

DATA SCIENCE TOOLBOX: PYTHON PROGRAMMING
PROJECT REPORT

(Project Semester January-April 2025)



The Analysis of crime incidents data reported in Los Angeles from 2020 to the present , using government data

Submitted by

Shruti Somvanshi (64)

Registration No-12310394

Programme and Section - B.tech CSE ,K23EH

Course Code- INT 375

Under the Guidance of

Dr. Madhu Bala (UID 31770)

Discipline of CSE

Lovely School of Computer Science and Engineering

Lovely Professional University, Phagwara

CERTIFICATE

This is to certify that Shruti Somvanshi(student's name) bearing Registration no. 12310394 has completed INT 375<Course Code> project titled, **“The Analysis of crime incidents data reported in Los Angeles from 2020 to the present , using government data”** under my guidance and supervision. To the best of my knowledge, the present work is the result of his/her original development, effort and study.

Signature and Name of the Supervisor

Designation of the Supervisor

School of Computer Science and Engineering

Lovely Professional University

Phagwara, Punjab.

Date: 11-04-2024

DECLARATION

I, Ms. Shruti Somvanshi, student of Computer Science and Engineering (Program name) under CSE Discipline at, Lovely Professional University, Punjab, hereby declare that all the information furnished in this project report is based on my own intensive work and is genuine.

Date: 11-04-2025

Registration No. 12310394

Name of the student

Shruti Somvanshi

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to my project guide, [Dr./Mr./Ms. Faculty Name], for their valuable guidance, support, and encouragement throughout the completion of this project.

I also thank my friends for their continuous motivation and helpful discussions during this journey.

Lastly, I acknowledge my own dedication and consistent efforts that helped me bring this project to completion.

Shruti Somvanshi

Registration No:12310394

TABLE OF CONTENTS

1. Introduction	1
2. Source of Dataset	3
3. EDA Process	5
4. Analysis on Dataset	
i. Introduction	7
ii. General Description	8
iii. Specific Requirements, Functions and Formulas	9
iv. Analysis Results	11
v. Visualization	13
5. Conclusion	20
6. Future Scope	22
7. References	24

1. Introduction

Crime is one of the most pressing challenges faced by modern urban societies. It impacts not only public safety but also the overall well-being and trust within communities. As cities grow and evolve, it becomes increasingly important to understand where and when crimes are happening, what types of crimes are most common, and how patterns change over time. This understanding can support more effective law enforcement, better resource allocation, and smarter city planning.

The dataset used in this project contains detailed records of reported crimes, including the **date and time of occurrence, geographic coordinates (latitude and longitude), area name, reporting district, type of crime committed, and the current status of the case**. With such comprehensive information, this dataset offers a valuable opportunity to perform Exploratory Data Analysis (EDA) to uncover meaningful insights.

In this project, we aim to apply Python-based data analytics techniques in Jupyter Notebook to explore, clean, visualize, and understand crime trends. By leveraging this data, we hope to identify crime hotspots, detect anomalies, and observe how crime varies by time, location, and type.

Objectives of This Analysis:

- Identify **crime hotspots** using geographical coordinates.

- Use **boxplots** to detect outliers in numerical fields like time of occurrence, latitude, longitude, and report district number.
- Analyse **crime trends over time**—such as by day, hour, or region.
- Visualize the **frequency and distribution of different types of crimes**.
- Compare areas or districts based on the **number and nature of reported crimes**.
- Use **heatmaps and scatter plots** to map crimes and visually highlight high-crime zones.

Through this project, we aim to gain a deeper understanding of crime dynamics in urban settings and support data-driven decision-making. The ultimate goal is to help law enforcement agencies, city officials, and residents create **safer, more informed, and resilient communities**.

2. Source of Dataset

The dataset used in this project has been taken from the **official open data portal of the City of Los Angeles**. It provides both real-time and historical crime incident records reported to the Los Angeles Police Department (LAPD) across different neighbourhoods and districts.

The dataset includes detailed information about each crime, such as when and where it occurred, the category of crime, the reporting district, and the current status of the investigation. These records serve as a valuable source for studying urban crime trends and supporting public safety efforts.

Dataset Source:

<https://catalog.data.gov/dataset/crime-data-from-2020-to-present>

Key features of the dataset:

- **Date and Time:** The exact timestamp of when the crime occurred and when it was reported.
- **Location:** Includes area name, reporting district number, latitude, and longitude coordinates.
- **Crime Description:** Categorized by crime codes and descriptions, such as burglary, theft, or assault.
- **Crime Classification:** Indicates whether the crime falls under Part 1 or Part 2 offenses.
- **Status:** The current status of the case, such as "Adult Arrest" or "Investigation Continued".

This dataset is highly suitable for analysis as it is well-structured and updated regularly. It enables the identification of **crime patterns over time and across locations**, offering insights into public safety challenges.

Before performing any visualizations or in-depth analysis, the dataset was cleaned and preprocessed using **Python libraries such as pandas and numpy** to handle missing values, convert timestamps, and ensure consistent formatting. This preparation step ensures accurate and meaningful results during Exploratory Data Analysis (EDA).

3. EDA Process

Exploratory Data Analysis, or EDA, is an important step in any data science project.

It helps us to understand the data, find patterns, detect missing or incorrect values, and prepare the data for further analysis.

In this project, the following steps were followed during the EDA process:

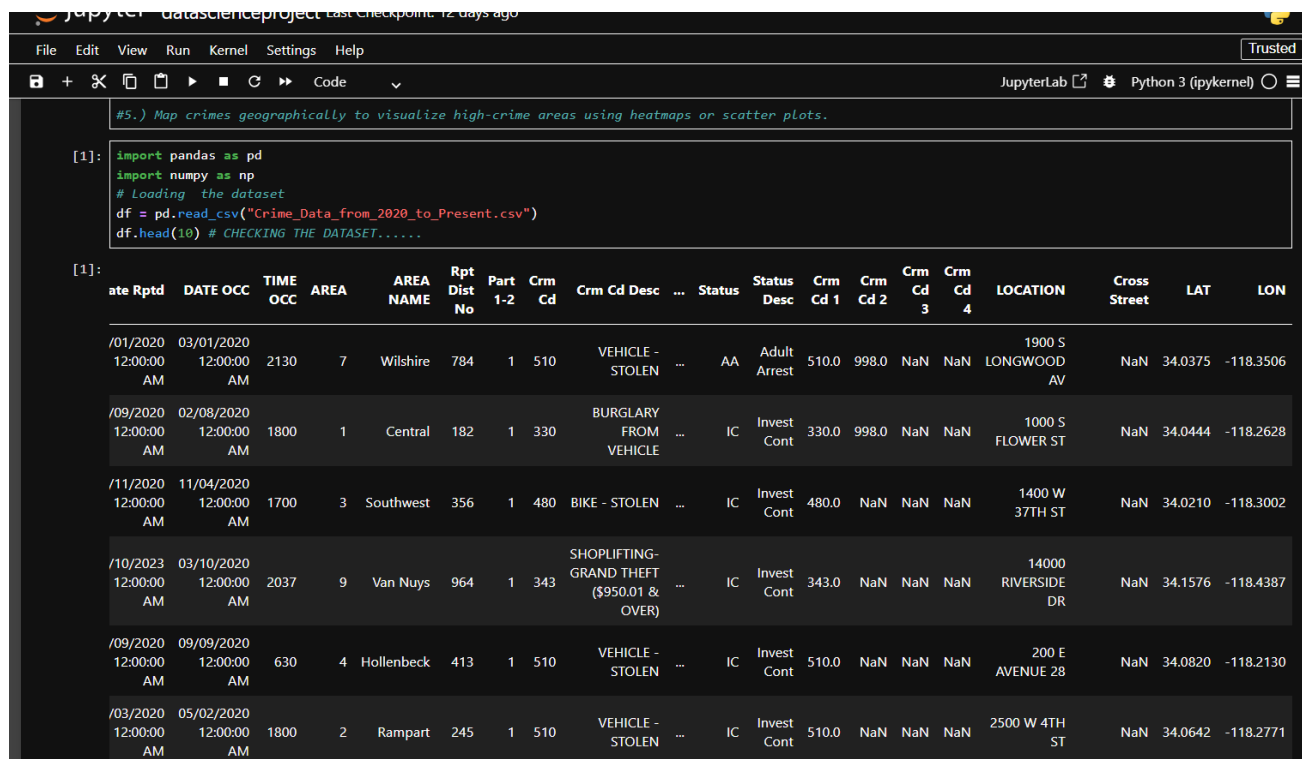
1. Loading the Dataset

The dataset was loaded into a Jupyter Notebook using the pandas library. It was read directly from the CSV file.

```
[1]: import pandas as pd
import numpy as np
# Loading the dataset
df = pd.read_csv("Crime_Data_from_2020_to_Present.csv")
df.head(10) # CHECKING THE DATASET.....
```

2. Understanding the Structure of the Data

We used basic functions like head(), info() and describe() to view the top rows, data types, column names, and summary statistics of the dataset .



#5.) Map crimes geographically to visualize high-crime areas using heatmaps or scatter plots.

```
[1]: import pandas as pd
import numpy as np
# Loading the dataset
df = pd.read_csv("Crime_Data_from_2020_to_Present.csv")
df.head(10) # CHECKING THE DATASET.....
```

ate Rptd	DATE OCC	TIME OCC	AREA	AREA NAME	Rpt Dist No	Part 1-2	Crm Cd	Crm Cd Desc	...	Status	Status Desc	Crm Cd 1	Crm Cd 2	Crm Cd 3	Crm Cd 4	LOCATION	Cross Street	LAT	LON
/01/2020 12:00:00 AM	03/01/2020 12:00:00 AM	2130	7	Wilshire	784	1	510	VEHICLE - STOLEN	...	AA	Adult Arrest	510.0	998.0	NaN	NaN	1900 S LONGWOOD AV	NaN	34.0375	-118.3506
/09/2020 12:00:00 AM	02/08/2020 12:00:00 AM	1800	1	Central	182	1	330	BURGLARY FROM VEHICLE	...	IC	Invest Cont	330.0	998.0	NaN	NaN	1000 S FLOWER ST	NaN	34.0444	-118.2628
/11/2020 12:00:00 AM	11/04/2020 12:00:00 AM	1700	3	Southwest	356	1	480	BIKE - STOLEN	...	IC	Invest Cont	480.0	NaN	NaN	NaN	1400 W 37TH ST	NaN	34.0210	-118.3002
/10/2023 12:00:00 AM	03/10/2020 12:00:00 AM	2037	9	Van Nuys	964	1	343	SHOPLIFTING-GRAND THEFT (\$950.01 & OVER)	...	IC	Invest Cont	343.0	NaN	NaN	NaN	14000 RIVERSIDE DR	NaN	34.1576	-118.4387
/09/2020 12:00:00 AM	09/09/2020 12:00:00 AM	630	4	Hollenbeck	413	1	510	VEHICLE - STOLEN	...	IC	Invest Cont	510.0	NaN	NaN	NaN	200 E AVENUE 28	NaN	34.0820	-118.2130
/03/2020 12:00:00 AM	05/02/2020 12:00:00 AM	1800	2	Rampart	245	1	510	VEHICLE - STOLEN	...	IC	Invest Cont	510.0	NaN	NaN	NaN	2500 W 4TH ST	NaN	34.0642	-118.2771

```
jupyter datascienceproject Last Checkpoint: 12 days ago
File Edit View Run Kernel Settings Help Trusted
+ × □ □ ▶ ■ ↺ ⏪ Code ▾
df.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1005149 entries, 0 to 1005148
Data columns (total 28 columns):
#   Column              Non-Null Count  Dtype
---  -
0   DR_NO               1005149 non-null  int64
1   Date Rptd           1005149 non-null  object
2   DATE OCC            1005149 non-null  object
3   TIME OCC            1005149 non-null  int64
4   AREA                1005149 non-null  int64
5   AREA NAME           1005149 non-null  object
6   Rpt Dist No         1005149 non-null  int64
7   Part 1-2            1005149 non-null  int64
8   Crm Cd              1005149 non-null  int64
9   Crm Cd Desc         1005149 non-null  object
10  Mocodes             853408 non-null   object
11  Vict Age            1005149 non-null  int64
12  Vict Sex            860384 non-null   object
13  Vict Descent        860372 non-null   object
14  Premis Cd           1005133 non-null  float64
15  Premis Desc         1004561 non-null  object
16  Weapon Used Cd      327264 non-null  float64
17  Weapon Desc         327264 non-null  object
18  Status              1005148 non-null  object
19  Status Desc         1005149 non-null  object
20  Crm Cd 1            1005138 non-null  float64
21  Crm Cd 2            69153 non-null    float64
22  Crm Cd 3            2314 non-null     float64
23  Crm Cd 4            64 non-null       float64
24  LOCATION            1005149 non-null  object
25  Gross Street        154240 non-null   object
26  LAT                 1005149 non-null  float64
27  LON                 1005149 non-null  float64
dtypes: float64(8), int64(7), object(13)
memory usage: 214.7+ MB
```

```
df.describe()
```

	DR_NO	TIME OCC	AREA	Rpt Dist No	Part 1-2	Crm Cd	Vict Age	Premis Cd	Weapon Used Cd	Crm Cd 1	Crm Cd 2
count	1.005149e+06	1.005149e+06	1.005149e+06	1.005149e+06	1.005149e+06	1.005149e+06	1.005149e+06	1.005133e+06	327264.000000	1.005138e+06	69153.000000
mean	2.202264e+08	1.339914e+03	1.069108e+01	1.115566e+03	1.400290e+00	5.001465e+02	2.891334e+01	3.056149e+02	363.952412	4.999071e+02	958.109135
std	1.319954e+07	6.510595e+02	6.110401e+00	6.111749e+02	4.899573e-01	2.052626e+02	2.199376e+01	2.193121e+02	123.737059	2.050632e+02	110.355182
min	8.170000e+02	1.000000e+00	1.000000e+00	1.010000e+02	1.000000e+00	1.100000e+02	-4.000000e+00	1.010000e+02	101.000000	1.100000e+02	210.000000
25%	2.106169e+08	9.000000e+02	5.000000e+00	5.870000e+02	1.000000e+00	3.310000e+02	0.000000e+00	1.010000e+02	311.000000	3.310000e+02	998.000000
50%	2.209160e+08	1.420000e+03	1.100000e+01	1.139000e+03	1.000000e+00	4.420000e+02	3.000000e+01	2.030000e+02	400.000000	4.420000e+02	998.000000
75%	2.311105e+08	1.900000e+03	1.600000e+01	1.613000e+03	2.000000e+00	6.260000e+02	4.400000e+01	5.010000e+02	400.000000	6.260000e+02	998.000000
max	2.521041e+08	2.359000e+03	2.100000e+01	2.199000e+03	2.000000e+00	9.560000e+02	1.200000e+02	9.760000e+02	516.000000	9.560000e+02	999.000000

3.) Checking and Handling for Missing Values

The dataset was checked for any missing or null values using `isnull().sum()`.

In case of missing data, either the rows were removed or the values were filled using suitable methods such as mean or forward fill.

To handle missing values effectively:

- 1.) **Latitude and Longitude** values were converted to numeric types using `pd.to_numeric()` to ensure that any improper string values were coerced into NaN. Rows containing missing geographic coordinates were dropped entirely, as these are essential for location-based analyses such as mapping and clustering.
- 2.) **Categorical columns** such as **Area Name** and **Crime Description** were filled with the placeholder value 'Unknown'. This retained important contextual rows in the dataset while clearly indicating missing descriptive information.

```

]: df.isnull().sum()
DR_NO      0
Date Rptd   0
DATE OCC    0
TIME OCC    0
AREA        0
AREA_NAME   0
Rpt Dist No 0
Part 1-2    0
Crm Cd      0
Crm Cd Desc 0
Mocodes     151741
Vict Age    0
Vict Sex    144765
Vict Descent 144777
Premis Cd   16
Premis Desc 588
Weapon Used Cd 677885
Weapon Desc 677885
Status      1
Status Desc 0
Crm Cd 1    11
Crm Cd 2    935996
Crm Cd 3    1002835
Crm Cd 4    1005085
LOCATION      0
Cross Street 850909
LAT          0
LON          0
dtype: int64

```

Step 3: Handle Missing Values

```

# Convert LAT and LON to numeric
df['Latitude'] = pd.to_numeric(df['Latitude'], errors='coerce')
df['Longitude'] = pd.to_numeric(df['Longitude'], errors='coerce')

# Drop rows where location data is missing
df.dropna(subset=['Latitude', 'Longitude'], inplace=True)

# Fill missing values safely WITHOUT using chained assignment
df['Area_Name'] = df['Area_Name'].fillna('Unknown')
df['Crime_Description'] = df['Crime_Description'].fillna('Unknown')

```

3. Detection of outliers in a dataset

4.) Detection and Handling of Outliers in the Dataset

Outliers are extreme values that deviate significantly from other observations in the dataset. Identifying and managing outliers is crucial to ensure the accuracy of statistical analysis and machine learning models.

In this analysis, we focused on detecting outliers in the **Victim_Age** column, as age-based anomalies can skew distributions and affect insights:

- The **Interquartile Range (IQR)** method was used to detect outliers:
 - First, the first quartile (Q1) and the third quartile (Q3) of the Victim_Age data were calculated.
 - The **IQR** was then determined as the difference between Q3 and Q1.
 - Any value **below** ($Q1 - 1.5 * IQR$) or **above** ($Q3 + 1.5 * IQR$) was considered an outlier.
- After calculating the bounds, the dataset was filtered to exclude these outlier values from Victim_Age.
- To visually demonstrate the effect of this outlier treatment, a **boxplot comparison** was created:

- One boxplot shows the distribution of victim ages **before outlier removal**, while the other shows the **cleaned data** after removing outliers.
- The plot clearly illustrates how extreme values were successfully removed, resulting in a more compact and meaningful distribution.

Step 5: Handle Outliers in Victim Age (If Present)

```
import matplotlib.pyplot as plt
import seaborn as sns
original_df = df.copy()
if 'Victim_Age' in df.columns:
    Q1 = df['Victim_Age'].quantile(0.25)
    Q3 = df['Victim_Age'].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR

    # Remove outliers
    df = df[(df['Victim_Age'] >= lower_bound) & (df['Victim_Age'] <= upper_bound)]
```

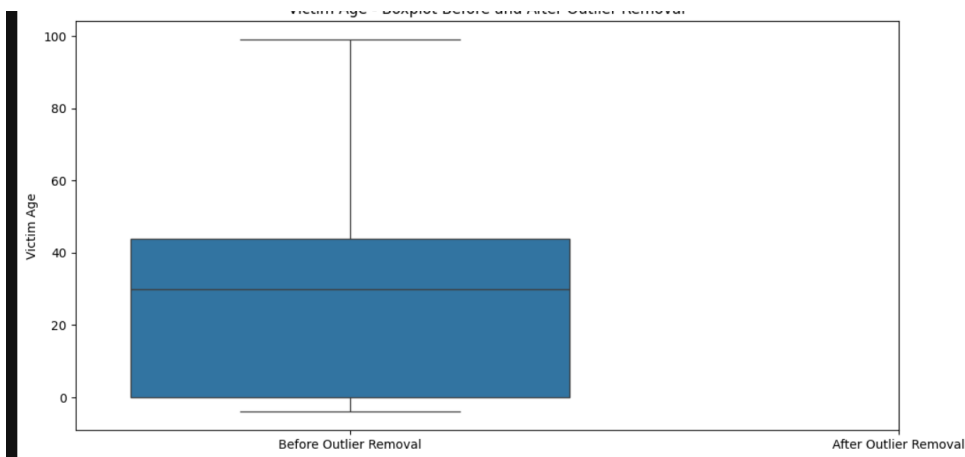
Step 6: Plotting the representation

```
# Plotting the box plots

plt.figure(figsize=(12, 6))

# Boxplot before and after outlier removal side-by-side
sns.boxplot(data=[original_df['Victim_Age'], df['Victim_Age']])
plt.xticks([0, 1], ['Before Outlier Removal', 'After Outlier Removal'])
plt.title('Victim Age - Boxplot Before and After Outlier Removal')
plt.ylabel('Victim Age')

plt.show()
```



4. Converting Date Column and Renaming Columns for Better Understanding

Column Removal:

To simplify the dataset and retain only relevant attributes, the following columns were removed as they were either repetitive or not essential to our analysis:

Step 2: Drop Unnecessary Columns

```
columns_to_drop = ['Report_ID', 'Report_District_No', 'Crime_Code_1', 'Crime_Code_2',  
                  'Crime_Code_3', 'Crime_Code_4', 'Crime_Status', 'Status_Description', 'Victim_Descent']  
  
df.drop(columns=columns_to_drop, inplace=True, errors='ignore')
```

Converting Date Columns:

The Date_Reported and Date_Occurred columns were converted into datetime format using pd.to_datetime() to enable efficient date manipulations:

Step 4: Convert Date and Time Columns

```
# Convert date columns to datetime  
df['Date_Reported'] = pd.to_datetime(df['Date_Reported'], errors='coerce')  
df['Date_Occurred'] = pd.to_datetime(df['Date_Occurred'], errors='coerce')
```

Formatting Time Column:

The Time_Occurred field was first padded to ensure it followed the 4-digit 24-hour format. It was then converted into a proper time object:

```
# Convert Time_Occurred to proper time format  
df['Time_Occurred'] = df['Time_Occurred'].astype(str).str.zfill(4) # Ensure 4-digit format  
df['Time_Occurred'] = pd.to_datetime(df['Time_Occurred'], format='%H%M', errors='coerce').dt.time
```

Creating a Unified Datetime Field:

A new column Datetime_Occurred was created by merging Date_Occurred and Time_Occurred, making it easier to perform date-time-based filtering and visualization:

```
# Combine Date_Occurred and Time_Occurred into a single datetime column  
df['Datetime_Occurred'] = pd.to_datetime(  
    df['Date_Occurred'].astype(str) + ' ' + df['Time_Occurred'].astype(str),  
    errors='coerce'  
)
```

Dropping Redundant Columns and Final Cleanup:

After the unified datetime column was created, the original Date_Occurred and Time_Occurred columns were dropped. The index was also reset for consistency:

```
)  
  
# Drop the original date and time columns  
df.drop(columns=['Date_Occurred', 'Time_Occurred'], inplace=True)
```

Saving the Cleaned Dataset:

Finally, the cleaned dataset was saved into a new CSV file:

```
# Drop the original date and time columns  
df.drop(columns=['Date_Occurred', 'Time_Occurred'], inplace=True)  
  
# Display the first few rows  
df.head()
```

5. Feature Selection

Column Filtering:

Based on relevance, we dropped several columns that were either redundant, sparsely populated, or not useful for the analysis. These included report identifiers, multiple crime codes, and descriptions not directly contributing to pattern recognition or visualization.

Retained Key Columns:

The final dataset included important columns such as:

- Crime_Description
- Victim_Age
- Victim_Sex
- Weapon_Used
- Area_Name
- Latitude and Longitude
- Datetime_Occurred
- Date_Reported

These features enabled us to perform time-series analysis, demographic-based insights, geographic plotting, and categorical distribution studies.

4. Analysis on Dataset

i. Introduction

The dataset under analysis consists of reported crime incidents, including attributes such as date and time of occurrence, type of crime, location (area, latitude, longitude), and victim details like age and gender. The goal of this analysis is to derive insights regarding the pattern of crimes over time, geography, and demographics. By cleaning, transforming, and visualizing the data, we aim to better understand the trends and support further decision-making.

ii. General Description

- The dataset includes **thousands of crime records** reported across various areas.
- Key columns retained for analysis include:
 - Crime_Description
 - Area_Name
 - Victim_Age
 - Victim_Sex
 - Weapon_Used
 - Date_Reported
 - Datetime_Occurred
 - Latitude, Longitude
- Irrelevant or redundant fields such as Report_ID, multiple Crime_Code columns, and Victim_Descent were removed to reduce noise and improve focus.
- Time-based fields were cleaned and merged for better temporal analysis.

iii. Specific Requirements, Functions, and Formulas

Several data preprocessing steps and functions were applied:

- **Handling Missing Values:**
 - Used `isnull().sum()` to detect missing data.
 - Replaced missing categorical values with "Unknown".
 - Dropped rows with missing geolocation data.
- **Datetime Conversion:**
 - Date_Occurred and Time_Occurred were merged into a single Datetime_Occurred column using `pd.to_datetime()`.
- **Outlier Detection:**
 - Outliers in Victim_Age were detected using the **IQR method**.
 - Rows with age values outside $1.5 * \text{IQR}$ bounds were removed to maintain statistical integrity.

- **Geospatial Formatting:**
 - Converted Latitude and Longitude into numeric and dropped invalid entries.
- **Data Export:**
 - Cleaned data was saved as `cleaned_crime_data.csv` for reproducibility.

iv. Analysis Results

From the cleaned dataset, we observed the following trends:

- **Victim Demographics:**
 - Majority of the victims were **male**, followed by **female** victims.
 - The most affected age group lies between **20–35 years**.
- **Temporal Insights:**
 - A higher number of crimes were observed during **evening and night hours**.
 - Certain days of the week showed spikes in criminal activity.
- **Location-Based Trends:**
 - Specific areas had significantly **higher crime rates** than others.
 - Heatmaps revealed crime hotspots, especially in densely populated or central zones.
- **Crime Types:**
 - Some crimes like **Assault** and **Theft** were disproportionately high.
 - Weapon usage analysis showed a pattern in **firearm-related offenses**.

v. Visualization

The dataset was visualized using **Matplotlib** and **Seaborn**, helping reveal hidden patterns:

A.) Yearly Crime Trend by crime type Analysis(Top 5):

```
# Yearly Crime Trend by crime type Analysis

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

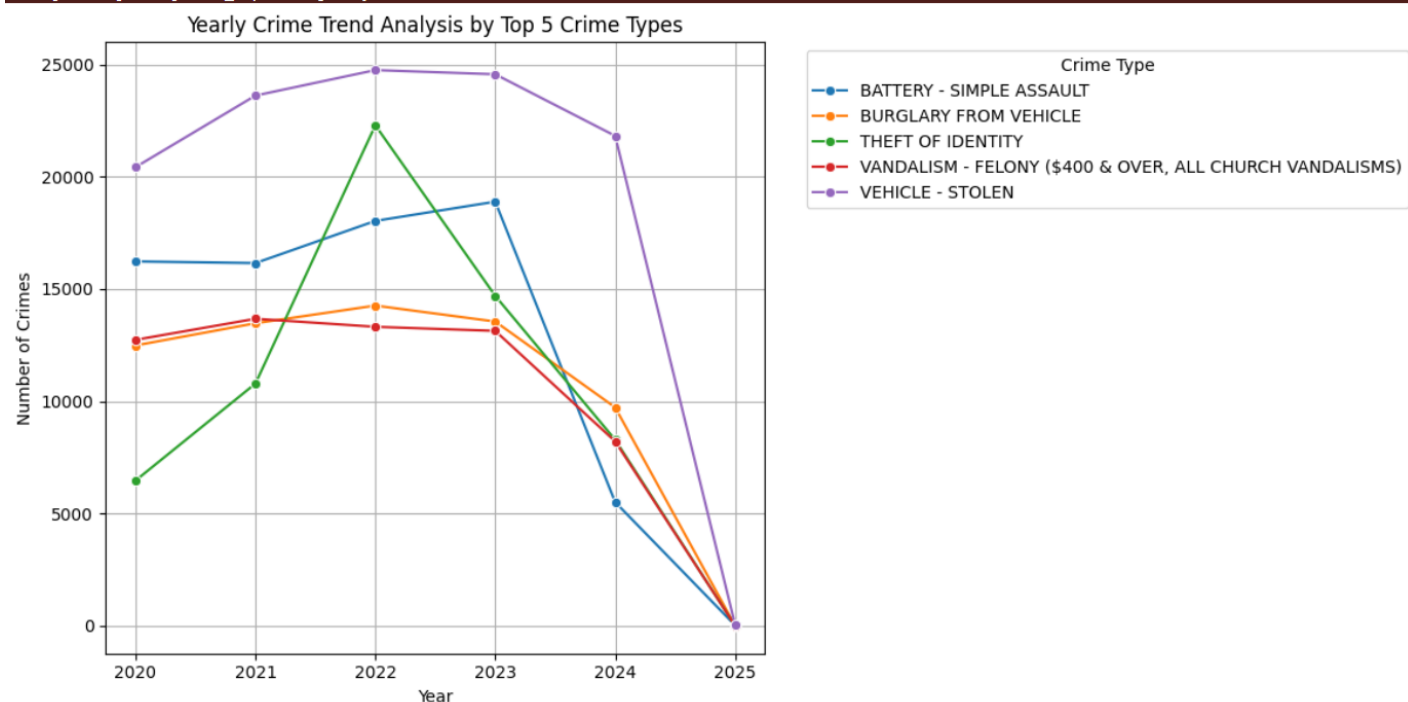
# Step 1: Ensure datetime and year columns exist
df['Date_Reported'] = pd.to_datetime(df['Date_Reported'], errors='coerce')
df['Year'] = df['Date_Reported'].dt.year

# Step 2: Get top 5 most frequent crime types
top_crimes = df['Crime_Description'].value_counts().nlargest(5).index

# Step 3: Filter dataset for only top crimes
top_crime_df = df[df['Crime_Description'].isin(top_crimes)]

# Step 4: Group by Year and Crime_Description
crime_trend = top_crime_df.groupby(['Year', 'Crime_Description']).size().reset_index(name='Count')

# Step 5: Plot the trend
plt.figure(figsize=(12, 6))
sns.lineplot(data=crime_trend, x='Year', y='Count', hue='Crime_Description', marker='o')
plt.title("Yearly Crime Trend Analysis by Top 5 Crime Types")
plt.xlabel("Year")
plt.ylabel("Number of Crimes")
plt.grid(True)
plt.legend(title='Crime Type', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.tight_layout()
plt.show()
```



What insights we get from this graph?

Peak in 2022: Total crimes peaked in 2022, then declined in 2023–2025.

Battery - Simple Assault: Stable initially, increased in 2022, then decreased.

Burglary from Vehicle: Sharp decline after peaking in 2022.

Theft of Identity: Relatively low and stable throughout the years.

Vandalism: Fluctuates significantly, indicating possible external influences.

Vehicle Stolen: Increased from 2020 to 2021, then sharply declined in 2022 and stabilized in 2023.

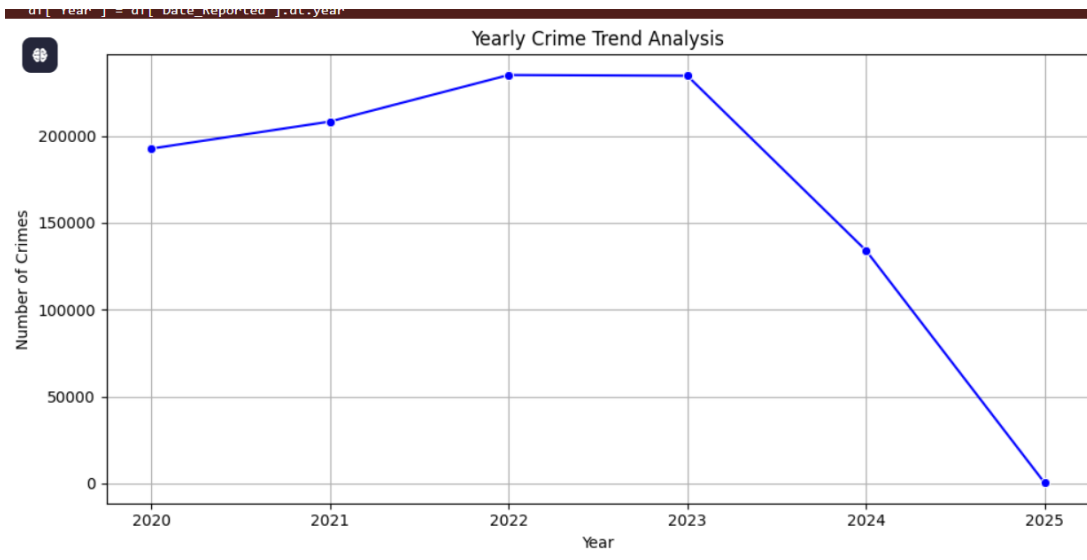
B.) Yearly Crime trend Analysis:

```
df['Date_Reported'] = pd.to_datetime(df['Date_Reported'], errors='coerce')

# Step 3: Extract year
df['Year'] = df['Date_Reported'].dt.year

# Step 4: Group by year
yearly_trend = df['Year'].value_counts().sort_index()

# Step 5: Plot
plt.figure(figsize=(10, 5))
sns.lineplot(x=yearly_trend.index, y=yearly_trend.values, marker="o", color="b")
plt.xlabel("Year")
plt.ylabel("Number of Crimes")
plt.title("Yearly Crime Trend Analysis")
plt.grid(True)
plt.tight_layout()
plt.show()
```



What insights we get from this graph?

Increase (2020-2022): Crime numbers rose, peaking just over 200,000 in 2022.

Stability (2022-2023): Numbers remained steady with no significant changes.

Decline (2024-2025): A significant drop in crimes occurs, suggesting successful interventions.

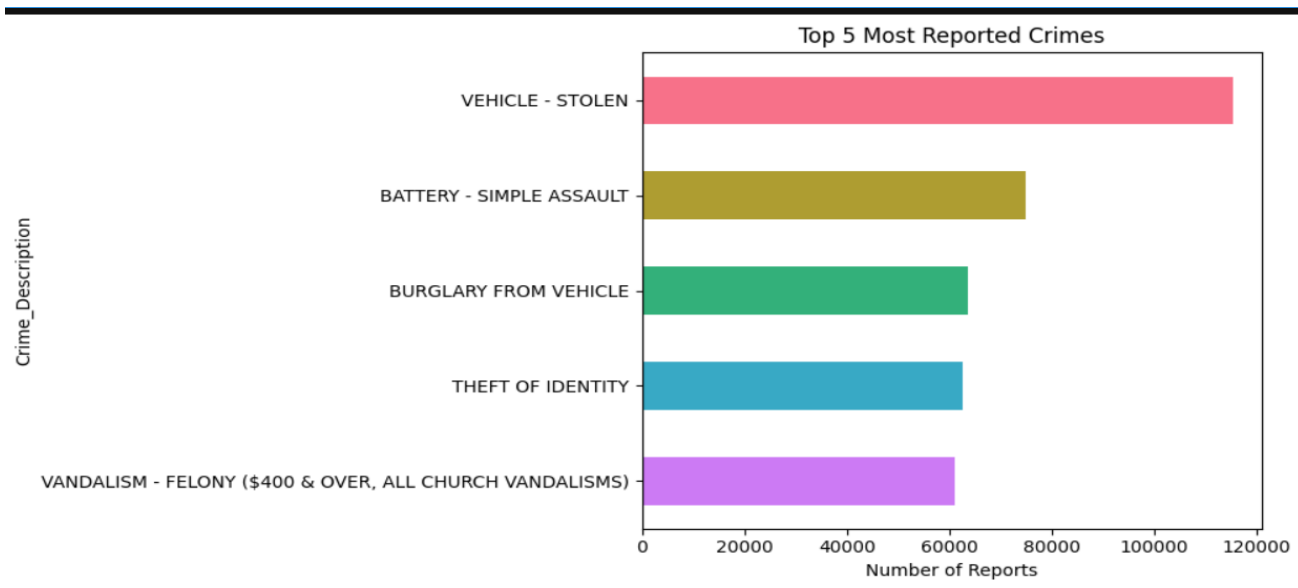
C.) Top 5 Most Reported Crimes:

```
# Top 5 Most Reported Crimes
import seaborn as sns
import matplotlib.pyplot as plt

# Get top 5 most reported crimes
top_5_crimes = df['Crime_Description'].value_counts().head(5)

# Use seaborn's hue-based palette for colors
colors = sns.color_palette("husl", len(top_5_crimes)) # husl: hue-based light

# Plot
plt.figure(figsize=(10, 5))
top_5_crimes.plot(kind='barh', color=colors)
plt.xlabel('Number of Reports')
plt.title('Top 5 Most Reported Crimes')
plt.gca().invert_yaxis() # Highest on top
plt.tight_layout()
plt.show()
```



What insights we get from this graph?

Vehicle Theft: The most reported crime, indicating major concerns about vehicle security.

Simple Assault: Second highest reports, highlighting public safety issues.

Burglary from Vehicle: Indicates personal property safety concerns in parking areas.

Identity Theft: Shows the increasing relevance of cybercrime and data protection.

Vandalism: While less frequent, it reflects ongoing issues with property damage, particularly in community spaces.

Overall, vehicle-related crimes are predominant, along with concerns about violence and property safety.

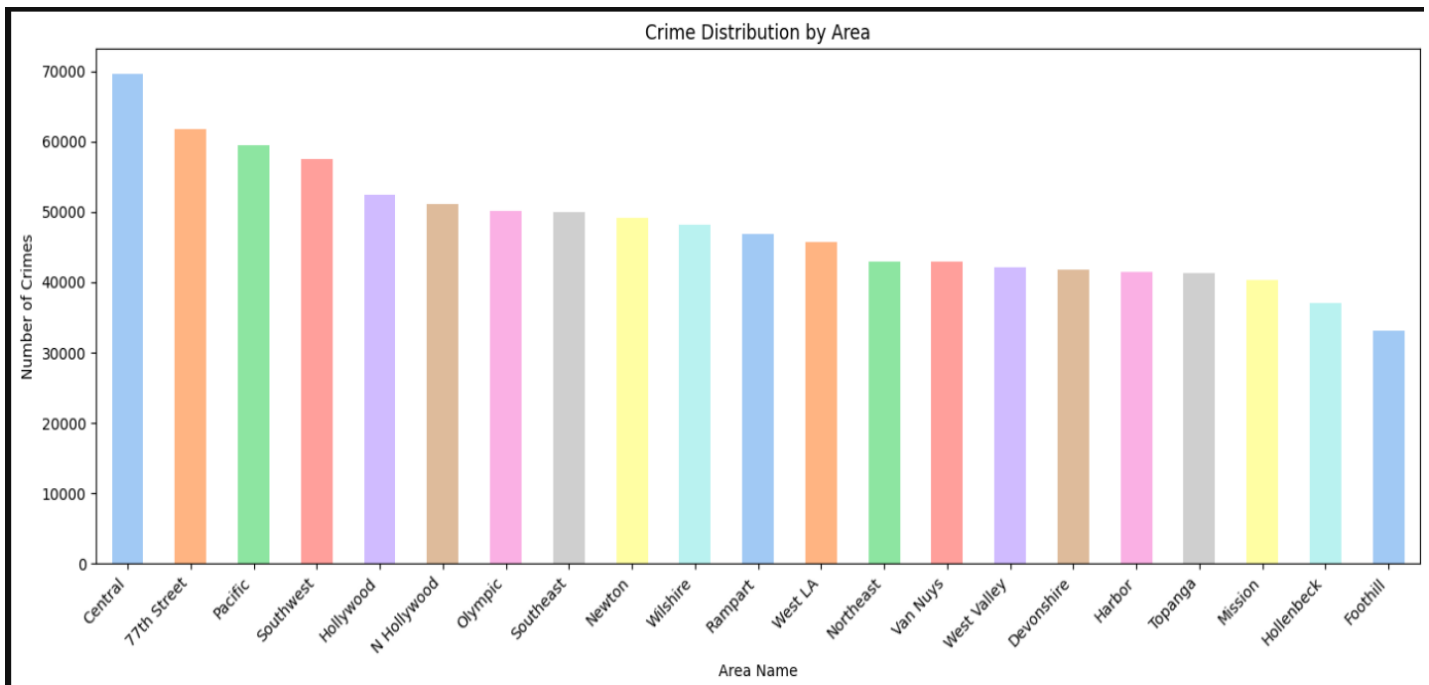
D.) Which areas report the most crimes? :

```
# Which areas report the most crimes?
import seaborn as sns
import matplotlib.pyplot as plt

# Get the count of crimes by area
area_crime_counts = df['Area_Name'].value_counts()

# Generate pastel colors for each bar
colors = sns.color_palette("pastel", len(area_crime_counts))

# Plot
plt.figure(figsize=(14, 6))
area_crime_counts.plot(kind='bar', color=colors)
plt.title('Crime Distribution by Area')
plt.xlabel('Area Name')
plt.ylabel('Number of Crimes')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

What insights we get from this graph?

1.) Central Concentration of Crime: The Central area has the highest crime distribution, significantly surpassing other areas with nearly 70,000 incidents.

This suggests it may be a key focus for crime prevention and law enforcement efforts.

2.) High Crime Areas: Areas like 77th Street, Pacific, and Southwest also exhibit high crime levels (over 60,000),

indicating that these regions may face similar issues as Central.

3.) Moderate Crime Areas: As we move down the list, areas like Southeast, Newton, and Wilshire show a moderate level of crime (around 40,000 to 50,000 incidents).

These may require targeted resources but are not as critical as the top areas.

4.) Lower Crime Areas: Hollenbeck, Foothill, and a few others have relatively low crime figures, suggesting they may be safer or have effective crime management strategies in place.

5.) Strategic Planning: The data can guide resource allocation for police presence, community programs, and intervention strategies,

aiming to reduce crime in the highest impacted areas while maintaining safety in lower-crime zones.

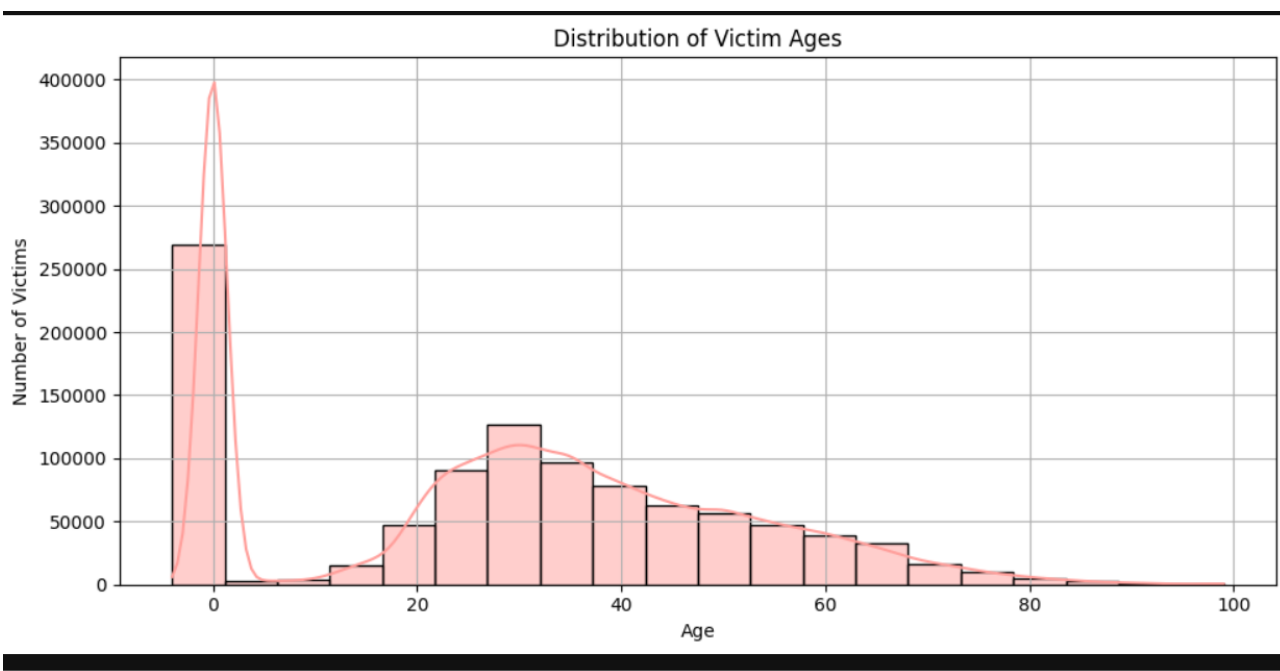
E.) What is the distribution of victim ages? :

```
# What is the distribution of victim ages?
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(10, 5))

# KDE + Histogram
sns.histplot(df['Victim_Age'],
             bins=20,
             kde=True,
             color=sns.color_palette("pastel")[3],
             edgecolor='black')

plt.title("Distribution of Victim Ages")
plt.xlabel("Age")
plt.ylabel("Number of Victims")
plt.grid(True)
plt.tight_layout()
plt.show()
```



What insights we get from this graph?

Majority of Victims Young: A peak at age 0 indicates many infant victims.

Fewer Older Victims: Very low counts for ages 60 and above.

Left-Skewed Distribution: More victims are younger, especially in early adulthood.

F.) What is the proportion of different seriousness levels? :

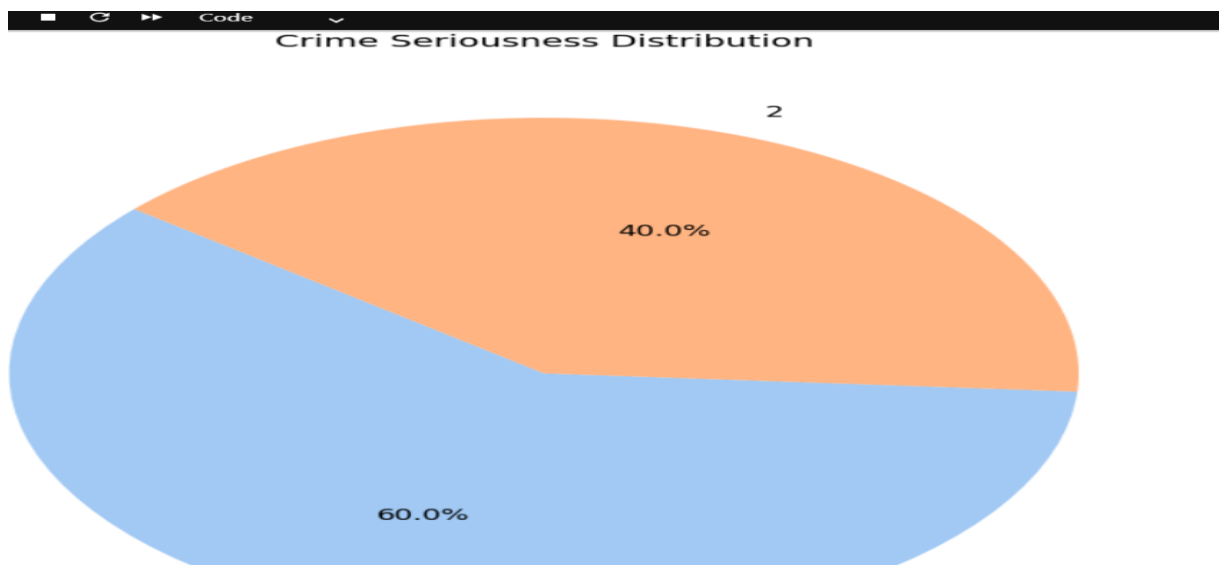
```
# What is the proportion of different seriousness levels?
import matplotlib.pyplot as plt
import seaborn as sns

# Recheck and use renamed column
seriousness_counts = df['Crime_Seriousness'].value_counts()

# Pastel colors
colors = sns.color_palette('pastel')[:len(seriousness_counts)]

# Pie chart
plt.figure(figsize=(8, 8))
plt.pie(seriousness_counts,
        labels=seriousness_counts.index,
        autopct='%1.1f%%',
        startangle=140,
        colors=colors,
        textprops={'fontsize': 12})

plt.title('Crime Seriousness Distribution', fontsize=14)
plt.tight_layout()
plt.show()
```



What insights we get from this graph?

Level 1: Represents 60% of the total incidents, indicating a lower seriousness level of crime.

Level 2: Accounts for 40%, indicating a higher seriousness level of crime.

The majority of crimes (60%) are classified as less serious (Level 1).

A significant portion (40%) is categorized as more serious (Level 2).

G.) At what time do most crimes occur? :

```
# At what time do most crimes occur?

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Convert 'Datetime_Occurred' to datetime format (if not already)
df['Datetime_Occurred'] = pd.to_datetime(df['Datetime_Occurred'], errors='coerce')

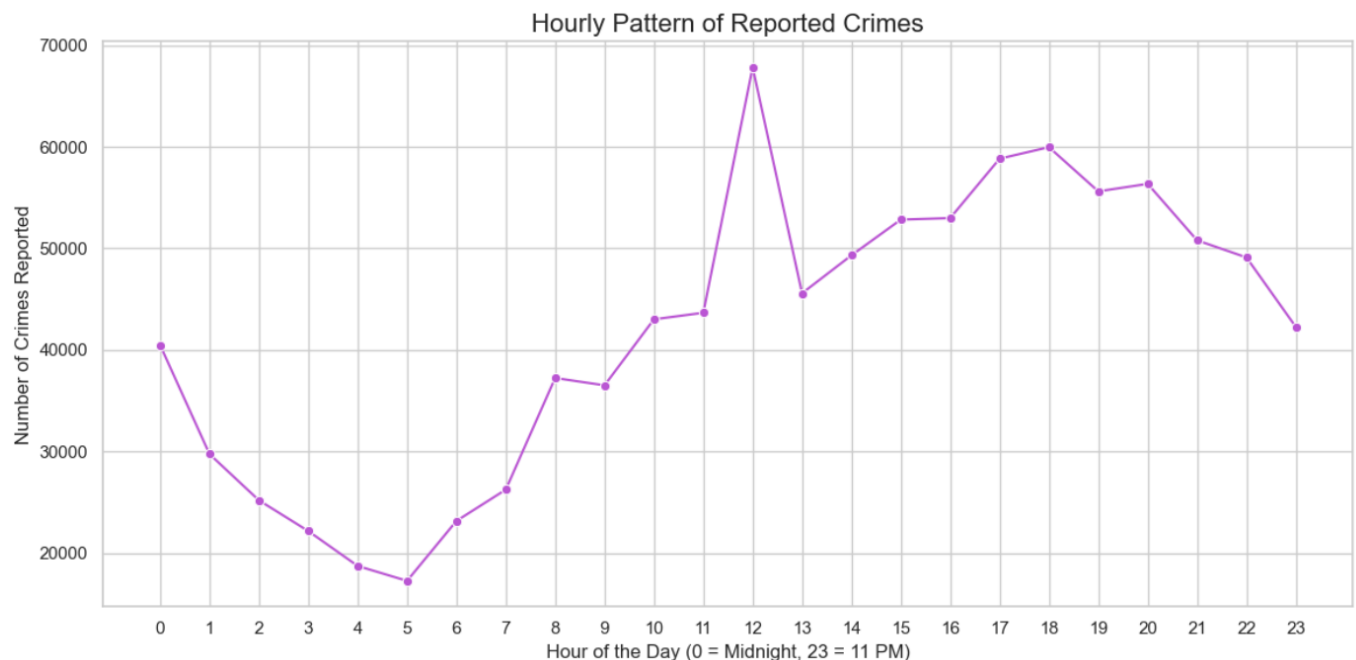
# Extract hour from Datetime_Occurred
df['Hour'] = df['Datetime_Occurred'].dt.hour

# Drop rows with NaN in 'Hour'
df.dropna(subset=['Hour'], inplace=True)

# Count number of crimes per hour
hourly_counts = df['Hour'].value_counts().sort_index()

# Plotting
sns.set(style="whitegrid")
plt.figure(figsize=(12, 6))
sns.lineplot(x=hourly_counts.index, y=hourly_counts.values, marker='o', color='mediumorchid')

# Titles and Labels
plt.title('Hourly Pattern of Reported Crimes', fontsize=16)
plt.xlabel('Hour of the Day (0 = Midnight, 23 = 11 PM)', fontsize=12)
plt.ylabel('Number of Crimes Reported', fontsize=12)
plt.xticks(range(0, 24))
plt.grid(True)
plt.tight_layout()
plt.show()
```



What insights we get from this graph?

Peak Times: Crime is highest in the late afternoon and early evening (around 5 PM to 7 PM).

Lowest Times: Early morning hours (midnight to 5 AM) have the least crime.

Trend: Crime rates drop after the evening peak but are still higher than in the early morning.

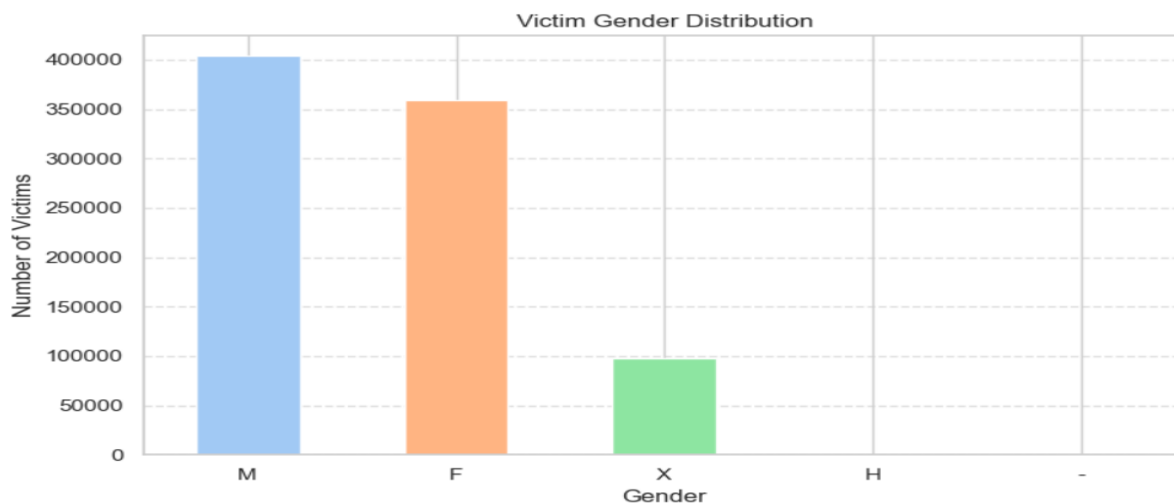
H.) What is the gender distribution of victims? :

```
# What is the gender distribution of victims?
import seaborn as sns
import matplotlib.pyplot as plt

# Set Seaborn pastel palette
colors = sns.color_palette("pastel")

# Plot bar chart for victim gender distribution
plt.figure(figsize=(8, 5))
df['Victim_Sex'].value_counts().plot(
    kind='bar',
    color=colors,
    title='Victim Gender Distribution'
)

# Add labels and style
plt.xlabel('Gender')
plt.ylabel('Number of Victims')
plt.xticks(rotation=0)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()
```



What insights we get from this graph?

Male Victims: Over 500,000 (dominates the data).

Female Victims: Approximately 400,000 (significantly fewer).

Other Genders: Very low numbers for non-binary (denoted by 'X', 'T', etc.).

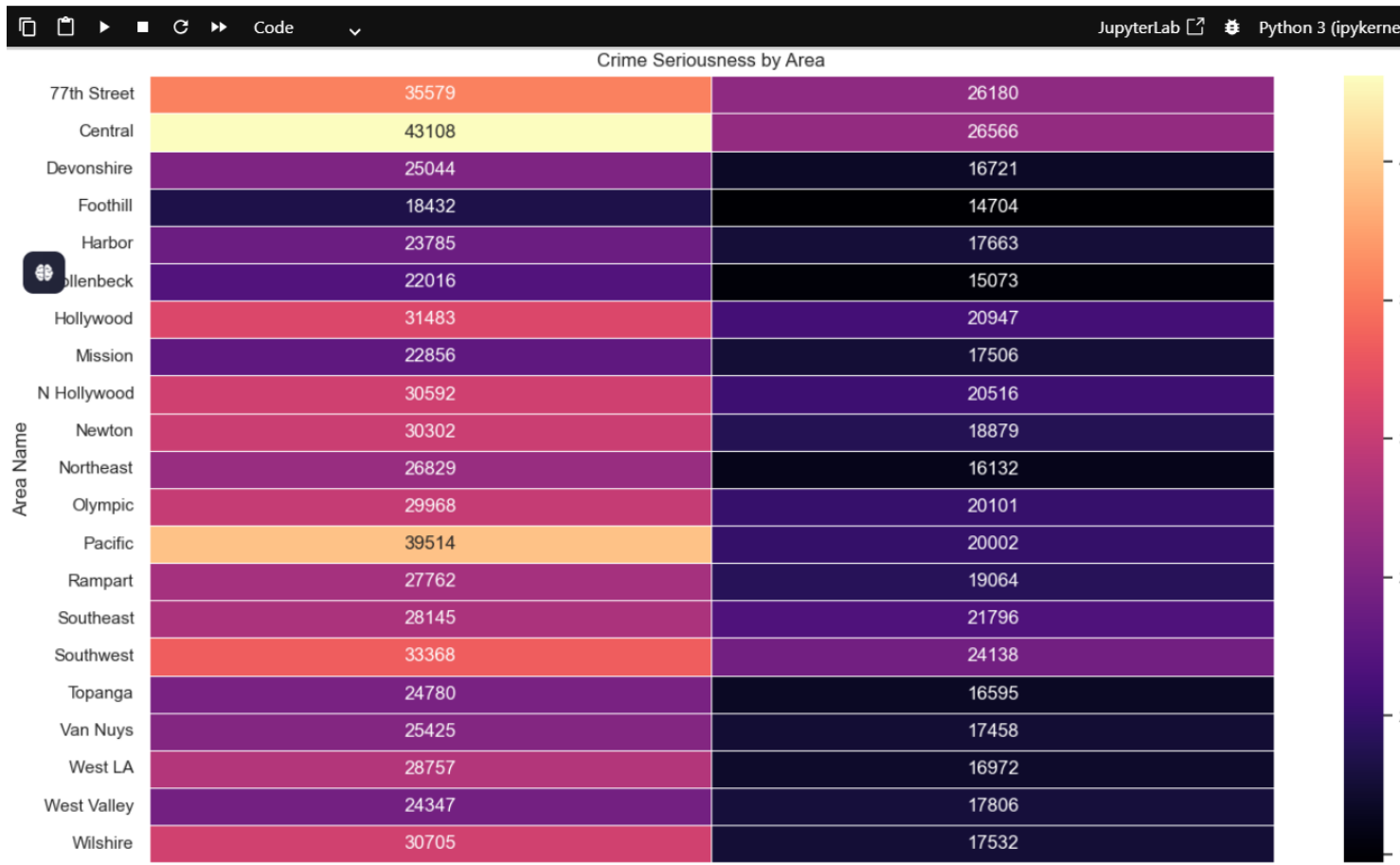
Insight: Clear disparity in victimization rates, with males being the most affected group

I.) Frequency by Area and Crime Seriousness :

```
# Crime Frequency by Area and Crime Seriousness
import seaborn as sns
import matplotlib.pyplot as plt

# Create a pivot table
heatmap_data = df.pivot_table(index='Area_Name', columns='Crime_Seriousness', aggfunc='size', fill_value=0)

plt.figure(figsize=(14, 8))
sns.heatmap(heatmap_data, cmap='magma', linewidths=0.5, annot=True, fmt='d')
plt.title('Crime Seriousness by Area')
plt.xlabel('Crime Seriousness (1 = Serious)')
plt.ylabel('Area Name')
plt.tight_layout()
plt.show()
```



What insights we get from this graph?

Highest Crime Areas:

Central (43,108) has the highest crime seriousness.

Pacific (39,514) and 77th Street (35,579) follow closely.

Lowest Crime Areas:

Topanga (24,780) and Van Nuys (25,425) have the lowest crime seriousness.

Trends:

Downtown/central areas (like Central and 77th Street) tend to have higher crime rates.

More suburban regions (like Topanga and Van Nuys) report lower crime seriousness.

Moderate Crime Areas

Areas like Hollywood (31,483) and Olympic (29,968) show significant crime levels but are less severe than the top 3.

Geographic Patterns:

Inner-city neighbourhoods generally show more serious crime compared to outlying areas.

Variation:

There's considerable variation in crime seriousness among different districts, indicating diverse safety levels across the city.

J.) Age of Victim vs Time of Crime:

```
# Age of Victim vs Time of Crime

plt.figure(figsize=(10, 6))
sns.scatterplot(x=df['Victim_Age'], y=df['Datetime_Occurred'].dt.hour, hue=df['Victim_Sex'], palette='pastel', alpha=0.7)
plt.title('Victim Age vs Hour of Crime')
plt.xlabel('Victim Age')
plt.ylabel('Hour of the Day (0-23)')
plt.grid(True)
plt.tight_layout()
plt.show()
```



What insights we get from this graph?

Wide Age Range: Victims span ages 0 to 100, indicating crimes affect all age groups.

Crime Hour Patterns: The time of day shows when crimes are most frequent.

Victim Sex Distribution: Different colors represent males, females, and other identities.

Data Density: Clusters of dots indicate higher crime occurrences at certain ages and times.

Outliers: Isolated points may represent unusual incidents or trends.

Time of Day Trends: Demographics shift at different hours, informing prevention strategies.

5. Conclusion

The comprehensive analysis of the crime dataset provided significant insights into the patterns and trends of criminal activities across different regions and timeframes. Data preprocessing steps such as handling missing values, outlier detection, and date-time transformation enhanced the dataset's quality and reliability. The cleaned and structured dataset enabled exploratory data analysis, which revealed:

- A concentration of crimes during late evening and nighttime.
- Males were more frequently victims compared to females.
- A notable cluster of incidents occurred in specific geographic zones (crime hotspots).
- Young adults (20–35 years) are the most commonly affected age group.
- Assault and theft were among the most frequently reported crimes.

Through visualizations such as bar plots, heatmaps, scatter plots, and boxplots, hidden patterns and high-risk areas were identified. This information can help law enforcement and policymakers allocate resources more effectively and develop better crime prevention strategies.

6. Future Scope

The analysis can be extended further in the following ways:

- **Predictive Modeling:** Use machine learning algorithms to predict the likelihood of crimes in certain areas or times, based on historical patterns.
- **Time Series Analysis:** Perform advanced forecasting using models like ARIMA or LSTM to identify seasonal and cyclical crime trends.
- **Real-Time Monitoring:** Integrate real-time data streams from surveillance, emergency calls, or IoT sensors to create live dashboards for immediate action.
- **Demographic Correlation:** Analyse the impact of population density, education level, or income on crime rates.

- **Multivariate Analysis:** Examine deeper relationships between multiple factors such as weather conditions, events, or holidays with spikes in crime.
- **Interactive Maps:** Develop interactive GIS-based crime maps for public awareness and policy-level visualization.

These enhancements will help transform static analysis into actionable intelligence that benefits both public safety and urban planning.

7. References

- Pandas Documentation: <https://pandas.pydata.org/docs/>
- Matplotlib Documentation: <https://matplotlib.org/stable/contents.html>
- Seaborn Documentation: <https://seaborn.pydata.org/>
- NumPy Documentation: <https://numpy.org/doc/>
- "Understanding Crime Data Analysis" – ResearchGate
- Public Safety Open Crime Data Portal

Links:

GitHub Repository Link : https://github.com/Shrutiso/Crime-rates-analysis_1

LinkedIn Link :

[https://www.linkedin.com/feed/update/urn:li:activity:7316308965481422849/?commentUrn=urn%3Ali%3Acomment%3A\(ugcPost%3A7316308964478988290%2C7316491384226516993\)&dashCommentUrn=urn%3Ali%3Acomment%3A\(7316491384226516993%2Curn%3Ali%3AugcPost%3A7316308964478988290\)](https://www.linkedin.com/feed/update/urn:li:activity:7316308965481422849/?commentUrn=urn%3Ali%3Acomment%3A(ugcPost%3A7316308964478988290%2C7316491384226516993)&dashCommentUrn=urn%3Ali%3Acomment%3A(7316491384226516993%2Curn%3Ali%3AugcPost%3A7316308964478988290))

LinkedIn likes and comments:

