Big Data in Law Enforcement 2

ENDTERM REPORT

Group: CS-2119

Member:

Aydar Amangeldy

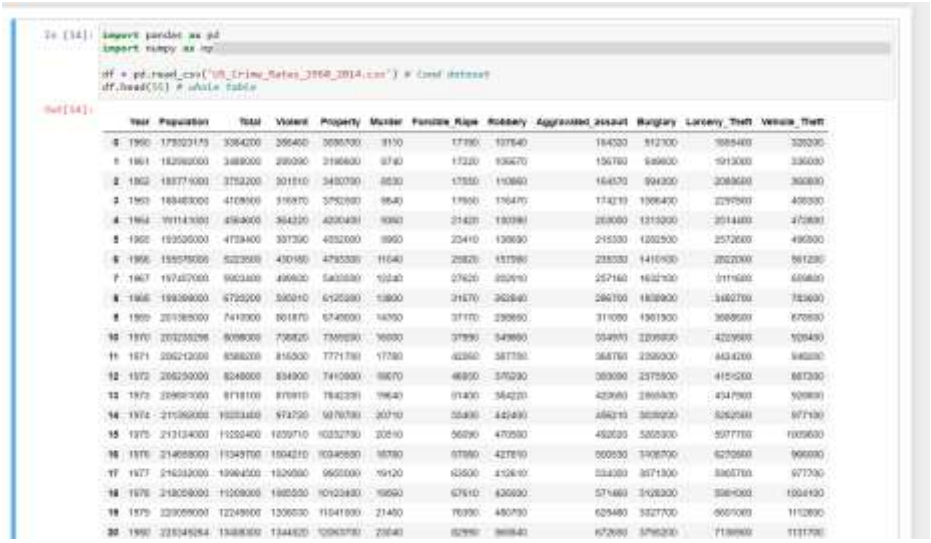
Introduction

The chosen dataset is US_Crime_Rates_1960_2014 from Kaggle website. This dataset is highly relevant to law enforcement, since it contains types of crimes occurred in USA from 1960 to 2014. By identifying patterns in the given dataset, we can project potential issues and improve law enforcement. The goal of the project is to identify key trends and patterns which will be helpful in decreasing the crime rate.

The link to Github: https://github.com/P1zzApple/bdle2_endterm

Data Preparation

1. Download dataset from Kaggle (https://www.kaggle.com/datasets/mahmoudshogaa/us-crime-rates-1960-2014?resource=download)
2. Open Jupyter Notebook
3. Extract dataset table in Jupyter(photo below)

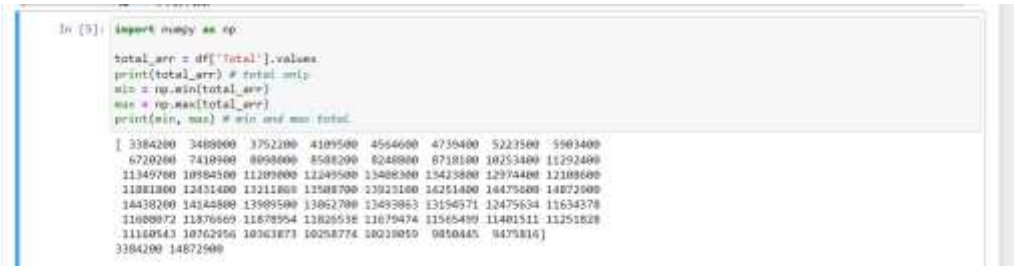


Analysis

After the preparation, we will try to look to trends and changes in this dataset.

To start, we can use df.describe() function to get basic data out of table:

Try to get in which years there were minimum/maximum total crimes. We will try using NumPy to sort (despite this data already existing in previous line)



```
In [5]: import numpy as np

total_arr = df['Total'].values
print(total_arr) # total only
min = np.min(total_arr)
max = np.max(total_arr)
print(min, max) # min and max total
```

```
[ 3384200  3488000  3752200  4109500  4564600  4739400  5223500  5503400
  6720200  7410900  8098000  8508200  8248800  8718100 10253400 11292400
 11349700 10984500 11309000 12249500 13408300 13423800 12974400 12108600
 11381800 12431400 13211800 13588700 13925100 14251400 14475600 14872900
 14438200 14144800 13909500 13862700 13493863 13194571 12475634 11634378
 11680072 11876669 11878954 11826538 11679474 11565499 11401511 11251828
 11160543 10762956 10363873 10258774 10210059  9850445  9475816]
3384200 14872900
```

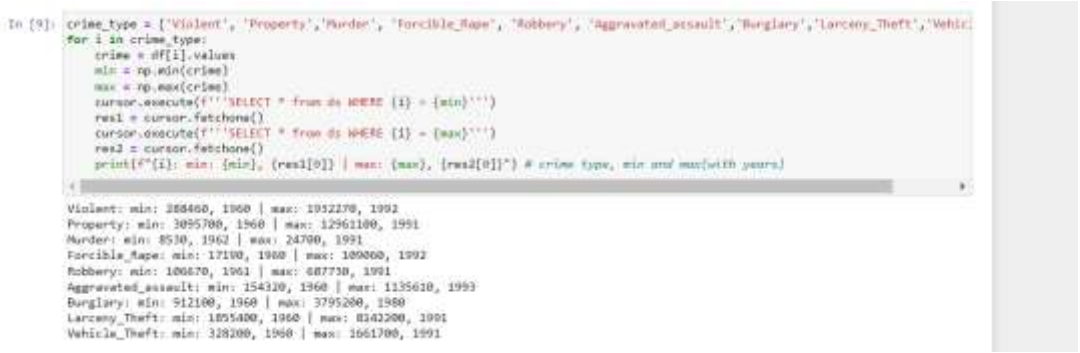
Use SQLite3 to find years via sql commands

```
In [7]: import sqlite3
conn = sqlite3.connect('and.db')
cursor = conn.cursor()

#df.to_sql("ds", conn)
cursor.execute(f'''
    SELECT * from ds WHERE Total = {min}
''')
#print(sdf)
res = cursor.fetchone()
print(f'Year: {res[0]}, Population: {res[1]}, Total crimes: {min}')
cursor.execute(f'''
    SELECT * from ds WHERE Total = {max}
''')
res = cursor.fetchone()
print(f'Year: {res[0]}, Population: {res[1]}, Total crimes: {max}')
```

```
Year: 1960, Population: 179323175, Total crimes: 3384200
Year: 1991, Population: 252177000, Total crimes: 14872900
```

Next, let's try to use this on all types of crimes. The command outputs the crime type, year with the minimum/maximum amount of said crimes.



```
In [9]: crime_type = ['Violent', 'Property','Murder', 'Forcible_Rape', 'Robbery', 'Aggravated_assault','Burglary','Larceny_Theft','Vehic
for i in crime_type:
    crime = df[i].values
    min = np.min(crime)
    max = np.max(crime)
    cursor.execute(f'''SELECT * from ds WHERE {i} = {min}''')
    res1 = cursor.fetchone()
    cursor.execute(f'''SELECT * from ds WHERE {i} = {max}''')
    res2 = cursor.fetchone()
    print(f"{i}: min: {min}, {res1[0]} | max: {max}, {res2[0]}") # crime type, min and max(with years)
```

```
Violent: min: 288460, 1960 | max: 1932270, 1992
Property: min: 3095700, 1960 | max: 12961100, 1991
Murder: min: 8530, 1962 | max: 24700, 1991
Forcible_Rape: min: 17190, 1960 | max: 109060, 1992
Robbery: min: 106670, 1961 | max: 687730, 1991
Aggravated_assault: min: 154320, 1960 | max: 1135610, 1993
Burglary: min: 912100, 1960 | max: 3795200, 1980
Larceny_Theft: min: 1855400, 1960 | max: 8142200, 1991
Vehicle_Theft: min: 328200, 1960 | max: 1661700, 1991
```

Furthermore, we will try to define top of committed crimes using Apache Spark.

```
In [33]: from pyspark.sql import SparkSession
         from pyspark.sql.functions import col, count, avg, desc, split, explode


         spark = SparkSession.builder.appName("Indians").getOrCreate()

         rdf = spark.read.option("header", "true").csv("US_Crime_Rates_1960_2014.csv")
         rdf = rdf.withColumn("Crime coefficient", col("Total") / col("Population"))
         rdf = rdf.orderBy(col("Crime coefficient").desc())
         top = rdf.limit(5)
         top.select("Year", "Crime coefficient").show()

+----+--------------------+
|Year|   Crime coefficient|
+----+--------------------+
|1980| 0.05950006330055414|
|1991| 0.05897801944700381|
|1981|0.058581864837265126|
|1990| 0.05828275578686014|
|1989| 0.05748099068130788|
+----+--------------------+

In [34]: rdf = rdf.orderBy("Crime coefficient")
         antitop = rdf.limit(5)
         antitop.select("Year", "Crime coefficient").show()

         spark.stop()

+----+--------------------+
|Year|   Crime coefficient|
+----+--------------------+
|1960|0.018872072725680884|
|1961|0.019060942554865786|
|1962|0.020197985032061752|
|1963|0.021003027328724606|
|1964| 0.02388080003766855|
+----+--------------------+
```

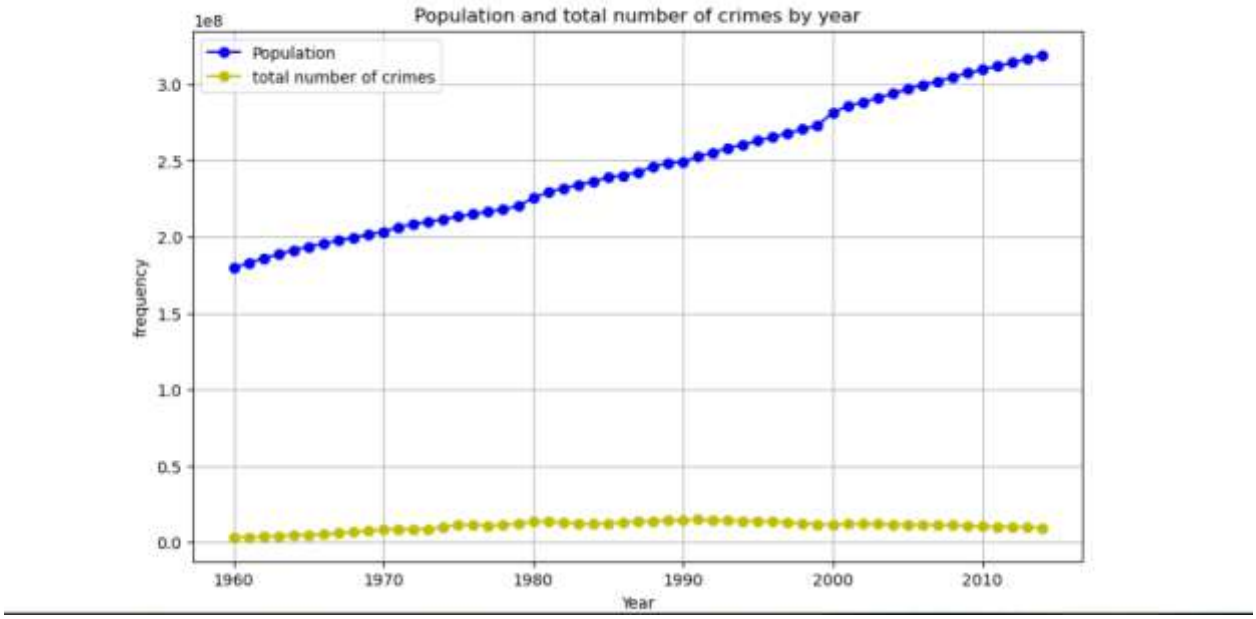
Lastly, to top it off, we can visualize data by using MatPlotLib. Visualize population and total crimes.

```
In [14]: # Visualize the data
         years = df['Year']
         population = df['Population']
         total = df['Total']

         plt.figure(figsize=(10, 6))
         plt.plot(years, population, label='Population', color='b', marker='o')
         plt.plot(years, total, label='total number of crimes', color='y', marker='o')


         plt.xlabel('Year')
         plt.ylabel('frequency')
         plt.title('Population and total number of crimes by year')
         plt.legend()
         plt.grid(True)


         plt.show()
```



Conclusion

To wrap up, the time of most crimes occurred were during the start of the 90s, the crimes coefficient is low compared to the population but this situation must be always cautiously monitored.