Crime Rate Prediction Using K-Means

Under the Guidance : Mrs.B.Annapoorna
Batch number : 05
Student Details :

1. **Ch.Sreenith**
2. **Ch.Suryakamal**
3. **D.Prathyusha**
4. **N.Vamshi krishna**

ABSTRACT

- Crime is an alarming aspect of our society, and its prevention is a vital task. Crime analysis is a well-organized way of detecting and examining patterns and trends in crime. It is of utmost importance to study reasons, consider different factors and determine the relationship among various crimes occurring and discover the best suitable methods to control crime.
- The primary objective of this project is to distinguish various crimes using clustering techniques based on the occurrences and regularity. Data mining is used for analysis, investigation and check patterns in crimes. In this project, a clustering approach is used to analyze the crime data; the stored data is clustered usi+ng the K-Means algorithm.

INTRODUCTION

- K-means algorithm is a clustering algorithm used to group data points into clusters based on their similarity. In the context of crime rate prediction, K-means algorithm can be used to identify patterns and clusters of criminal activities in a given area or region.
- To use K-means algorithm for crime rate prediction, we first need to collect data on criminal activities such as theft, burglary, assault, etc. in a particular area. The data should include the location, date, and time of the criminal activity.
- Next, we need to preprocess the data by removing any outliers and normalizing the data to ensure that all variables are on the same scale. We then apply the K-means algorithm to the preprocessed data to identify clusters of criminal activities.

EXISTING SYSTEM

- It is only within the last few decades that the technology made spatial data mining a practical solution for broad audiences of Law enforcement officials which is affordable and available. Even today all our data is available in the form of paper-based format. We can digitise this great data record for creating a criminal record database. So the primary challenge in front of us is developing a better, efficient crime pattern detection tool to identify crime patterns effectively.

- In countries like England, Cambridge Police Department have done a similar one named Series Finder for finding the patterns in a burglary. For achieving this, they used the modus operandi of an offender, and they extracted some crime patterns which were followed by the offender. The predicted result showed more than 80% accuracy. We are applying a similar concept in this study is to examine the use of clustering technology.

PROPOSED SYSTEM

- The precision of the program depends on the correctness of the training set. Finding the patterns and trends in crime is a challenging factor. To identify a pattern, crime analysts takes much time, scanning through data to find whether a particular crime fits into a known pattern. If it does not fit into an existing pattern, then the data must be classified as a new pattern.
- After detecting a pattern, it can be used to predict, anticipate and prevent crime before this clustering algorithms have been used for crime analysis. For instance, one site it is revealed that suspect has black hair and from next site/witness, it is revealed that suspect is youth and from third one reveals that the offender has a tattoo on his left arm. Through the offender details, we can obtain a basic picture of various crime incidents. These days most of the work is manually done with the assistance of different reports which detectives obtain from data analysts and old crime logs.

IMPLEMENTATION

Data Collection:

- Gather historical crime data for the chosen geographic area.
- Acquire additional relevant data such as demographic, economic, and geographic information.
- Ensure data quality and consistency.

Input Dataset Preprocessing:

- Perform initial data cleaning
- Handle missing values, duplicates, and outliers.
- Standardize units and formats across datasets.
- Merge and integrate different datasets if necessary.

Classification:

- Based on your project's objectives, classify crime data into categories. For example:
- Low, medium, and high crime rate areas.
- Different types of crime clusters (e.g., property crimes, violent crimes).

Clustering:

- Choose the number of clusters (K) based on your classification results or domain knowledge.
- Apply K-Means clustering to group similar areas based on features:
- Use location coordinates (latitude and longitude) and engineered features.
- Include crime-related features like crime type frequencies.
- Assign each area to a cluster based on K-Means results

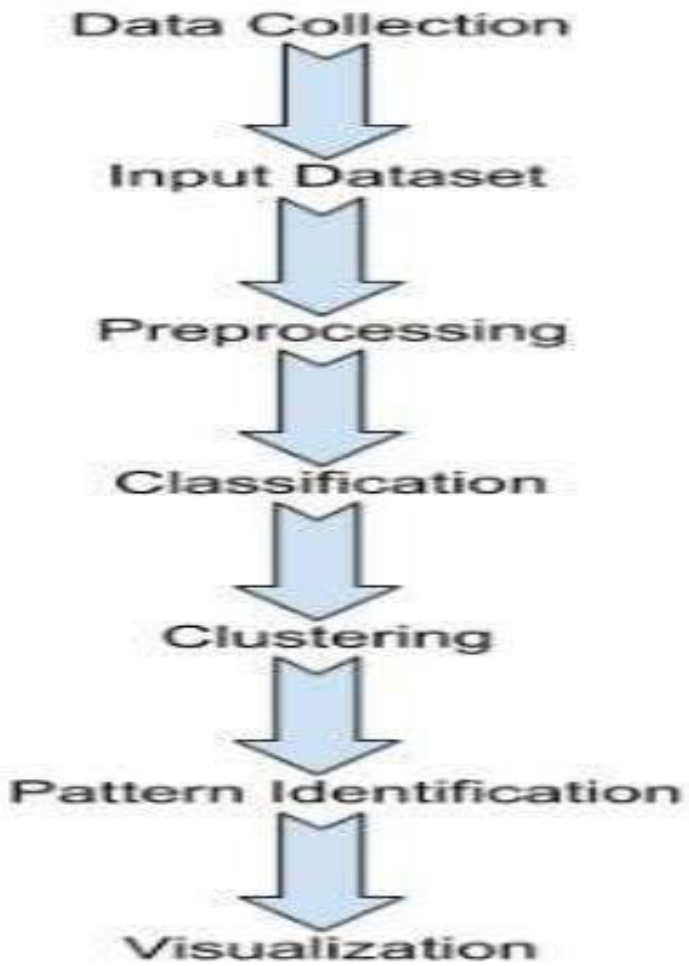
Pattern Identification:

- Determine which areas have high or low crime rates within each cluster.
- Identify common features or attributes that define each cluster.
- Assess how well the clusters align with your initial classification.

Visualization:

- Create visualizations to help stakeholders understand the results:
- Generate maps with colored clusters to show areas with similar crime characteristics.
- Create bar charts or heatmaps to illustrate how crime rates vary within and between clusters.
- Plot cluster centroids to visualize the central tendencies of each cluster.

FLOWCHART


Data Collection
Input Dataset
Preprocessing
Classification
Clustering
Pattern Identification
Visualization

CODE

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
plt.style.use('seaborn')


df = pd.read_csv("combo_crime_data.csv")
df.dropna(); ## drop rows with missing data
# convert dates to pandas datetime format
#df.Date = pd.to_datetime(df.Date, format='%m/%d/%Y %I:%M:%S %p')
df.index = pd.DatetimeIndex(df.Date)


x = df.sample(30000) ##sampling a part of the dataset
```

```
x['Primary_Type'].value_counts().plot.bar()
plt.title("Crimes")
plt.show()

x_theft = x[x['Primary_Type'] == "THEFT"]
x_battery = x[x['Primary_Type'] == "BATTERY"]
x_cd = x[x['Primary_Type'] == "CRIMINAL DAMAGE"]
x_narc = x[(x['Primary_Type'] == "NARCOTICS")]

x_theft['Description'].value_counts(normalize=True).plot.bar()
plt.title("Theft Types")
plt.show()

x_battery['Description'].value_counts(normalize=True).plot.bar()
plt.title("Battery Types")
plt.show()

x_cd['Description'].value_counts(normalize=True).plot.bar()
plt.title("Criminal Damage Types")
plt.show()
```

```
x_narc['Description'].value_counts(normalize=True).plot.bar()
plt.title("Narcotics Types")
plt.show()

x['Arrest'].value_counts(normalize=True).plot.bar()
plt.title("Arrests")
plt.show()

x[x.Arrest == True]['Primary_Type'].value_counts(normalize=True).plot.bar()
plt.title("Arrests per crime type")
plt.show()


color = (0.2, 0.4, 0.6, 0.6)
x.groupby([x.index.month]).size().plot(kind='barh', color=color)
plt.ylabel('Month')
plt.xlabel('Number of crimes')
plt.title('Number of crimes by month of year')
plt.show()
```

CODE EXPLANATION

Importing Libraries:

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns
```

“Here, the code imports the necessary libraries for data manipulation (Pandas and NumPy) and data visualization (Matplotlib and Seaborn). Seaborn is used for setting the plotting style”.

Loading Data:

```
df = pd.read_csv("combo_crime_data.csv")
```

“The code reads a CSV file named "combo_crime_data.csv" into a Pandas DataFrame called df. This DataFrame likely contains crime-related data”.

Dropping Missing Data:

df.dropna() ## drop rows with missing data

“This line of code attempts to drop rows with missing data (NaN values) from the DataFrame. However, it's not entirely correct because it doesn't actually modify the DataFrame in place. You should assign the result of the drop operation back to df if you want to remove rows with missing data”.

Converting Dates:

convert dates to pandas datetime format

df.Date = pd.to_datetime(df.Date, format='%m/%d/%Y %I:%M:%S %p')

“This line is commented out, but it suggests an attempt to convert a column named 'Date' to Pandas datetime format. The format specified is commented out as well”.

Setting Datetime Index:

df.index = pd.DatetimeIndex(df.Date)

“This code sets the index of the DataFrame to be a DatetimeIndex based on the 'Date' column, assuming the 'Date' column contains timestamps”.

Data Sampling:

x = df.sample(30000) ##sampling a part of the dataset

“A random sample of 30,000 rows is taken from the DataFrame df and stored in a new DataFrame called ‘x’”.

Plotting Crime Types:

```
x['Primary_Type'].value_counts().plot.bar()
plt.title("Crimes")
plt.show()
```

“This code plots a bar chart showing the frequency of different crime types in the 'Primary_Type' column of the DataFrame x. It also sets a title and displays the plot using Matplotlib”.

Subsetting Data by Crime Type:

```
x_theft = x[x['Primary_Type'] == "THEFT"]
x_battery = x[x['Primary_Type'] == "BATTERY"]
x_cd = x[x['Primary_Type'] == "CRIMINAL DAMAGE"]
x_narc = x[(x['Primary_Type'] == "NARCOTICS")]
```

“This code subsets the DataFrame x into separate DataFrames based on specific crime types”.

Plotting Crime Types for Subsets:

The following code chunks plot the normalized counts of crime descriptions for each crime type subset (e.g., 'Theft Types', 'Battery Types', etc.) and display them using Matplotlib.

Plotting Arrests:

```
x['Arrest'].value_counts(normalize=True).plot.bar()
plt.title("Arrests")
plt.show()
```

“This code plots the normalized counts of arrests in the DataFrame ‘x’ and displays the results in a bar chart”.

Plotting Arrests by Crime Type:

```
x[x.Arrest == True]['Primary_Type'].value_counts(normalize=True).plot.bar()
plt.title("Arrests per crime type")
plt.show()
```

“This code plots the normalized counts of arrests for each crime type and displays the results in a bar chart”.

Plotting Number of Crimes by Month:

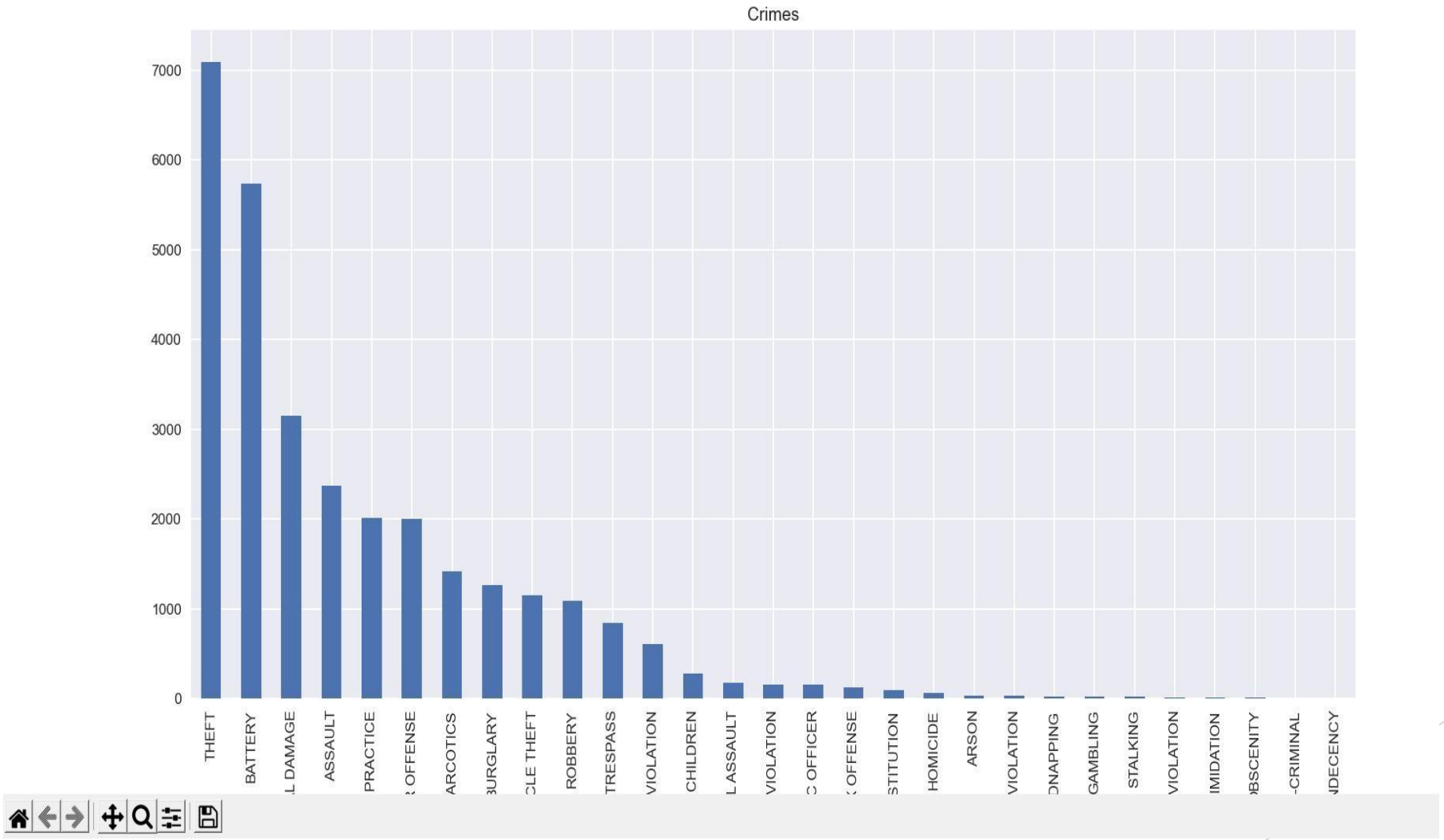
```
color = (0.2, 0.4, 0.6, 0.6)
x.groupby([x.index.month]).size().plot(kind='barh', color=color)
plt.ylabel('Month')
plt.xlabel('Number of crimes')
plt.title('Number of crimes by month of year')
plt.show()
```

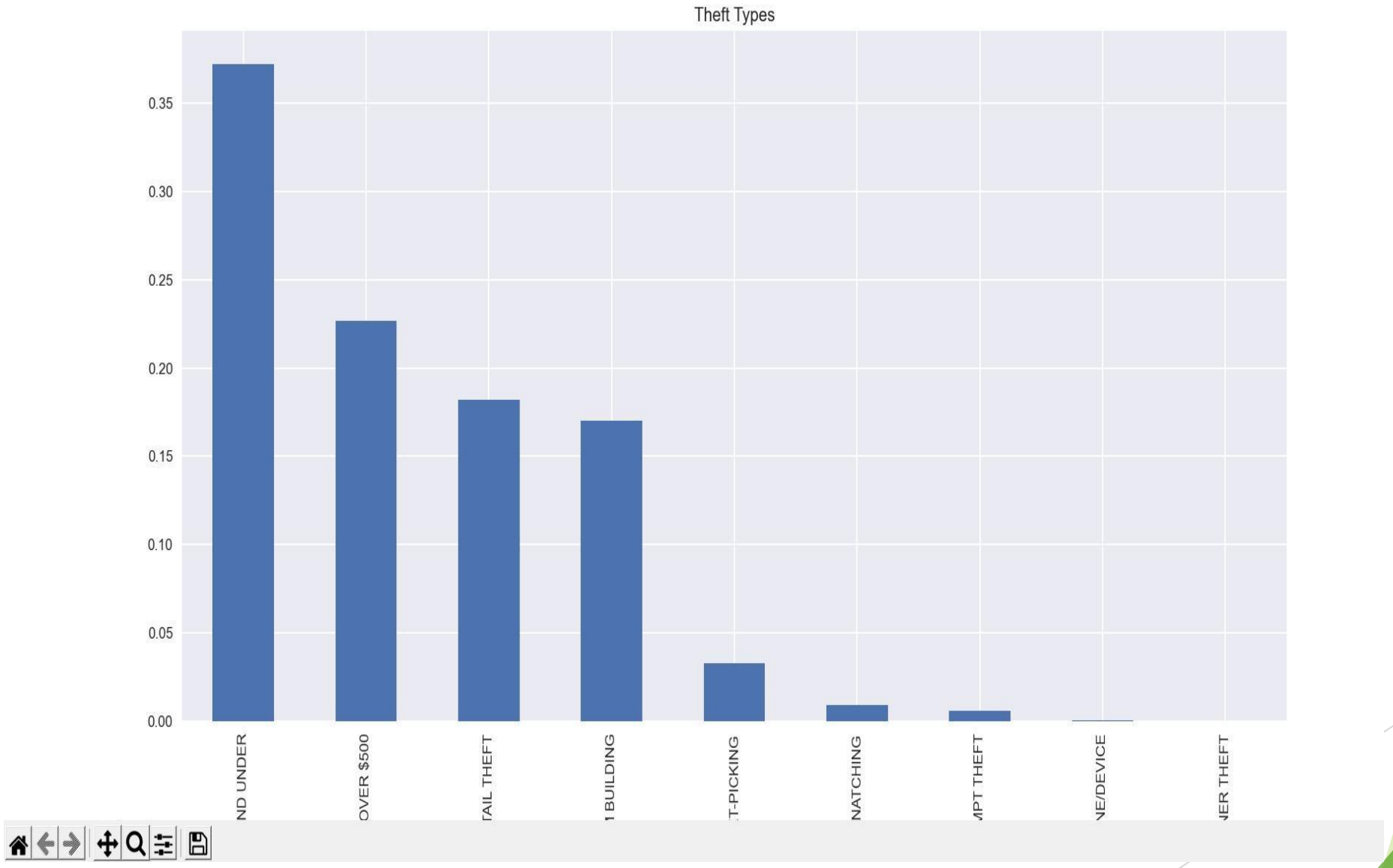
“This code groups the data by the month of the year, calculates the size of each group (number of crimes), and plots a horizontal bar chart to visualize the number of crimes by month. It also sets labels and a title for the plot”.

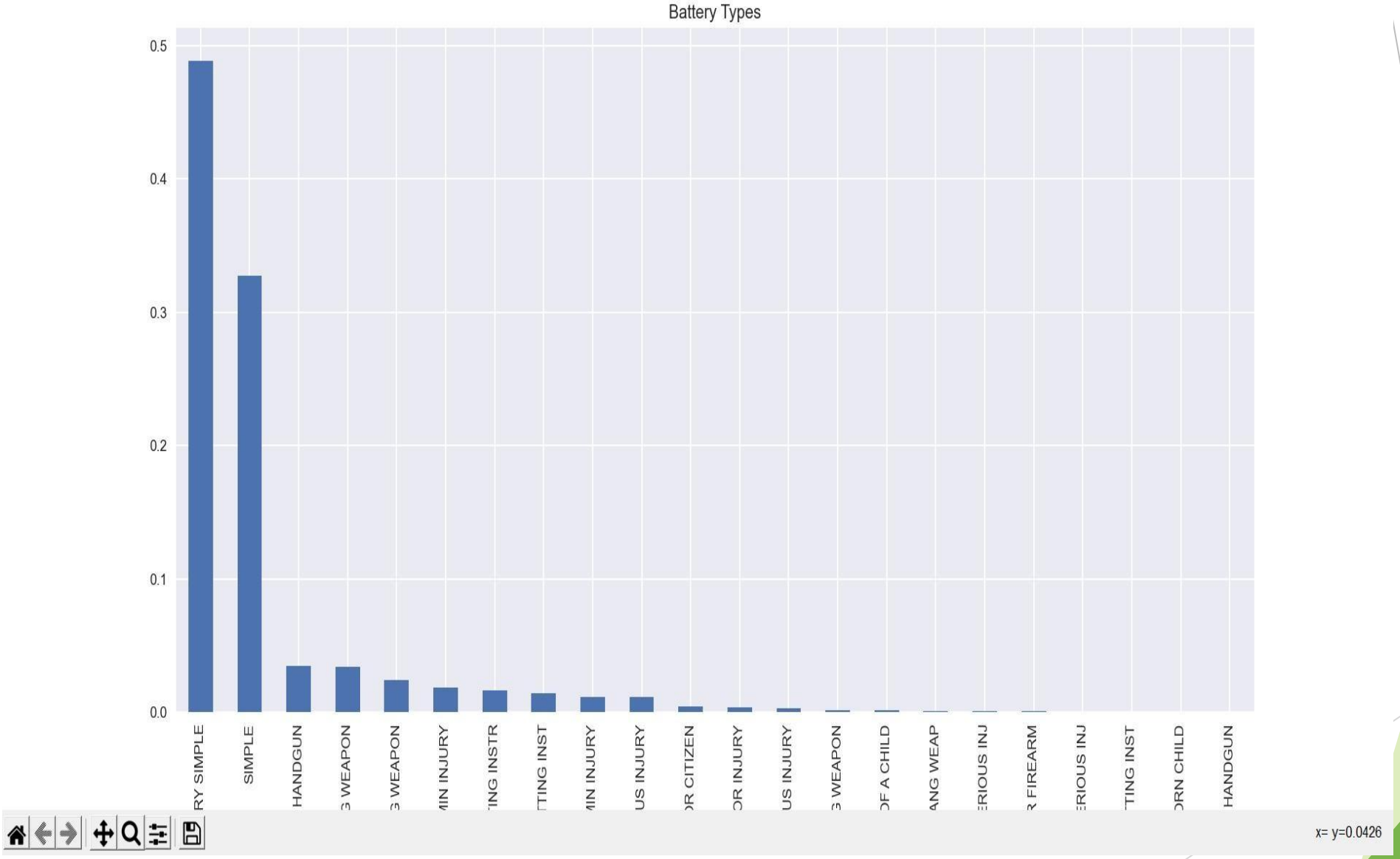
EXECUTION STEPS

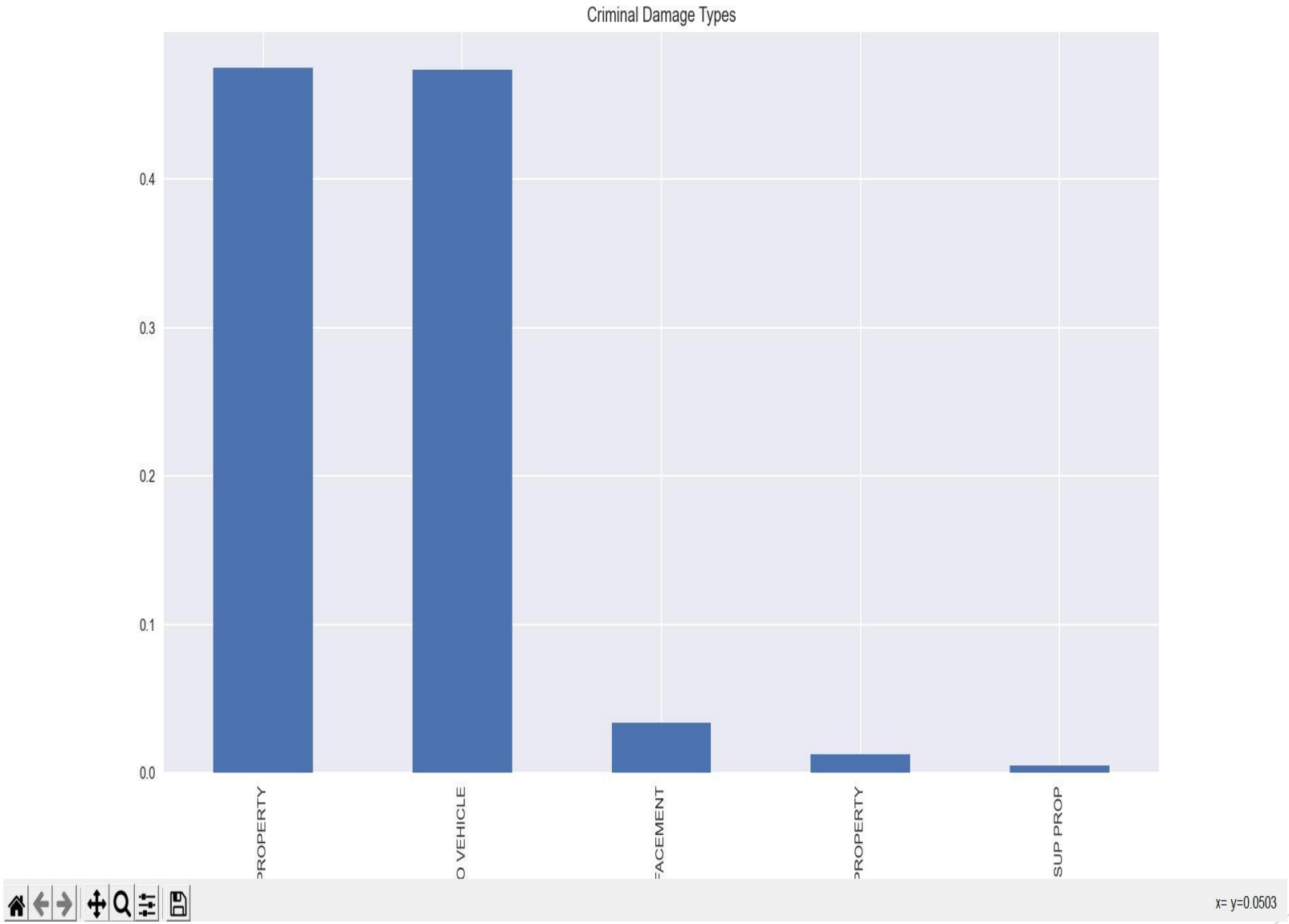
- Open Python IDLE(Version 3.7.1)
- Open the project folder(Mini project) in Python IDLE.
- Select the file(crime analysis.py).
- Click on Run-Run Module.
- The output of the Project displayed in Graphical Representation.

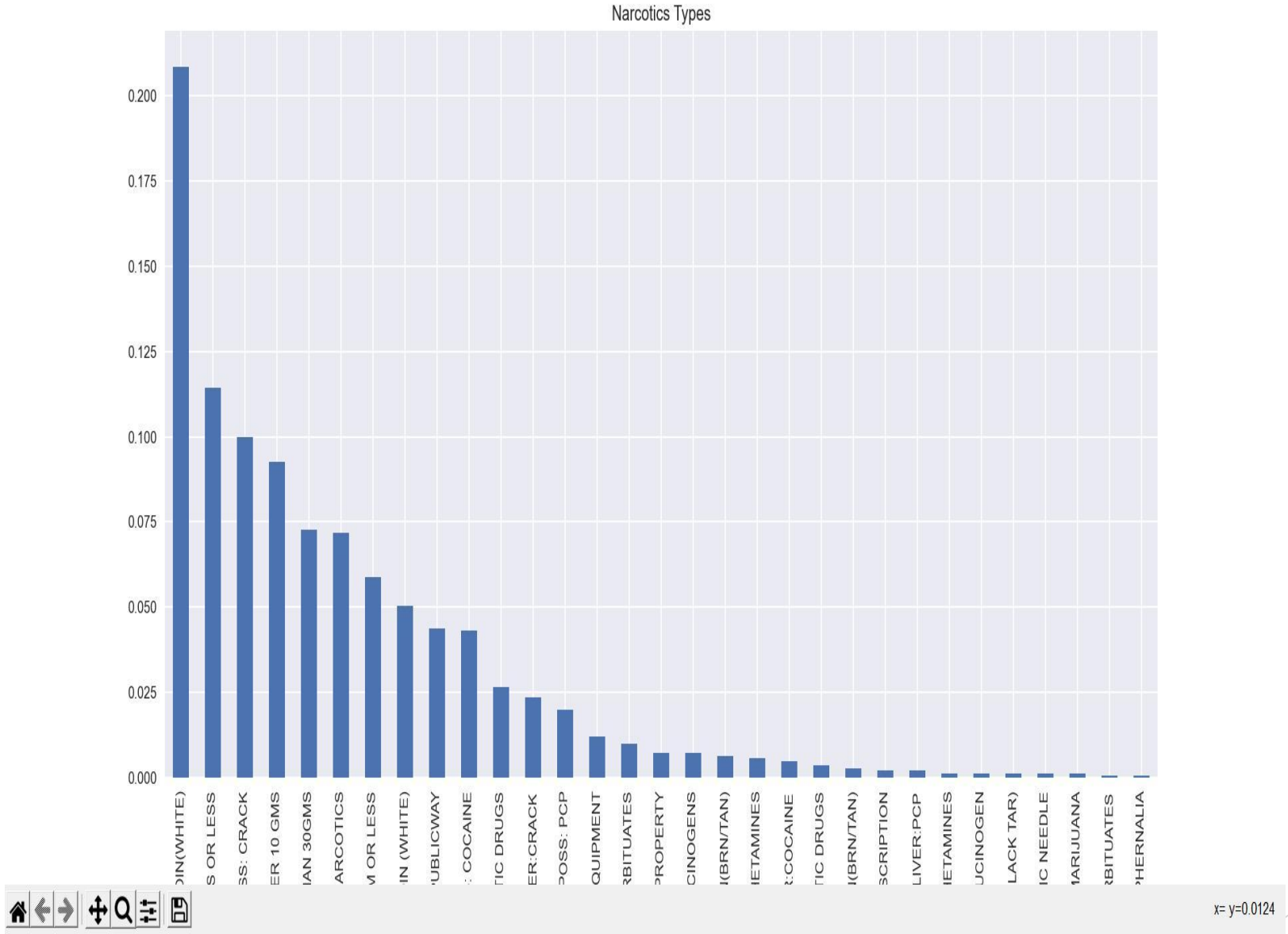
SCREENSHOTS

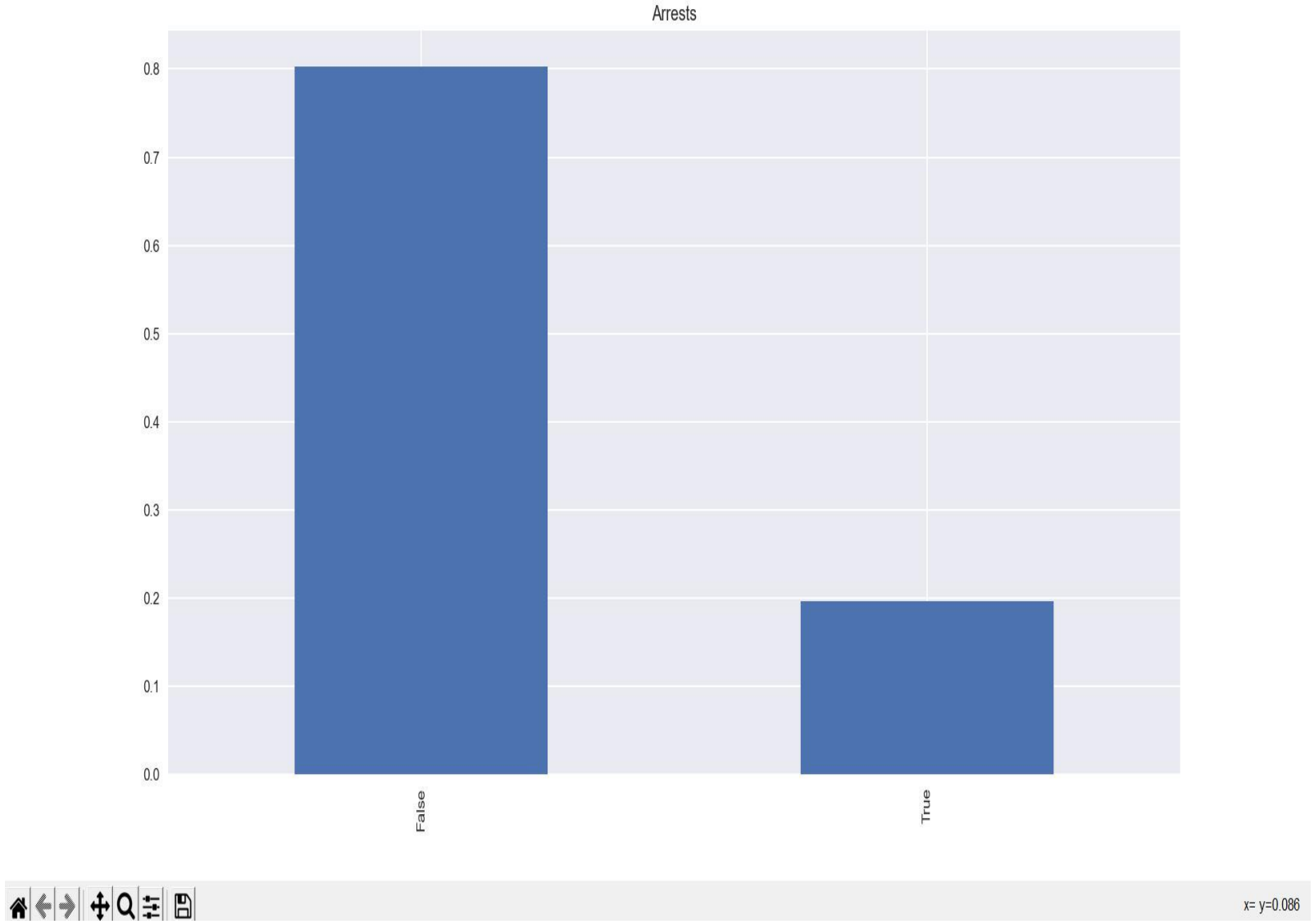


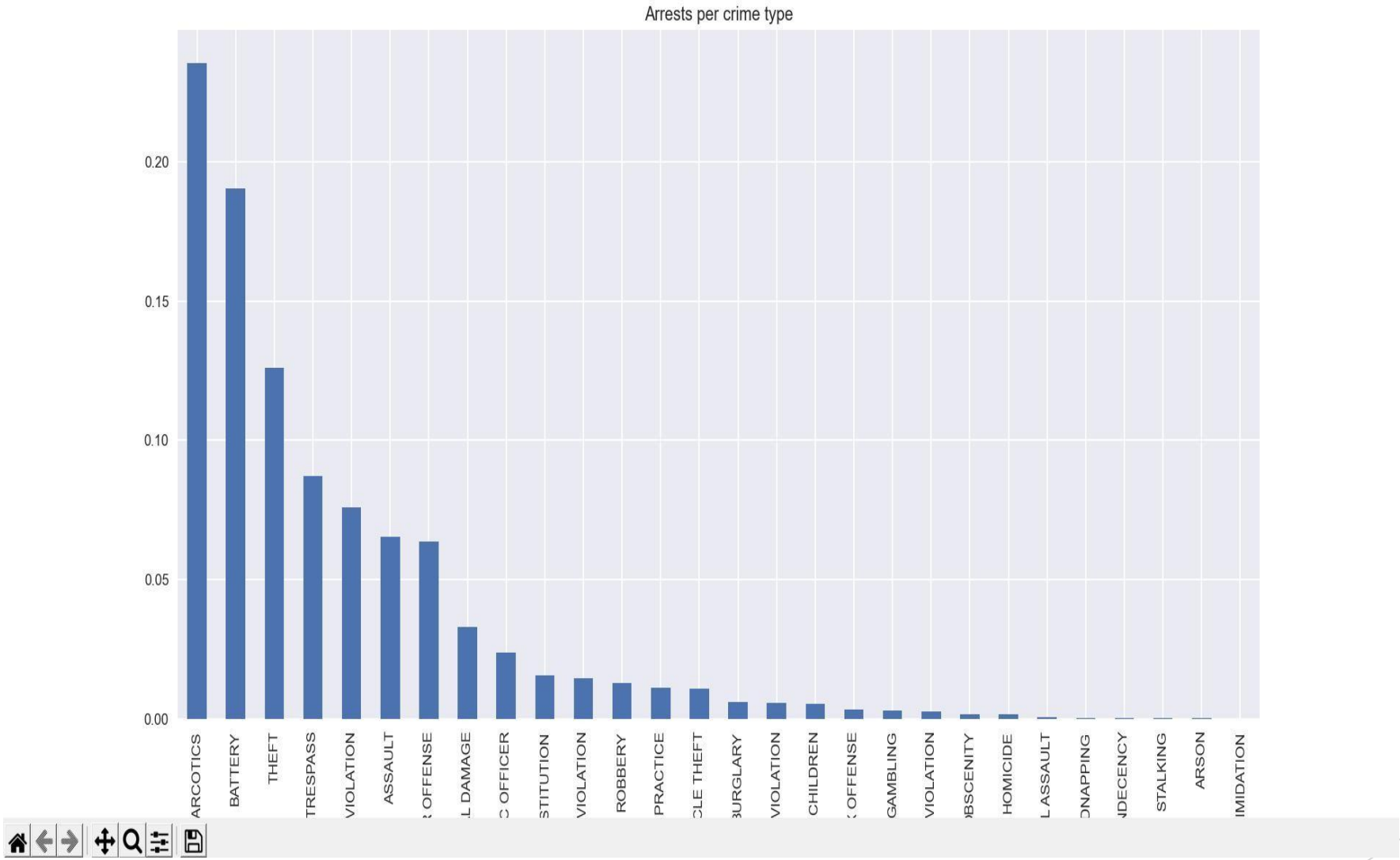

Theft Types
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
ND UNDER
OVER $500
TAIL THEFT
BUILDING
T-PICKING
NATCHING
MPT THEFT
NE/DEVICE
NER THEFT

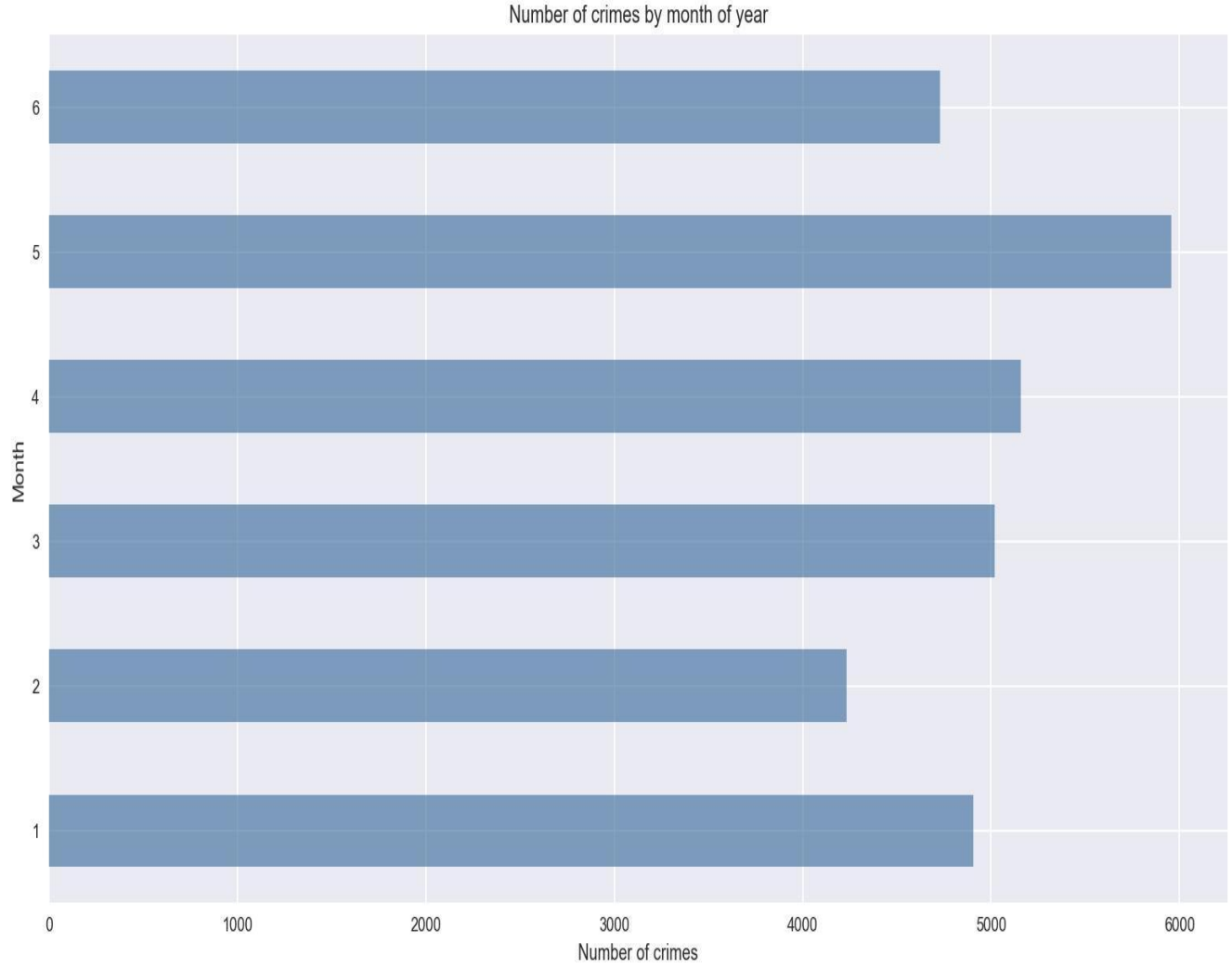

Battery Types
0.5
0.4
0.3
0.2
0.1
0.0
RY SIMPLE
SIMPLE
HANDGUN
G WEAPON
G WEAPON
MIN INJURY
TING INSTR
TTING INST
MIN INJURY
US INJURY
OR CITIZEN
OR INJURY
US INJURY
G WEAPON
OF A CHILD
ANG WEAP
ERIOUS INJ
R FIREARM
ERIOUS INJ
TTING INST
ORN CHILD
HANDGUN
x= y=0.0426


Criminal Damage Types
0.4
0.3
0.2
0.1
0.0
ROPERTY
O VEHICLE
ACEMENT
ROPERTY
SUP PROP
x= y=0.0503


Narcotics Types
0.200
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
x= y=0.0124


Arrests
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
False
True
x= y=0.086


Arrests per crime type
0.20
0.15
0.10
0.05
0.00
ARCOTICS
BATTERY
THEFT
TRESPASS
VIOLATION
ASSAULT
OFFENSE
L DAMAGE
OFFICER
STITUTION
VIOLATION
ROBBERY
PRACTICE
CLE THEFT
BURGLARY
VIOLATION
CHILDREN
OFFENSE
GAMBLING
VIOLATION
BSCENITY
HOMICIDE
L ASSAULT
DNAPPING
NDECENCY
STALKING
ARSON
IMIDATION


Number of crimes by month of year
Month
6
5
4
3
2
1
0
1000
2000
3000
4000
5000
6000
Number of crimes

x=4372. y=

CONCLUSION

In this system, we get to classify and cluster to improve the accuracy of location and pattern-based crimes. This software predicts frequently occurring crimes and its locations, especially for particular state, region and occasions. Also, time is an essential factor in the occurrence of crimes, and we shall predict the time as well in this application.

THANK YOU