates     14181
         Seat belt               12774
         Name: violation, dtype: int64
```

```
In [22]: ri.groupby(['driver_gender'])['violation'].value_counts(normalize = True)

Out[22]: driver_gender  violation
         F              Speeding               0.657308
                        Moving violation       0.136581
                        Equipment              0.107055
                        Registration/plates    0.043077
                        Other                  0.028908
                        Seat belt              0.027071
         M              Speeding               0.522364
                        Moving violation       0.206948
                        Equipment              0.135102
                        Other                  0.058450
                        Registration/plates    0.040581
                        Seat belt              0.036555
         Name: violation, dtype: float64
```

```
In [43]: ri.groupby("driver_gender")["violation"].value_counts(normalize = True).plot.bar(figsize = (12, 7));
```



Now, let us compare the **outcomes** of police stops due to **speeding**. When a driver is pulled over for speeding, many people believe that gender has an impact on whether the driver will receive a ticket or a warning. Once again, two separate datasets are prepared for speeding: for male and female drivers.

```
In [31]: male_and_speeding = ri[(ri['driver_gender'] == 'M') & (ri['violation'] == 'Speeding')]
```

```
In [36]: print('Violations in % for Female Drivers')
         print()
         print(female_and_speeding.stop_outcome.value_counts(normalize = True))
         print()
         print()
         print()
         print('Violations in % for Male Drivers')
         print()
         print(male_and_speeding.stop_outcome.value_counts(normalize = True))

         Violations in % for Female Drivers

         Citation            0.953247
         Warning             0.039003
         Arrest Driver       0.005290
         Arrest Passenger    0.001033
         N/D                 0.000905
         No Action           0.000522
         Name: stop_outcome, dtype: float64



         Violations in % for Male Drivers

         Citation            0.944636
         Warning             0.036086
         Arrest Driver       0.015767
         Arrest Passenger    0.001265
         N/D                 0.001183
         No Action           0.001063
         Name: stop_outcome, dtype: float64
```

Interestingly enough, once the driver is stopped for speeding, there is a slightly higher possibility for female drivers to get a ticket when compared to the male drivers (95.32 % for female drivers vs. 94.46 % for male drivers). On the other hand, there may be other effective factors, but percentage of arrest for male drivers is almost three times when compared to female drivers (0.53 % for female drivers vs. 1.58 % for male drivers).

During a traffic stop, the police officer sometimes **conducts a search** of the vehicle. Calculating the percentage of the stops that result in a full vehicle search will give us some valuable data. Figure below shows that around 3.7 % of the police stops end with a car search.

```
In [33]: ri.search_conducted.value_counts()

Out[33]: False    462822
         True      17762
         Name: search_conducted, dtype: int64

In [36]: ri.search_conducted.value_counts(normalize = True)

Out[36]: False    0.963041
         True     0.036959
         Name: search_conducted, dtype: float64

In [37]: ri.search_conducted.mean()

Out[37]: 0.036959199640437465
```

In [44]:
```
ri.groupby('driver_gender')['search_conducted'].mean()
```

Out[44]:
```
driver_gender
F    0.018751
M    0.043792
Name: search_conducted, dtype: float64
```

When we check the effect of gender on the percentage of searches conducted, we get an important result; male drivers' cars are subject to search by a huge difference (1.88 % for female drivers vs. 4.38 % for male drivers).

The following results combine all types of violations, driver gender and the resulting police search and it is easy from the details to see that, regardless of the type of violation, male drivers are always unlucky when it comes to a possible police search:

In [47]:
```
ri.groupby(['violation', 'driver_gender'])['search_conducted'].mean()
```

Out[47]:
```
violation            driver_gender
Equipment            F                0.040245
                     M                0.070916
Moving violation     F                0.038021
                     M                0.059156
Other                F                0.045898
                     M                0.046120
Registration/plates  F                0.054700
                     M                0.103589
Seat belt            F                0.017746
                     M                0.031705
Speeding             F                0.007738
                     M                0.026630
Name: search_conducted, dtype: float64
```

Conclusion

In this article, I have assessed the effect of gender on the police stops in Rhode Island. The analysis provides evidence that:

- Once a driver is stopped for speeding, there is a slightly higher possibility for female drivers to get a ticket when compared to the male drivers (95.32 % for female drivers vs. 94.46 % for male drivers).
- Percentage of arrest for male drivers is almost three times when compared to female drivers (0.53 % for female drivers vs. 1.58 % for male drivers).
- Male drivers' cars are subject to search by a huge difference (1.88 % for female drivers vs. 4.38 % for male drivers).

Thank you for reading. I am planning to publish articles from the same dataset in the near future.