PREDICTING THE CRITICAL HOT SPOTS OF CRIMINAL ACTIVITY USING MACHINE LEARNING WITH PYTHON

A project report submitted in partial fulfillment of the requirements for the award of the degree in

M.Sc. Data Science

Submitted by

JAI CHAUDHRANI

(VP21CSCI0200072)

Under the Esteemed Guidance of

Dr. G. BABU RAO

(Assistant Professor)

Department of Computer Science

GITAM School Of Science

GITAM (Deemed to be University)

Visakhapatnam-530045, A.P.

(2022- 23)

CERTIFICATE

This is to certify that the project report entitled "**PREDICTING THE CRITICAL HOT SPOTS OF CRIMINAL ACTIVITY USING MACHINE LEARNING WITH PYTHON**" is bonafide record of the project work done by **JAI CHAUDHRANI(VP21CSCI0200072)** from **December 2022** to **April 2023** in partial fulfillment of the requirement for the award of the degree of **M.Sc. Data Science** in the Department of Computer Science, GITAM School of Science, GITAM (Deemed to be University), Visakhapatnam.

Internal Guide

Dr. G. Babu Rao
Assistant Professor
Department of Computer Science

Head of the Department

Dr. T. Uma Devi
Associate Professor
Department of Computer Science
Gitam School of Science
GITAM

DECLARATION

I **JAI CHAUDHRANI(VP21CSCI0200072)** hereby declare that the project entitled "**PREDICTING THE CRITICAL HOT SPOTS OF CRIMINAL ACTIVITY USING MACHINE LEARNING WITH PYTHON**" is an original work done in the partial fulfillment of the requirements for the award of the degree of **M.Sc. Data Science** in **GITAM School of Science, GITAM (Deemed to be University)**, **Visakhapatnam**. I assure you that this project work has not been submitted towards any other degree or diploma in any other colleges or universities.

JAI CHAUDHRANI
(VP21CSCI0200072)

ACKNOWLEDGEMENT

It is my prime duty to express my sincere gratitude to all those who have helped me to successfully complete this project.

I also express my respectful and sincere thanks to my Project Guide Dr. G. Babu Rao, Assistant Professor who has given timely advice and supported me in my work.

I express respectful and sincere thanks to our Project In charge Mrs. B. Satyasaivani, Associate Professor who has mentored us throughout our project . Your tireless effort and dedication have made this project successful. I would like to acknowledge your expertise, hard work, and commitment that have been invaluable in delivering the project on time and meeting all the objectives. Your contribution to the project has been crucial, and without it, I would not have been able to achieve the desired outcome.

I express respectful and sincere thanks to Dr. T. Uma Devi, our Head of the Department and Associate Professor. I express my sincere thanks to our respectful faculty members of our department for the valuable cooperation, guidance and continuous support rendered by them to me throughout my project work.

I express my gratitude to Prof. K. Vedavathi, our beloved Principal, for the facilities provided by her throughout the course.

I express my gratitude to Prof.Balkumar Marthi, our beloved Dean, for the invaluable support and guidance provided by him throughout the course.

Finally, I would like to thank all my friends and my parents for giving full advice and giving full support for the completion of this project.

JAI CHAUDHRANI

(VP21CSCI0200072)

INDEX

ABSTRACT

The aim of this project is to create a model which involves systematic analysis for trends and patterns identification. The prime objective is to study all the history of crime regarding arrest of a criminal, crime history of the criminal, when the police were called for help to identify patterns and trends and to make a decision rapidly. Since, crime is one of the biggest social problems in today's world and it is increasing at the fire pace which is a major cause for concern, hence there is a need to monitor and keep track of all the crimes so that it can be used by police department to investigate the cases easily and quickly. Crime – a term which is just like havoc in today's world. It is a disastrous act for entire humanity and an obstacle in the way of development. Simply, a crime can be defined as a criminal offense against any person or an organization with an intent to harm them directly or indirectly that is illegal and punishable under the country law. As crimes are lifting high and high so there is a need to control them. Thus, there should be a system which can analyse crime risk prediction and police department can make use of this technology that can make their task easier to investigate the case on the basis of different trends for years. The LightGBM (Gradient Boosting Machine) algorithm is used for this project to build the prediction model.

1.INTRODUCTION

The increase in crime data recording coupled with data analytics resulted in the growth of research approaches aimed at extracting knowledge from crime records to better understand criminal behavior and ultimately prevent future crimes. Crime is a complex social phenomenon that has grown due to major changes in society. Law enforcement agencies need to learn the factors that lead to an increase in crime tendency. To curb this, there is always a need for strategies and policies to prevent crime. As a result of technology development, science and information, data mining and artificial intelligence tools are increasingly prevalent in the law enforcement community. Law enforcement agencies face a large volume of data that needs to be processed and turned into useful information, and data mining can improve crime analysis by helping to predict and prevent it. By processing criminal data, law enforcement agencies can use models that may be important in the crime prevention process.

1.1 Background

Crime is a social phenomenon as old as societies themselves, and although there will never be a free from crime society - just because it would need everyone in that society to think and act in the same way - societies always look for a way to minimize it and prevent it. In the modern United States history, crime rates increased after World War II, peaking from the 1970s to the early 1990s. Violent crime nearly quadrupled between 1960 and its peak in 1991. Property crime more than doubled over the same period. Since the 1990s, however, crime in the United States has declined steadily. Until recently crime prevention was studied based on strict behavioural and social methods, but the recent developments in Data Analysis have allowed a more quantitative approach in the subject. The dataset of nearly 12 years of crime reports from across all of San Francisco's neighbourhoods is explored, and a model that predicts the category of crime that occurred, given the time and location is

created. To build the model the LightGBM's (Gradient Boosting Machine) Python API is used to predict the category of crimes that occurred in the city by the bay.

1.2 Problem Statement

Crimes now a days are increasing day by day and with different level of intensity and versatility. The result is a great loss to society in terms of monitory loss, social loss and further it enhances the level of threat against the smooth livelihood in the society. To overcome this problem, the computing era can help to reduce the crime or even may be helpful in predicting the crime so that sufficient measures can be taken to minimize the loss to property and life. The crime rate prediction strategies can be applied on historical data available in the police records by examining the data at various angles like reason of crime, frequency of similar kind of crimes at specific location with other parameters to prepare the model crime prediction. It is a major challenge to understand the versatile data available with us, then model it to predict the future incidence with acceptable accuracy and further to reduce the crime rate.

1.3 Existing system

Many researchers have gone through this problem regarding the criminal cases being unsolved for a long period. They proposed different crime prediction algorithms. In all these models the accuracy will surely vary depending on the data set and the features or attributes which are selected during data pre-processing because it is discovered that many machine learning algorithms are implemented on data sets consisting of different places having distinctive features, so predictions are changing in all cases.

Drawbacks:

- By this methodology they had less accuracy in prediction.
- By this methodology the results are not perfect.

1.4 Proposed system

The advanced deep learning algorithms like XG Boost and Gradient Boost are used which gives better accuracy. The libraries used are Sklearn, Keras which has more understanding about the algorithm and are more accurate with the data produced. This project consists of processing steps that involves data cleaning, feature selection, dropping null values, data scaling by normalizing and standardizing. After these steps, the data is free of null values which may alter the accuracy of the model significantly and feature selection is used to select only the required features that won't affect the accuracy of model. The LightGBM(Gradient Boosting Machine) algorithm is used for good performance of the proposed model. The proposed solution can potentially help people to stay away from the locations (crime hotspot) at a certain time of the day along with saving lives. The police forces can use this solution to increase the level of crime prediction and prevention for police resources allocation. It can help in the distribution of police at most likely crime places for any given time, to grant an efficient usage of police resources. By having all of this information available, it will help to make our community safer for the people living there and also for others who will travel there.

Advantages:

- A high accuracy is resulted in this model prediction methodology.
- In this project, LightGBM (Gradient Boosting Machine) algorithm is used which will help to predict the crimes patterns and fast up the process of solving crime.
- The results are perfect and accurate using this technology.

1.5 Aim of the project

Crimes are a common social problem affecting the quality of life and the economic growth of a society. A crime or criminal offence is an act harmful not only to some individual but also to a community, society, or the state. Crimes could occur everywhere. It is common that criminals work on crime opportunities they face in most familiar areas for them. With the increase of crimes, law enforcement agencies are continuing to demand advanced geographic information systems and new data mining approaches to improve crime analytics and better protect their communities. By providing a data mining approach to determine the most criminal hotspots and find the type, location and time of committed crimes, there's a hope to raise people's awareness regarding the dangerous locations in certain time periods. By hand, these are difficult tasks but AI categorization shifting through massive amounts of visual data along with AI/ML algorithms can eliminate human errors especially in witness identification and therefore increasing arrest accuracy.

1.6 Scope of the project

The scope of using different methods for crime prediction and prevention can change the scenario of law enforcement agencies. Using a combination of ML and computer vision can substantially impact the overall functionality of law enforcement agencies. In the near future, by combining ML and computer vision, along with security equipment such as surveillance cameras and spotting scopes, a machine can learn the pattern of previous crimes, understand what crime actually is, and predict future crimes accurately without human intervention. Law enforcement agencies can be warned and prevent crime from occurring by implementing more surveillance within the prediction zone. This complete automation can overcome the drawbacks of the current system, and law enforcement agencies can depend more on these techniques in the near future. Designing a machine to anticipate and identify patterns of such crimes will be the starting point of our future study.

1.7 Objectives

- To find spatial and temporal criminal hotspots. Forecast crime using a set of real-world datasets of crimes.

- To locate the most likely crime locations and their frequent occurrence time.

- To predict what type of crime might occur next in a specific location within a particular time.

- To provide an analysis study by combining our findings of a particular crime dataset with its demographic's information.

2. SYSTEM REQUIREMENT SPECIFICATIONS

2.1 Feasibility Analysis

Feasibility Study analyses whether the proposed ideas will succeed or fail. It determines the practicality by assessing the opportunities and threats of the proposed plan. A feasibility study can help choose the best available alternative by assessing the opportunity cost. Feasibility study is necessary to determine that the proposed system is Feasible by considering the technical, Operational, and Economical factors. By having a detailed feasibility study the management will have a clear-cut view of the proposed system. There are different types of studies to check feasibility, such as technical feasibility, economic feasibility and operational feasibility that help a company determine the viability of a business plan. The feasibilities that are considered for the project in order to ensure that the project is variable and it does not have any major obstructions are as follows:

- Technical feasibility: This involves evaluating the technical capabilities and requirements of the project, such as the availability of relevant data sources, the complexity of the machine learning algorithms required, and the integration of the system with existing software development processes.
- Economic feasibility: Economic feasibility allows an organization to determine cost-benefit analysis. It gives details about the investment that has to go in to get the desired level of benefit (profit). Factors such as total cost and expenses are considered to arrive simultaneously.
- Operational feasibility: This involves evaluating the practicality and ease of use of the system, including how well it integrates with existing development processes, how easy it is to use and maintain, and how well it meets the needs and expectations of stakeholders.

2.2 Hardware Requirements

- Processor: Intel dual core, i3 or higher
- Speed: 2.2 GHz
- RAM: 4 GB or higher
- Hard Disk: 20 GB or higher

2.3 Software Requirements

- Operating System: Windows 8 or higher
- Programming Language: Python
- IDE: Kaggle Notebook or Jupyter Notebook

2.4 Functional Requirements

- The relevant crime data is collected before fitting the model and pre-processing is done by eliminating the outliers.
- The model should provide visualised data to identify the patterns, trends, and anomalies.
- The model developed should give predictions on accuracy to detect the crime patterns.
- The model should be evaluated and validated so that the final accuracy will be predicted.

2.5 Non-Functional Requirements

- Usability: It should be easy to use as well as read the data to prevent data loss.
- Performance: The model should generate accurate values even if the data is large.
- Scalability: The model should be able to handle large amount of data and users.
- Cost: The system should be cost effective including software, hardware.

3. ABOUT THE SOFTWARE

- Python programming language is a high-level, general-purpose and a very popular programming language.
- Python programming language is used in this project to predict the critical hotspots of criminal activity.
- The Kaggle notebook from Kaggle website is used for this project. The reason to choose Kaggle notebook facility is that when compared to Jupyter notebook, it gives faster outputs since it uses GPU and TPU accelerators to run the notebook cells.
- There is a need to be connected to the internet to run the code since the Kaggle notebook is not available to install on local device.
- A high-speed internet connection is preferred.
- Also, it should be made sure to fulfill the the mentioned hardware and software requirements to run the project.

Python is huge collection of standard libraries which can be used in this project for the following:

- Machine Learning
- GUI Applications (like Kaggle)
- Image processing (like Structured data, sk-learn)
- Test frameworks

Libraries Used in Project:

➢ Matplotlib

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib is used to show various visualizations and plots of this project like world plot, donut chart, bar plot, etc.

- **Pandas**

  Pandas is a Python library used for working with data sets. It has functions for analyzing, cleaning, exploring, and manipulating data. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc. A Pandas DataFrame is a 2-dimensional data structure, like a 2-dimensional array, or a table with rows and columns.

- **Seaborn**

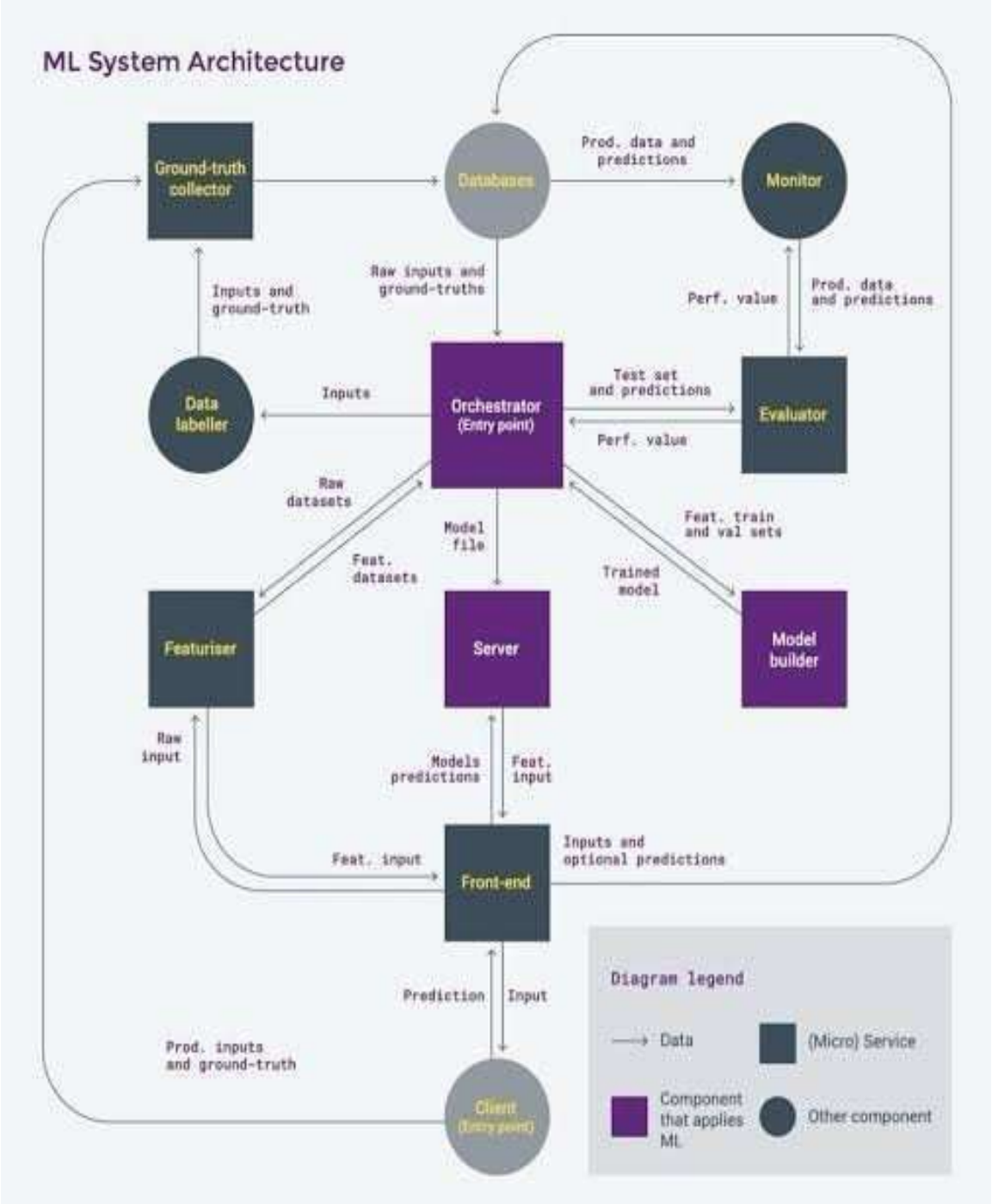
  Seaborn is a library for making statistical graphics in Python. It builds on top of matplotlib and integrates closely with pandas data structures. It plots pairwise relationships in a dataset. By default, a grid of Axes such that each numeric variable in data will be shared across the y-axes across a single row and the x-axes across a single column will be created. Using the pairplot function of the seaborn library, graphs are plotted for each pair of numerical features. This creates a nice visualisation and helps us understand the data by summarising a large amount of data in a single figure.

4.SYSTEM ANALYSIS AND REQUIREMENTS DOCUMENTATION

4.1 Overview

The prediction of critical hotspots of criminal activity is a systematic approach for finding the crime patterns and trends. This system can predict region which have high probability for crime occurrences and visualize crime prone area. It uses historical data and machine learning algorithms to identify areas of repeated crime. The system generates a high accuracy machine learning model. The proposed solution can potentially help people to stay away from the locations (crime hotspot) at a certain time of the day along with saving lives.

4.2 System Architecture



4.3 Modules

Module-1: Crime Data Acquisition

The crime data acquisition module involves the data cleaning, filling the missing values and data transformation. The preprocessing steps are performed to make the data suitable to visualize it. It is used to reduce the complexity of the dataset and improve the performance of machine learning models by selecting only the most important features. The features are ranked based on their relevance to the target variable. The top-ranked features are selected for the model. This can be done using various selection methods. The selected features are used to train the machine learning model. The feature selection process is iterative, and the steps are repeated until the desired level of accuracy or performance is achieved.

Module-2: Crime History Analysis

Crime analysis is a law enforcement function that involves systematic analysis for identifying and analysing patterns and trends in crime and disorder. Information on patterns can help law enforcement agencies deploy resources in a more effective manner, and assist detectives in identifying and apprehending suspects. Crime analysis also plays a role in devising solutions to crime problems, and formulating crime prevention strategies. A profession and process in which a set of quantitative and qualitative techniques are used to analyse data valuable to police agencies and their communities. Crime history data set is used to know, what type of data it is, Data Visualization and co relations are used to visualize the data. This involves analysis of the criminal activity happened in the past which will be helpful to understand the crime patterns and hotspots.

Module-3: Data EDA

EDA (Exploratory Data Analysis) is primarily used to see what data can reveal beyond the formal modeling or hypothesis testing task and provides a provides a

better understanding of data set variables and the relationships between them. It can also help determine if the statistical techniques you are considering for data analysis are appropriate. Exploratory Data Analysis refers to the critical process of performing initial investigations on data so as to discover patterns, to spot anomalies, to test hypothesis and to check assumptions with the help of summary statistics and graphical representations.

Module-4: Implementation of the Model

It is used to recognize about the model with more accuracy. The module implemented in this project is LightGBM (Gradient Boosting Machine). The implementation model represents how a system (application, service, interface, etc.) works. It is often described with system diagrams and pseudocode to be later translated into real code. This involves model fitting techniques and building process of LightGBM (Gradient Boosting Machine) model.

Module-5: Applying the Light GBM Algorithm

LightGBM(Gradient Boosting Machine) is an ensembling algorithm which is used for faster training speed and higher efficiency. LightGBM (Gradient Boosting Machine) uses histogram- based algorithm. It also replaces continuous values to discrete bins which result in lower memory usage. LightGBM (Gradient Boosting Machine) is a gradient boosting ensemble method that is used by the Train using AutoML tool and is based on decision trees. As with other decision tree-based methods, LightGBM (Gradient Boosting Machine) can be used for both classification and regression. LightGBM (Gradient Boosting Machine) is optimized for high performance with distributed systems. LightGBM (Gradient Boosting Machine) creates decision trees that grow leaf wise, which means that given a condition, only a single leaf is split, depending on the gain. Leaf-wise trees can sometimes overfit especially with smaller

datasets. Limiting the tree depth can help to avoid overfitting. It gives the generated model in a pickle file.

Module-6: Prediction of Crime Hotspots

Prediction of crime is a systematized method that classifies and examines the crime patterns. There exists various clustering algorithms for crime analysis and pattern prediction but they do not reveal all the possible hotspots of crime. Crime forecasting refers to the basic process of predicting crimes before they occur.

5. UML DIAGRAMS

UML stands for Unified Modelling Language and is a standardized visual language used to represent and design software systems. UML diagrams provide a way to visually represent various aspects of a software system, including its structure, behaviour, and interactions.

UML diagrams are categorized into two main groups: structural diagrams and behavioural diagrams

5.1 Use Case Diagram

Shows the interactions between the system and its actors. The Use-case diagram of this project shows how the system and the actor goes through the process of interaction. Use case diagrams model behaviour within a system and help the developers understand what the user requires. The stick man represents what's called an actor.

A use case diagram can be useful for getting an overall view of the system and clarifying who can do it and more importantly what they can't do.

- The purpose is to show the interactions between the use case and actor.
- To represent the system requirements from user's perspective.
- An actor could be the end-user of the system or an external system.

A Use case of Criminal hotspot prediction analysis includes Importing the necessary packages, Data Cleaning or preprocessing, Visualization of the data, Generation of the model that is going to be used, Building the model and Prediction.

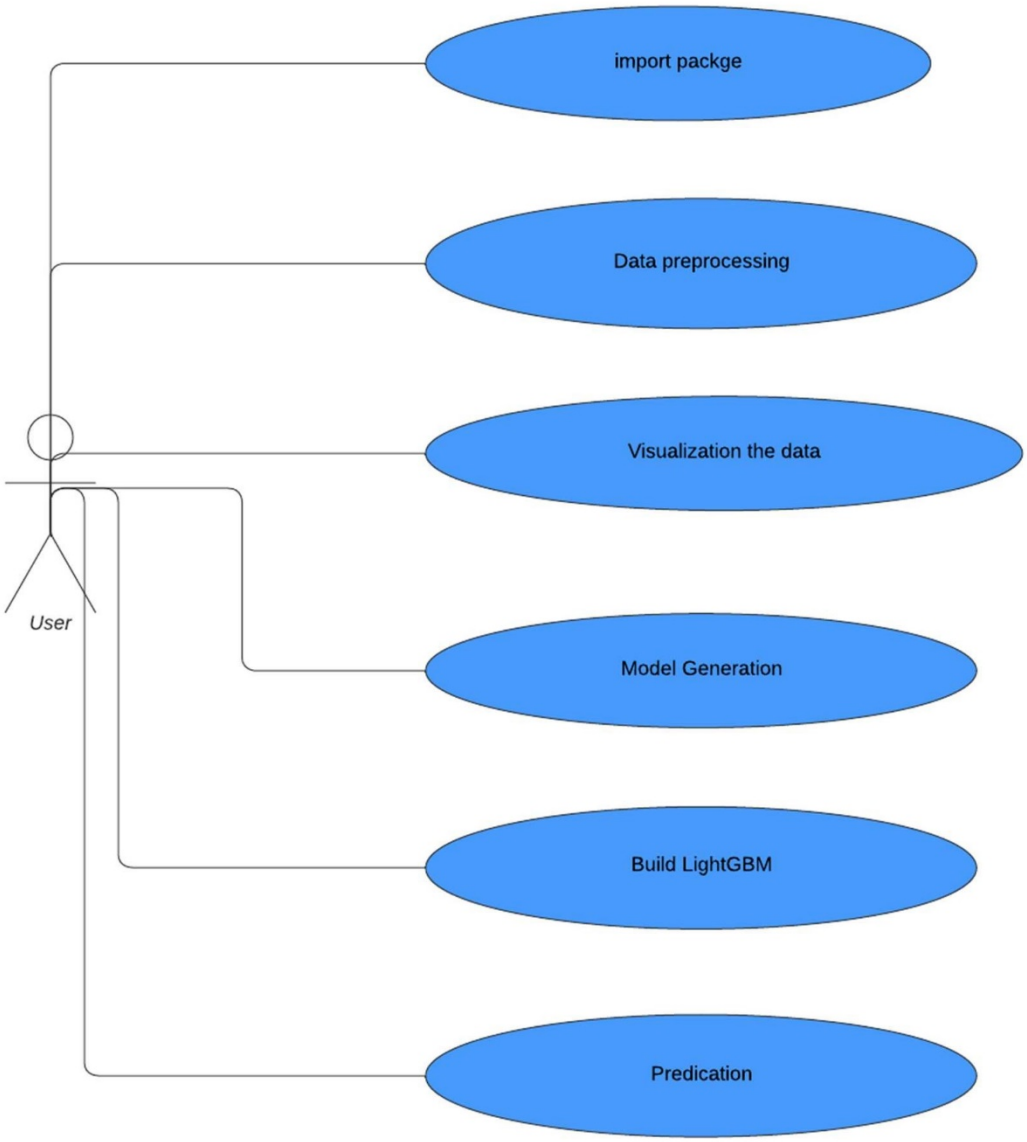


Fig-1: Use case diagram of criminal hotspot prediction

5.1.1 Collaboration Diagram

- Shows the involvement of various entities and interconnection between them in the system.
- A collaboration diagram groups together the interactions between different objects.The collaboration diagram helps to identify all the possible

interactions that each object has with other objects. The user interacts with application and dataset to know prediction the criminal hotspots.

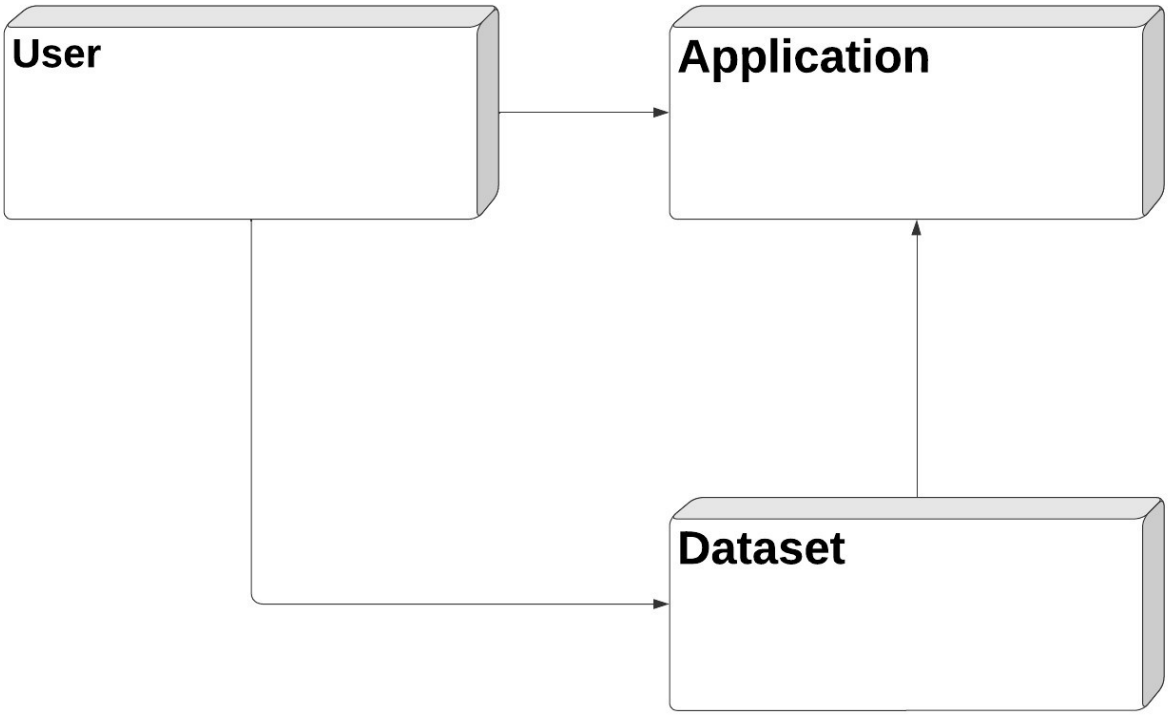


Fig-2: Collaboration Diagram

5.1.2 Activity Diagram

An activity diagram for the Crime hotspot prediction project can help to illustrate the flow of activities involved in the model's functioning.

- The first activity in the diagram is the user opening an application for accessing the dataset for the pre-processing.
- Once the pre-processing is done, validation of the crime is done. This involves the normalization and splitting of data.
- The model is generated after the validation is done. Data is then sent for training and testing states while applying the LightGBM (Gradient Boosting Machine) algorithm such that the model is built.

- Finally, the prediction model is built which shows the prediction of the criminal hotspots and accuracy is calculated.

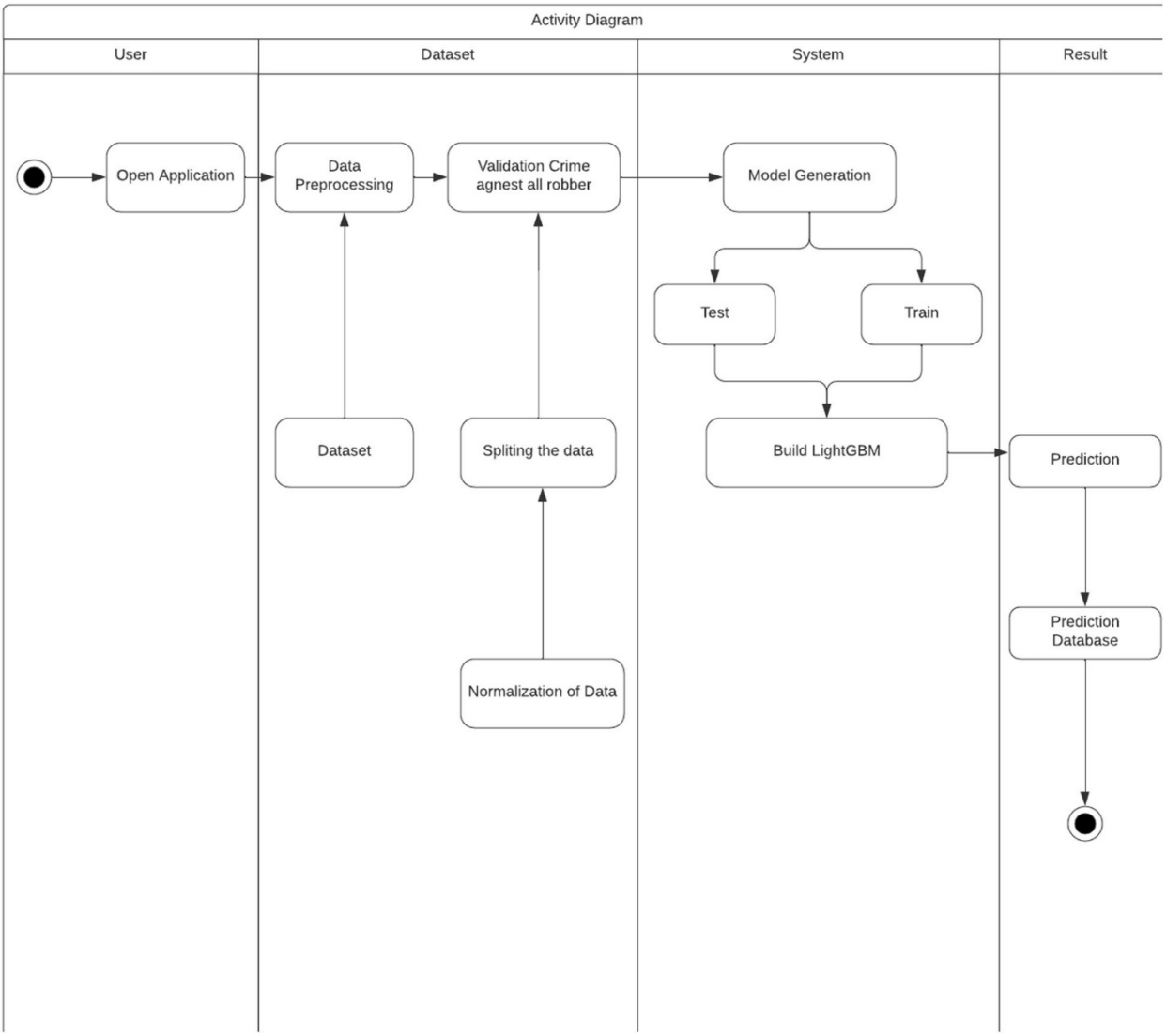


Fig-3: Activity Diagram

5.1.3 Deployment Diagram

- Shows the deployment of the system.
- Deployment diagram captures the configuration of the runtime elements of the application. This diagram is by far most useful when a system is built and ready to be deployed. The crime application code helps to generate a model which can be distributed among clients.

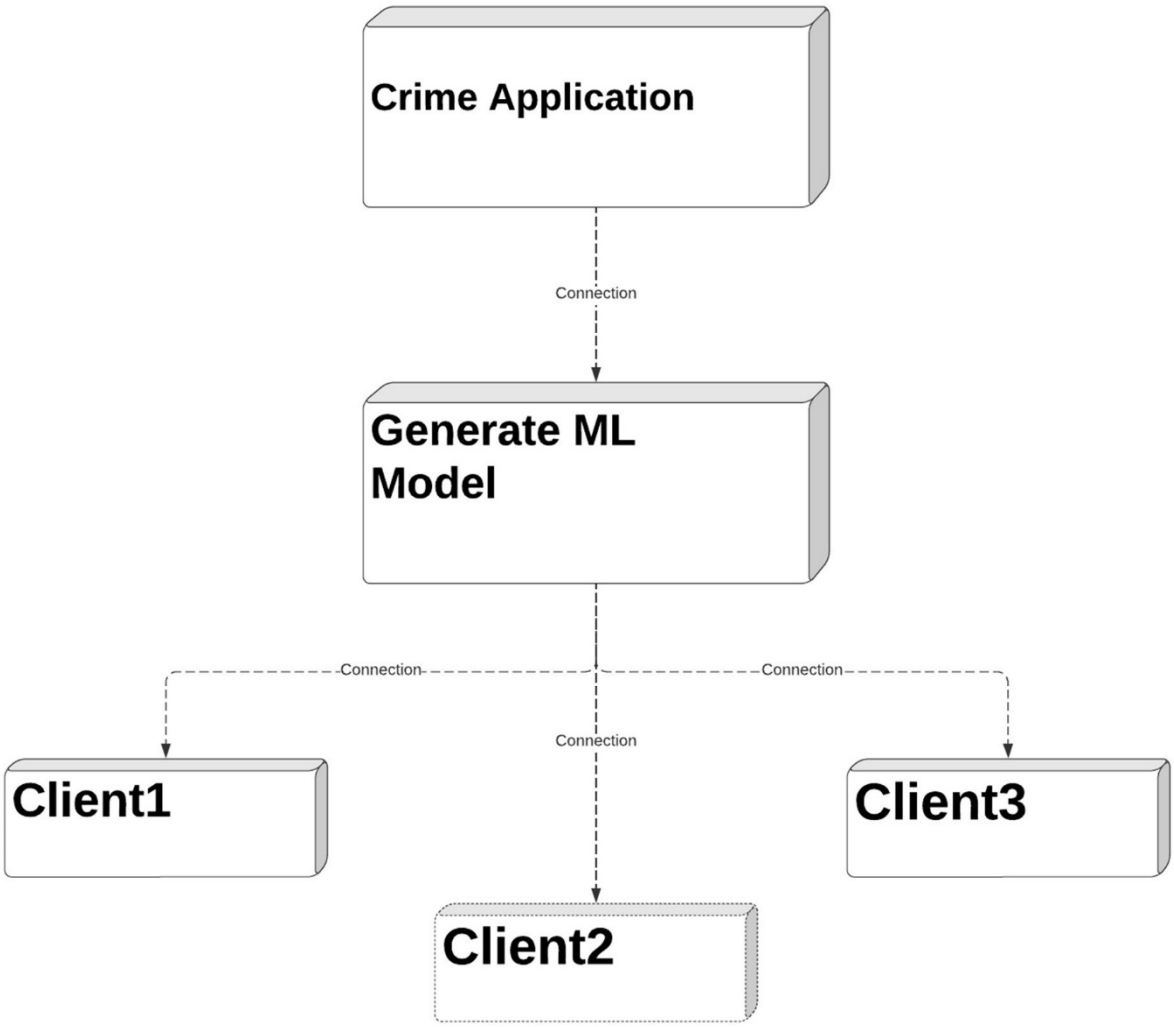


Fig-4: Deployment Diagram

5.2 Structured Diagram

5.2.1 Class Diagram

Shows the classes and their relationships in the system.

The class diagram for the Crime prediction project includes several classes that represent different modules and components of the prediction model. These classes are interconnected and work together to provide the model's functionality. The class diagram is used to refine the use case diagram and define a detailed design of the system. The class diagram classifies the actors defined in the use case diagram into a set of interrelated classes. The relationship or association between the classes can be either an "is-a" or "has-a" relationship. Each class in the class diagram may be capable of providing certain functionalities. These functionalities provided by the class are termed "methods" of the class. Apart from this, each class may have certain "attributes" that uniquely identify the class.

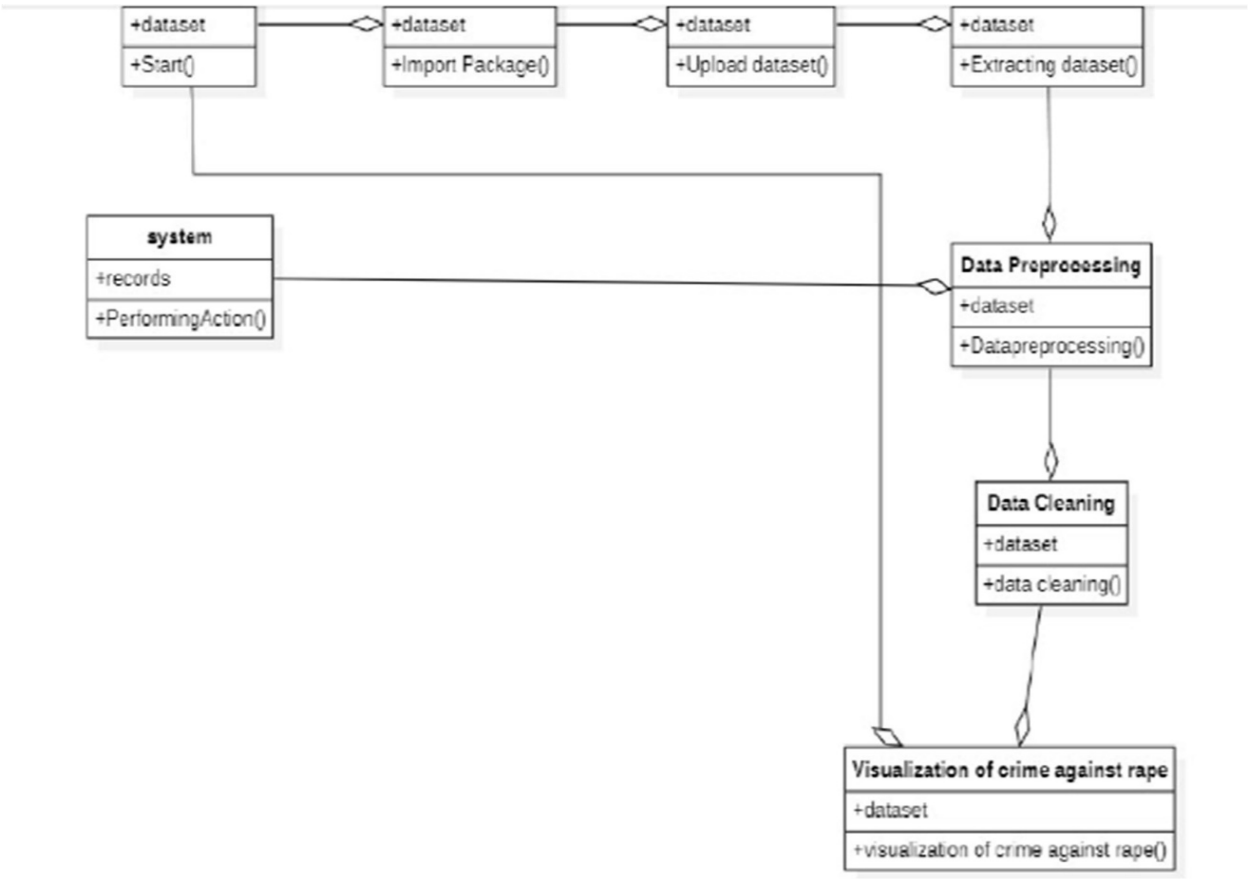


Fig-5: Class Diagram

5.2.2 Object Diagram

- Shows the objects and methods involved in the system.
- The object diagram is a special kind of class diagram. An object is an instance of a class. This essentially means that an object represents the state of a class at a given point of time while the system is running. The object diagram captures the state of different classes in the system and their relationships or associations at a given point of time. The object diagram for this project consists of user doing all the necessary ML steps for the ML algorithms to generate and build a model and predict crime hotspots.

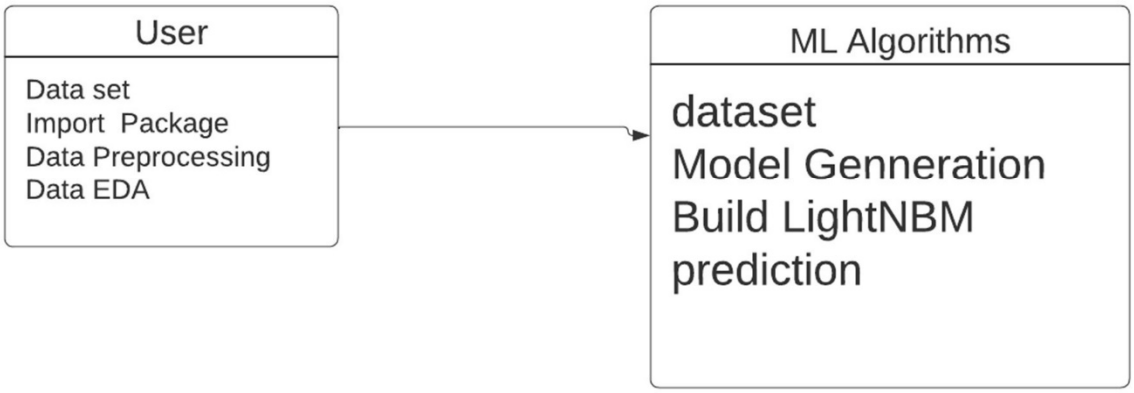


Fig-6: Object Diagram

6. FLOW CHART

It Shows the flowing of steps involved in the system.

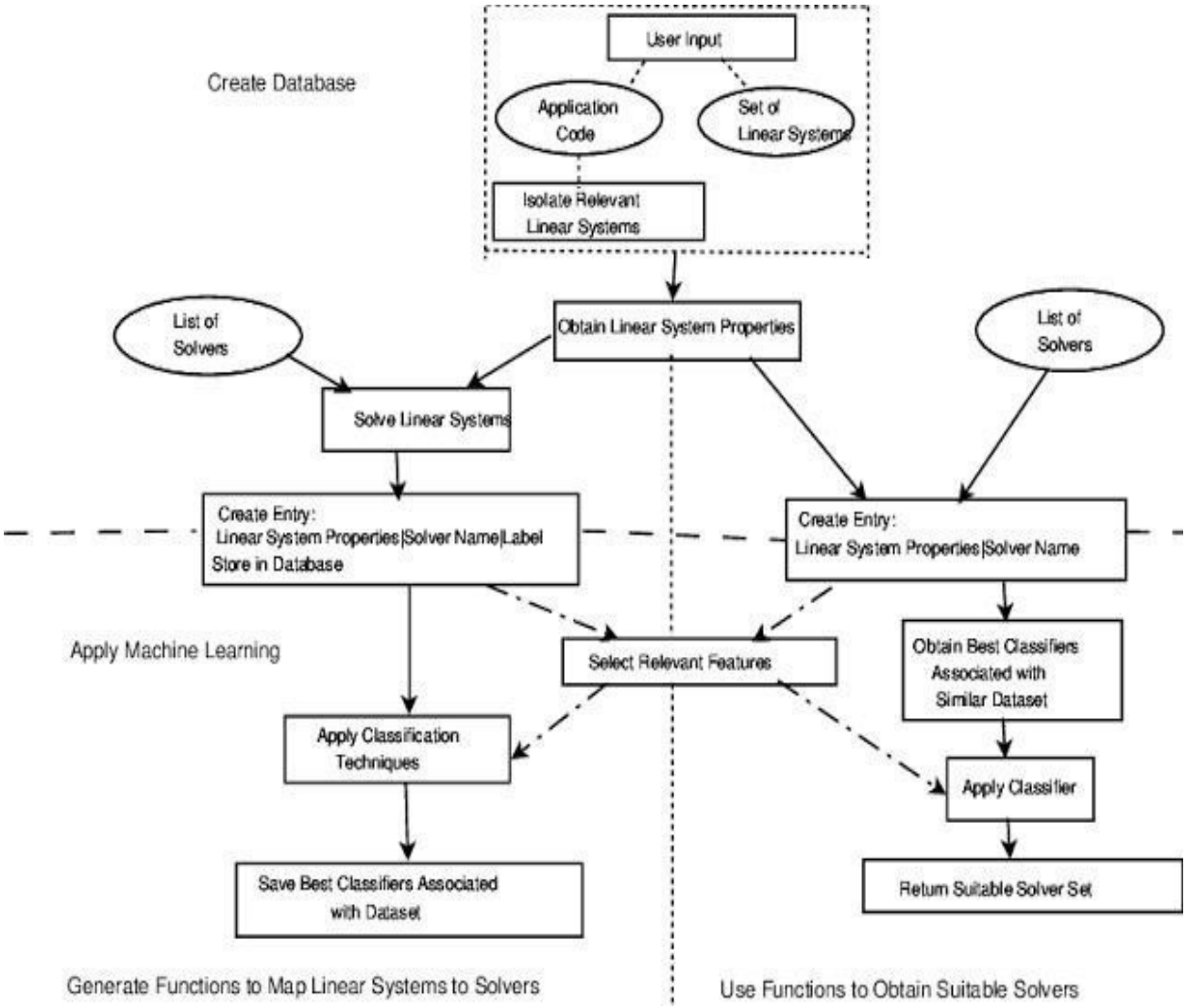


Fig-7: Flow Chart

7.SYSTEM DESIGN AND DOCUMENTATION

7.1 Data Collection: For solving machine learning problems firstly, raw data is needed because without raw data machine learning problems cannot be solved. Raw data is fetched from further discussion of the problem with client and data scientist team but the focus on data is that a data integration and data integration is a very difficult task because collection of data is done from multiple resources like structure data unstructured data, web scraping, etc. Collected data is stored in data warehouse and data is fetched from a data warehouse and as a data scientist one should know how to collect data from multiple resources.

```
[ ]:
df_test = pd.read_csv('/kaggle/input/sf-crime/test.csv.zip', parse_dates=['Dates'])
df_test.head()
```

```
[ ]:
df_test.info()
```

7.2 Data Analysis: After collecting data, the second step is data analysis which involves data cleaning. Data cleaning means either removing or using the imputer function for null values. In the data set there are so many null values present and how much of cleaning is done is unknown, that means removing of null values; but there is some other technique is that is imputer function for integer values only. When the imputer function is used which means that removing of null values and replacing the mean, median or mode function, int value is replaced in null value. But this technique is useful only for integer data set not string.

[ ]:
```
col = sns.color_palette()
plt.figure(figsize=(12, 8))
sns.kdeplot(data=data, shade=True, color='g')
plt.title('Number of crimes per day', fontdict={'fontsize': 16})
plt.xlabel('Quantity')
plt.ylabel('Frequency')
plt.show()
```

```
week = df_train['DayOfWeek'].value_counts()
week.plot(kind="bar",figsize=(12,8), table=True,  color='g')
plt.xticks([])
plt.xlabel('days and weeks',fontsize=15,labelpad=30)
plt.ylabel('Number of criminals',fontsize=25)
plt.title('interpolation by day of the week',fontsize=25)
plt.show()
week.describe()
```

+ Code + Markdown

[ ]:
```
dist = df_train["PdDistrict"].value_counts()
dist.plot(kind="bar",figsize=(12,8), table=True,  color='g')
plt.xticks([])
plt.xlabel('area',fontsize=15,labelpad=30)
plt.ylabel('Number of criminals',fontsize=25)
plt.title('Precipitation by regions',fontsize=25)
plt.show()
dist.describe()
```

[ ]:
```
kind = df_train['Category'].value_counts()
kind.plot(kind="barh",figsize=(20,12),  color='g')
plt.ylabel('Type',fontsize=15)
plt.xlabel('Number of criminals',fontsize=15)
plt.title('Videos of criminals',fontsize=25)
plt.show()
```

[ ]:
```
kind.describe()
```

[ ]:
```
kind
```

```
[ ]:
plt.figure(figsize=(12, 8))
sns.kdeplot(data=kind, shade=True, color='g')
plt.title('The total number of crimes per district', fontdict={'fontsize': 16})
plt.xlabel('Quantity')
plt.ylabel('Frequency')
plt.show()
```

```
[ ]:
df_train.Descript.describe()
```

```
[ ]:
f = plt.figure(figsize=(15, 12))
plt.matshow(df_train.corr(), fignum=f.number)
plt.xticks(range(df_train.select_dtypes(['number']).shape[1]), df_train.select_dtypes(['number']).columns, fontsize=14, rotation=45)
plt.yticks(range(df_train.select_dtypes(['number']).shape[1]), df_train.select_dtypes(['number']).columns, fontsize=14)
cb = plt.colorbar()
cb.ax.tick_params(labelsize=14)
plt.title('Correlation Matrix', fontsize=16);
```

```
[ ]:
sns.pairplot(df_train, hue='Category')
!pip install contextily
import contextily as ctx
```

7.3 Splitting of data: The next step is splitting of data in which data is split for training and testing. Almost 80% of data is for training and 20% for testing which is a basic rule in the machine learning.

```
[35]:
df_train.loc[(df_train.Hour >= 20) & (df_train.Hour <= 23)].count()
```

7.4 Training the data: In this step, data is trained for machine analysis itself and one more step which is done is a validation of training data because training data set will generate either overfitting or under fitting problem that means false positive output or true negative output.

```
[ ]:
df_train = pd.read_csv('/kaggle/input/sf-crime/train.csv.zip', parse_dates=['Dates'])
df_train.head()
```

```
[ ]:
df_train.info()
```

```
[ ]: df_train.duplicated().sum()
```

+ Code + Markdown

```
[ ]: df_train.drop_duplicates(inplace=True)
df_train = df_train.drop(df_train[(df_train.Y > 50)].index)
df_train.shape
```

7.5 Testing the data: In the testing phase, the model is tested using cross-validation. The model is checked to know if it is going is right or not. Some techniques of cross-validation and confusion matrix is used for checking model performance.

```
[ ]: df_test = pd.read_csv('/kaggle/input/sf-crime/test.csv.zip', parse_dates=['Dates'])
df_test.head()
```

```
[ ]: df_test.info()
```

```
[14]: df_test.duplicated().sum()
```

```
[11]: print(df_test.loc[df_test.Y > 50].count()[0])
df_test.loc[df_test.Y > 50].sample(3)
```

7.6 Evaluating the data: The data is then evaluated and the visualization results are shown.

[61]:
```
print("TRAIN")
y_pred = clf.predict(X_train)
print(f"accuracy:  {accuracy_score(y_train, y_pred)}")
print(f"precision: {precision_score(y_train, y_pred, average=None)}")
print(f"recall:    {recall_score(y_train, y_pred, average=None)}")
print("TEST")
y_pred = clf.predict(X_test)
print(f"accuracy:  {accuracy_score(y_test, y_pred)}")
print(f"precision: {precision_score(y_test, y_pred, average=None)}")
print(f"recall:    {recall_score(y_test, y_pred, average=None)}")
```

```
booster1.fit(X_train, y_train, eval_set=[(X_test, y_test),], )
print("Test  Accuracy: %.2f"%booster1.score(X_test, y_test))
print("Train Accuracy: %.2f"%booster1.score(X_train, y_train))
```

7.7 Deploying the model: The last step is model deployment which means after saving the model either pickle file is used in web development or s/w.

[72]:
```
import joblib
# Saving model
!wget https://www.dropbox.com/s/jqfncd3z33xdvei/lgb_fin.pkl?dl=0 -O lgb_fin.pkl
bst = joblib.load('lgb_fin.pkl')
```

The final output will be a csv file showing each attribute's accuracy.

```
# Store the test dataset in the variable submission
submission = pd.DataFrame(
    preds_proba,
    columns = le2.inverse_transform(np.linspace(0, 38, 39, dtype='int16')),
    index = test.index)
submission.head()
```

```
# Save the test data set in the file "lgb_model_fin.csv"
submission.to_csv(
    'lgb_model_fin.csv', index_label='Id')
```

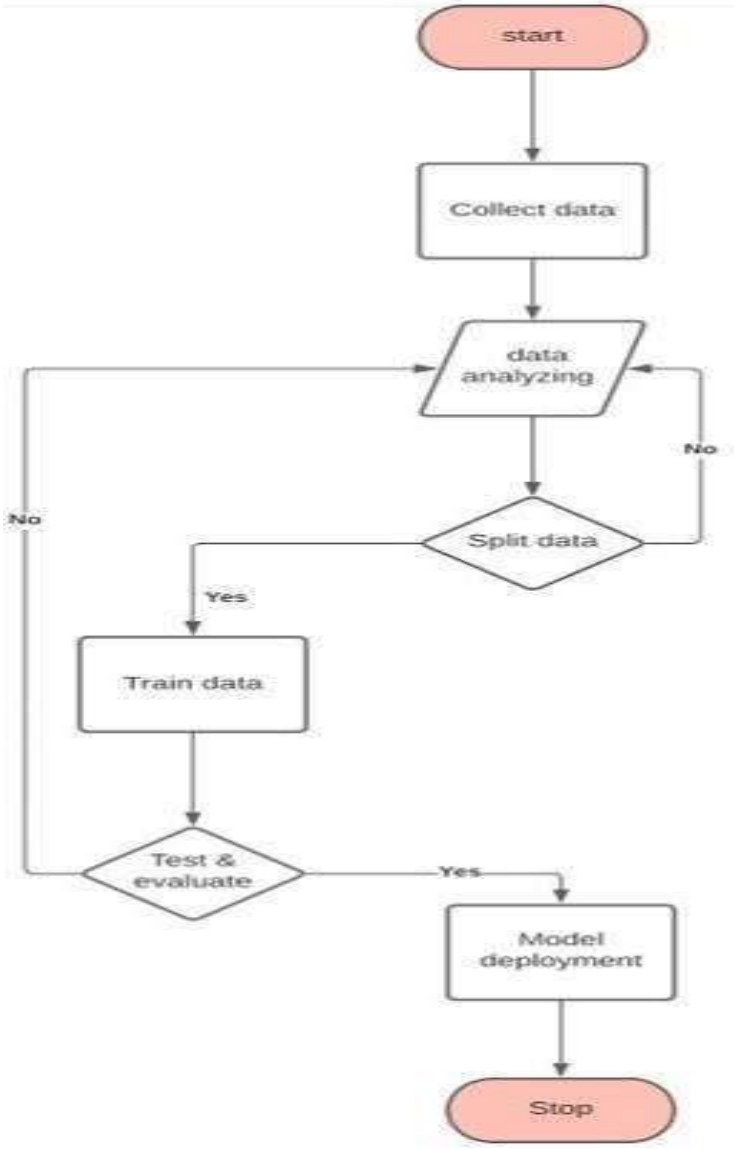


Fig-8: System Design

8. CODE

1.Importing the modules and installing the necessary packages:

```
[ ]:
!unzip "/kaggle/input/sf-crime/test.csv.zip" -d data
!unzip "/kaggle/input/sf-crime/train.csv.zip" -d data
# Import the required libraries
import os
import numpy as np
import pandas as pd

import warnings
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px
warnings.filterwarnings("ignore")
pd.set_option("display.max_rows",None)
from sklearn import preprocessing
import matplotlib
matplotlib.style.use('ggplot')
from sklearn.preprocessing import LabelEncoder
```

+ Code + Markdown

```
[ ]:
# Import the necessary libraries so that you can build geographic maps:

!pip install geopandas
!pip uninstall Fiona==1.9 --yes
!pip install Fiona==1.8
import geopandas as gpd
from shapely.geometry import  Point
sns.set(style="whitegrid", palette="pastel", color_codes=True)
sns.mpl.rc("figure", figsize=(20,20))
from shapely import wkt
%matplotlib inline
```

2.Reading the dataset files and creating a geo data frame:

[ ]:
```
df_test = pd.read_csv('/kaggle/input/sf-crime/test.csv.zip', parse_dates=['Dates'])
df_test.head()
```

[ ]:
```
df_test.info()
```

[ ]:
```
df_train = pd.read_csv('/kaggle/input/sf-crime/train.csv.zip', parse_dates=['Dates'])
df_train.head()
```

[ ]:
```
df_train.info()
```

[ ]:
```
print(f"first date covert: {df_train.Dates.sort_values().min()}")
print(f"last date: {df_train.Dates.sort_values().max()}")
df_train.shape
```

[ ]:
```
df_train.describe().T
```

[ ]:
```
def create_gdf(df):
    gdf = df.copy()
    gdf['Coordinates'] = list(zip(gdf.X, gdf.Y))
    gdf.Coordinates = gdf.Coordinates.apply(Point)
    gdf = gpd.GeoDataFrame(
        gdf, geometry='Coordinates', crs={'init': 'epsg:4326'})
    return gdf

train_gdf = create_gdf(df_train)

world = gpd.read_file(gpd.datasets.get_path('naturalearth_lowres'))
ax = world.plot(figsize=(14,10), color='green', edgecolor='black', alpha=0.5)

train_gdf.plot(ax=ax, color='red')
plt.show()
```

[ ]:
```
print(train_gdf.loc[train_gdf.Y > 50].count()[0])
train_gdf.loc[train_gdf.Y > 50].sample(5)
```

3.Performing data cleaning and data visualization:

```
[ ]: df_train.duplicated().sum()
```

+ Code + Markdown

```
[ ]: df_train.drop_duplicates(inplace=True)
df_train = df_train.drop(df_train[(df_train.Y > 50)].index)
df_train.shape
```

```
[ ]: df_test.duplicated().sum()
```

```
[ ]: train_gdf.loc[(train_gdf.PdDistrict == 'TENDERLOIN') & (train_gdf.Address == '7THSTNORTH ST / MCALLISTER ST')]
```

```
[ ]: train_gdf.loc[(train_gdf.Y > 50)].sort_values(['PdDistrict','Address']).sample(5)
```

```
[ ]: df_train['Date'] = df_train.Dates.dt.date
data = df_train.groupby('Date').count().iloc[:, 0]
data.describe()
```

```
[ ]: col = sns.color_palette()
plt.figure(figsize=(12, 8))
sns.kdeplot(data=data, shade=True, color='g')
plt.title('Number of crimes per day', fontdict={'fontsize': 16})
plt.xlabel('Quantity')
plt.ylabel('Frequency')
plt.show()
```

```
week = df_train['DayOfWeek'].value_counts()
week.plot(kind="bar",figsize=(12,8), table=True,  color='g')
plt.xticks([])
plt.xlabel('days and weeks',fontsize=15,labelpad=30)
plt.ylabel('Number of criminals',fontsize=25)
plt.title('interpolation by day of the week',fontsize=25)
plt.show()
week.describe()
```

+ Code + Markdown

[ ]:
```
dist = df_train["PdDistrict"].value_counts()
dist.plot(kind="bar",figsize=(12,8), table=True,  color='g')
plt.xticks([])
plt.xlabel('area',fontsize=15,labelpad=30)
plt.ylabel('Number of criminals',fontsize=25)
plt.title('Precipitation by regions',fontsize=25)
plt.show()
dist.describe()
```

[ ]:
```
kind = df_train['Category'].value_counts()
kind.plot(kind="barh",figsize=(20,12),  color='g')
plt.ylabel('Type',fontsize=15)
plt.xlabel('Number of criminals',fontsize=15)
plt.title('Videos of criminals',fontsize=25)
plt.show()
```

[ ]:
```
kind.describe()
```

[ ]:
```
kind
```

```
[ ]:
plt.figure(figsize=(12, 8))
sns.kdeplot(data=kind, shade=True, color='g')
plt.title('The total number of crimes per district', fontdict={'fontsize': 16})
plt.xlabel('Quantity')
plt.ylabel('Frequency')
plt.show()
```

```
[ ]:
df_train.Descript.describe()
```

```
[ ]:
f = plt.figure(figsize=(15, 12))
plt.matshow(df_train.corr(), fignum=f.number)
plt.xticks(range(df_train.select_dtypes(['number']).shape[1]), df_train.select_dtypes(['number']).columns, fontsize=14, rotation=45)
plt.yticks(range(df_train.select_dtypes(['number']).shape[1]), df_train.select_dtypes(['number']).columns, fontsize=14)
cb = plt.colorbar()
cb.ax.tick_params(labelsize=14)
plt.title('Correlation Matrix', fontsize=16);
```

```
[ ]:
sns.pairplot(df_train, hue='Category')
!pip install contextily
import contextily as ctx
```

```
[ ]:
train_gdf.head()
```

```
[ ]:
geometry2 = [Point(xy) for xy in zip(df_train['X'], df_train['Y'])]
```

```
[ ]:
! wget https://www2.census.gov/geo/tiger/TIGER2017//ROADS/tl_2017_06075_roads.zip -q
```

```
[ ]:
!unzip "./tl_2017_06075_roads.zip" -d shapefiles
```

+ Code + Markdown

```
[ ]:
geo_df = gpd.read_file('shapefiles/tl_2017_06075_roads.shp')
geo_df = geo_df.to_crs(epsg=3857)
crime_geo_df = gpd.GeoDataFrame(df_train, geometry=geometry2)
crime_geo_df.crs = "EPSG:4326"
crime_geo_df = crime_geo_df.to_crs(epsg = 3857)
sns.set_context("paper", font_scale=2)
fig, ax = plt.subplots(figsize=(20,20))
geo_df.plot(ax=ax, alpha = .1)
crime_geo_df.plot(ax=ax, column = 'Category', cmap = 'brg', marker = '.', \
                  markersize=100, edgecolor = 'black', alpha=1, legend = True)
ctx.add_basemap(ax, )
ax.set_axis_off()
fig.tight_layout()
plt.title("Transmigration to San Francisco")
plt.savefig('SF_crime_areas.png');
```

```
[33]:
#Seaborn - Count Plot
plt.figure(figsize=(25,15))
ax = sns.countplot(x="DayOfWeek", data=df_train,
                   facecolor=(0, 0, 0, 0),
                   linewidth=5,
                   edgecolor=sns.color_palette("dark", 24))
```

[ ]:
```
import plotly.express as px
cats = df_train["Category"].value_counts()[:10]
cats['OTHER 29 pos'] = df_train["Category"].value_counts()[11:].sum()
label = cats[0]
cats = pd.DataFrame(cats).reset_index()
cats.rename(columns={'index': 'Category', 'Category': 'Count'}, inplace=True)
cats.index += 1
label = cats['Category']
fig = px.pie(names=label, values=cats['Count'], height=600, color_discrete_sequence=px.colors.sequential.RdBu)
fig.update_traces(textposition='inside', textinfo='percent+label', textfont_size=12, )
fig.update_layout(title_text = 'TOP-10 Category of criminals',
                  title_font = dict(size=20, family='Verdana', color='black'))
fig.show()
```

[ ]:
```
df_train['Hour'] = df_train.Dates.dt.hour
```

[ ]:
```
df_train.loc[(df_train.Hour >= 20) & (df_train.Hour <= 23)].count()
```

[ ]:
```
data = df_train.groupby(['Hour', 'Date', 'Category'],
                        as_index=False).count().iloc[:, :4]
data.rename(columns={'Dates': 'Incidents'}, inplace=True)
data = data.groupby(['Hour', 'Category'], as_index=False).mean()

sns.set_style("whitegrid")
fig, ax = plt.subplots(figsize=(14, 10))
ax = sns.lineplot(x='Hour', y='Incidents', data=data, hue='Category')
ax.legend(loc='upper center', bbox_to_anchor=(0.5, 1.15), ncol=6)
plt.suptitle('Number of crimes in an interval of time')
fig.tight_layout(rect=[0, 0.03, 1, 0.95])
plt.show()
```

[ ]:
```
average_frequency = (df_train.groupby('Category').count())/(df_train.shape[0])
average_frequency.Dates
```

[ ]:
```
average_frequency.mean().mean()
```

[ ]:
```
from pandas.tseries.holiday import USFederalHolidayCalendar as calendar
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import log_loss
```

[ ]:
```
df_train['Date'] = df_train.Dates.dt.date
df_train['Day'] = df_train.Dates.dt.day
df_train['Month'] = df_train.Dates.dt.month
df_train['Year'] = df_train.Dates.dt.year
df_train.head(5)
```

[ ]:
```
dates = pd.to_datetime(df_train["Date"])
holidays = calendar().holidays(start=df_train["Date"].min(), end=df_train["Date"].max())
df_train["Holiday"] = dates.astype("datetime64").isin(holidays)
df_train.Holiday.value_counts()
```

[ ]:
```
le = LabelEncoder()
df_train["Category_Label"] = le.fit_transform(df_train["Category"])
df_train["DayOfWeek_Label"] = le.fit_transform(df_train["DayOfWeek"])
df_train["PdDistrict_Label"] = le.fit_transform(df_train["PdDistrict"])
df_train.head(5)
```

[ ]:
```
df_train["Holiday_Label"] = le.fit_transform(df_train["Holiday"])
df_train.head(5)
```

[ ]:
```
df_train = df_train.drop(columns=["Dates", "Category", "Descript", "DayOfWeek", "PdDistrict","Resolution", "Address", "Date", "Holiday"])
df_train.dropna(inplace=True)
```

[ ]:
```
df_train = df_train.drop(columns=['geometry'])
df_train.head()
```

```
[ ]: df_train.shape
```

+ Code + Markdown

```
[ ]: f = plt.figure(figsize=(15, 12))
plt.matshow(df_train.corr(), fignum=f.number)
plt.xticks(range(df_train.select_dtypes(['number']).shape[1]), df_train.select_dtypes(['number']).columns, fontsize=14, rotation=45)
plt.yticks(range(df_train.select_dtypes(['number']).shape[1]), df_train.select_dtypes(['number']).columns, fontsize=14)
cb = plt.colorbar()
cb.ax.tick_params(labelsize=14)
plt.title('Correlation Matrix', fontsize=16);
```

```
[ ]: from sklearn.model_selection import train_test_split
X = df_train.drop(columns='Category_Label')
y = df_train['Category_Label']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, stratify = y)
```

```
[ ]: X_train.head()
```

```
[ ]: def model_result(model):
    train_acc = model.score(X_train, y_train)
    train_loss = log_loss(y_train, model.predict_proba(X_train))
    test_acc = model.score(X_test, y_test)
    test_loss = log_loss(y_test, model.predict_proba(X_test))
    print(f'{model} train_score = {train_acc}')
    print(f'{model} log_loss = {train_loss}')
    print(f'{model} test_score = {test_acc}')
    print(f'{model} test_log_loss = {test_loss}')
    return print()
```

```
[ ]: !wget https://data.sfgov.org/api/views/a2rp-pwkh/rows.csv?accessType=DOWNLOAD -O LandUse.csv
```

[ ]:
```
land_use = pd.read_csv('./LandUse.csv')
land_use.dropna(inplace=True)
mask = (land_use['LANDUSE'] == 'MISSING DATA') | (land_use['LANDUSE'] == 'Right of Way')
lu = land_use[~mask]
land_cat = lu['LANDUSE'].unique()
print('Plots of land in San Francisco are divided into the following categories:\n', land_cat)
```

+ Code + Markdown

[ ]:
```
mp_data = lu['the_geom']
mp_geo = mp_data.apply(wkt.loads)
centroids = gpd.GeoSeries(mp_geo).centroid
coords = list(zip(centroids.x, centroids.y))
```

[ ]:
```
lu['LANDUSE'].shape
```

[ ]:
```
lu_df = pd.DataFrame(coords, columns=['Land_X', 'Land_Y'])
lu_df = pd.concat([lu_df, lu['LANDUSE']], axis=1)
lu_df.dropna(inplace=True)
lu_df.head()
```

[ ]:
```
from scipy.spatial import cKDTree
def nearest_land(df1, df2, land_coords_cols=['Land_X', 'Land_Y'],
                 coords_cols=['X', 'Y'], land_use_col='LANDUSE'):
    df1_sub = df1[land_coords_cols]
    df2_sub = df2[coords_cols]
    tree = cKDTree(df1_sub)
    distances, indices = tree.query(df2_sub)
    nearest_neighbors = df2.iloc[indices, :].copy(deep=True)
    land_types = df1.iloc[indices].copy(deep=True)
    land_types = land_types.reset_index(drop=True)
    df2 = df2.reset_index(drop=True)
    print(df2.shape)
    print(land_types.shape)
    df2_new = pd.concat([df2, land_types], axis=1)
    land_dum = pd.get_dummies(df2_new[land_use_col])
    df2_land = pd.concat([(df2_new.drop(columns=[land_use_col])), land_dum], axis=1)
    print(df2_land.shape)
    return df2_land
```

[ ]:
```
df_train_land = nearest_land(lu_df, df_train)
```

[ ]:
```
# LightGBM
import lightgbm as lgb
clf = lgb.LGBMClassifier()
```

4. Applying the LightGBM (Gradient Boosting Machine) algorithm

```
[ ]:
# LightGBM
import lightgbm as lgb
clf = lgb.LGBMClassifier()
```

+ Code + Markdown

```
[ ]:
#  learning LightGBM:
clf.fit(X_train, y_train)
print(model_result(clf))
```

```
[ ]:
# metrics on task classification
from sklearn.metrics import precision_score, recall_score, accuracy_score
```

```
[ ]:
print("TRAIN")
y_pred = clf.predict(X_train)
print(f"accuracy:  {accuracy_score(y_train, y_pred)}")
print(f"precision: {precision_score(y_train, y_pred, average=None)}")
print(f"recall:    {recall_score(y_train, y_pred, average=None)}")
print("TEST")
y_pred = clf.predict(X_test)
print(f"accuracy:  {accuracy_score(y_test, y_pred)}")
print(f"precision: {precision_score(y_test, y_pred, average=None)}")
print(f"recall:    {recall_score(y_test, y_pred, average=None)}")
```

```
[ ]:
# Let's look at the importance of features (feature_importances_) in the classification model and analyze the top features
sorted(zip(X_train.columns, clf.feature_importances_), key=lambda x: -x[1])
```

```
[ ]:
def feature_engineering(data):
    data['Date'] = pd.to_datetime(data['Dates'].dt.date)
    data['n_days'] = (
        data['Date'] - data['Date'].min()).apply(lambda x: x.days)
    data['Day'] = data['Dates'].dt.day
    data['DayOfWeek'] = data['Dates'].dt.weekday
    data['Month'] = data['Dates'].dt.month
    data['Year'] = data['Dates'].dt.year
    data['Hour'] = data['Dates'].dt.hour
    data['Minute'] = data['Dates'].dt.minute
    data.drop(columns=['Dates','Date','Address'], inplace=True)
    return data
```

5. Buiding the model:

[ ]:
```
!pip uninstall lightgbm -y
!pip install lightgbm==2.2.3
```

[ ]:
```
import lightgbm as lgb
lgb.__version__
```

[ ]:
```
# Upload the required files:
train = pd.read_csv('./data/train.csv', parse_dates=['Dates'])
test = pd.read_csv('./data/test.csv', parse_dates=['Dates'], index_col='Id')
# We clean the data:
train.drop_duplicates(inplace=True)
train.drop(train[(train.Y > 50)].index)
test.drop(test[(test.Y > 50)].index)
# Let's use our service function to add date columns and new features
train = feature_engineering(train)
train.drop(columns=['Descript','Resolution'], inplace=True)
test = feature_engineering(test)
# We code categorical columns:
le1 = LabelEncoder()
train['PdDistrict'] = le1.fit_transform(train['PdDistrict'])
test['PdDistrict'] = le1.transform(test['PdDistrict'])
le2 = LabelEncoder()
X = train.drop(columns=['Category'])
y= le2.fit_transform(train['Category'])
```

[ ]:
```
train.head()
```

[ ]:
```
df_train = nearest_land(lu_df, train)
df_train.head()
```

[ ]:
```
# Let's split the data on the sign and the target
X = df_train.drop(columns='Category')
y = df_train['Category']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, stratify = y)
```

[ ]:
```
print(X_train.shape, y_train.shape)
print(X_test.shape, y_test.shape)
```

6. Evaluating the train and test data:

```
[ ]: booster1 = lgb.LGBMClassifier(objective='multiclass', n_estimators=100, num_class=39,
                                   max_delta_step=0.9,
                                   min_data_in_leaf=21,
                                   learning_rate=0.4,
                                  )
     booster1.fit(X_train, y_train, eval_set=[(X_test, y_test),], )
     print('Test  Accuracy: %.2f'%booster1.score(X_test, y_test))
     print('Train Accuracy: %.2f'%booster1.score(X_train, y_train))
```

+ Code + Markdown

```
[ ]: import joblib
     # Saving model
     !wget https://www.dropbox.com/s/jqfncd3z33xdvei/lgb_fin.pkl?dl=0 -O lgb_fin.pkl
     bst = joblib.load('lgb_fin.pkl')
```

```
[ ]: print('Importance of features in LGBM:')
     sorted(zip(X_train.columns, booster1.feature_importances_), key=lambda x: -x[1])
```

```
[ ]: # Comment to download file model
     booster1 = bst
```

```
[ ]: from sklearn.metrics import RocCurveDisplay
     from sklearn.metrics import roc_auc_score
     y_score = booster1.predict_proba(X_test)
     macro_roc_auc_ovr = roc_auc_score(
         y_test,
         y_score,
         multi_class="ovr",
         average="macro",
     )
```

```
[ ]: macro_roc_auc_ovr
```

```
[ ]: import matplotlib.pyplot as plt
     from sklearn.metrics import RocCurveDisplay
```

```
[ ]: from itertools import combinations
     pair_list = list(combinations(np.unique(y), 2))
```

7.Applying and visualizing the ROC curve which compares each crime:

[ ]:
```
from sklearn.preprocessing import LabelBinarizer
label_binarizer = LabelBinarizer().fit(y_train)
y_onehot_test = label_binarizer.transform(y_test)
y_onehot_test.shape  # (n_samples, n_classes)
```

[ ]:
```
from sklearn.metrics import roc_auc_score
```

[ ]:
```
from sklearn.metrics import roc_curve, auc
fpr, tpr, roc_auc = dict(), dict(), dict()
# Calculating micro curve ROC and ROC area
fpr["micro"], tpr["micro"], _ = roc_curve(y_onehot_test.ravel(), y_score.ravel())
roc_auc["micro"] = auc(fpr["micro"], tpr["micro"])
print(f"Micro-averaged One-vs-Rest ROC AUC score:\n{roc_auc['micro']:.2f}")
```

[ ]:
```
for i in range(39):
    fpr[i], tpr[i], _ = roc_curve(y_onehot_test[:, i], y_score[:, i])
    roc_auc[i] = auc(fpr[i], tpr[i])
fpr_grid = np.linspace(0.0, 1.0, 1000)
# Interpolation of all ROC curves at these points
mean_tpr = np.zeros_like(fpr_grid)
for i in range(39):
    mean_tpr += np.interp(fpr_grid, fpr[i], tpr[i])
# AUC is averaged and calculated
mean_tpr /= 39
fpr["macro"] = fpr_grid
tpr["macro"] = mean_tpr
roc_auc["macro"] = auc(fpr["macro"], tpr["macro"])
print(f"Macro-averaged One-vs-Rest ROC AUC score:\n{roc_auc['macro']:.2f}")
```

[ ]:
```
macro_roc_auc_ovr = roc_auc_score(
    y_test,
    y_score,
    multi_class="ovr",
    average="macro",
)

print(f"Macro-averaged One-vs-Rest ROC AUC score:\n{macro_roc_auc_ovr:.2f}")
```

[ ]:

```
pair_scores = []
mean_tpr = dict()
for ix, (label_a, label_b) in enumerate(pair_list):
    a_mask = y_test == label_a
    b_mask = y_test == label_b
    ab_mask = np.logical_or(a_mask, b_mask)
    a_true = a_mask[ab_mask]
    b_true = b_mask[ab_mask]
    idx_a = np.flatnonzero(label_binarizer.classes_ == label_a)[0]
    idx_b = np.flatnonzero(label_binarizer.classes_ == label_b)[0]
    fpr_a, tpr_a, _ = roc_curve(a_true, y_score[ab_mask, idx_a])
    fpr_b, tpr_b, _ = roc_curve(b_true, y_score[ab_mask, idx_b])
    mean_tpr[ix] = np.zeros_like(fpr_grid)
    mean_tpr[ix] += np.interp(fpr_grid, fpr_a, tpr_a)
    mean_tpr[ix] += np.interp(fpr_grid, fpr_b, tpr_b)
    mean_tpr[ix] /= 2
    mean_score = auc(fpr_grid, mean_tpr[ix])
    pair_scores.append(mean_score)
    fig, ax = plt.subplots(figsize=(6, 6))
    plt.plot(
        fpr_grid,
        mean_tpr[ix],
        label=f"Mean {label_a} vs {label_b} (AUC = {mean_score :.2f})",
        linestyle=":",
        linewidth=4,
    )
    RocCurveDisplay.from_predictions(
        a_true,
        y_score[ab_mask, idx_a],
        ax=ax,
        name=f"{label_a} as positive class",
    )
    RocCurveDisplay.from_predictions(
        b_true,
        y_score[ab_mask, idx_b],
        ax=ax,
        name=f"{label_b} as positive class",
    )
    plt.plot([0, 1], [0, 1], "k--", label="chance level (AUC = 0.5)")
    plt.axis("square")
    plt.xlabel("False Positive Rate")
    plt.ylabel("True Positive Rate")
    plt.title(f"{label_a} vs {label_b} ROC curves")
    plt.legend()
    plt.show()
print(f"Macro-averaged One-vs-One ROC AUC score:\n{np.average(pair_scores):.2f}")
```

[ ]:

```
from sklearn.metrics import classification_report
# Get the report of classification transitions in classification_report Actual tags and predicted
print("TEST")
print(classification_report(y_test, booster1.predict(X_test)))
```

[ ]:

```
print("TRAIN")
print(classification_report(y_train, booster1.predict(X_train)))
```

8. Saving the model:

[ ]:
```
# Convert the format of the data in the test data set to the format of the model
le = LabelEncoder()
test = nearest_land(lu_df, test)
test.info()
```

+ Code + Markdown

[ ]:
```
X_train.columns.sort_values()
```

[ ]:
```
test.columns.sort_values()
```

[ ]:
```
X_train.head()
```

[ ]:
```
# Let's convert the order of the columns in the test data to the order of the trained model

cols = list(X_train.columns)
test = pd.DataFrame(test, columns=cols)
test.head()
```

[ ]:
```
print(X_train.shape, test.shape)
```

[ ]:
```
# We get the meaning of predicted classes
preds_class = booster1.predict(test)
```

[ ]:
```
# We get the predicted probabilities — the probability for each class
preds_proba = booster1.predict_proba(test)
```

[ ]:
```
preds_proba.shape
```

[ ]:
```
# Store the test dataset in the variable submission
submission = pd.DataFrame(
    preds_proba,
    columns = le2.inverse_transform(np.linspace(0, 38, 39, dtype='int16')),
    index = test.index)
submission.head()
```

```
# Save the test data set in the file "lgb_model_fin.csv"
submission.to_csv(
    'lgb_model_fin.csv', index_label='Id')
```

9.TESTING

Unit Testing

Test Case Description	Expected Outcome	Actual Outcome	Pass/Fail
Test loading dataset from file.	Loaded dataset matches original dataset file	Loaded dataset matches original dataset file	Pass
Test splitting dataset into training and testing sets	Training set and testing set have expected sizes	Training set and testing set have expected sizes	Pass
Test feature selection	Selected features have expected correlation with software defectiveness	Selected features have expected correlation with software defectiveness	Pass
Test algorithm training	Trained algorithm has expected accuracy, precision, recall, and F-measure on training set	Trained algorithm has expected accuracy, precision, recall, and F-measure on training set	Pass
Test algorithm testing	Tested algorithm has expected accuracy, precision, recall, and F-measure on testing set.	Tested algorithm has expected accuracy, precision, recall, and F-measure on testing set	Pass

10. SCREENSHOTS

The datasets that used in our project:

	Id	Dates	DayOfWeek	PdDistrict	Address	X	Y
0	0	2015-05-10 23:59:00	Sunday	BAYVIEW	2000 Block of THOMAS AV	-122.399588	37.735051
1	1	2015-05-10 23:51:00	Sunday	BAYVIEW	3RD ST / REVERE AV	-122.391523	37.732432
2	2	2015-05-10 23:50:00	Sunday	NORTHERN	2000 Block of GOUGH ST	-122.426002	37.792212
3	3	2015-05-10 23:45:00	Sunday	INGLESIDE	4700 Block of MISSION ST	-122.437394	37.721412
4	4	2015-05-10 23:45:00	Sunday	INGLESIDE	4700 Block of MISSION ST	-122.437394	37.721412

	Dates	Category	Descript	DayOfWeek	PdDistrict	Resolution	Address	X	Y
0	2015-05-13 23:53:00	WARRANTS	WARRANT ARREST	Wednesday	NORTHERN	ARREST, BOOKED	OAK ST / LAGUNA ST	-122.425892	37.774599
1	2015-05-13 23:53:00	OTHER OFFENSES	TRAFFIC VIOLATION ARREST	Wednesday	NORTHERN	ARREST, BOOKED	OAK ST / LAGUNA ST	-122.425892	37.774599
2	2015-05-13 23:33:00	OTHER OFFENSES	TRAFFIC VIOLATION ARREST	Wednesday	NORTHERN	ARREST, BOOKED	VANNESS AV / GREENWICH ST	-122.424363	37.800414
3	2015-05-13 23:30:00	LARCENY/THEFT	GRAND THEFT FROM LOCKED AUTO	Wednesday	NORTHERN	NONE	1500 Block of LOMBARD ST	-122.426995	37.800873
4	2015-05-13 23:30:00	LARCENY/THEFT	GRAND THEFT FROM LOCKED AUTO	Wednesday	PARK	NONE	100 Block of BRODERICK ST	-122.438738	37.771541

Visualization:

World plot showing where the crimes have been committed:

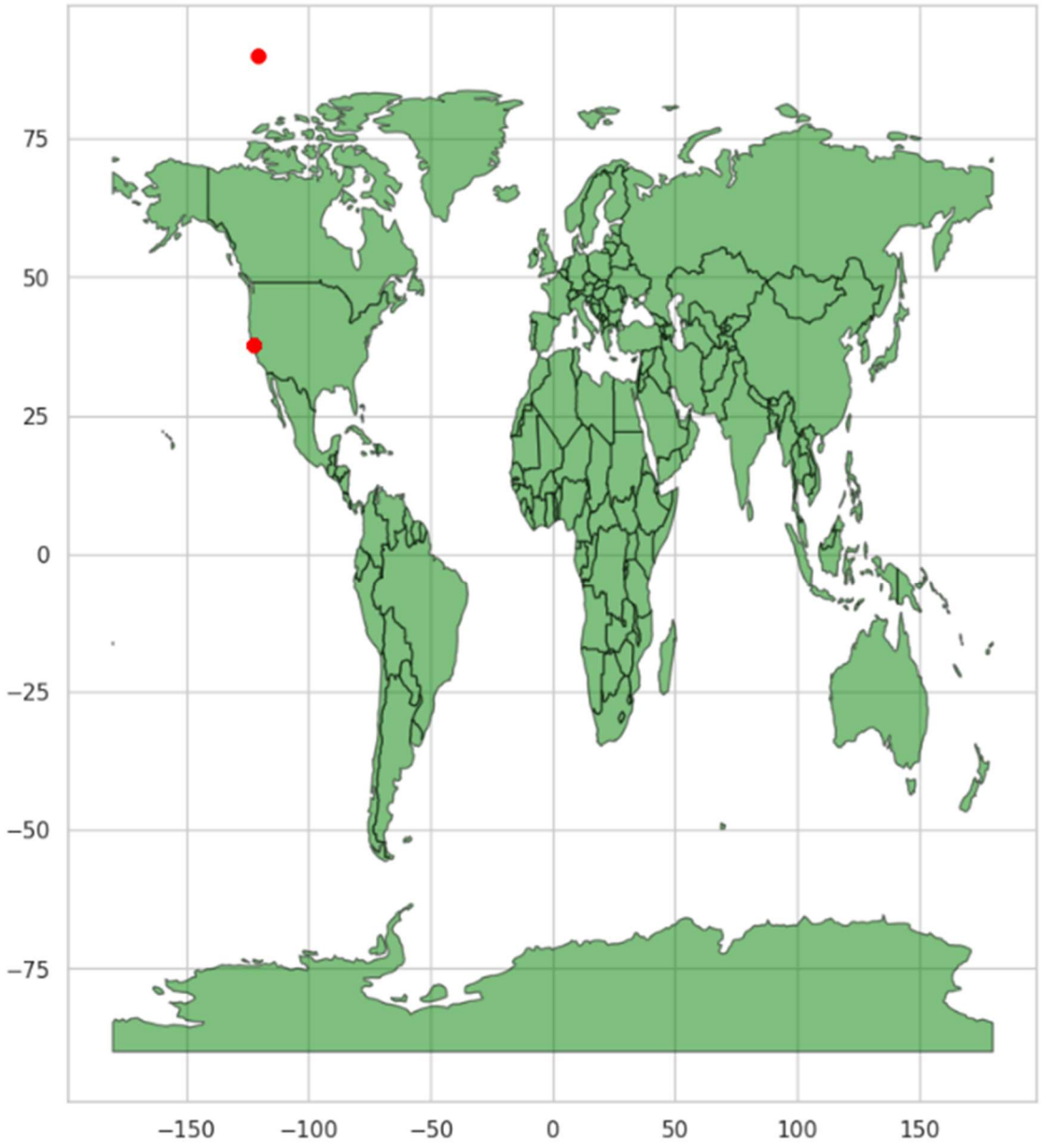


Fig-9: World Plot

KDE plot showing normal distribution for number of crimes per day:

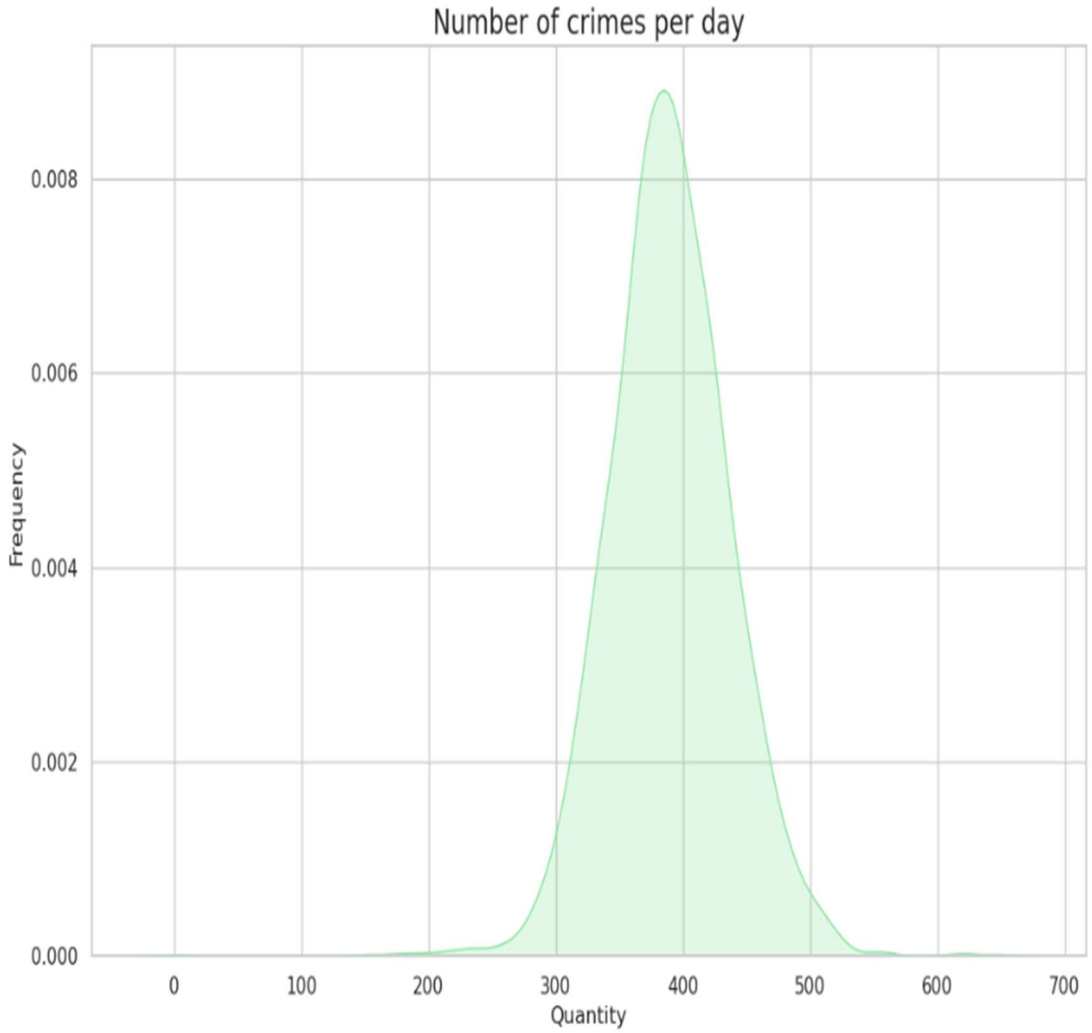


Fig-10: KDE Plot(1)

Bar plot showing most crimes committed on a particular day of the week:

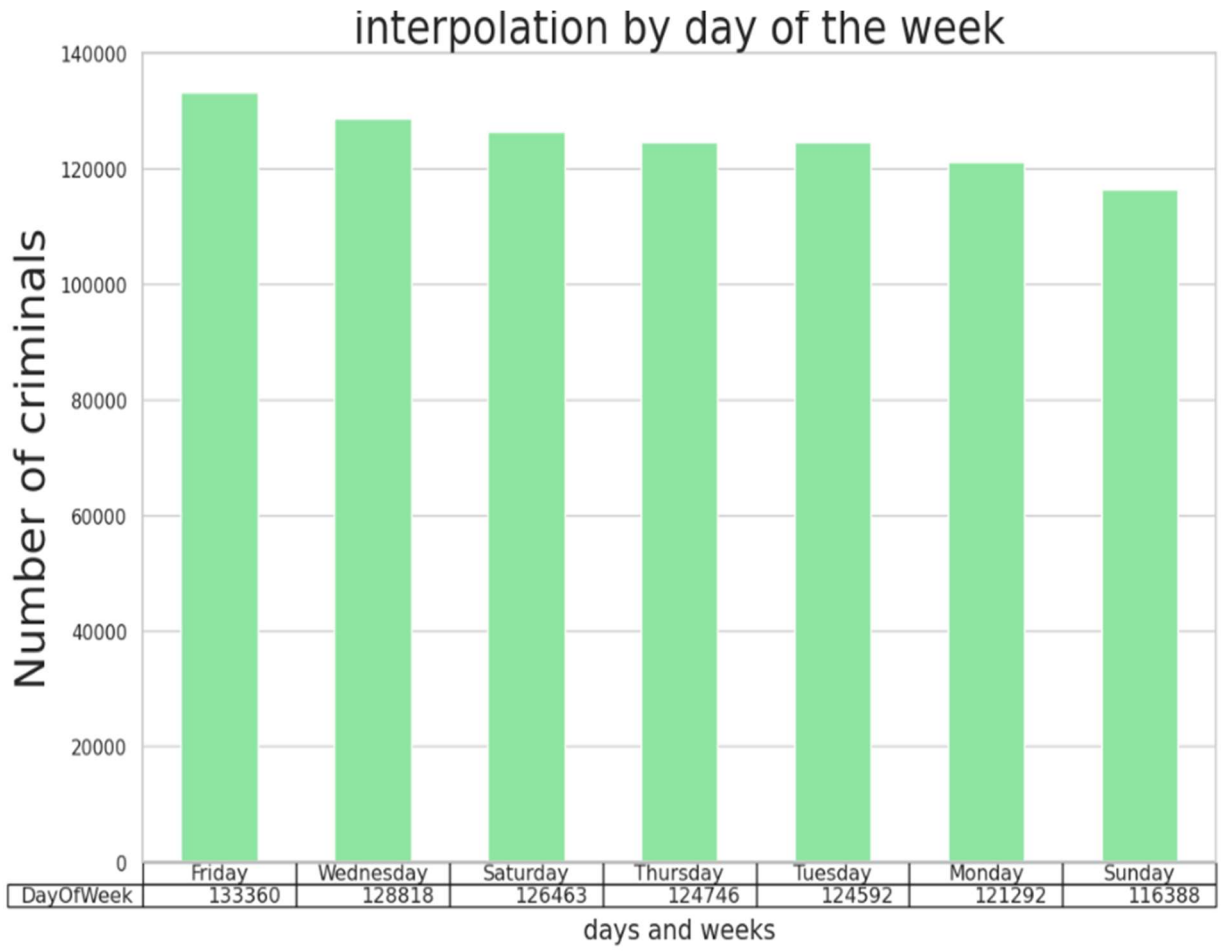


Fig-11: Bar Plot(1)

Bar plot showing number of crimes by district:

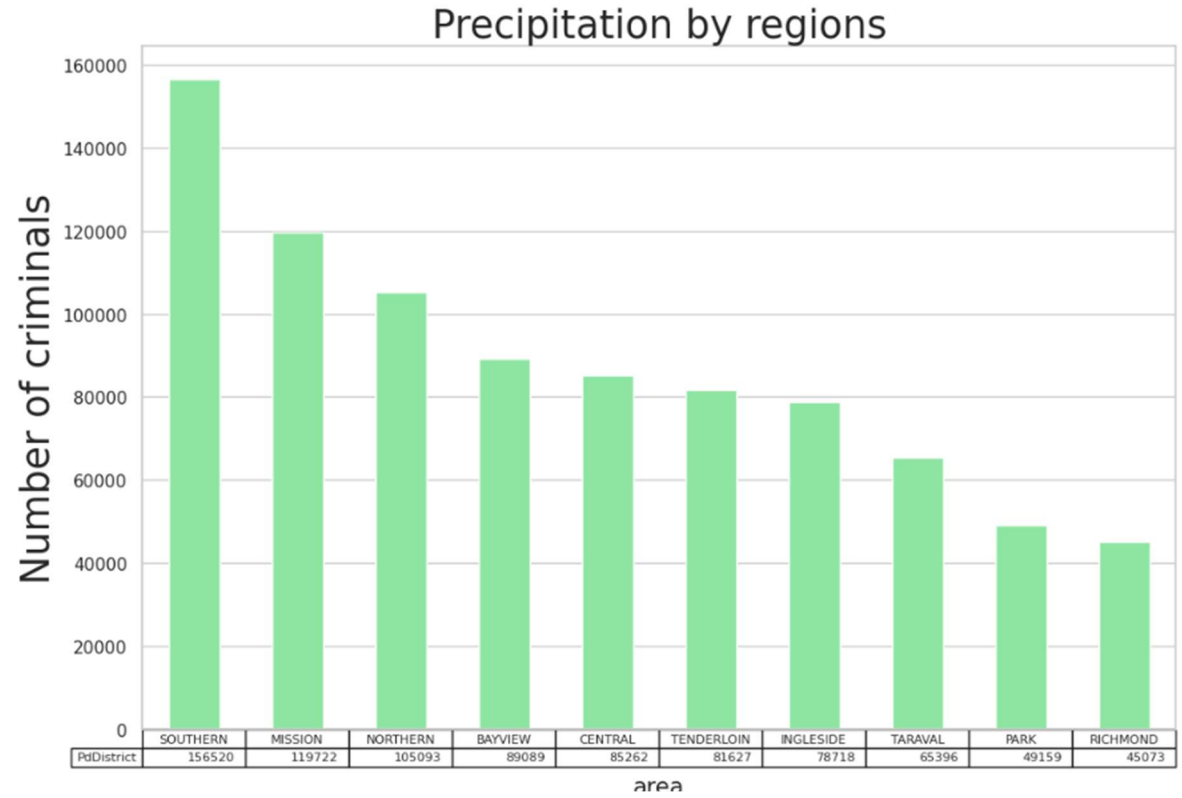


Fig-12: Bar Plot(2)

Horizontal bar plot showing which crime occur most often:

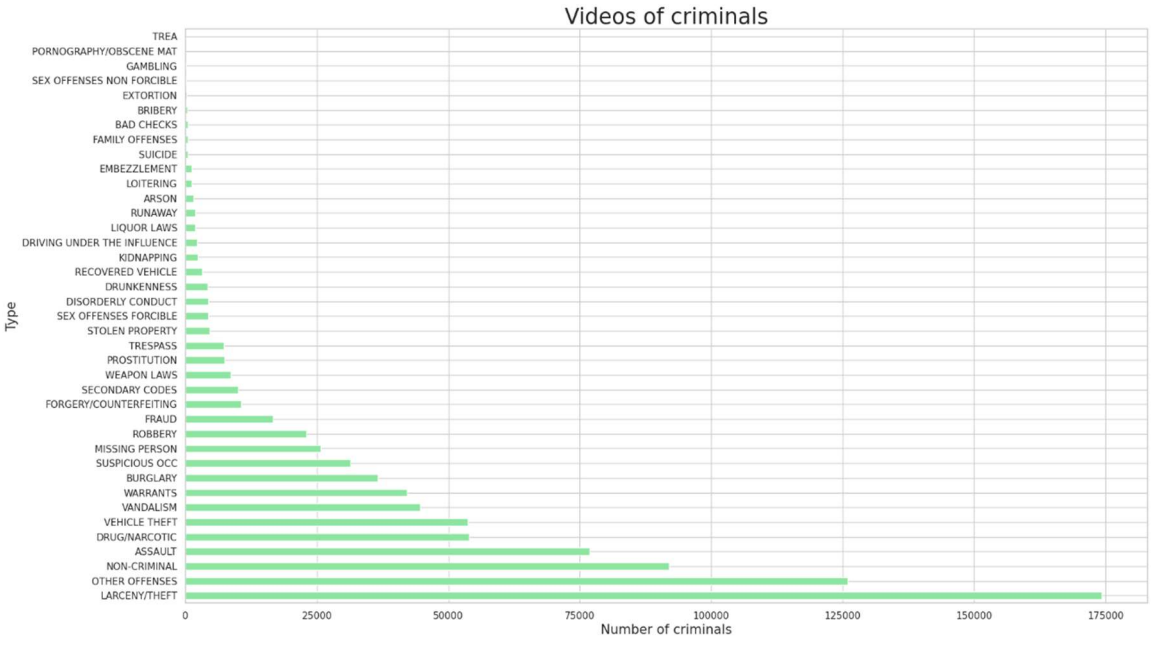


Fig-13: Horizontal Bar Plot

KDE plot showing total crimes per district:

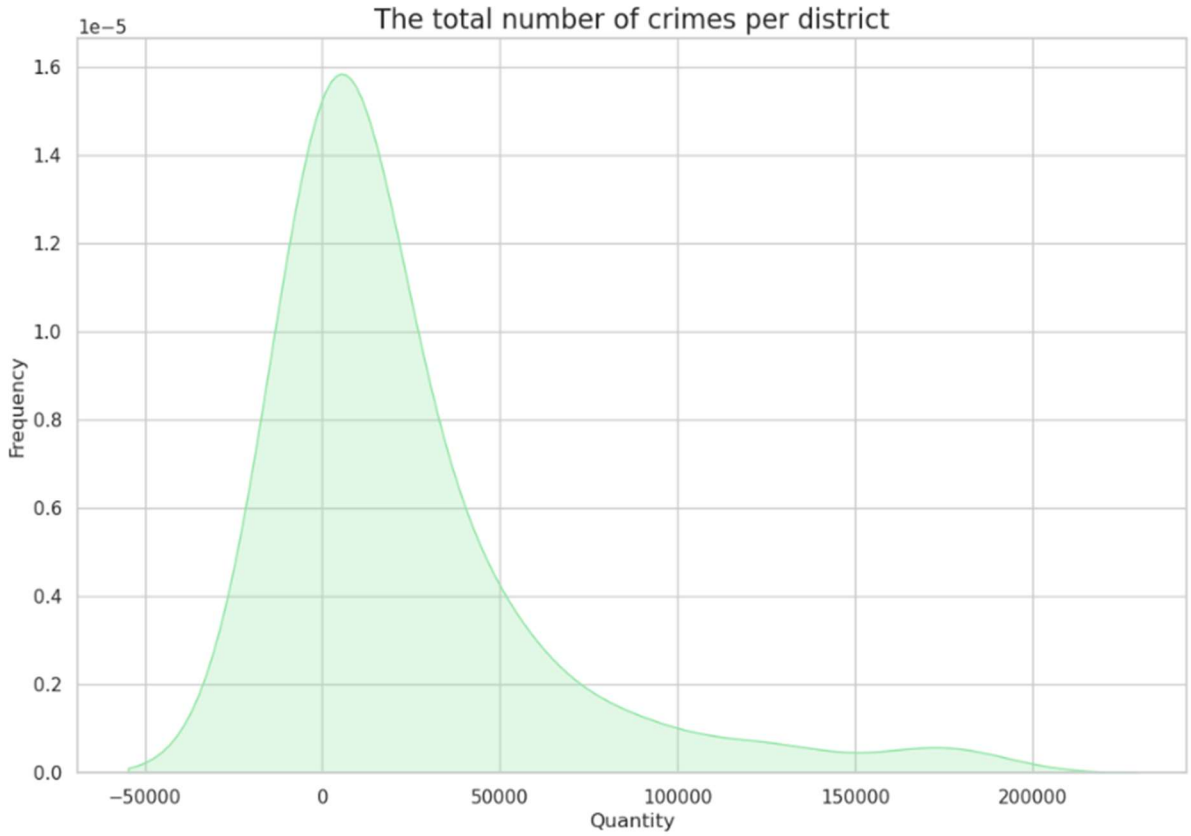


Fig-14: KDE Plot(2)

Converting shape file to geopandas geodataframe:

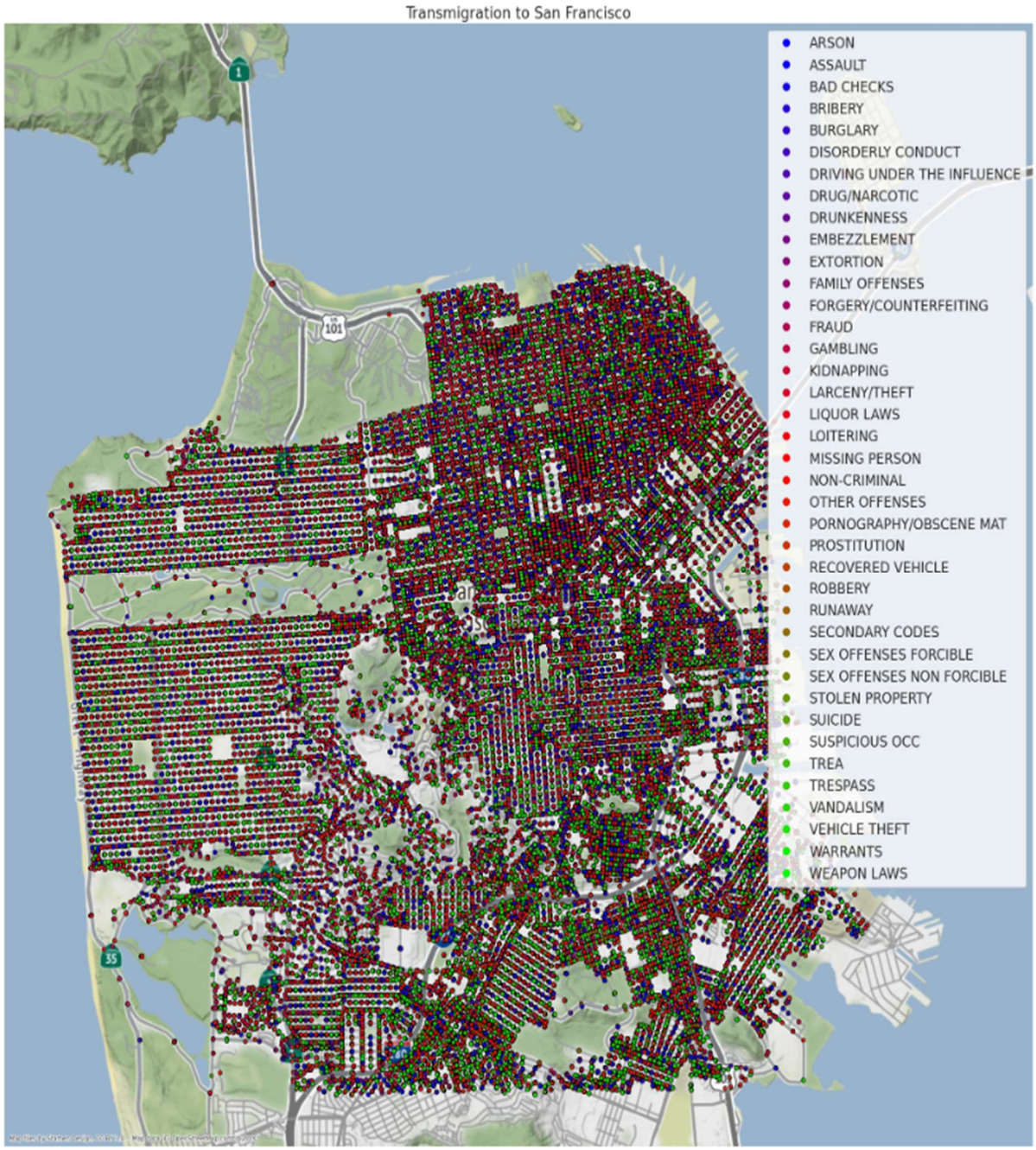


Fig-15: Geopandas Geodataframe

Correlation matrix:

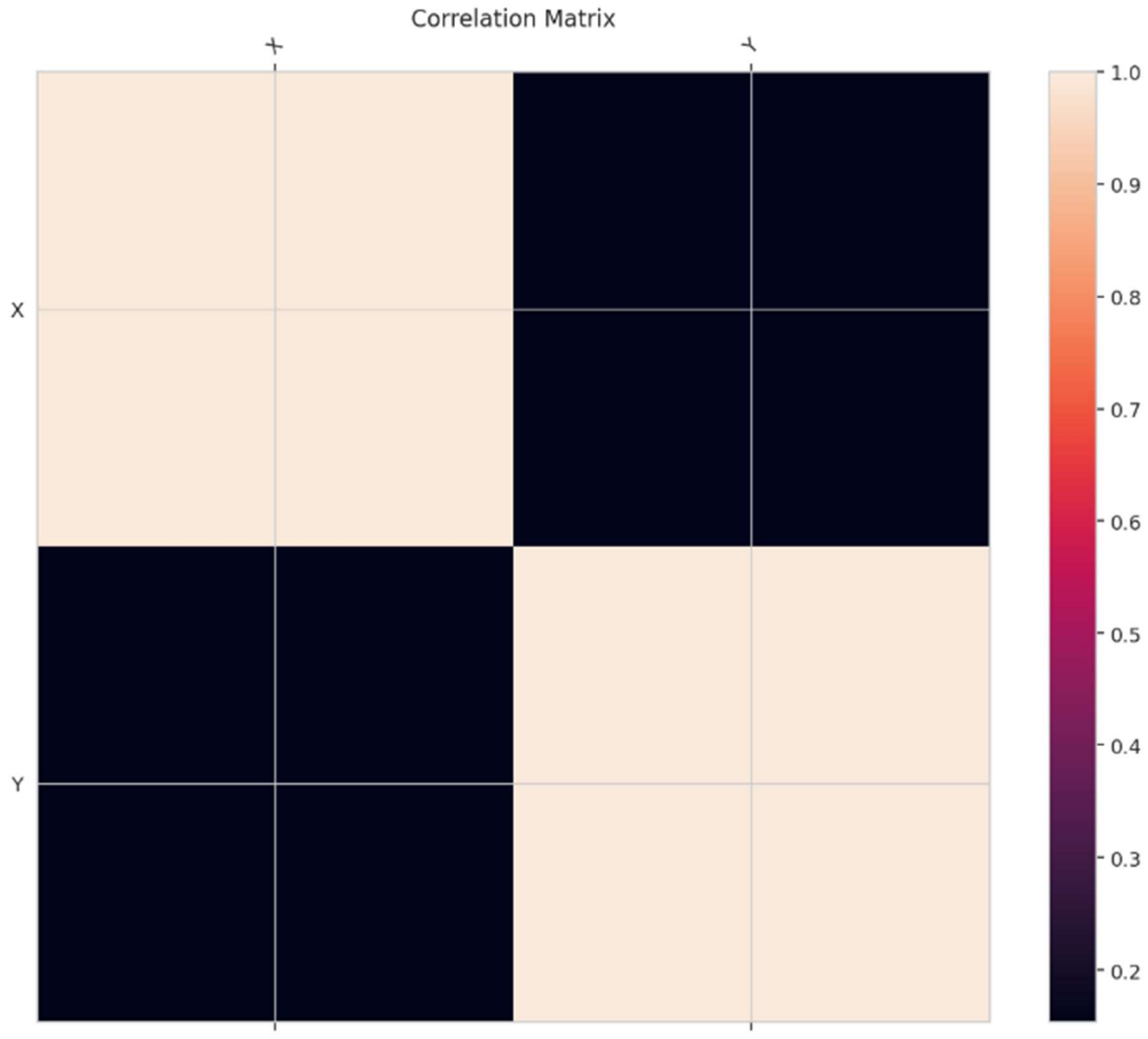


Fig-16: Correlation Matrix(1)

Pairplot:

Fig-17: Pairplot

Donut chart:

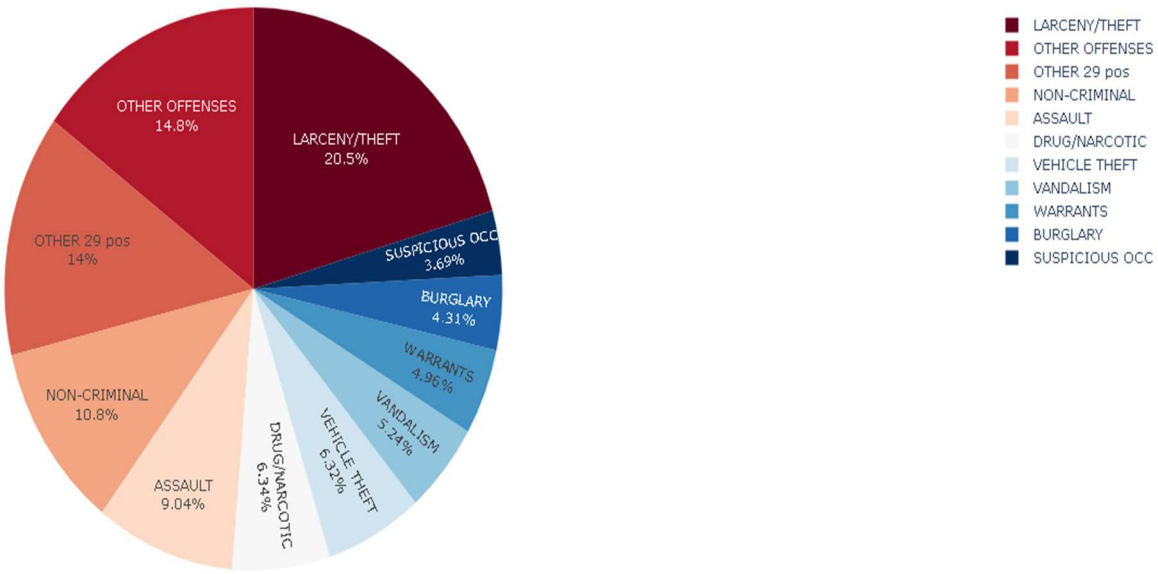


Fig-18: Donut Chart

Line plot showing different categories occurred in different time intervals:

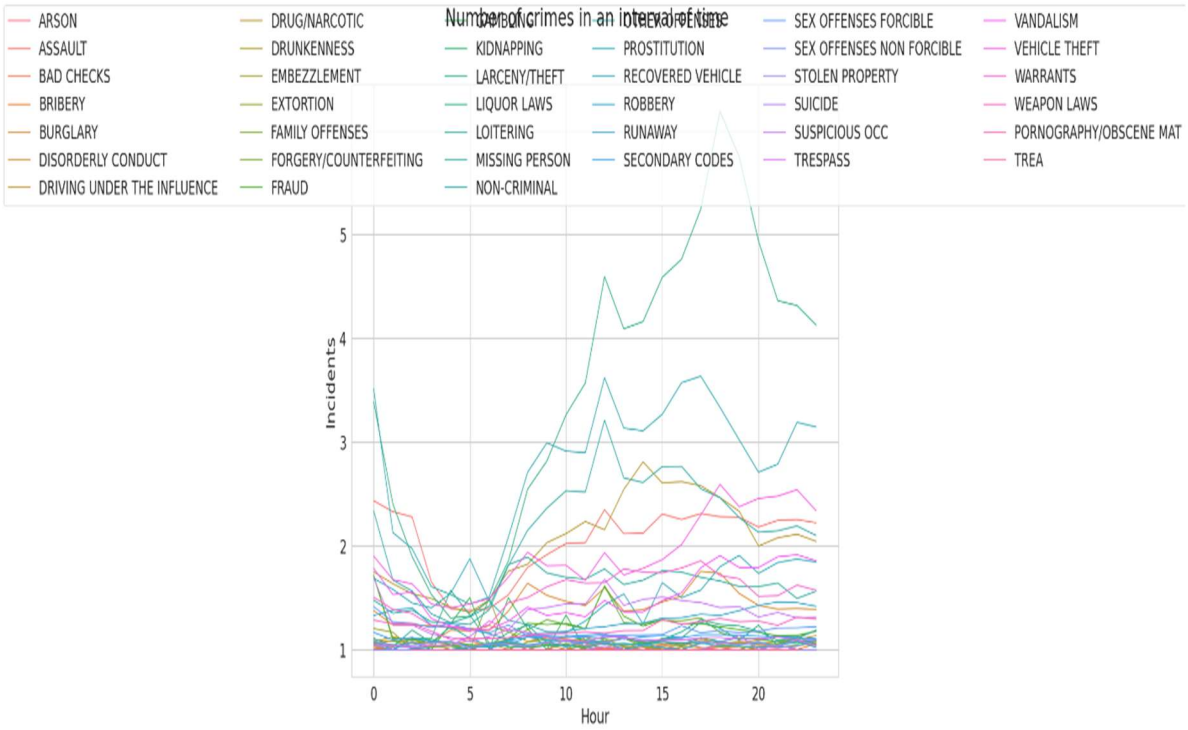


Fig-19: Line Plot

Correlation matrix:

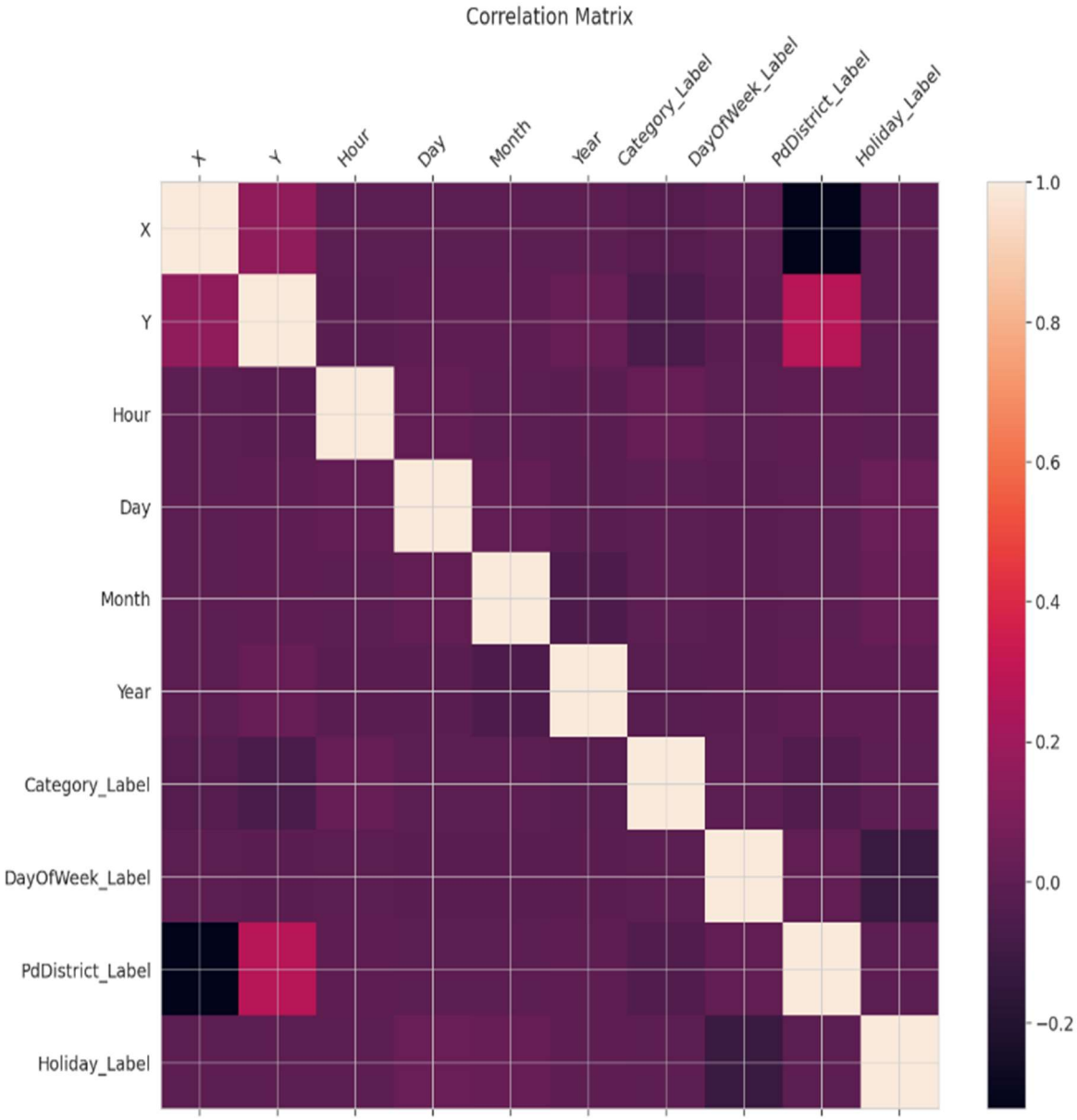


Fig-20: Correlation Matrix(2)

ROC curve comparing each criminal category:

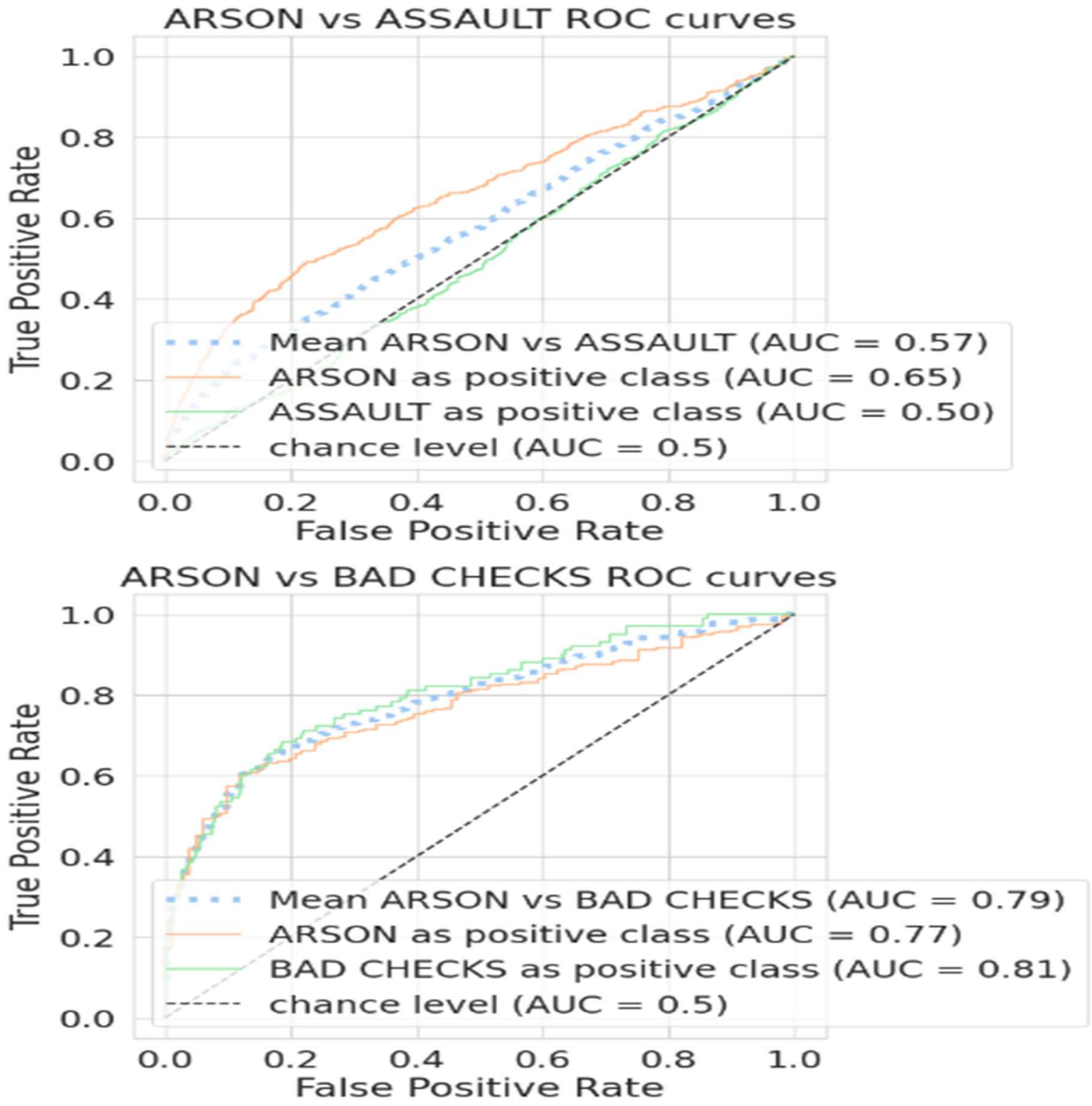


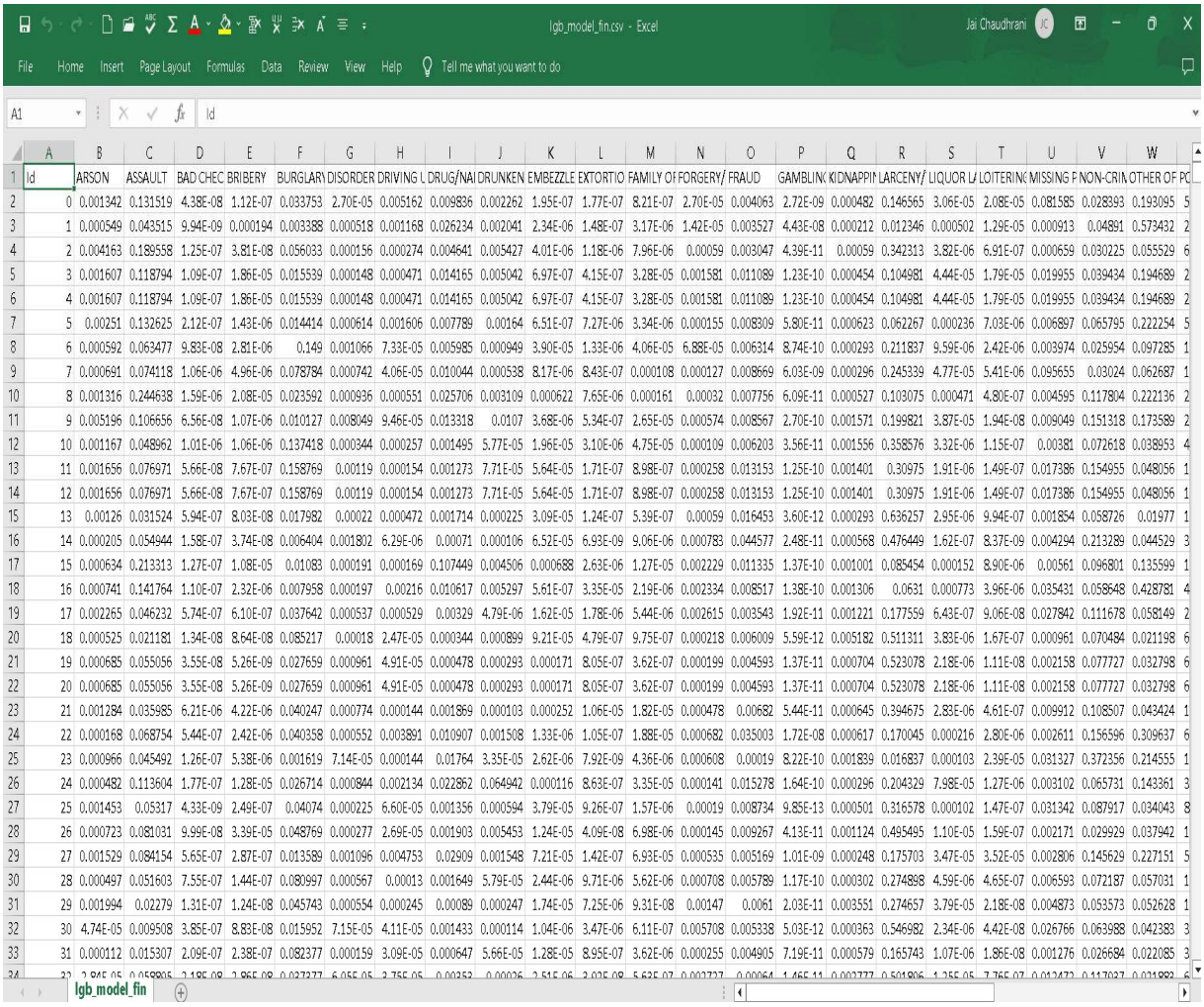
Fig-21: ROC Curve comparison

ARSON	ASSAULT	BAD CHECKS	BRIBERY	BURGLARY	DISORDERLY CONDUCT	DRIVING UNDER THE INFLUENCE	DRUG/NARCOTIC	DRUNKENNESS	E
0.000139	0.163422	2.759362e-10	0.000002	0.011857	0.000415	0.024503	0.014112	0.003217	
0.000309	0.080713	1.424533e-09	0.000001	0.001055	0.000958	0.010187	0.022690	0.004404	
0.002395	0.142848	1.602038e-07	0.000002	0.015520	0.000219	0.003255	0.032587	0.011640	
0.000140	0.153123	1.891387e-07	0.000002	0.008314	0.000381	0.007258	0.008025	0.019423	
0.000140	0.153123	1.891387e-07	0.000002	0.008314	0.000381	0.007258	0.008025	0.019423	

ows × 39 columns

Fig-22: Accuracy visualization

Final Output:



lgb_model_fin.csv - Excel

Id	ARSON	ASSAULT	BAD CHEC	BRIBERY	BURGLARY	DISORDER	DRIVING L	DRUG/NAI	DRUNKEN	EMBEZZLE	EXTORTIO	FAMILY OF	FORGERY/	FRAUD	GAMBLIN	KIDNAPPI	LARCENY/	LIQUOR LA	LOITERIN	MISSING P	NON-CRIM	OTHER OF	PO
0	0.001342	0.131519	4.38E-08	1.12E-07	0.033753	2.70E-05	0.005162	0.009836	0.002262	1.95E-07	1.77E-07	8.21E-07	2.70E-05	0.004063	2.72E-09	0.000482	0.146565	3.06E-05	2.08E-05	0.081585	0.028393	0.193095	5
1	0.000549	0.043515	9.94E-09	0.000194	0.003388	0.000518	0.001168	0.026234	0.002041	2.34E-06	1.48E-07	3.17E-06	1.42E-05	0.003527	4.43E-08	0.000212	0.012346	0.000502	1.29E-05	0.000913	0.04891	0.573432	2
2	0.004163	0.189558	1.25E-07	3.81E-08	0.056033	0.000156	0.000274	0.004641	0.005427	4.01E-06	1.18E-06	7.96E-06	0.00059	0.003047	4.39E-11	0.00059	0.342313	3.82E-06	6.91E-07	0.000659	0.030225	0.055529	6
3	0.001607	0.118794	1.09E-07	1.86E-05	0.015539	0.000148	0.000471	0.014165	0.005042	6.97E-07	4.15E-07	3.28E-05	0.001581	0.011089	1.23E-10	0.000454	0.104981	4.44E-05	1.79E-05	0.019955	0.039434	0.194689	2
4	0.001607	0.118794	1.09E-07	1.86E-05	0.015539	0.000148	0.000471	0.014165	0.005042	6.97E-07	4.15E-07	3.28E-05	0.001581	0.011089	1.23E-10	0.000454	0.104981	4.44E-05	1.79E-05	0.019955	0.039434	0.194689	2
5	0.00251	0.132625	2.12E-07	1.43E-06	0.014414	0.000614	0.001606	0.007789	0.00164	6.51E-07	7.27E-06	3.34E-06	0.000155	0.008309	5.80E-11	0.000623	0.062267	0.000236	7.03E-06	0.006897	0.065795	0.222254	5
6	0.000592	0.063477	9.83E-08	2.81E-06	0.149	0.001066	7.33E-05	0.005985	0.000949	3.90E-05	1.33E-06	4.06E-05	6.88E-05	0.006314	8.74E-10	0.000293	0.211837	9.59E-06	2.42E-06	0.003974	0.025954	0.097285	1
7	0.000691	0.074118	1.06E-06	4.96E-06	0.078784	0.000742	4.06E-05	0.010044	0.000538	8.17E-06	8.43E-07	0.000108	0.000127	0.008669	6.03E-09	0.000296	0.245339	4.77E-05	5.41E-06	0.095655	0.03024	0.062687	1
8	0.001316	0.244638	1.59E-06	2.08E-05	0.023592	0.000936	0.000551	0.025706	0.003109	0.000622	7.65E-06	0.000161	0.00032	0.007756	6.09E-11	0.000527	0.103075	0.000471	4.80E-07	0.004595	0.117804	0.222136	2
9	0.005196	0.106656	6.56E-08	1.07E-06	0.010127	0.008049	9.46E-05	0.013318	0.0107	3.68E-06	5.34E-07	2.65E-05	0.000574	0.008567	2.70E-10	0.001571	0.199821	3.87E-05	1.94E-08	0.009049	0.151318	0.173589	2
10	0.001167	0.048962	1.01E-06	1.06E-06	0.137418	0.000344	0.000257	0.001495	5.77E-05	1.96E-05	3.10E-06	4.75E-05	0.000109	0.006203	3.56E-11	0.001556	0.358576	3.32E-06	1.15E-07	0.00381	0.072618	0.038953	4
11	0.001656	0.076971	5.66E-08	7.67E-07	0.158769	0.00119	0.000154	0.001273	7.71E-05	5.64E-05	1.71E-07	8.98E-07	0.000258	0.013153	1.25E-10	0.001401	0.30975	1.91E-06	1.49E-07	0.017386	0.154955	0.048056	1
12	0.001656	0.076971	5.66E-08	7.67E-07	0.158769	0.00119	0.000154	0.001273	7.71E-05	5.64E-05	1.71E-07	8.98E-07	0.000258	0.013153	1.25E-10	0.001401	0.30975	1.91E-06	1.49E-07	0.017386	0.154955	0.048056	1
13	0.00126	0.031524	5.94E-07	8.03E-08	0.017982	0.00022	0.000472	0.001714	0.000225	3.09E-05	1.24E-07	5.39E-07	0.00059	0.016453	3.60E-12	0.000293	0.636257	2.95E-06	9.94E-07	0.001854	0.058726	0.01977	1
14	0.000205	0.054944	1.58E-07	3.74E-08	0.006404	0.001802	6.29E-06	0.00071	0.000106	6.52E-05	6.93E-09	9.06E-06	0.000783	0.044577	2.48E-11	0.000568	0.476449	1.62E-07	8.37E-09	0.004294	0.213289	0.044529	3
15	0.000634	0.213313	1.27E-07	1.08E-05	0.01083	0.000191	0.000169	0.107449	0.004506	0.000688	2.63E-06	1.27E-05	0.002229	0.011335	1.37E-10	0.001001	0.085454	0.000152	8.90E-06	0.00561	0.096801	0.135599	1
16	0.000741	0.141764	1.10E-07	2.32E-06	0.007958	0.000197	0.00216	0.010617	0.005297	5.61E-07	3.35E-05	2.19E-06	0.002334	0.008517	1.38E-10	0.001306	0.0631	0.000773	3.96E-06	0.035431	0.058648	0.428781	4
17	0.002265	0.046232	5.74E-07	6.10E-07	0.037642	0.000537	0.000529	0.00329	4.79E-06	1.62E-05	1.78E-06	5.44E-06	0.002615	0.003543	1.92E-11	0.001221	0.177559	6.43E-07	9.06E-08	0.027842	0.111678	0.058149	2
18	0.000525	0.021181	1.34E-08	8.64E-08	0.085217	0.00018	2.47E-05	0.000344	0.000899	9.21E-05	4.79E-07	9.75E-07	0.000218	0.006009	5.59E-12	0.005182	0.511311	3.83E-06	1.67E-07	0.000961	0.070484	0.021198	6
19	0.000685	0.055056	3.55E-08	5.26E-09	0.027659	0.000961	4.91E-05	0.000478	0.000293	0.000171	8.05E-07	3.62E-07	0.000199	0.004593	1.37E-11	0.000704	0.523078	2.18E-06	1.11E-08	0.002158	0.077727	0.032798	6
20	0.000685	0.055056	3.55E-08	5.26E-09	0.027659	0.000961	4.91E-05	0.000478	0.000293	0.000171	8.05E-07	3.62E-07	0.000199	0.004593	1.37E-11	0.000704	0.523078	2.18E-06	1.11E-08	0.002158	0.077727	0.032798	6
21	0.001284	0.035985	6.21E-06	4.22E-06	0.040247	0.000774	0.000144	0.001869	0.000103	0.000252	1.06E-05	1.82E-05	0.000478	0.00682	5.44E-11	0.000645	0.394675	2.83E-06	4.61E-07	0.009912	0.108507	0.043424	1
22	0.000168	0.068754	5.44E-07	2.42E-06	0.040358	0.000552	0.003891	0.010907	0.001508	1.33E-06	1.05E-07	1.88E-05	0.000682	0.035003	1.72E-08	0.000617	0.170045	0.000216	2.80E-06	0.002611	0.156596	0.309637	6
23	0.000966	0.045492	1.26E-07	5.38E-06	0.001619	7.14E-05	0.000144	0.01764	3.35E-05	2.62E-06	7.92E-09	4.36E-06	0.000608	0.00019	8.22E-10	0.001839	0.016837	0.000103	2.39E-05	0.031327	0.372356	0.214555	1
24	0.000482	0.113604	1.77E-07	1.28E-05	0.026714	0.000844	0.002134	0.022862	0.064942	0.000116	8.63E-07	3.35E-05	0.000141	0.015278	1.64E-10	0.000296	0.204329	7.98E-05	1.27E-06	0.003102	0.065731	0.143361	3
25	0.001453	0.05317	4.33E-09	2.49E-07	0.04074	0.000225	6.60E-05	0.001356	0.000594	3.79E-05	9.26E-07	1.57E-06	0.00019	0.008734	9.85E-13	0.000501	0.316578	0.000102	1.47E-07	0.031342	0.087917	0.034043	8
26	0.000723	0.081031	9.99E-08	3.39E-05	0.048769	0.000277	2.69E-05	0.001903	0.005453	1.24E-05	4.09E-08	6.98E-06	0.000145	0.009267	4.13E-11	0.001124	0.495495	1.10E-05	1.59E-07	0.002171	0.029929	0.037942	1
27	0.001529	0.084154	5.65E-07	2.87E-07	0.013589	0.001096	0.004753	0.02909	0.001548	7.21E-05	1.42E-07	6.93E-05	0.000535	0.005169	1.01E-09	0.000248	0.175703	3.47E-05	3.52E-05	0.002806	0.145629	0.227151	5
28	0.000497	0.051603	7.55E-07	1.44E-07	0.080997	0.000567	0.00013	0.001649	5.79E-05	2.44E-06	9.71E-06	5.62E-06	0.000708	0.005789	1.17E-10	0.000302	0.274898	4.59E-06	4.65E-07	0.006593	0.072187	0.057031	1
29	0.001994	0.02279	1.31E-07	1.24E-08	0.045743	0.000554	0.000245	0.00089	0.000247	1.74E-05	7.25E-06	9.31E-08	0.00147	0.0061	2.03E-11	0.003551	0.274657	3.79E-05	2.18E-08	0.004873	0.053573	0.052628	1
30	4.74E-05	0.009508	3.85E-07	8.83E-08	0.015952	7.15E-05	4.11E-05	0.001433	0.000114	1.04E-06	3.47E-06	6.11E-07	0.005708	0.005338	5.03E-12	0.000363	0.546982	2.34E-06	4.42E-08	0.026766	0.063988	0.042383	3
31	0.000112	0.015307	2.09E-07	2.38E-07	0.082377	0.000159	3.09E-05	0.000647	5.66E-05	1.28E-05	8.95E-07	3.62E-06	0.000255	0.004905	7.19E-11	0.000579	0.165743	1.07E-06	1.86E-08	0.001276	0.026684	0.022085	3

11.CONCLUSION

The input is given through the csv format of a data to the algorithm, and it processes it and gives the output in a csv format. The algorithm used here is Light GBM (Gradient Boosting Machine). Light GBM (Gradient Boosting Machine) algorithm is used here for prediction of the csv format input data and processing it and catapulting the output in csv format. The csv data provided in the input will be same during output but the major difference between the input and output data will be accuracy level which is achieved by the light GBM (Gradient Boosting Machine) algorithm will be around 80-85%.

12. FUTURE SCOPE

From the encouraging results, it is believed that the crime prediction has a promising future for increasing the effectiveness and efficiency of criminal and intelligence analysis. Visual and intuitive criminal and intelligence investigation techniques can be developed for crime pattern. As the LightGBM (Gradient Boosting Machine) is applied for crime analysis, the other techniques of data mining can also be performed.

13. BIBLIOGRAPHY

Textbooks:

Programming Python, Mark Lutz

Head First Python, Paul Barry

Core Python Programming, R. Nageswara Rao

Learning with Python, Allen B. Downey

Journals:

1. McClendon, Lawrence, and Natarajan Meghanathan. "Using machine learningalgorithms to analyze crime data." Machine Learning and Applications: An International Journal (MLAIJ) 2.1 (2015): 1-12.
2. Alkesh Bharati, Dr Sarvanaguru RA. K," Crime Prediction and Analysis Using Machine Learning" in International Research Journal of Engineering and Technology (IRJET), Volume: 05 Issue: 09 | September 2018
3. Sathyadevan, Shiju. "Crime analysis and prediction using data mining." 2014 FirstInternational Conference on Networks & Soft Computing (ICNSC2014). IEEE, 2014.

Websites:

https://www.w3schools.com/python/

https://www.javatpoint.com/python-tutorial

https://www.learnpython.org/

https://www.pythontutorial.net