

# LOCATION BASED CRIME TYPE PREDICTION USING PYTORCH NON-LINEAR DEEP-2-LAYER MODEL WITH STREAMLIT WEBAPP

*Major project report submitted  
in partial fulfillment of the requirement for award of the degree of*

**Bachelor of Technology  
in  
Computer Science & Engineering**

**By**

<b>SIVA PRASAD REDDY BANDI</b>	<b>(19UECS0104)</b>	<b>(VTU14477)</b>
<b>T. NIKHIL</b>	<b>(19UECS0986)</b>	<b>(VTU14506)</b>
<b>K. SAI KIRAN</b>	<b>(19UECS0402)</b>	<b>(VTU15321)</b>

*Under the guidance of  
Dr. K. SEETHALAKSHMI, M.E, Ph.D.,  
ASSOCIATE PROFESSOR*



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING  
SCHOOL OF COMPUTING**

**VEL TECH RANGARAJAN DR. SAGUNTHALA R&D INSTITUTE OF  
SCIENCE & TECHNOLOGY**

**(Deemed to be University Estd u/s 3 of UGC Act, 1956)**

**Accredited by NAAC with A++ Grade  
CHENNAI 600 062, TAMILNADU, INDIA**

**April, 2023**

# **LOCATION BASED CRIME TYPE PREDICTION USING PYTORCH NON-LINEAR DEEP-2-LAYER MODEL WITH STREAMLIT WEBAPP**

*Major project report submitted  
in partial fulfillment of the requirement for award of the degree of*

**Bachelor of Technology  
in  
Computer Science & Engineering**

**By**

<b>SIVA PRASAD REDDY BANDI</b>	<b>(19UECS0104)</b>	<b>(VTU14477)</b>
<b>T. NIKHIL</b>	<b>(19UECS0986)</b>	<b>(VTU14506)</b>
<b>K. SAI KIRAN</b>	<b>(19UECS0402)</b>	<b>(VTU15321)</b>

*Under the guidance of  
Dr. K. SEETHALAKSHMI, M.E, Ph.D.,  
ASSOCIATE PROFESSOR*



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING  
SCHOOL OF COMPUTING**

**VEL TECH RANGARAJAN DR. SAGUNTHALA R&D INSTITUTE OF  
SCIENCE & TECHNOLOGY**

**(Deemed to be University Estd u/s 3 of UGC Act, 1956)**

**Accredited by NAAC with A++ Grade  
CHENNAI 600 062, TAMILNADU, INDIA**

**April, 2023**

# CERTIFICATE

It is certified that the work contained in the project report titled LOCATION BASED CRIME TYPE PREDICTION USING PYTORCH NON-LINEAR DEEP-2-LAYER MODEL WITH STREAMLIT WEBAPP by SIVA PRASAD REDDY BANDI (19UECS0104), T. NIKHIL (19UECS0986), K. SAI KIRAN (19UECS0402) has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

**Signature of Supervisor**

**Dr. K. Seethalakshmi**

**Associate Professor**

**Computer Science & Engineering**

**School of Computing**

**Vel Tech Rangarajan Dr.Sagunthala R&D**

**Institute of Science & Technology**

**April, 2023**

**Signature of Head of the Department**

**Dr. M.S. Muralidhar**

**Associate Professor**

**Computer Science & Engineering**

**School of Computing**

**Vel Tech Rangarajan Dr. Sagunthala R&D**

**Institute of Science & Technology**

**April, 2023**

**Signature of the Dean**

**Dr. V. Srinivasa Rao**

**Professor & Dean**

**Computer Science & Engineering**

**School of Computing**

**Vel Tech Rangarajan Dr. Sagunthala R&D**

**Institute of Science & Technology**

**April, 2023**

# DECLARATION

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

(Signature)

(SIVA PRASAD REDDY BANDI)

Date:        /        /

(Signature)

(T. NIKHIL)

Date:        /        /

(Signature)

(K. SAI KIRAN)

Date:        /        /

# APPROVAL SHEET

This project report entitled LOCATION BASED CRIME TYPE PREDICTION USING PYTORCH NON-LINEAR DEEP-2-LAYER MODEL WITH STREAMLIT WEBAPP by SIVA PRASAD REDDY BANDI (19UECS0104), T. NIKHIL (19UECS0986), K. SAI KIRAN (19UECS0402) is approved for the degree of B.Tech in Computer Science & Engineering.

**Examiners**

**Supervisor**

Dr. K. Seethalakshmi,M.E,Ph.D.,

**Date:**     /     /

**Place:**

# ACKNOWLEDGEMENT

We express our deepest gratitude to our respected **Founder Chancellor and President Col. Prof. Dr. R. RANGARAJAN B.E. (EEE), B.E. (MECH), M.S (AUTO),D.Sc., Foundress President Dr. R. SAGUNTHALA RANGARAJAN M.B.B.S.** Chairperson Managing Trustee and Vice President.

We are very much grateful to our beloved **Vice Chancellor Prof. S. SALIVAHANAN**, for providing us with an environment to complete our project successfully.

We record indebtedness to our **Professor & Dean, Department of Computer Science & Engineering, School of Computing, Dr. V. SRINIVASA RAO, M.Tech., Ph.D.**, for immense care and encouragement towards us throughout the course of this project.

We are thankful to our **Head, Department of Computer Science & Engineering, Dr.M.S. MURALI DHAR, M.E., Ph.D.**, for providing immense support in all our endeavors.

We also take this opportunity to express a deep sense of gratitude to our Internal Supervisor **Dr. K. SEETHALAKSHMI, M.E.,Ph.D.**, for her cordial support, valuable information and guidance, she helped us in completing this project through various stages.

A special thanks to our **Project Coordinators Mr. V. ASHOK KUMAR, M.Tech., Ms. C. SHYAMALA KUMARI, M.E.**, for their valuable guidance and support throughout the course of the project.

We thank our department faculty, supporting staff and friends for their help and guidance to complete this project.

<b>SIVA PRASAD REDDY BANDI</b>	<b>(19UECS0104)</b>
<b>T. NIKHIL</b>	<b>(19UECS0986)</b>
<b>K. SAI KIRAN</b>	<b>(19UECS0402)</b>

## ABSTRACT

Finding both spatial and temporal criminal hotspots is the primary emphasis of this work. It examines several datasets based on crimes that have occurred in the actual world. After that, it elucidates how it implemented an algorithm in order to develop Binary classifications for crime hotspots. In addition, the research demonstrates how the utilized logistic regression to make predictions for possible sorts of criminal activity. This research presents an analytical study that combines findings of crime's dataset with its demographics information in order to capture the aspects that can affect the safety of neighborhoods. The purpose of this study is to further analyze crimes' datasets, and it is introduced. The results of implementing this method might be utilized to increase people's awareness of the dangerous regions and to assist agencies in predicting future crimes in a certain location within a specific time frame. Both outcomes would be beneficial. To find spatial and temporal criminal hotspots using a set of real-world datasets of crimes. It will try to locate the most likely crime locations and their frequent occurrence time. In addition, we will predict what type of crime might occur next in a specific location within a particular time. Finally, intend to provide an analysis study by combining our findings of a particular crimes dataset with its demographics information.

**Keywords:** Binary classifications, Hotspots, Demographics, Frequent occurrence time, Logistic regression, Spatial and Temporal.

# LIST OF FIGURES

4.1	<b>Framework of Location Based Crime Prediction</b>	14
4.2	<b>Data Flow Diagram</b>	15
4.3	<b>Use Case Daagram</b>	16
4.4	<b>Class Diagram</b>	17
4.5	<b>Sequence Diagram</b>	18
4.6	<b>Collaboration Diagram</b>	19
4.7	<b>Activity Diagram</b>	20
5.1	<b>Crime Prediction Input Design</b>	28
5.2	<b>Crime Prediction Output Design</b>	29
5.3	<b>Unit Testing Output</b>	31
5.4	<b>Integration Testing Output</b>	33
5.5	<b>System Testing Output</b>	35
5.6	<b>Types of Crime Occurred</b>	36
5.7	<b>Reported Crimes by hour</b>	37
6.1	<b>Input Data to Predict Crime</b>	43
6.2	<b>Probablity of Types and Acts of Crime</b>	43
6.3	<b>Occurrence of crime in Act</b>	44
6.4	<b>Relationship Between Year And Crime Category</b>	44
6.5	<b>Position of Data Points Using Coordinates</b>	45
6.6	<b>Relationship between Day, Year, and Crime</b>	45
8.1	<b>Siva Prasad Reddy Bandi Internship Offer Letter</b>	50
8.2	<b>T Nikhil Internship Offer Letter</b>	51
8.3	<b>K Sai Kiran Internship Offer Letter</b>	52
8.4	<b>Project Commencement Form</b>	53
8.5	<b>Internship Completion Certificate</b>	54
9.1	<b>Plagiarism Report</b>	55
10.1	<b>Poster Presentation</b>	61



# LIST OF ACRONYMS AND ABBREVIATIONS

CD	Criminal Dataset
CH	Criminal Hotspots
DI	Demographics Information
D2LM	Deep 2-Layer Model
DS	Data Set
DS	K nearest neighbor
NL	Non-Linear
LB	Location Based
PT	PyTorch
LR	Logistic Regression
ST	Streamlit
SVM	Support vector machine

# TABLE OF CONTENTS

	Page.No
<b>ABSTRACT</b>	<b>v</b>
<b>LIST OF FIGURES</b>	<b>vi</b>
<b>LIST OF ACRONYMS AND ABBREVIATIONS</b>	<b>vii</b>
<b>1 INTRODUCTION</b>	<b>1</b>
1.1 Introduction . . . . .	1
1.2 Aim of the project . . . . .	2
1.3 Project Domain . . . . .	2
1.4 Scope of the Project . . . . .	3
<b>2 LITERATURE REVIEW</b>	<b>4</b>
<b>3 PROJECT DESCRIPTION</b>	<b>8</b>
3.1 Existing System . . . . .	8
3.2 Proposed System . . . . .	8
3.3 Feasibility Study . . . . .	9
3.3.1 Economic Feasibility . . . . .	10
3.3.2 Technical Feasibility . . . . .	10
3.3.3 Social Feasibility . . . . .	11
3.4 System Specification . . . . .	11
3.4.1 Hardware Specification . . . . .	12
3.4.2 Software Specification . . . . .	12
3.4.3 Standards and Policies . . . . .	13
<b>4 METHODOLOGY</b>	<b>14</b>
4.1 General Architecture . . . . .	14
4.2 Design Phase . . . . .	15
4.2.1 Data Flow Diagram . . . . .	15
4.2.2 Use Case Diagram . . . . .	16
4.2.3 Class Diagram . . . . .	17

4.2.4	Sequence Diagram . . . . .	18
4.2.5	Collaboration diagram . . . . .	19
4.2.6	Activity Diagram . . . . .	20
4.3	Algorithm & Pseudo Code . . . . .	21
4.3.1	Binary Classification Using Logistic Regression . . . . .	21
4.3.2	Pseudo Code . . . . .	21
4.4	Module Description . . . . .	23
4.4.1	Data Acquisition and Pre-Processing . . . . .	23
4.4.2	The Model Creation . . . . .	23
4.4.3	Making A Prediction Function and Creating A Web App . .	24
4.5	Steps to execute/run/implement the project . . . . .	24
4.5.1	Checking Requirements and Installation . . . . .	24
4.5.2	Installing Packages . . . . .	26
4.5.3	Executing Project . . . . .	27

## **5 IMPLEMENTATION AND TESTING 28**

5.1	Input and Output . . . . .	28
5.1.1	Input Design . . . . .	28
5.1.2	Output Design . . . . .	29
5.2	Testing . . . . .	29
5.3	Types of Testing . . . . .	30
5.3.1	Unit testing . . . . .	30
5.3.2	Unit Testing Result . . . . .	31
5.3.3	Unit Testing Description . . . . .	31
5.3.4	Integration testing . . . . .	32
5.3.5	Integration Testing Result . . . . .	33
5.3.6	Integration Testing Description . . . . .	33
5.3.7	System testing . . . . .	34
5.3.8	System Testing Result . . . . .	35
5.3.9	System Testing Description . . . . .	35
5.3.10	Test Result . . . . .	36
5.3.11	Test Result Description . . . . .	37

## **6 RESULTS AND DISCUSSIONS 38**

6.1	Efficiency of the Proposed System . . . . .	38
6.2	Comparison of Existing and Proposed System . . . . .	39

6.3	Sample Code . . . . .	40
<b>7</b>	<b>CONCLUSION AND FUTURE ENHANCEMENTS</b>	<b>46</b>
7.1	Conclusion . . . . .	46
7.2	Future Enhancements . . . . .	47
<b>8</b>	<b>INDUSTRY DETAILS</b>	<b>49</b>
8.1	Industry name . . . . .	49
8.1.1	Duration of Internship (From Date - To Date) . . . . .	49
8.1.2	Duration of Internship in months . . . . .	49
8.1.3	Industry Address . . . . .	49
8.2	Internship offer letter . . . . .	50
8.3	Project Commencement Form . . . . .	53
8.4	Internship Completion Certificate . . . . .	54
<b>9</b>	<b>PLAGIARISM REPORT</b>	<b>55</b>
<b>10</b>	<b>SOURCE CODE &amp; POSTER PRESENTATION</b>	<b>56</b>
10.1	Source Code . . . . .	56
10.2	Poster Presentation . . . . .	61
	<b>References</b>	<b>61</b>

# Chapter 1

## INTRODUCTION

### 1.1 Introduction

Crime reduction is becoming one of the most important social issues in enormous metropolitan areas because it influences people's concerns about their personal safety, the development of children, and individuals socioeconomic standing. Crimes are influenced by organizations and other places occurred frequently in a society. This is due to the fact that larger cities have higher crime rates than smaller cities. The crime rate forecasting method is a method that determines the crime rate by using a variety of algorithms and basing their findings on previous information. Because of the nature of our work, that are required to travel to a large number of locations on a daily basis. Crimes are serious threats to human society, safety, and sustainable development and are thus meant to be controlled. Investigation authorities often demand computational predictions and predictive systems that improve crime analytics to further enhance the safety and security of cities and help to prevent crimes. It will try to locate the most likely crime locations occurrence time.

The main purpose of this project is implement a completely integrated and compact system is developed that can be used by the common man as well as the police and this system would be like a situation for both of them. Google Maps may provide us one, two, or even more routes to reach our destination nonetheless, The almost always take the shortest route, even though this indicates that not fully understand the path condition. This research introduces the creation and implementation of a plan based on historical crime data and evaluates the crime rate in previous places at separate moments. Because of the uncertainty over whether or not it is actually secure, they are forced to deal a number of unfavorable scenarios. Using a collection of information taken from the real world, the purpose of this research is to identify both temporal and spatial hotspots for criminal activity. The core objective of this project is to design a system to help the officers to speed up the process of investigation and track status of predicting the suspects.

## **1.2 Aim of the project**

The main objective of the project is to find spatial and temporal criminal hotspots using a set of real-world datasets of crimes. Based on this Information the officials can take charge and try to reduce the crime rate. The goal of this challenge is to apply machine learning procedures to find a criminal sample with the aid of its category with the given precise time and location. The major aim of this mission is to expect which category of crime is most probably to take place at a detailed time and places. Crime analysis based on available information to extract crime patterns. The core objective of this project is to design a system to help the officers to speed up the process of investigation and track status of ongoing case by predicting out the suspects. The major aim of this mission is to expect which category of crime is most probably to take place at a detailed time and places.

## **1.3 Project Domain**

Digital transformation based on artificial intelligence technology and big data has received considerable attention and has become a huge trend. In addition, AI technology is being utilized in various research areas such as smart communications, medical research, natural language process, and robots. Crime is considered an essential factor that determines whether or not people move to a new city and what places should be avoided when they travel. 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. Our study aims to find spatial and temporal criminal hotspots using a set of real-world datasets of crimes. It will predict what type of crime might occur next in a specific location within a particular time.

The Location Based Crime Type Prediction Using PyTorch Non-Linear Deep-2-layer Model With Streamlit WebApp is implemented using deep learning which is used to is a subset of machine learning, which is essentially a neural network with three or more layers. The PyTorch is used to implement the a machine learning framework based on the Torch library, used for applications such as computer vision and natural language processing, originally developed by Meta AI and now part of the Linux Foundation umbrella. By using Stramlit is an open-source Python

library that makes it easy to build custom web apps for machine learning and data science. To describe something as non-linear, that means it does not progress or develop smoothly from one stage to the next in a logical way. In order to build and make the project user-friendly, deep learning will be used in the project's domain while using Streamlit to create the web application. The goal of this challenge is to apply machine learning procedures to find a criminal sample with the aid of its category with the given precise time and location.

## **1.4 Scope of the Project**

The main scope of the project is to analyze the crime patterns and identify the crime predicted areas based on the crime rate using the crime data. Using the fuzzy clustering obtain the crime patterns to know in which area the crime will occur frequently. The approach will help regulation enforcement agencies to have assumption ahead about the possibility of crime, which could arise at a place in a given time. This will assist them to resolve the instances faster than earlier. In addition, having this kind of knowledge would help people to improve their living place choices. On the other hand, police forces can use this solution to increase the level of crime prediction and prevention. It can see that the events happened all over town. Using the latitude and longitude of these events, it can find a centroid for optimal resource allocation. To explore this further we need to configure two prediction models one for latitude and one for longitude.

This project will be widely used in the future by the police department, the common man, security agencies and even hospitals for accident and assault victims. The greatest strength of this project is that it offers new features as well as retaining the original characteristics of the existing systems for example: Criminal Database. Should also convert occurrence column to fit new date, time, and day of week columns with date-time schema. It should also remove any unfounded events from our dataset as well. We can see that data is almost bimodal on the Latitude axis, and that the time of day does not significantly predict a shift in direction. The core objective of this project is to design a system to help the officers to speed up the process of investigation and track status of ongoing case by predicting out the suspects. The new features extraction method adapts satisfactorily to the changing datasets with updates according to the expected strategy of the concerned criminal group.

## Chapter 2

# LITERATURE REVIEW

**Charlie Catlett et al.,-2018 [1], have developed A Data-Driven Approach for Spatio-Temporal Crime Predictions in Smart Cities.**

The main goal of there research consists in detecting the most significant crime dense regions and discovering a predictive model for each one to forecast the number of crimes that will happen in the future in each area. Presented a general algorithm for Spatio Temporal Crime Prediction in urban areas, that takes advantages from the partitioning of the whole analyzed area by detecting crime dense regions (of arbitrary shapes).Such regions are then analyzed and a different forecasting auto regressive model is tailored specifically for each detected region. Experimental evaluation, performed on crime data of a wide area of Chicago, showed that the proposed methodology can forecast the number of crimes with an high accuracy.

**Imarn Shafi et al.,-2021 [2] have implemented the Adaptable Reduced-Complexity Approach Based on State Vector Machine for Identification of Criminal Activists on Social Media.**

This Experimental results reveal that the proposed approach in the current study performs well on the collected dataset with the extra characteristics of better flexibility and adaptability to the targeted criminal group’s changing interests and strategies. The proposed method can identify the textual malicious contents on social media with high accuracy and less computational complexity.The new features extraction method adapts satisfactorily to the changing datasets with updates according to the expected strategy of the concerned criminal group.

**Lavanya Elluri et al.,-2019 [3] have developed a Developing Machine Learning Based Predictive Models for Smart Policing.**

The focus of this study is to predict the type of crime based on the weather,



temporal data, and relevant crime data. For this purpose, we relabeled samples in the dataset. Aim to predict the crime category based on weather attributes. In this study to evaluate machine learning models we primarily compare four performance metrics. In this work, they have used 2018 NYPD crime and New York city weather data set to check if the weather-related attributes play a significant role, or not, by performing various feature selection techniques.

**Myung-Sun Baek et al.,-2021 [4] they have developed the Smart Policing Technique With Crime Type and Risk Score Prediction Based on Machine Learning for Early Awareness of Risk Situation.**

Myung-Sun Baek designs and validates a crime type and associated risk prediction technique based on machine learning technology. The KICS data format is used for text-based criminal case summary data, which is actual policing data containing information about criminal cases. For the crime type, the system considers 21 representative types of crimes; thus, the system can predict one of the 21 types of criminal activity for each criminal case.

**R. Iqbal et al.,-2021 [5] was focused on An experimental study of classification algorithms for crime prediction.**

In this research, Vancouver crime data for the last 15 years was used in two different dataset approaches. Machine Learning predictive models KNN and boosted decision tree were used to obtain crime-prediction accuracy between 39 accuracy, complexity, and training time of algorithms were slightly different for different approaches and algorithms. Machine-learning-based crime analysis usually involves data collection, classification, pattern identification, prediction, and visualization. Traditional data mining techniques.

**Sai Tarlekar et al.,-2021 [6], they proposed “Geographical Crime Rate Prediction System.**

This system is primarily concerned with determining where the crime will occur rather than identifying the criminal. The existing system employed naive bayes classification. The fuzzy C-Means algorithm will be used in the current system to

cluster crime data for total cognizable crimes such as murder, theft, cheating, kidnapping, crime against women, robbery, and other such crimes. The Crime analysis and prediction is a systematic method for categorizing the kinds of crimes committed, the purpose of the crime, and forecasting future crimes. The system can calculate hotspot areas by analyzing crime reports. The goal of this proposed system is to investigate datasets and analyze crimes that have been committed, and then predict crimes using the Random Forest Algorithm.

**Saroj Kumar Dash et.al.,-2018 [7] was developed a Spatio-temporal prediction of crimes using network analytic approach.**

The objective is to predict crime levels for different types of crimes and a given year based on previous years criminal activity and other social aspects. It used the spatial resolution of the community area in our experiments to connect various types of data. Because the reasons of illegal activities depend on social factors, it is clear that two communities might share similarity in crime patterns if they are neighbors to each other since social factors typically do not change drastically between neighboring communities.

**Umair Muneer Butt, et al.,-2020 [8] has implemented a Spatio-Temporal Crime Hotspot Detection and Prediction.**

In this study, N Kanimozhi implements such a crime pattern analysis by utilizing crime data obtained from the Kaggle open source, which is then used to predict the most recently occurring crimes. Some machine learning algorithms, such as Naive Bayes, are used in this work to classify various crime patterns, and the accuracy achieved is comparable to pre-composed works. Thus the crime that occur the most could be predicted and spotted using Naïve Bayesian Classification.

**Vrushali Pednekar et al.,-2018 [9] Crime Rate Prediction using KNN”, International Journal on Recent and Innovation Trends in Computing and Communication.**

This system is primarily concerned with determining where the crime will occur rather than identifying the criminal. The existing system employed naive bayes clas-

sification. The fuzzy C-Means algorithm will be used in the current system to cluster crime data for total cognizable crimes such as murder, robbery, other crimes. This paper is useful to identify the crime areas based on the clustering technique. Crime patterns are also identified they are not static. We have identified the crime areas and which type of crime occurred very frequently in which place. It is very useful for investigators to solve the crime rate. Based on the fuzzy clustering technique the crime prone areas are identified it takes a less time.

**Wajiha Safat et al.,-2021 [10] they have developed “Empirical Analysis for Crime Prediction and Forecasting Using Machine Learning and Deep Learning Techniques.**

This research presents the to better fit the crime data, Wajiha Safat used a variety of machine learning algorithms, including logistic regression, support vector machine (SVM), Nave Bayes, k-nearest neighbors (KNN), decision tree, multilayer perceptron (MLP), random forest, and eXtreme Gradient Boosting (XGBoost), as well as time series analysis using long-short term memory (LSTM) and autoregressive integrated moving average (ARIMA). Crimes are serious threats to human society, safety, and sustainable development and are thus meant to be controlled.

**Xu Zhang 1,2 et al.,-2020 [11] they have developed the Comparison of Machine Learning Algorithms for Predicting Crime Hotspots.**

This paper presents crime prediction is of great significance to the formulation of policing strategies and the implementation of crime prevention and control. Machine learning is the current mainstream prediction method. In this paper, six machine learning algorithms are applied to predict the occurrence of crime hotspots in a town in the southeast coastal city of China. models have improved prediction accuracies, compared with other models. In empirical research on the prediction of crime hotspots using three models of logistic regression, neural network, and the combination of logistic regression and neural network.

## Chapter 3

# PROJECT DESCRIPTION

### 3.1 Existing System

The CNN-based crime type prediction model outperforms SVM and naïve Bayes algorithms by 7 percent and 8 percent, respectively. The Random Forest technique for classification and regression displays the final prediction interface along with a demo input and output with less accuracy than the current model. The classifier's use of categorical values, which produces a biased result for the nominal qualities with higher values, accounts for the low accuracy. The algorithm might not be able to correctly forecast future crimes if there isn't much information available about previous crimes in a specific area.

Deep neural networks are vulnerable to bias in the training data and overfitting, which can lead to incorrect forecasts and unjust treatment of certain populations. Overfitting is when the model gets too complicated and is matched to the training data too tightly, leading to reduced accuracy and poor generalisation. Deep neural networks are also criticised for being unintelligible and requiring a high level of technical proficiency to build and maintain, making it difficult for non-technical people to comprehend and use the system properly. Limited Scope is also a concern, as the model's ability to forecast crime in other places or types of crime may be constrained by its scope, which may be restricted to certain types of crimes or geographic regions.

### 3.2 Proposed System

The location-based crime type prediction system that is being presented uses a PyTorch non-linear deep 2-layer model with the Streamlit web app. Data gathering, cleaning, preprocessing, and feature engineering are necessary for modelling and deployment. is an intriguing concept that has the potential to be useful for law

enforcement and public safety. The model would analyse past crimes to discover patterns and relationships between variables and different sorts of crime, allowing it to forecast the likelihood that a given crime will occur in a specific location at a specific time. The system would be user-friendly, with a web app interface that would let law enforcement personnel and other users enter pertinent information and get real-time predictions about future crimes in a certain location. This could lead to improved prevention and reaction plans as well as predictions that are more accurate.

Due to the complexity and heterogeneity of crime patterns, predicting the type of crime that will occur where a person will live can be difficult. However, a powerful system for determining the type of crime based on location can be created using a PyTorch-based non-linear deep 2-layer model. The proposed system of location-based crime type prediction utilising PyTorch non-linear deep 2-layer model with Streamlit web app has the potential to be an important tool for law enforcement and public safety officials, but data quality and model upkeep should be considered. It could lead to a decline in crime rates and an improvement in general public safety, improved resource allocation, crime monitoring in real time, model customization, user-friendly interface, and increased forecast accuracy. The technique may be adopted more widely and become more effective.

### **3.3 Feasibility Study**

To ascertain the viability and use of such a system, a feasibility study of location-based crime type prediction using a PyTorch non-linear deep 2-layer model with Streamlit web app can be carried out. The model would be created to account for the various elements that can influence the occurrence of various sorts of crimes in diverse settings. Once the model has been trained, a Streamlit web application can be created that will let users to input location data and receive predictions about the crimes that are most likely to occur there based on the model's analysis.

This system's viability would be influenced by a few variables. To effectively train the model, the data would first need to be sufficiently available and of high quality. The Streamlit web app development would need to be user-friendly and open to a variety of people. A feasibility study of location-based crime type prediction using a PyTorch non-linear deep 2-layer model with Streamlit web app is a challenging

project that would need for careful planning, data preparation, and evaluation. But if it works, system might have a big impact on assisting law enforcement and enhancing public safety.

### **3.3.1 Economic Feasibility**

The economic feasibility of a location-based crime type prediction project using a PyTorch non-linear deep 2-layer model with a Streamlit web app will depend on the costs involved in acquiring and processing the data, building and running the model, deploying the model, maintaining and supporting the model, and generating revenue. The cost of acquiring and processing the data required for training the PyTorch model can vary depending on the source and quality of the data. Hardware and software costs include the computer, GPU, RAM, and other hardware components, as well as software costs such as PyTorch, Streamlit, and other libraries or tools.

Development and deployment costs can vary depending on the size and complexity of the project, and whether it is deployed on-premise or in the cloud. The project will require ongoing maintenance and support to ensure the model continues to function correctly. The project should consider maintenance and support costs, revenue potential, and monetization to ensure it is economically feasible. The economic feasibility of a location-based crime type prediction project using a PyTorch non-linear deep 2-layer model and Streamlit web app will depend on the costs involved and potential revenue generated.

### **3.3.2 Technical Feasibility**

The use of PyTorch, a non-linear deep 2-layer model, and a Streamlit web application to create a location-based crime type prediction system is technically viable. Many deep learning models, including non-linear 2-layer models, can be created using PyTorch, a potent open-source machine learning framework. A web application framework called Streamlit enables programmers to create interactive and user-friendly web applications with little to no coding. It offers a simple interface for building web programmes that can interact with PyTorch models and display the model's results.

To build the location-based crime type prediction system, you would need to

gather and preprocess crime data, make a PyTorch model, train the model on the data, and then publish the model to the Streamlit web app. The technical viability of creating a location-based crime type prediction system utilising PyTorch and Streamlit ultimately rests on the availability and calibre of the crime data, the skill of the developers, and the resources accessible to support the development process.

### **3.3.3 Social Feasibility**

Using a PyTorch non-linear deep 2-layer model with Streamlit web app, location-based crime type prediction may be socially feasible depending on a number of variables. These are some things to think about: The Ethical Considerations is one of the most crucial things to take into account is ethics. In order to prevent biases or discrimination based on race, gender, or socioeconomic position from being perpetuated, it is imperative to make sure that the use of such a model and web app respects the privacy of individuals and communities.

The Legal Framework it's crucial to make sure that the use of such a model and web app complies with the laws of the area in question. Assist guarantee that the system is in line with the needs and values of community and also contribute to system trust-building. Careful consideration of factors and the participation of stakeholders and the community in the development and implementation of system are essential for social viability of location-based crime type prediction using a PyTorch non-linear deep 2-layer model with Streamlit web application. Transparency: Crucial to make the model's inner workings clear and accessible to the broader audience.

## **3.4 System Specification**

- **The System Overview**

The system, which is a web application, predicts the most likely sort of crime that will occur in a given location given a location as input. Based on a PyTorch non-linear deep 2-layer model trained on historical crime data, the prediction was made.

- **Components of a System**

The system must gather historical crime data to train a PyTorch non-linear deep 2-layer model, with a clear user interface and prediction of the type of crime.

Deployment should be done on a server or cloud service.

- **Structure of the System**

A PyTorch non-linear deep 2-layer model is trained using preprocessed data and a Streamlit framework is used to create a web application and trained model on a cloud provider.

- **System prerequisites**

The location-based crime type prediction system requires data storage, memory, a web server, and an internet connection. The PyTorch model must be trained on a contemporary CPU or GPU, and enough memory must be available to store preprocessed data and historical crime data.

### **3.4.1 Hardware Specification**

- Processor: Intel Core i5 or higher / AMD Ryzen 5 or higher
- RAM: 8GB or higher
- Storage: At least 256GB SSD or higher for fast data access
- Graphics Card: NVIDIA GeForce GTX 1660 or higher
- Screen Resolution: 1920x1080 or higher

Hardware specifications for deep learning models vary depending on the size of the dataset and complexity of the model, so it is recommended to upgrade hardware and install software packages and dependencies.

### **3.4.2 Software Specification**

- Operating System: Windows 10, Linux or macOS
- Python 3.7 or above
- PyTorch deep learning framework
- Streamlit web application framework
- Git Version Control System



- Integrated Development Environment (IDE) such as PyCharm or Visual Studio Code.

PyTorch is a deep learning framework used to build and train neural networks, while Streamlit is an open-source application framework used for building and deploying data science and machine learning applications.

### **3.4.3 Standards and Policies**

The Standards and Policies for location based crime type prediction using pytorch non-linear deep-2-layer model with streamlit webapp is:

#### **Python 3.10.10: (Standard Used: : ISO/IEC WD TR 24772-4)**

Python is a high-level, interpreted programming language known for its readability, versatility, and simplicity, used for web development frameworks, data science and analytics, and deep learning and machine learning. Python is used to automate operations, create desktop apps, and teach computer science and programming.

#### **Anaconda: (Standard Used: ISO/IEC 10918-1:1994)**

Anaconda is a type of command line interface which explicitly deals with the ML( Machine Learning) modules.And navigator is available in all the Windows,Linux and MacOS.The anaconda prompt has many number of IDE's which make the coding easier. The UI can also be implemented in python. pre-built packages ,tools Python scientific computing and data.

#### **Visual Studio: (Standard Used: : IS 16007 : 2013 ISO 13492 : 2007)**

Visual Studio is Microsoft's integrated development environment (IDE) used to create, debug, test, and deploy software across multiple platforms. It offers a complete collection of tools and functionalities, including C++, C, Python, JavaScript, Type Script, and others. It also offers resources for database administration, web design, cloud computing, game design, and AI development.

## Chapter 4

# METHODOLOGY

### 4.1 General Architecture

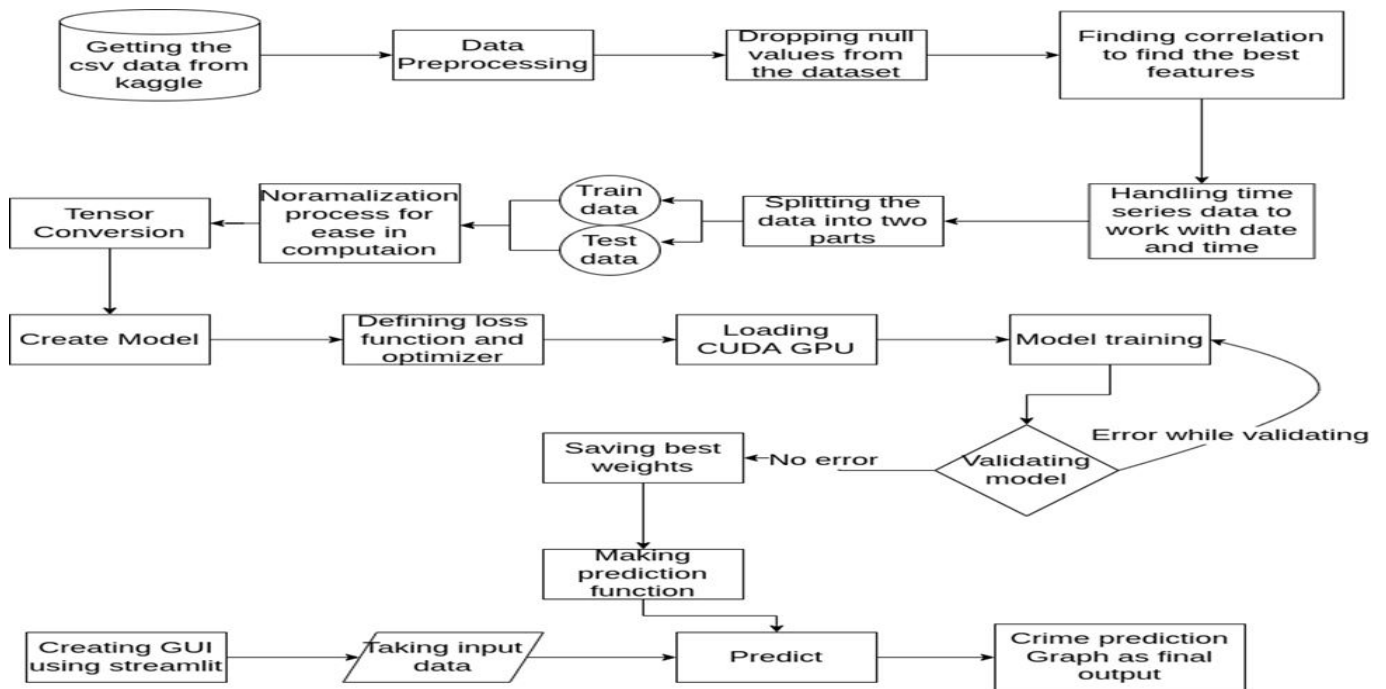


Figure 4.1: Framework of Location Based Crime Prediction

#### Description:

The Figure 4.1 begins with a source of crime data, which may be a database or a stream of real-time crime reports. The location of crimes is included in the geospatial data repository, which is utilised to enhance the crime data. In order to prepare the data for the PyTorch model, the stream processor ingests the criminal data stream and performs real-time feature engineering. Based on geography and other relevant features, the PyTorch model is trained on historical crime data to forecast different sorts of crimes. In the Figure 4.1 Real-time predictions are made using the trained model on the criminal data stream, and the predictions are output to a separate stream so that downstream applications can use them.

## 4.2 Design Phase

### 4.2.1 Data Flow Diagram

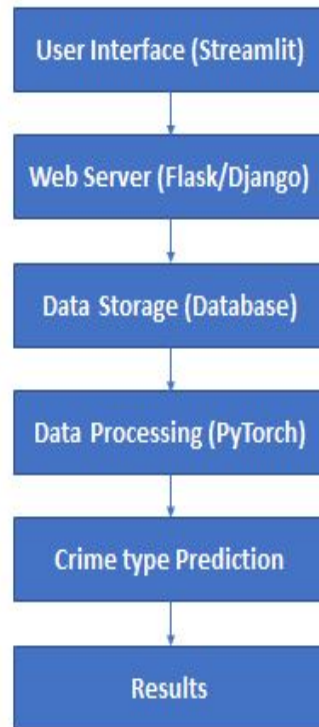


Figure 4.2: **Data Flow Diagram**

#### Description:

The Figure 4.2 provides a high-level overview of the system architecture for a location-based crime type prediction system, but other parts and procedures may be needed depending on the system's requirements. A Streamlit web app serves as the system's initial user interface, allowing users to enter their location and other pertinent information. A web server receives this information and is in charge of managing incoming requests and providing answers. The data is fed into a PyTorch non-linear deep 2-layer model, which uses the input data to predict the most likely type of crime. The user interface then displays the outcomes of the prediction process for viewing by the user.

#### 4.2.2 Use Case Diagram

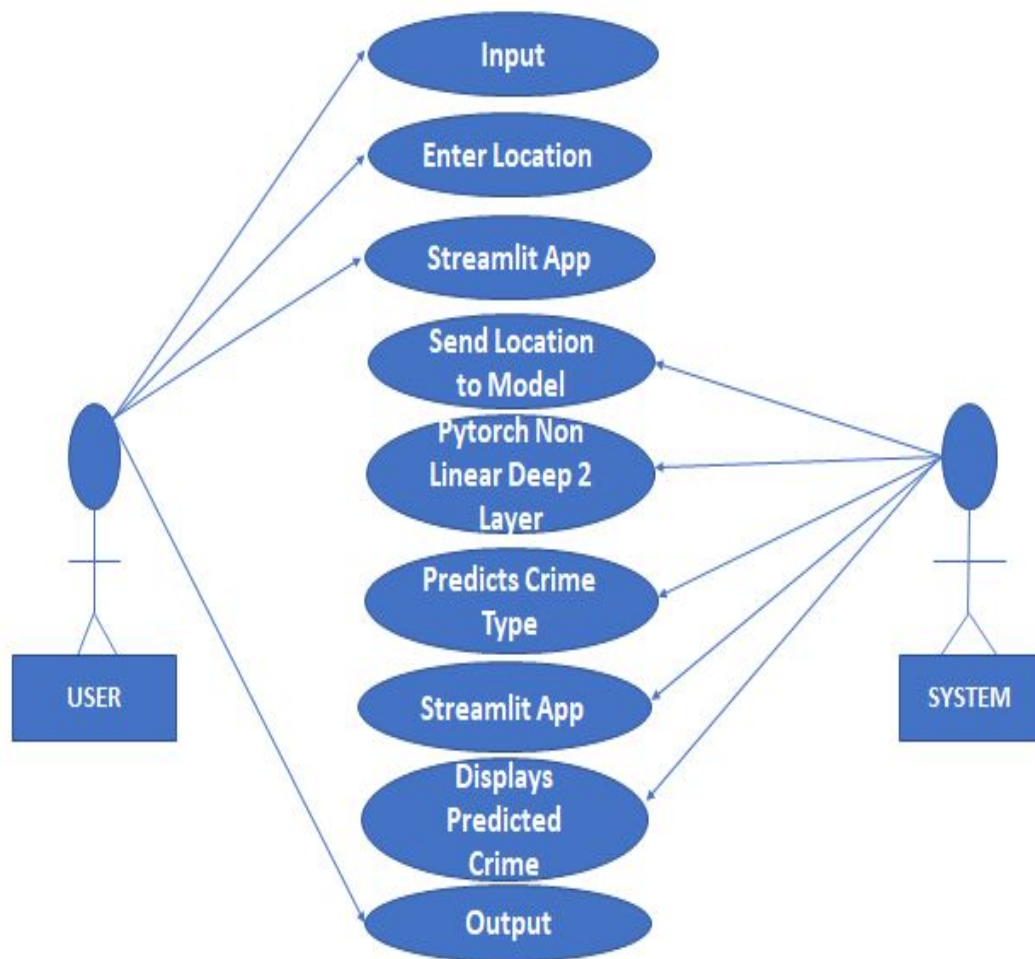


Figure 4.3: Use Case Diagram

#### Description:

In this Figure 4.3, the user enters location information using a Streamlit web application. The PyTorch non-linear deep-2-layer model receives the location information from the Streamlit app and uses it to forecast the type of crime. Based on the geographical information, the programme forecasts the sort of crime and relays the findings to the Streamlit app. The projected crime type is then shown to the user by the Streamlit app. Use-case diagrams describe the high-level functions and scope of a system, but not how it operates internally. The Figure 4.3 illustrate and define the context and requirements of an entire system or important parts. We can model a complex system with a single use-case diagram, or create many use-case diagrams to model the components of the system.

### 4.2.3 Class Diagram

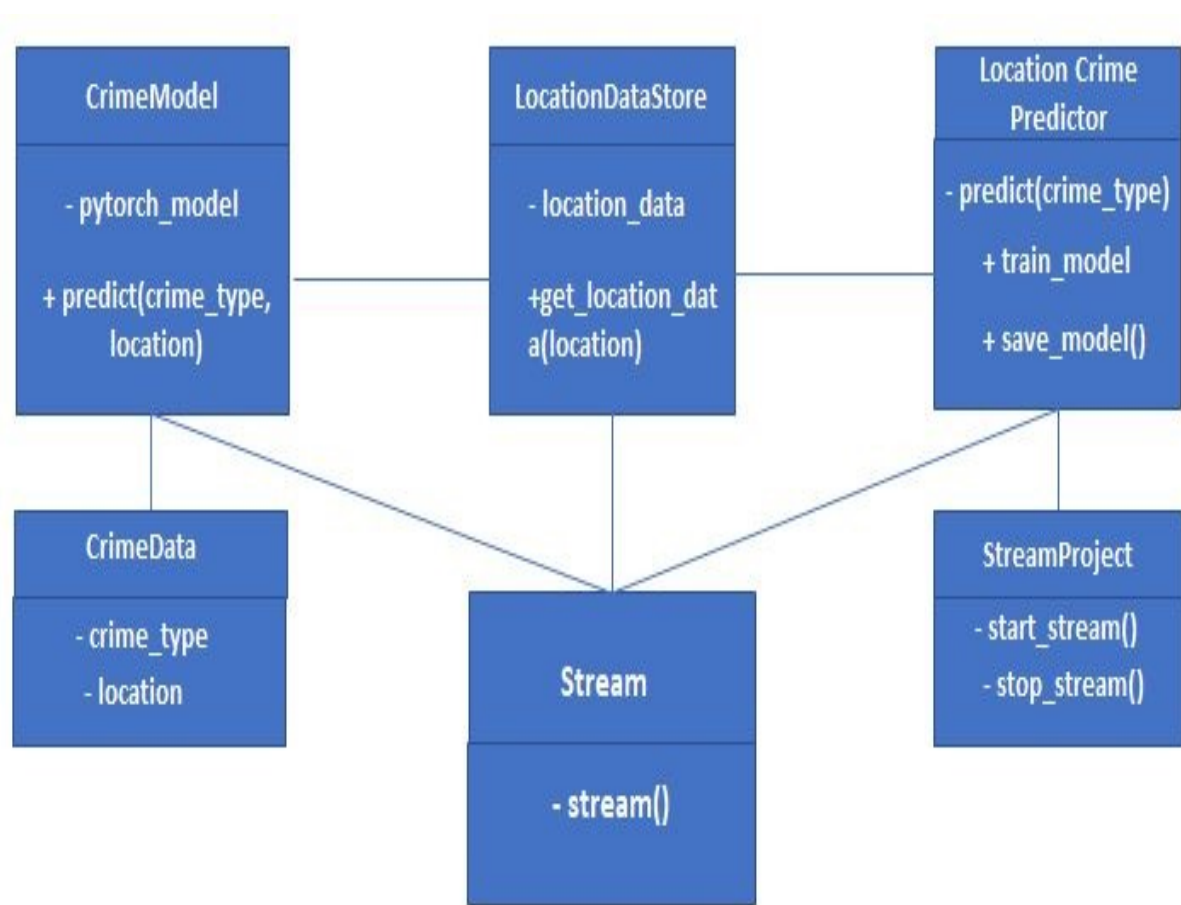


Figure 4.4: **Class Diagram**

Description:

Crime Model is a PyTorch non-linear deep 2-layer model used for crime type prediction. Location Data Store is a representation of the location data store, with a `get location data()` method and location data attribute. Location Crime Predictor is a location-based crime prediction system with two methods: `predict()` and `train model()`. Figure 4.4 is a representation of the system's crime data, while Stream Project is the data stream used to collect real-time location information. The location data store receives data from the stream using the `stream()` method. Crime Data is a representation of the system's crime data, while Stream Project is the data stream used to collect real-time location information. The location data store receives data from the stream using the `stream()` method.

#### 4.2.4 Sequence Diagram

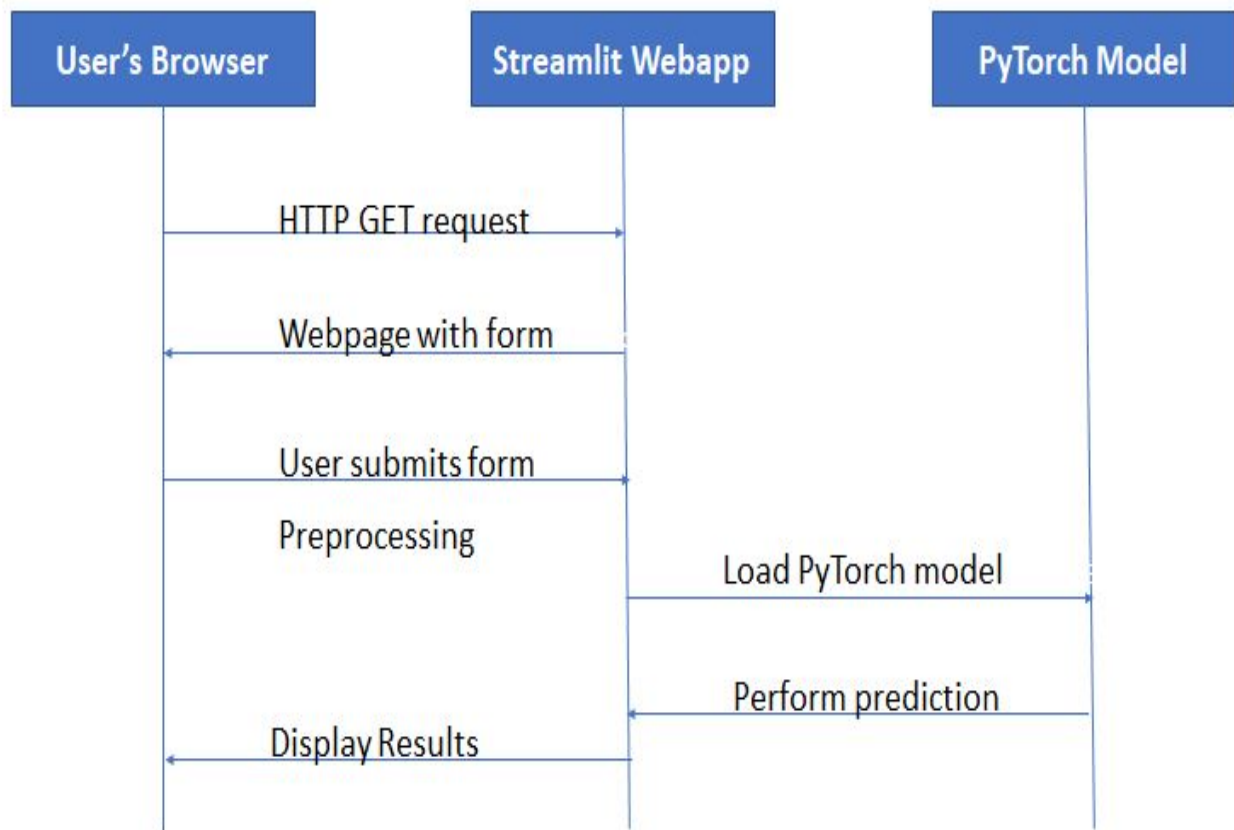


Figure 4.5: Sequence Diagram

#### Description :

Through their browser, the user connects to the web application and requests the web page from the server. After receiving the request, the Streamlit web application sends the user's browser the requested web page. The user fills out the form and interacts with the website. When a form is submitted, the Streamlit web application gets it and preprocesses the user input. The PyTorch model is loaded by the Streamlit website application. On the preprocessed input, Figure 4.5, the PyTorch model makes the prediction. The Streamlit web application receives the prediction from the PyTorch model. The user can view the findings in their browser thanks to the Streamlit web application. It should be noted that this simplified sequence diagram excludes error handling and other information that might be necessary for a real-world implementation.

#### 4.2.5 Collaboration diagram

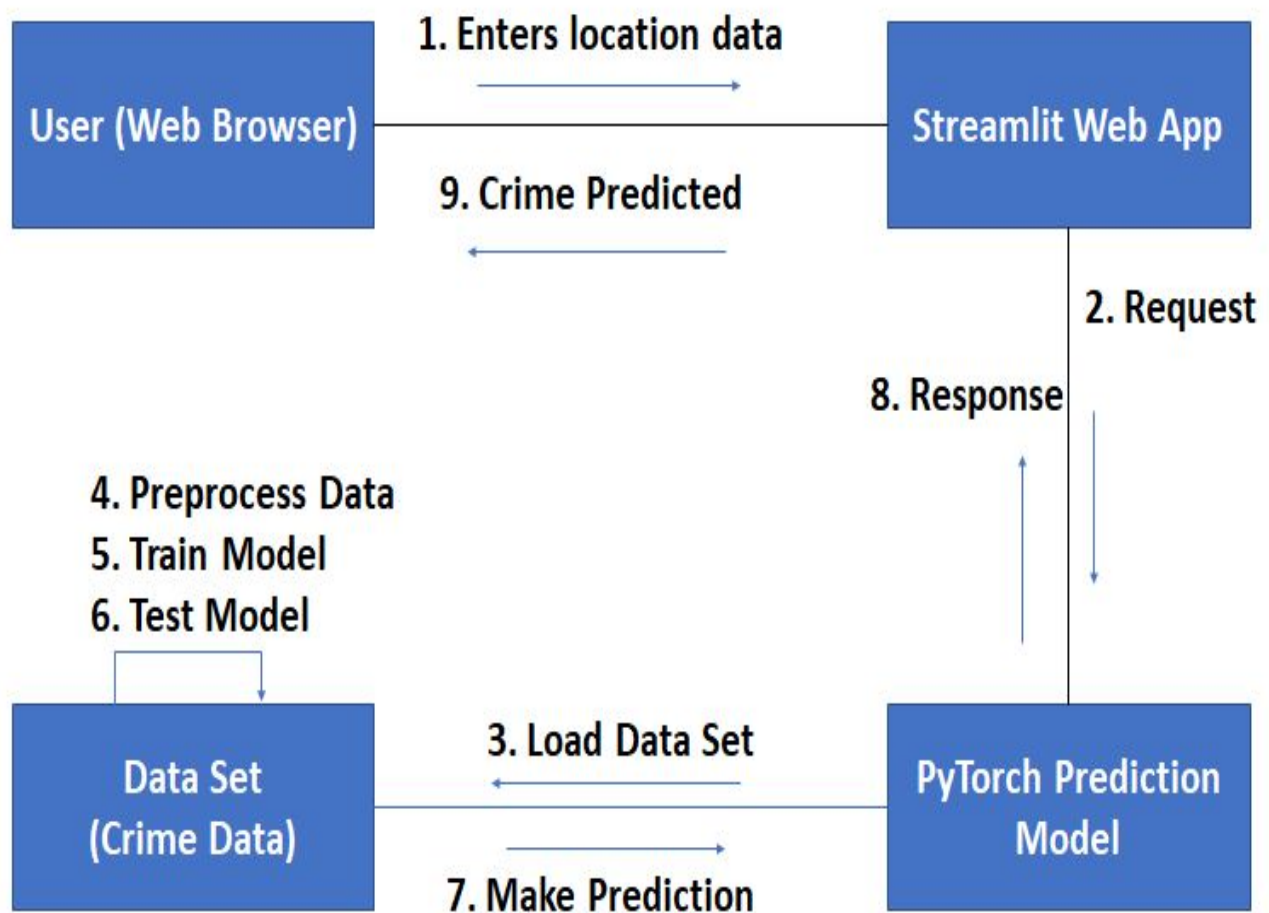


Figure 4.6: Collaboration Diagram

#### Description:

The interactions between objects or components in a system are depicted using a collaboration diagram, sometimes referred to as a communication diagram. The PyTorch Deep Learning Model is a neural network model used to forecast the types of crimes based on location data. The Location Dataset is used to train the model's two non-linear layers, and the stream processor is used to process incoming data and make predictions. Figure 4.6 Input data will be processed before being fed into the PyTorch model. The collaboration diagram illustrates how the system's many parts cooperate to predict crime categories based on location information. Data from the Location Dataset is preprocessed and supplied into the PyTorch model for training, and data from the stream processor is fed into the model during testing to produce output predictions.

#### 4.2.6 Activity Diagram

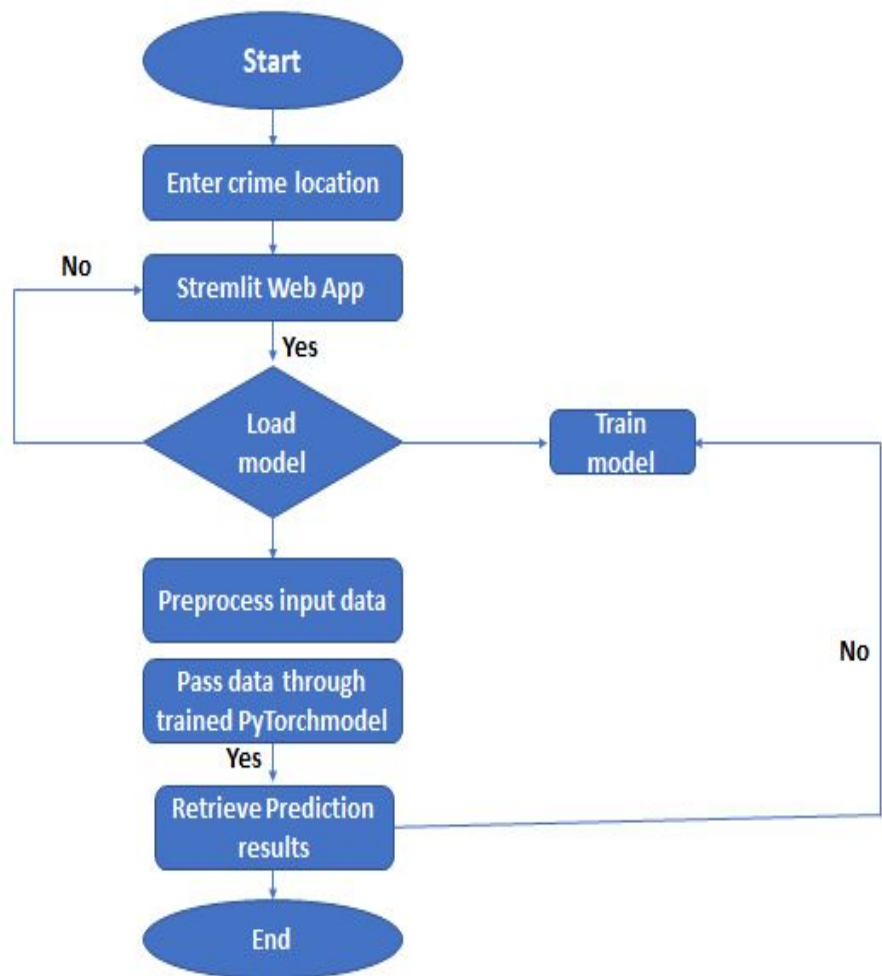


Figure 4.7: Activity Diagram

#### Description :

The user enters the crime location through the Streamlit web app to begin the activity diagram. After preprocessing the input data, the web application loads the pre-trained PyTorch non-linear deep 2-layer model. The model is then used to extract the results of the predictions. The user is then shown the prediction results using the web app. In the Unified Modeling Language (UML), an activity diagram, Figure 4.7 is a form of behavioural diagram that shows how activities flow through a system or business process. It is frequently employed to represent software system workflows, business workflows, and other intricate systems. Nodes and edges make up activity diagrams, where nodes stand for activities and edges for interactions between activities. A start node, activity node, decision node, merge node, control flow, object flow, and final node represent the process or activity.



## 4.3 Algorithm & Pseudo Code

### 4.3.1 Binary Classification Using Logistic Regression

The Algorithm used for Location based crime type prediction using pytorch non-linear deep-2-layer model with streamlit webapp is Binary classification is a supervised learning task in machine learning that aims to predict one of two potential outputs for a given input. The two possible outcomes are positive and negative, and the model's input is a group of features or predictors. Binary classification techniques use support vector machines, decision trees, and logistic regression to measure accuracy, precision, recall, and F1 score. This algorithm is a statistical method that is to analyze a data set one or more independent variables that determine the outcome of the dependent variable. Logistic regression can be used to detect fraudulent transactions by creating a model that calculates the probability of a transaction being fraudulent based on its features. Cross-entropy loss function is the most widely used cost function in logistic regression, used in applications such as spam filtering, fraud detection, and medical diagnosis.

### 4.3.2 Pseudo Code

- Bring in the necessary libraries, including PyTorch, Streamlit, pandas, numpy, and others.
- Preprocess the crime dataset by loading it and carrying out the appropriate feature engineering, cleaning, and normalisation.
- Create training and validation sets from the dataset.
- The PyTorch model architecture should be defined. We will apply a nonlinear deep 2-layer model with ReLU activation in this scenario.
- Utilize the training set to develop the model, and the validation set to assess its effectiveness.
- Using the user's location information entered into the Streamlit web app, utilise the trained model to forecast the sort of crime.
- Create the user interface for the Streamlit online application, which includes a map widget that allows users to enter their location and receive anticipated crime

types. Publish the web application to a server so that users can view it.

```
1 # Import necessary libraries
2 import torch
3 import pandas as pd
4 import numpy as np
5 import streamlit as st
6
7 # Load the trained model
8 model = torch.load('trained_model.pt')
9
10 # Define function to preprocess data
11 def preprocess_data(data):
12     # Perform necessary data cleaning and feature engineering
13     # Convert data to tensor and normalize the values
14     return preprocessed_data
15
16 # Define function to predict crime type
17 def predict_crime_type(data):
18     # Preprocess the input data
19     preprocessed_data = preprocess_data(data)
20
21     # Make prediction using the trained model
22     outputs = model(preprocessed_data)
23     _, predicted = torch.max(outputs.data, 1)
24     predicted_crime_type = predicted[0]
25
26     return predicted_crime_type
27
28 # Create Streamlit web app interface
29 st.title('Location-Based Crime Type Prediction')
30 st.write('Enter the details of the crime location below:')
31
32 # Collect input from the user
33 latitude = st.number_input('Latitude:')
34 longitude = st.number_input('Longitude:')
35 time_of_day = st.selectbox('Time of day:', ['Morning', 'Afternoon', 'Evening', 'Night'])
36 day_of_week = st.selectbox('Day of week:', ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday',
37     'Saturday', 'Sunday'])
38
39 # Create a dictionary from the input data
40 input_data = {'latitude': latitude, 'longitude': longitude, 'time_of_day': time_of_day, 'day_of_week': day_of_week}
41
42 # Convert the dictionary to a Pandas DataFrame
43 input_df = pd.DataFrame(input_data, index=[0])
44
45 # Make prediction using the model
46 predicted_crime_type = predict_crime_type(input_df)
```

```
47 # Display the predicted crime type
48 st.write(f'The predicted crime type is: {predicted_crime_type}')
```

## 4.4 Module Description

### 4.4.1 Data Acquisition and Pre-Processing

We will be using a csv dataset with 18 columns and 2100 rows from Kaggle on crime prediction. Data preprocessing changes the data into a format that can be processed in machine learning, and other data science tasks more quickly and efficiently.

- Data Acquisition: We collected a csv dataset with 18 columns and 2100 rows from Kaggle on crime prediction.
- Data Pre-Processing: We have chosen to concentrate on the following measures for our dataset: Making The Dataset A Time-Series Dataset Eliminating Null Elements Finding out Correlation Extracting Important Features
- Splitting The Dataset Into Training and Testing: To divide our dataset into train and test, the pandas and sklearn packages are imported.
- Data Normalization: Data is normalized to put all of the attributes on the same scale.
- Convert The Data into Tensor: The TensorFlow library's `tf.convert to tensor()` method is used to convert a NumPy array into a Tensor.

### 4.4.2 The Model Creation

- Making The Model:  
The algorithms used for our model are binary classification with logistic regression. In our model, the data's street addresses were mapped, and the coordinates of each incident were determined by creating a grid reference system. An epicenter can be determined based on the concentration of events within each grid by annotating the number of events per grid.
- Defining the Optimizer And Loss Function:  
Loss function and optimizer are two most deciding factors for a model being

a great model or an ill-trained model, we have used Mean Square Error Loss as Loss function and Adam as optimizer. A larger MSE indicates that the data points are widely dispersed around the central moment (mean), whereas a smaller MSE indicates the opposite. A lower MSE is preferable because it indicates that your data points are clustered closely around its central point (mean). It reflects your data's centralized distribution, the fact that it is not skewed, and, most importantly, it has fewer errors.

#### **4.4.3 Making A Prediction Function and Creating A Web App**

- **Making A Prediction Function:**

On the basis of the trained model, we may predict the labels of the data values using the Python `predict()` method. The data to be tested is typically the sole argument that the `predict()` function accepts. Based on the learnt or trained data generated from the model, it returns the labels of the data supplied as an argument. As a result, the `predict()` function uses the learnt label to map and predict the labels for the test data on top of the trained model.

- **Creating UI Using Streamlit:**

An open-source Python framework called Streamlit is used to create web applications for machine learning and data science. Build interactive web-based user interfaces. Using Streamlit, we quickly design and launch web applications. Working on the interactive cycle of coding and watching outcomes on the web app is made simple by Streamlit. To use Streamlit to create apps using Python.

### **4.5 Steps to execute/run/implement the project**

#### **4.5.1 Checking Requirements and Installation**

- Installing python software in the computer or laptop
- Select Version of Python to Install: Python has various versions available with differences between the syntax and working of different versions of the language. We need to choose the version which we want to use or need. install only Python 3.
- Download Python Executable Installer: On the web browser, in the official site of python ([www.python.org](http://www.python.org)), move to the Download for Windows section.

- All the available versions of Python will be listed. Select the version required by you and click on Download. Let suppose, we chose the Python 3.9.1 version. On clicking download, various available executable installers shall be visible with different operating system specifications. Choose the installer which suits your system operating system and download the installer. Let suppose, we select the Windows installer(64 bits).
- Select Version of Python to Install: Python has various versions available with differences between the syntax and working of different versions of the language. We need to choose the version which we want to use or need. install only Python 3.
- Run Executable Installer, and Download Python Executable Installer: We downloaded the Python 3.9.1 Windows 64 bit installer. Run the installer. Make sure to select both the checkboxes at the bottom and then click Install New. On the web browser, in the official site of python ([www.python.org](http://www.python.org)), move to the Download for Windows section.
- On clicking the Install Now, The installation process starts: All the available versions of Python will be listed. Select the version required by you and click on Download. Let suppose, we chose the Python 3.9.1 version. On clicking download, various available executable installers shall be visible with different operating system specifications. Choose the installer which suits your system operating system and download the installer. Let suppose, we select the Windows installer(64 bits).
- The installation process will take few minutes to complete and once the installation is successful, the following screen is displayed.
- Run Executable Installer: We downloaded the Python 3.9.1 Windows 64 bit installer. Run the installer. Make sure to select both the check boxes at the bottom and then click Install New. Verify Python is installed on Windows:
- On clicking the Install Now, The installation process starts: To ensure if Python is successfully installed on your system. Follow the given steps
- Open the command prompt: The installation process will take few minutes to complete and once the installation is successful, the following screen is displayed. Type 'python' and press enter the version of the python which you have

installed will be displayed if the python is successfully installed on your windows. Verify Python is installed on Windows.

- Verify Pip was installed: Pip is a powerful package management system for Python software packages. Thus, make sure that you have it installed. To ensure if Python is successfully installed on your system. Follow the given steps, Open the command prompt. To verify if pip was installed, follow the given steps Type 'python' and press enter.
- The version of the python which you have installed will be displayed if the python is successfully installed on your windows. Open the command prompt.
- Verify Pip was installed: Enter pip -V to check if pip was installed. Pip is a powerful package management system for Python software packages. Thus, make sure that you have it installed.
- The following output appears if pip is installed successfully. To verify if pip was installed, follow the given steps, Open the command prompt, Enter pip -V to check if pip was installed, The following output appears if pip is installed successfully.

#### 4.5.2 Installing Packages

- open file manager and click on the crime predictio file, open the file in command propmpt using command propmt and check whether python is installed.
- installing numpy: In the terminal, use the pip command to install numpy package. Once the package is installed successfully, type python to get into python prompt. Notice the python version is displayed too. Use the import command to include numpy package and use it. You can also set an alias name (shortcut) for package.
- installing matplotlib: Make sure Python and pip is preinstalled on your system
- python -version If python is successfully installed, the version of python installed on your system will be displayed.
- To check pip pip -V. The version of pip will be displayed, if it is successfully installed on your system.

- Install Matplotlib, `pip install matplotlib`. This command will start downloading and installing packages related to the matplotlib library. Once done, the message of successful installation will be displayed.
- Check if it is installed successfully: To verify that matplotlib is successfully installed on your system, execute the following command in the command prompt. If matplotlib is successfully installed, the version of matplotlib installed will be displayed.

### 4.5.3 Executing Project

- Firstly need to check requirements in the command prompt and install requirements `c:\Users\tnikh\onwdrive\Desktop\crime prediction\pip install -r requirements.txt`
- Release of pip available: 22.3.1 & 23.0.1
- `Python.exe -m pip install --upgrade pip` successfully completed
- open command prompt
- In the terminal that appears, type: `pip install streamlit`
- Test that the installation worked: `streamlit hello` Running the command
- Streamlit Crime prediction.py

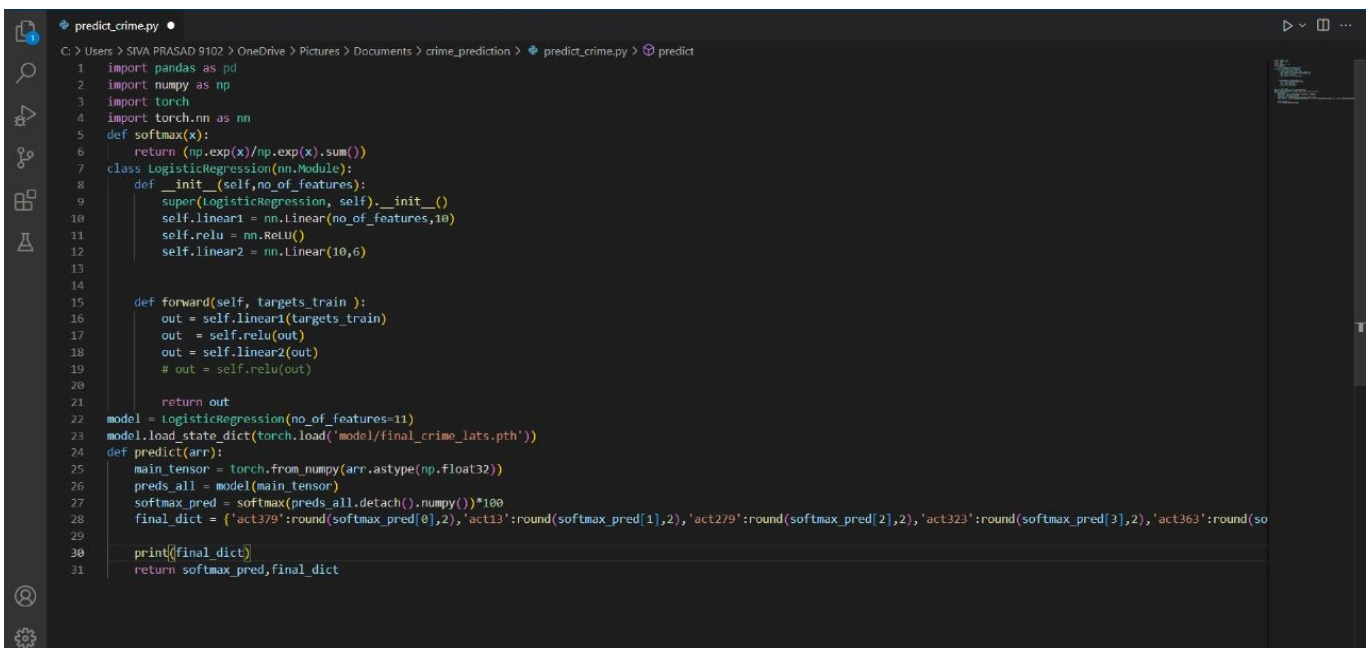
The project execution of a location-based crime type prediction using PyTorch non-linear deep-2-layer model involves data collection, cleaning, exploration, model development, web app development, and deployment. Web app development involves developing a Streamlit web app that can take input from the user and predict the type of crime that is likely to occur. Deployment involves using cloud-based services such as Streamlit. Install PyTorch, NumPy, pandas, scikit-learn, and Streamlit libraries, collect crime dataset, clean, transform, and normalise it, create a PyTorch model with a nonlinear activation function, two hidden layers, and two hidden layers. Analyze the model, create a Streamlit app, and deploy the model.

# Chapter 5

## IMPLEMENTATION AND TESTING

### 5.1 Input and Output

#### 5.1.1 Input Design



```
predict_crime.py
C:\Users> SIVA PRASAD 9102 > OneDrive > Pictures > Documents > crime_prediction > predict_crime.py > predict

1 import pandas as pd
2 import numpy as np
3 import torch
4 import torch.nn as nn
5 def softmax(x):
6     return (np.exp(x)/np.exp(x).sum())
7 class LogisticRegression(nn.Module):
8     def __init__(self,no_of_features):
9         super(LogisticRegression, self).__init__()
10        self.linear1 = nn.Linear(no_of_features,10)
11        self.relu = nn.ReLU()
12        self.linear2 = nn.Linear(10,6)
13
14
15    def forward(self, targets_train ):
16        out = self.linear1(targets_train)
17        out = self.relu(out)
18        out = self.linear2(out)
19        # out = self.relu(out)
20
21        return out
22 model = LogisticRegression(no_of_features=11)
23 model.load_state_dict(torch.load('model/final_crime_lats.pth'))
24 def predict(arr):
25     main_tensor = torch.from_numpy(arr.astype(np.float32))
26     preds_all = model(main_tensor)
27     softmax_pred = softmax(preds_all.detach().numpy())*100
28     final_dict = {'act379':round(softmax_pred[0],2),'act13':round(softmax_pred[1],2),'act279':round(softmax_pred[2],2),'act323':round(softmax_pred[3],2),'act363':round(so
29
30     print(final_dict)
31     return softmax_pred,final_dict
```

Figure 5.1: Crime Prediction Input Design

There are various phases involved in implementing a location-based crime type prediction utilising a PyTorch non-linear deep 2-layer model with a Streamlit web app project is Data Gathering and Preprocessing to get the crime statistics for the area of interest. The data is preprocessed by being cleaned, reorganised, and put into a structured format that is appropriate for machine learning. Feature engineering is the process of taking preprocessed data and extracting useful features that may be utilised to train a machine learning model. This step involves identifying key elements that have a major impact on a particular crime type's chance to occur in a particular setting. Train a PyTorch nonlinear deep 2-layer model using preprocessed and manipulated data. Construct a Streamlit web application that gets location-based inputs and generates a forecasted crime category based on the model.



### 5.1.2 Output Design

The image shows a web application interface titled "Location Based Crime Prediction". At the top, there is a header image featuring a magnifying glass held by a gloved hand over yellow crime scene tape that reads "DO NOT ENTER". Below the header, the interface consists of several input fields for user data: "Enter Date" (with sub-fields for Day, Month, and Year), "Hour", "Minutes", and "Seconds". There are also "Latitude" and "Longitude" input fields. At the bottom, there is a "Predict Crime" button. The entire interface is set against a dark background.

Figure 5.2: Crime Prediction Output Design

The suggested system makes use of real-time information and takes advantage of a number of variables, including place, time, date, and percentage of crime rate. The suggested system makes use of multi-modal machine learning. The employment of numerous classifiers for class prediction results in greater accuracy. The suggested model performs well when tested against test and train data, and it will also produce improved results when tested against real-time data. A comparison of the suggested model with other models using cutting-edge techniques revealed that our model outperformed the others in terms of effectiveness, accuracy, and processing speed. It is a quick, easy, affordable, and extremely accurate way to spot patterns. The current system would cluster crime data for all cognizable crimes, such as murder, theft, cheating, abduction, crime against women, robbery, and other similar crimes, using the binary classification with logistic regression technique.

## 5.2 Testing

Testing is a collection of methods to evaluate an application's suitability for use in accordance with a predetermined script, however testing is not able to detect every application flaw. Testing is a process used to identify every conceivable flaw or weak spot in a piece of merchandise. There are many different kinds of tests.

## 5.3 Types of Testing

### 5.3.1 Unit testing

A type of software testing called unit testing examines the individual components or add-ons of a software programme. The goal is to ensure that every piece of software programme code functions as intended. Verify the model's ability to accurately load and preprocess the dataset. Test the layers have the necessary number of nodes and that the model architecture has been implemented correctly.

#### Input

```
1 import pandas as pd
2 import os
3 data_frame_main = pd.read_csv("data.csv")
4 dataset=pd.read_csv('data.csv')
5 data_frame_main['timestamp'] = pd.to_datetime(data_frame_main['timestamp'], errors='coerce')
6 data_frame_main['timestamp'] = pd.to_datetime(data_frame_main['timestamp'], format = '%d/%m/%Y %H:%M:%S')
7 column_1 = data_frame_main.iloc[1:,0]
8 db=pd.DataFrame({"year": column_1.dt.year,
9                  "month": column_1.dt.month,
10                 "day": column_1.dt.day,
11                 "hour": column_1.dt.hour,
12                 "dayofyear": column_1.dt.dayofyear,
13                 "week": column_1.dt.week,
14                 "weekofyear": column_1.dt.weekofyear,
15                 "dayofweek": column_1.dt.dayofweek,
16                 "quarter": column_1.dt.quarter, })
17 date_time = " ".join([main_date,main_time])
18 data1=pd.c os.listdir(os.getcwd()):
19     inputs_arr = np.array(inputs)
20 file = "main_data.csv"
21 if file insv(file,index=False)
22     print("File saved")
23     data = df.groupby('weekday').count().iloc[:, 0]
24 data = data.reindex([ 1.0, 2.0, 3.0, 4.0, 5.0, 6.0,7.0])
25 plt.figure(figsize=(10, 5))
26 with sns.axes_style("whitegrid"):
27     ax = sns.barplot(data.index, (data.values / data.values.sum()) * 100,orient='v',palette=cm.
28                        ScalarMappable(cmap='Reds').to_rgba(data.values))
29 plt.title('Incidents per Weekday', fontdict={'fontsize': 16})
30 plt.xlabel('Weekday')
31 plt.show()
32 plt.savefig('images/weekday_graph.png')
```

### 5.3.2 Unit Testing Result

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	year	month	day	hour	dayofyear	week	weekofyear	dayofweek	weekday	quarter	act379	act13	act279	act323	act363	act302	latitude	longitude
2	2018	2	28	21	59	9	9	2	2	1	1	0	0	0	0	0	22.72099	75.87608
3	2018	2	28	10	59	9	9	2	2	1	0	0	1	0	0	0	22.73668	75.88317
4	2018	2	28	10	59	9	9	2	2	1	0	0	1	0	0	0	22.74653	75.88714
5	2018	2	28	10	59	9	9	2	2	1	0	0	1	0	0	0	22.76953	75.88877
6	2018	2	28	14	59	9	9	2	2	1	0	0	0	1	0	0	22.73522	75.91337
7	2018	2	28	8	59	9	9	2	2	1	0	0	0	0	1	0	22.73677	75.90858
8	2018	2	28	3	59	9	9	2	2	1	0	0	0	1	0	0	22.73677	75.90858
9	2018	2	28	1	59	9	9	2	2	1	1	0	0	0	0	0	22.72225	75.91529
10	2018	2	28	19	59	9	9	2	2	1	0	0	0	1	0	0	22.72225	75.91529
11	2018	2	28	14	59	9	9	2	2	1	0	0	0	1	0	0	22.74166	75.86914
12	2018	2	28	14	59	9	9	2	2	1	0	0	0	1	0	0	22.74166	75.86914
13	2018	2	28	22	59	9	9	2	2	1	0	1	0	0	0	0	22.74688	75.88317
14	2018	2	28	22	59	9	9	2	2	1	0	0	0	1	0	0	22.74453	75.93142
15	2018	2	28	22	59	9	9	2	2	1	0	0	0	1	0	0	22.76577	75.87046
16	2018	2	28	12	59	9	9	2	2	1	1	0	0	0	0	0	22.76781	75.87598
17	2018	2	28	13	59	9	9	2	2	1	0	0	0	0	1	0	22.71521	75.87005
18	2018	2	28	3	59	9	9	2	2	1	0	0	0	1	0	0	22.69834	75.91076
19	2018	2	28	23	59	9	9	2	2	1	0	0	0	1	0	0	22.69411	75.90422
20	2018	2	28	19	59	9	9	2	2	1	1	0	0	0	0	0	22.71154	75.86011
21	2018	2	28	11	59	9	9	2	2	1	0	0	1	0	0	0	22.71512	75.85666
22	2018	2	28	11	59	9	9	2	2	1	0	0	0	1	0	0	22.71996	75.84366
23	2018	2	28	18	59	9	9	2	2	1	0	0	1	0	0	0	22.71996	75.84366
24	2018	2	28	1	59	9	9	2	2	1	0	0	0	1	0	0	22.72725	75.85321
25	2018	2	28	0	59	9	9	2	2	1	1	0	0	0	0	0	22.72692	75.85158
26	2018	2	28	22	59	9	9	2	2	1	0	0	1	0	0	0	22.72692	75.85158
27	2018	2	28	22	59	9	9	2	2	1	0	1	0	0	0	0	22.72692	75.85158

Enter Date

29 - +

Enter Month

7 - +

Enter Year

2022 - +

Hour

23 - +

Minutes

15 - +

Seconds

0 - +

Latitude

22.720992 - +

Longitude

75.876083 - +

Predict Crime

Figure 5.3: Unit Testing Output

### 5.3.3 Unit Testing Description

Unit testing is the design of test cases to validate that internal program logic is functioning properly and that program inputs produce valid outputs. The Figure 5.3

ensures that each unique path of a business process performs accurately and contains clearly defined inputs and expected results. To make sure the system is operating as intended, each component should be checked separately. This involves evaluating the Streamlit web application, the PyTorch model, and any other auxiliary libraries.

### 5.3.4 Integration testing

Test the web application's ability to retrieve input data from the user interface, pass it to the model, display the predicted crime type, and handle errors and exceptions. Errors may be caused by incompatibility between software modules.

#### Input

```
1 class LogisticRegression(nn.Module):
2     def __init__(self, no_of_features):
3         super(LogisticRegression, self).__init__()
4         self.linear1 = nn.Linear(no_of_features, 10)
5         self.relu = nn.ReLU()
6         self.linear2 = nn.Linear(10, 6)
7     def forward(self, targets_train):
8         out = self.linear1(targets_train)
9         out = self.relu(out)
10        out = self.linear2(out)
11        return out
12 model = LogisticRegression(no_of_features=no_features)
13     with torch.no_grad():
14         targets_preds = model(inputs_test)
15         targets_preds_cls = targets_preds.round()
16 torch.save(model.state_dict(), 'model/final_crime_lats.pth')
17     plt.plot(loss_hist)
18 for i in inputs_train:
19     pred = model(i)
20     print(pred)
21 plt.axvline(x=data.median(), ymax=0.95, linestyle='--', color=col[1])
22 plt.annotate('Median: ' + str(data.median()),
23             xy=(data.median(), 0.004),
24             xytext=(200, 0.005),
25             arrowprops=dict(arrowstyle='->', color=col[1], shrinkB=10))
26 plt.title(
27     'Distribution of number of incidents per day', fontdict={'fontsize': 16})
28 plt.xlabel('Incidents')
29 plt.ylabel('Density')
30 plt.show()
31 plt.savefig("distribution.png")
```

### 5.3.5 Integration Testing Result

```
... Output exceeds the size limit. Open the full output data in a text editor
Epoch: 10, Loss: 60.6170, ACC: 0.45169082283973694
Epoch: 20, Loss: 56.8374, ACC: 0.3840579688549042
Epoch: 30, Loss: 53.2638, ACC: 0.36231884360313416
Epoch: 40, Loss: 49.9021, ACC: 0.35748791694641113
Epoch: 50, Loss: 46.7511, ACC: 0.35748791694641113
Epoch: 60, Loss: 43.8046, ACC: 0.36231884360313416
Epoch: 70, Loss: 41.0563, ACC: 0.3478260934352875
Epoch: 80, Loss: 38.4860, ACC: 0.3478260934352875
Epoch: 90, Loss: 36.0678, ACC: 0.32125604152679443
Epoch: 100, Loss: 33.7819, ACC: 0.28743961453437805
Epoch: 110, Loss: 31.6358, ACC: 0.2753623127937317
Epoch: 120, Loss: 29.6267, ACC: 0.3550724685192108
Epoch: 130, Loss: 27.7454, ACC: 0.4855072498321533
Epoch: 140, Loss: 25.9764, ACC: 0.5483092069625854
Epoch: 150, Loss: 24.3140, ACC: 0.5410627722740173
Epoch: 160, Loss: 22.7611, ACC: 0.483091801404953
Epoch: 170, Loss: 21.3161, ACC: 0.4227053225040436
Epoch: 180, Loss: 19.9736, ACC: 0.4468598961830139
Epoch: 190, Loss: 18.7182, ACC: 0.43719807267189026
Epoch: 200, Loss: 17.5437, ACC: 0.5048308968544006
Epoch: 210, Loss: 16.4502, ACC: 0.5531401038169861
Epoch: 220, Loss: 15.4310, ACC: 0.5869565010070801
Epoch: 230, Loss: 14.4820, ACC: 0.6376811861991882
Epoch: 240, Loss: 13.6031, ACC: 0.6545893549919128
Epoch: 250, Loss: 12.7915, ACC: 0.7028985619544983
...
Epoch: 14970, Loss: 0.1128, ACC: 5.043478488922119
Epoch: 14980, Loss: 0.1128, ACC: 5.043478488922119
Epoch: 14990, Loss: 0.1127, ACC: 5.043478488922119
Epoch: 15000, Loss: 0.1127, ACC: 5.043478488922119
```

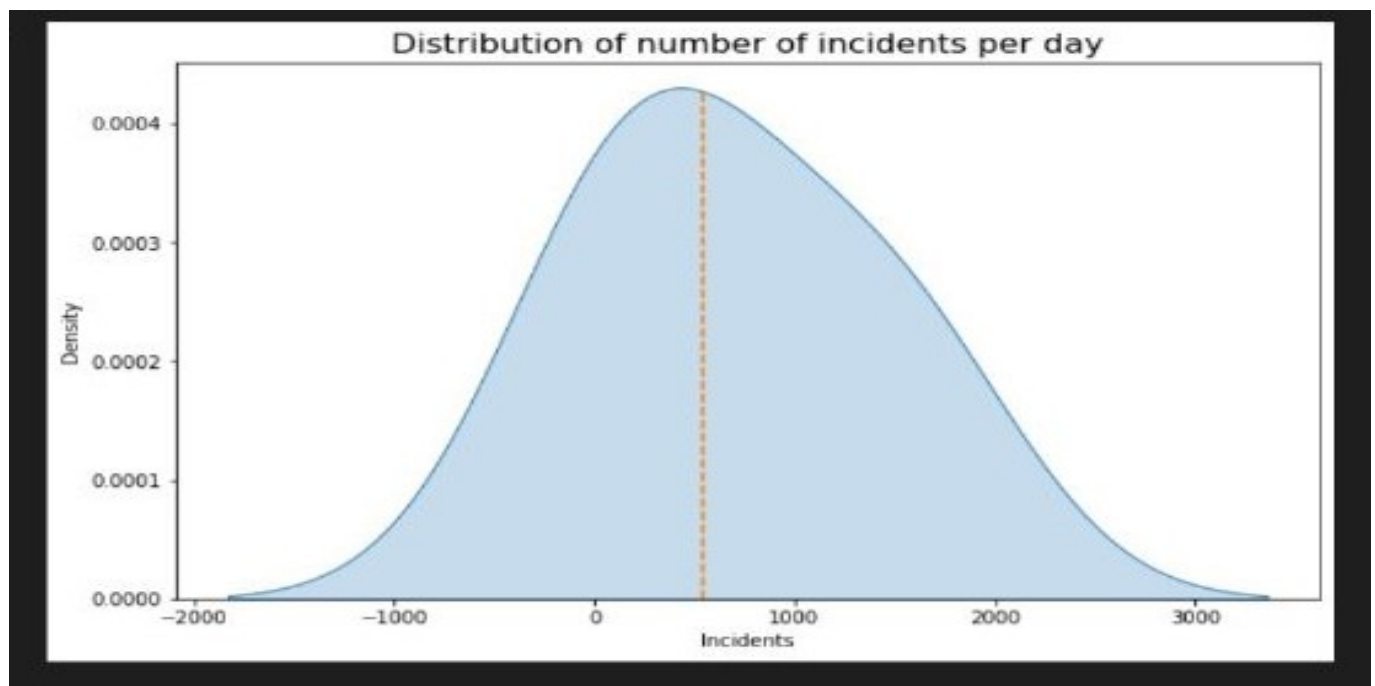


Figure 5.4: Integration Testing Output

### 5.3.6 Integration Testing Description

Integration tests are designed to test integrated software components to ensure that they run as one program. The Figure 5.4 that shows the necessary to ensure that the data these technologies accept is accurate, as modules often interface with third-party APIs or tools. To make sure that the system's many parts function as a coherent

whole, the components should be evaluated in concert. This involves evaluating the interactions and data transfer between the Streamlit web application and the PyTorch model.

### 5.3.7 System testing

It is a sort of software program checking out wherein the device is examined in opposition to the useful necessities/specifications. Functional checking out guarantees that the necessities are nicely glad through the application. The Figure 5.5, it attempts to execute the check instances and examine the outcomes and test the accuracy. Establish the test objectives and create the test environment. The test environment should be configured to resemble the production environment, with the web application, software and dependencies, and PyTorch model trained and prepared for use.

#### Input

```
1 import pandas as pd
2 def softmax(x):
3     return (np.exp(x)/np.exp(x).sum())
4 class LogisticRegression(nn.Module):
5     def __init__(self, no_of_features):
6         super(LogisticRegression, self).__init__()
7         self.linear1 = nn.Linear(no_of_features,10)
8         self.relu = nn.ReLU()
9         self.linear2 = nn.Linear(10,6)
10 model = LogisticRegression(no_of_features=11)
11 model.load_state_dict(torch.load('model/final_crime_lats.pth'))
12 def predict(arr):
13     main_tensor = torch.from_numpy(arr.astype(np.float32))
14     final_dict= {'act379':round(softmax_pred[0],2), 'act13':round(softmax_pred[1],2), 'act279':round(
15         softmax_pred[2],2), 'act323':round(softmax_pred[3],2), 'act363':round(softmax_pred[4],2), '
16         act302':round(softmax_pred[5],2)}
17     print(final_dict)
18     return softmax_pred, final_dict
19 data = df.groupby(['hour', 'day', 'type'],
20                 as_index=False).count().iloc[:, :4]
21 data.rename(columns={'Dates': 'Incidents'}, inplace=True)
22 data = data.groupby(['hour', 'type'], as_index=False).mean()
23 data = data.loc[data['type'].isin(
24     ['act379', 'act13', 'act279', 'act323', 'act363', 'act302'])]
25 sns.set_style("whitegrid")
26 fig, ax = plt.subplots(figsize=(14, 4))
```

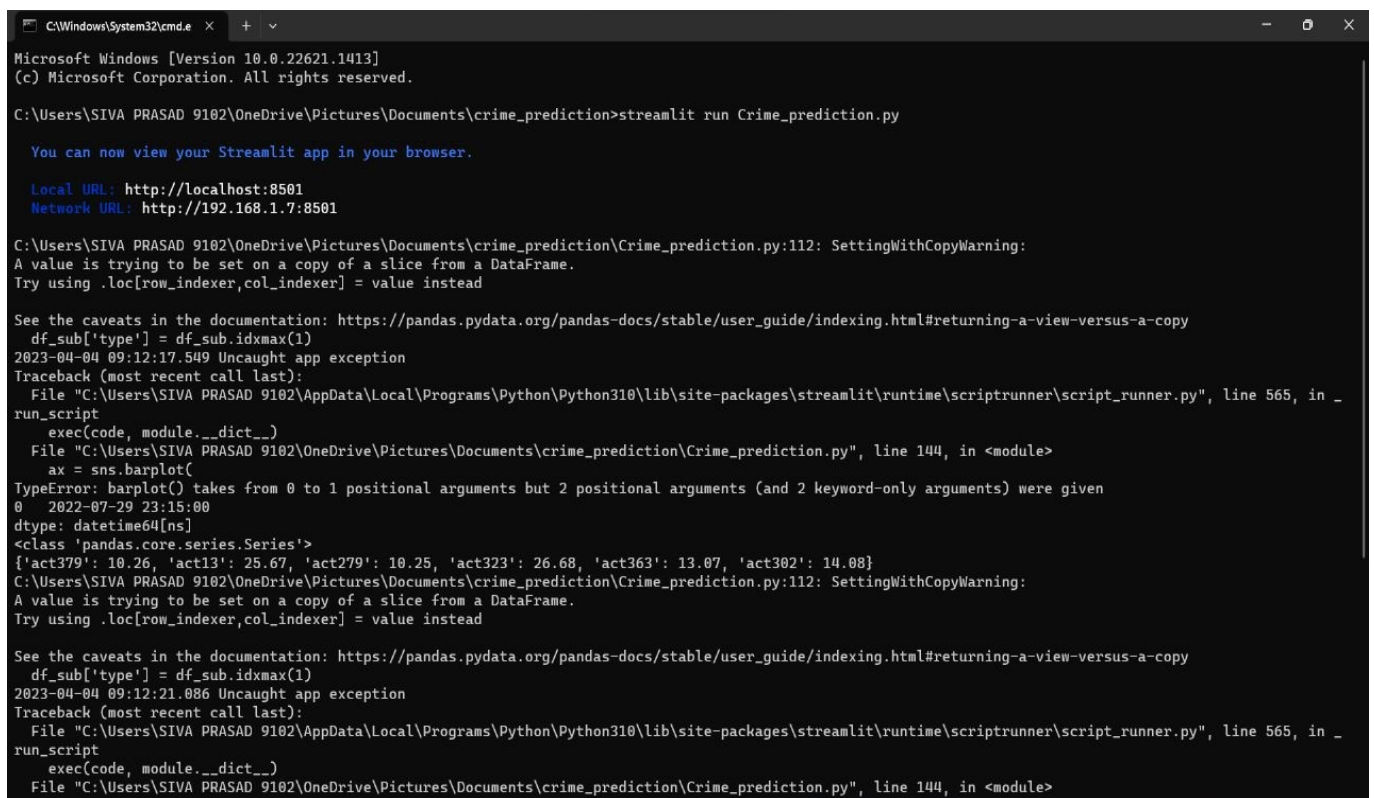


```

26 ax = sns.lineplot(x='hour', y='day', data=data, hue='type')
27 ax.legend(loc='upper center', bbox_to_anchor=(0.5, 1.15), ncol=6)
28 plt.suptitle('Average number of incidents per hour')
29 fig.tight_layout(rect=[0, 0.03, 1, 0.95])
30 plt.show()
31 ## How to setup ?
32 '''bash
33 cd path/to/dir
34 python -m pip install -r requirements.txt
35 '''
36 ## How to run ?
37 '''bash
38 python -m streamlit run Crime_prediction.py
39 ''' streamlit run Crime_prediction.py

```

### 5.3.8 System Testing Result



```

C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.22621.1413]
(c) Microsoft Corporation. All rights reserved.

C:\Users\SIVA PRASAD 9102\OneDrive\Pictures\Documents\crime_prediction>streamlit run Crime_prediction.py

You can now view your Streamlit app in your browser.

Local URL: http://localhost:8501
Network URL: http://192.168.1.7:8501

C:\Users\SIVA PRASAD 9102\OneDrive\Pictures\Documents\crime_prediction\Crime_prediction.py:112: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
df_sub['type'] = df_sub.idxmax(1)
2023-04-04 09:12:17.549 Uncaught app exception
Traceback (most recent call last):
  File "C:\Users\SIVA PRASAD 9102\AppData\Local\Programs\Python\Python310\lib\site-packages\streamlit\runtime\scriptrunner\script_runner.py", line 565, in _
run_script
    exec(code, module.__dict__)
  File "C:\Users\SIVA PRASAD 9102\OneDrive\Pictures\Documents\crime_prediction\Crime_prediction.py", line 144, in <module>
    ax = sns.barplot(
TypeError: barplot() takes from 0 to 1 positional arguments but 2 positional arguments (and 2 keyword-only arguments) were given
0 2022-07-29 23:15:00
dtype: datetime64[ns]
<class 'pandas.core.series.Series'>
{'act379': 10.26, 'act13': 25.67, 'act279': 10.25, 'act323': 26.68, 'act363': 13.07, 'act302': 14.08}
C:\Users\SIVA PRASAD 9102\OneDrive\Pictures\Documents\crime_prediction\Crime_prediction.py:112: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
df_sub['type'] = df_sub.idxmax(1)
2023-04-04 09:12:21.086 Uncaught app exception
Traceback (most recent call last):
  File "C:\Users\SIVA PRASAD 9102\AppData\Local\Programs\Python\Python310\lib\site-packages\streamlit\runtime\scriptrunner\script_runner.py", line 565, in _
run_script
    exec(code, module.__dict__)
  File "C:\Users\SIVA PRASAD 9102\OneDrive\Pictures\Documents\crime_prediction\Crime_prediction.py", line 144, in <module>

```

Figure 5.5: System Testing Output

### 5.3.9 System Testing Description

System testing ensures that the entire integrated software system meets requirements. System testing is based on process descriptions and flows, emphasizing pre-

driven process links and integration points Create test cases, run the tests, and Examine the outcomes, Reporting and resolving problems After the problems are fixed retest. Here in the figure 5.5 System testing and performance testing should be done to ensure the system is operating as intended and can handle large requests without crashing or slowing down. User acceptance testing should be conducted to ensure the system meets the needs and expectations of its intended users.

### 5.3.10 Test Result

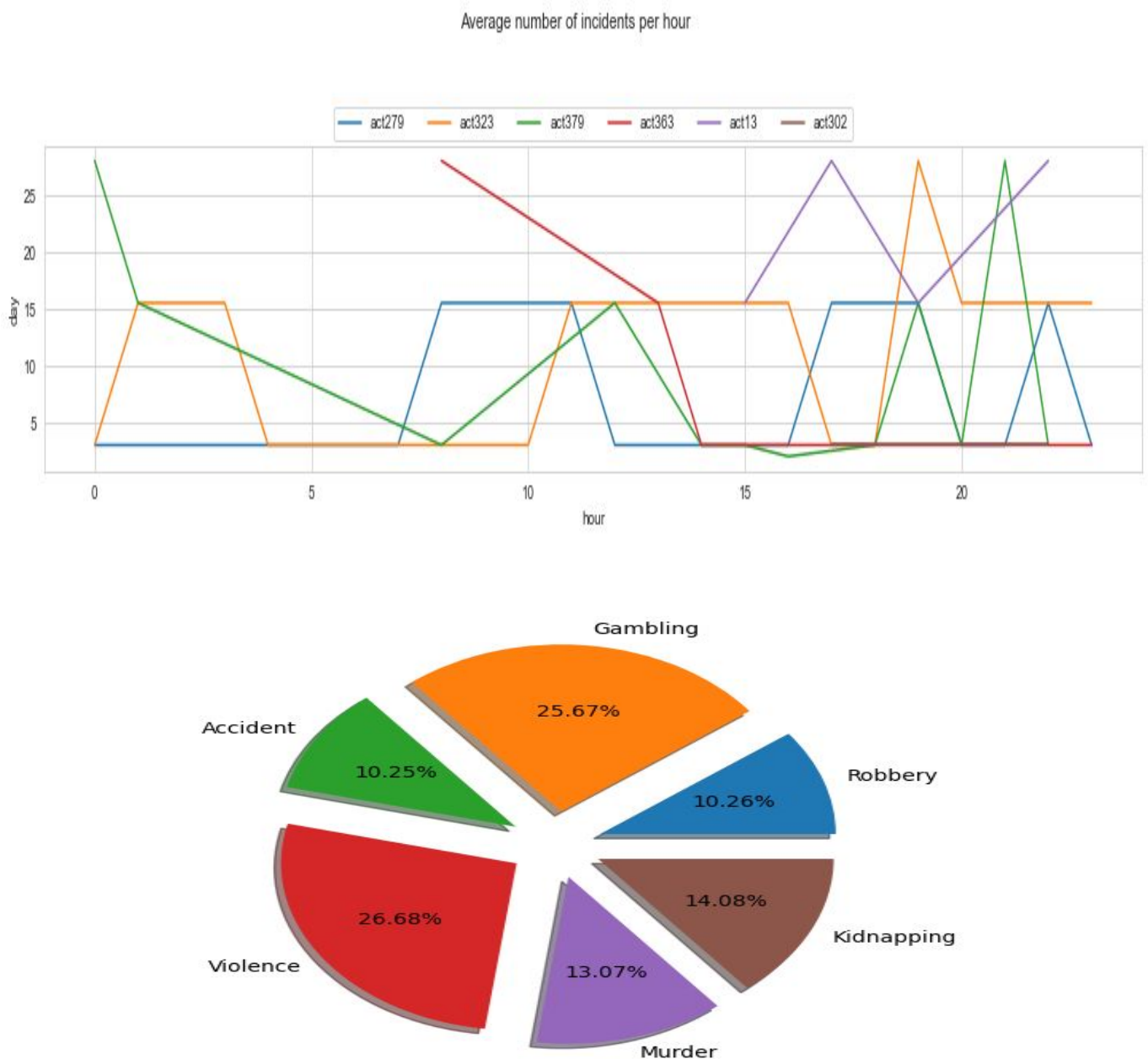


Figure 5.6: Types of Crime Occurred



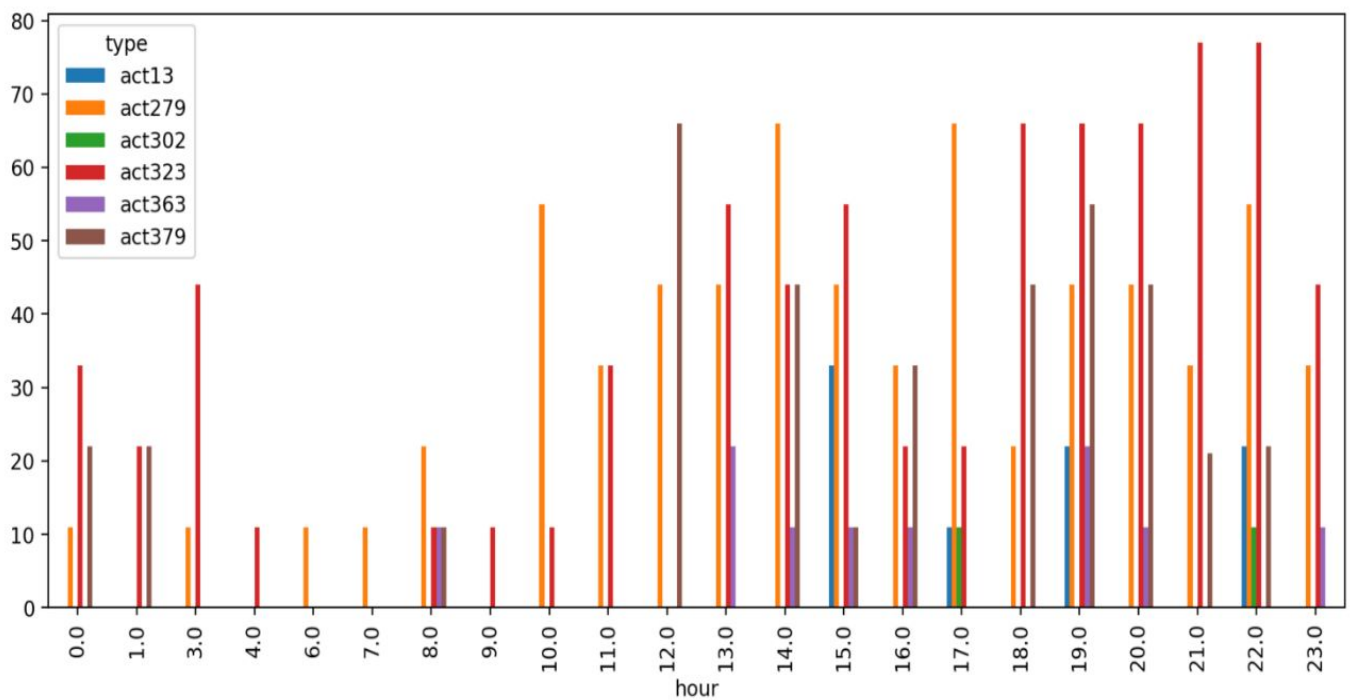


Figure 5.7: **Reported Crimes by hour**

### 5.3.11 Test Result Description

The most important details in this text are the processes involved in location-based crime type prediction combining a PyTorch non-linear deep 2-layer model with a Streamlit web application. Data must be gathered and preprocessed to gather information about the time and place of crimes, as well as the nature of the crime. To train a machine learning model, the data must be cleaned and processed, and the PyTorch deep learning model must be developed to provide the neural network's architecture. The preprocessed data are then used to train the model. Once the model is trained, it can be deployed as a Streamlit web app. The app allows users to input a location and receive a prediction of the most likely type of crime to occur in that location. The app may also provide additional information, such as the probability of the prediction as shown in the figure 5.6 and a visualization of the data also. The typical crime occurred at the specific time when the crime could have been committed. Overall, this approach can be useful for predicting crime patterns and helping law enforcement agencies allocate resources more effectively. The crime occurred at a specific period of time, and the act of crime that is mandated by law for the specific crime, according to the figure 5.7, which allows individuals to demonstrate this. The PyTorch deep learning model and Streamlit web app can be used to predict crime.

## Chapter 6

# RESULTS AND DISCUSSIONS

### 6.1 Efficiency of the Proposed System

The proposed system leverages a multi-modal machine learning approach. It proves to be more accurate due to the use of multiple classifiers for class prediction. The proposed model works well against train data and test data further this model will provide better results for real-time data. It is divided into a trained data set and a test data set. Both the training and testing datasets are used to train the model. To make classification easier, the crime type, year, month, time, date, and location are all mapped to integers. The proposed system leverages a multi-modal machine learning approach. Should set up the Streamlit which is an open-source Python library that makes it easy to build custom web apps for machine learning and data science. The first step is to install the Streamlit library, which can be accomplished with the pip command. The using of a Python virtual environment to separate your dependencies for each project. Streamlit makes it easy to work on the interactive cycle of coding and viewing results on the web app. Python code is used to build apps with Streamlit.

The proposed model works well against train data and test data further this model will provide better results for real-time data. This obtained a 2100-row, 18-column CSV dataset on crime prediction from Kaggle. For our dataset, we have decided to focus on the metrics by Creating a time-series dataset from the data, Getting Rid of Null Elements, Identifying Correlation and Extracting Key Attributes. It reflects your data's centralized distribution, it has fewer errors. To put all of the attributes on the same scale, data is normalised. A NumPy array is transformed into a Tensor using the `tf.convert to tensor()` method of the TensorFlow library. Using the Python `predict()` method, we can forecast the labels of the data values based on the trained model. The `predict()` function normally only accepts one input, the test data. In order to map and predict the labels for the test data on top of the training model, the `predict()` function employs the learned label. Web applications for machine learning and data science are developed using the open-source Python framework Streamlit.

## 6.2 Comparison of Existing and Proposed System

### **Existing system:(SVM, Naive Bayes and Random Forest model)**

In the currently used methods, such SVM and naive Bayes algorithms. The implied prediction's performance is then assessed in order to obtain a very low degree of accuracy when compared to the previously employed model. In particular, the CNN-based crime type prediction model performs more accurately than SVM and naive Bayes algorithms by 7 percent and 8 percent respectively. It offers a basic framework for subsequent classification procedures. The crime prediction aids in foreseeing future instances of any kind of criminal activity and assists the authorities in swiftly resolving them. This data were used to train a Random Forest model. With less accuracy in the present model, the Random Forest approach for classification and regression shows the final prediction interface along with a demo input and output. The low accuracy of the pre-existing works is explained by the classifier's usage of categorical values, which leads to a biased result for the nominal qualities with higher values.

### **Proposed system:(Binary Classification with Logistic Regression)**

In the proposed system is done using real-time data sets and also we are leveraging various factors like location, time, date and rate of crimes percentage as well. A comparative analysis of the proposed model with state-of-the-art methods assessed that our model was able to outperform the other models in terms of efficiency, accuracy, and processing time. It is a fast, cost-efficient, quick, and highly accurate method to identify patterns which set up in the Streamlit. This model's methods combine logistic regression and binary classification Concept of multi of novel pattern extraction against specified datasets is carried out. The prediction can be involved it is used to determine the crime rate and contrast it with the crime rate for each year based on the data this can be done in order to lower the crime rate. Crime visualisation is done using crime prediction using binary classification and logistic regression algorithms. Using the streamlit tool, an algorithm that is more effective than the previous classification algorithms is required to categorise the data Different algorithms efficiency of prediction accuracy is comparative better than those previously published, indicating improved performance.

## 6.3 Sample Code

```
1 import pandas as pd
2 import numpy as np
3 import torch
4 import torch.nn as nn
5 def softmax(x):
6     return (np.exp(x)/np.exp(x).sum())
7 class LogisticRegression(nn.Module):
8     def __init__(self, no_of_features):
9         super(LogisticRegression, self).__init__()
10        self.linear1 = nn.Linear(no_of_features,10)
11        self.relu = nn.ReLU()
12        self.linear2 = nn.Linear(10,6)
13    def forward(self, targets_train ):
14        out = self.linear1(targets_train)
15        out = self.relu(out)
16        out = self.linear2(out)
17        out = self.relu(out)
18        return out
19 model = LogisticRegression(no_of_features=11)
20 model.load_state_dict(torch.load('model/final_crime_lats.pth'))
21 def predict(arr):
22     final_dict = {'act379':round(softmax_pred[0],2), 'act13':round(softmax_pred[1],2), 'act279':round(
23         softmax_pred[2],2), 'act323':round(softmax_pred[3],2), 'act363':round(softmax_pred[4],2), '
24         act302':round(softmax_pred[5],2)}
25     print(final_dict)
26     return softmax_pred, final_dict
27 st.title("Location Based Crime Prediction")
28 st.image("https://hhsadvocate.com/wp-content/uploads/2021/12/TrueCrime-2.jpeg")
29 st.markdown("***")
30 a,b,c = st.columns(3)
31 year = c.number_input(label="Enter Year", value=2022, min_value=1900, max_value=3000)
32 date_in = a.number_input("Enter Date", value=29, max_value=31, min_value=1)
33 month_in = b.number_input("Enter Month", value=7, max_value=12, min_value=1)
34 min_in = b.number_input(label="Minutes", value=15, max_value=59, min_value=0)
35 hour_in = a.number_input(label="Hour", value=23, max_value=23, min_value=0)
36 sec_in = c.number_input(label="Seconds", value=0, max_value=59, min_value=0)
37 c,d = st.columns(2)
38 latitude = c.number_input(label="Latitude", value=22.720992, format="%f")
39 longitude = d.number_input(label="Longitude", value=75.876083, format="%f")
40 enter = st.button(label="Predict Crime")
41 st.markdown("***")
42 if enter:
43     if len(str(month_in)) == 1:
44         month_in = "0"+str(month_in)
45     if len(str(hour_in)) == 1:
46         hour_in = "0"+str(hour_in)
47     if len(str(sec_in)) == 1:
```

```

46     sec_in = "0"+str(sec_in)
47     main_date = '-'.join([str(year),str(month_in),str(date_in)])
48     main_time = ':'.join([str(hour_in),str(min_in),str(sec_in)])
49     date_time = " ".join([main_date,main_time])
50     sr = pd.Series([date_time])
51     sr = pd.to_datetime(sr)
52     print(sr)
53     print(type(sr.dt.month))
54     main_ins = {
55         "month": sr.dt.month.tolist()[0],
56         "day": sr.dt.day.tolist()[0],
57         "hour": sr.dt.hour.tolist()[0],
58         "dayofyear": sr.dt.dayofyear.tolist()[0],
59         "week": sr.dt.week.tolist()[0],
60         "weekofyear": sr.dt.weekofyear.tolist()[0],
61         "dayofweek": sr.dt.dayofweek.tolist()[0],
62         "weekday": sr.dt.weekday.tolist()[0],
63         "quarter": sr.dt.quarter.tolist()[0],
64     }
65     inputs = list(main_ins.values())
66     inputs.append(latitude)
67     inputs.append(longitude)
68     inputs_arr = np.array(inputs)
69     labels = ['Robbery','Gambling','Accident','Violence','Murder','Kidnapping']
70     vals,dicts = predict(inputs_arr)
71 df = pd.read_csv("main_data.csv")
72 st.markdown("Crime on hourly basis plotted yearly")
73 data = df.groupby(['hour', 'day', 'type'], as_index=False).count().iloc[:, :4]
74 data.rename(columns={'Dates': 'Incidents'}, inplace=True)
75 data = data.groupby(['hour', 'type'], as_index=False).mean()
76 data = data.loc[data['type'].isin(['act379', 'act13', 'act279','act323','act363', 'act302'])]
77 ax = sns.lineplot(x='hour', y='day', data=data, hue='type')
78 ax.legend(loc='upper center', bbox_to_anchor=(0.5, 1.15), ncol=6)
79 plt.suptitle('Average number of incidents per hour')
80 fig.tight_layout(rect=[0, 0.03, 1, 0.95])
81 df = pd.read_csv("main_data.csv")
82 x = df['year']
83 y = df['act279']
84 plt.plot(x,y)
85 plt.show()
86 plt.plot(df['year'])
87 def create_gdf(df):
88     gdf = df.copy()
89     gdf['Coordinates'] = list(zip(gdf.latitude , gdf.longitude))
90     gdf.Coordinates = gdf.Coordinates.apply(Point)
91     gdf = gpd.GeoDataFrame(
92         gdf, geometry='Coordinates', crs={'init': 'epsg:4326'})
93     return gdf
94 geo_df = create_gdf(df)
95 ax = world.plot(color='white', edgecolor='black')

```


```

96 geo_df.plot(ax=ax, color='red')
97 plt.show()
98 df_sub = df[['act379', 'act13', 'act279', 'act323', 'act363', 'act302']]
99 df_sub.head()
100 df_sub['type'] = df_sub.idxmax(1)
101 df_sub.head()
102 type_column = df_sub.type
103 data = df.groupby('weekday').count().iloc[:, 0]
104 data = data.reindex([1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0])
105 plt.figure(figsize=(10, 5))
106 with sns.axes_style("whitegrid"):
107     ax = sns.barplot(
108         data.index, (data.values / data.values.sum()) * 100,
109         orient='v',
110         palette=cm.ScalarMappable(cmap='Reds').to_rgba(data.values))
111 plt.title('Incidents per Weekday', fontdict={'fontsize': 16})
112 plt.xlabel('Weekday')
113 plt.ylabel('Incidents (%)')
114 plt.savefig('images/weekday-graph.png')
115 col = sns.color_palette()
116 plt.figure(figsize=(10, 6))
117 data = df.groupby('day').count().iloc[:, 0]
118 sns.kdeplot(data=data, shade=True)
119 plt.axvline(x=data.median(), ymax=0.95, linestyle='--', color=col[1])
120 plt.savefig("distribution.png")
121 data = df.groupby(['hour', 'day', 'type'], as_index=False).count().iloc[:, :4]
122 data.rename(columns={'Dates': 'Incidents'}, inplace=True)
123 data = data.groupby(['hour', 'type'], as_index=False).mean()
124 data = data.loc[data['type'].isin(['act379', 'act13', 'act279', 'act323', 'act363', 'act302'])]
125 sns.set_style("whitegrid")
126 fig, ax = plt.subplots(figsize=(14, 4))
127 ax = sns.lineplot(x='hour', y='day', data=data, hue='type')
128 ax.legend(loc='upper center', bbox_to_anchor=(0.5, 1.15), ncol=6)
129 plt.suptitle('Average number of incidents per hour')
130 fig.tight_layout(rect=[0, 0.03, 1, 0.95])
131 df.type.value_counts()
132 df.type.describe()
133 plt.figure(figsize = [4,4])
134 df.type.value_counts(normalize = True).plot.barh()
135 df.groupby(['year', 'type']).size().plot(kind='barh', figsize=(20, 20), grid=True)
136 a = df.pivot_table(index='hour', columns='type', values='year', aggfunc='size')
137 a.plot(kind='bar', figsize=(12, 5))
138 model = LogisticRegression(no_of_features=11)
139 final_dict = {'act379': round(softmax_pred[0], 2), 'act13': round(softmax_pred[1], 2), 'act279': round(
    softmax_pred[2], 2), 'act323': round(softmax_pred[3], 2), 'act363': round(softmax_pred[4], 2), 'act302':
    round(softmax_pred[5], 2)}
140 print(final_dict)
141 return softmax_pred, final_dict

```

## Output

### Location Based Crime Prediction



Enter Date: 19 -- + Enter Month: 6 -- + Enter Year: 2021 -- +

Hour: 21 -- + Minutes: 12 -- + Seconds: 2 -- +

Latitude: 17.385 -- + Longitude: 78.4867 -- +

[Predict Crime](#)

Figure 6.1: Input Data to Predict Crime

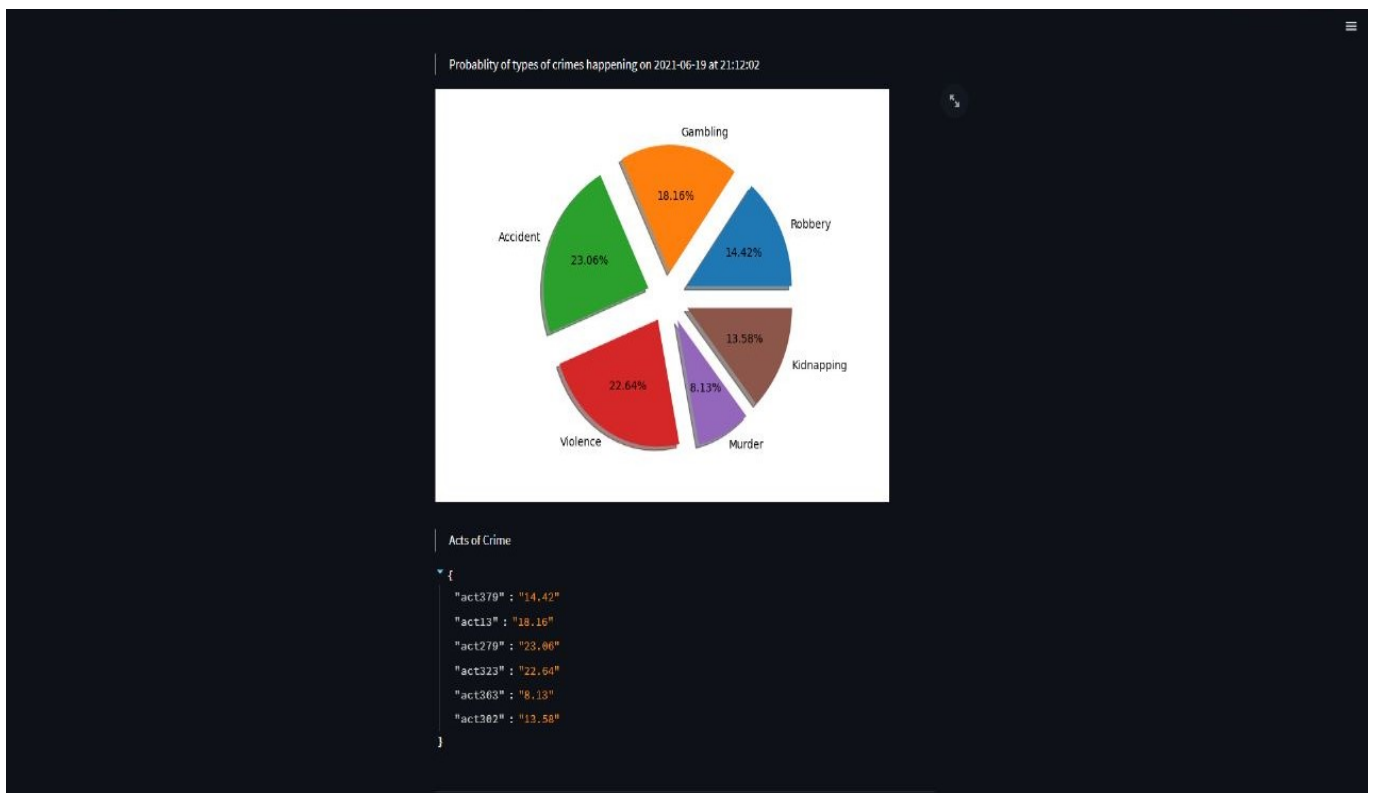


Figure 6.2: Probability of Types and Acts of Crime

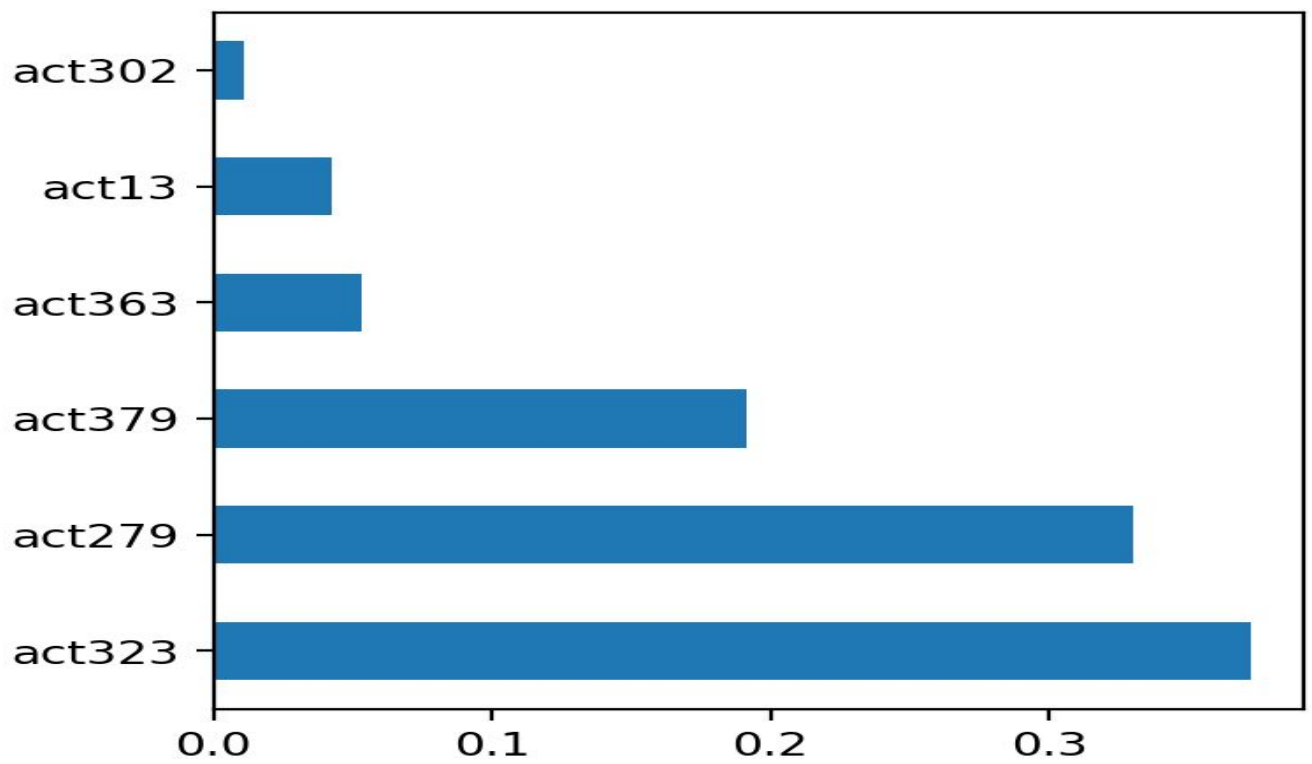


Figure 6.3: Occurrence of crime in Act

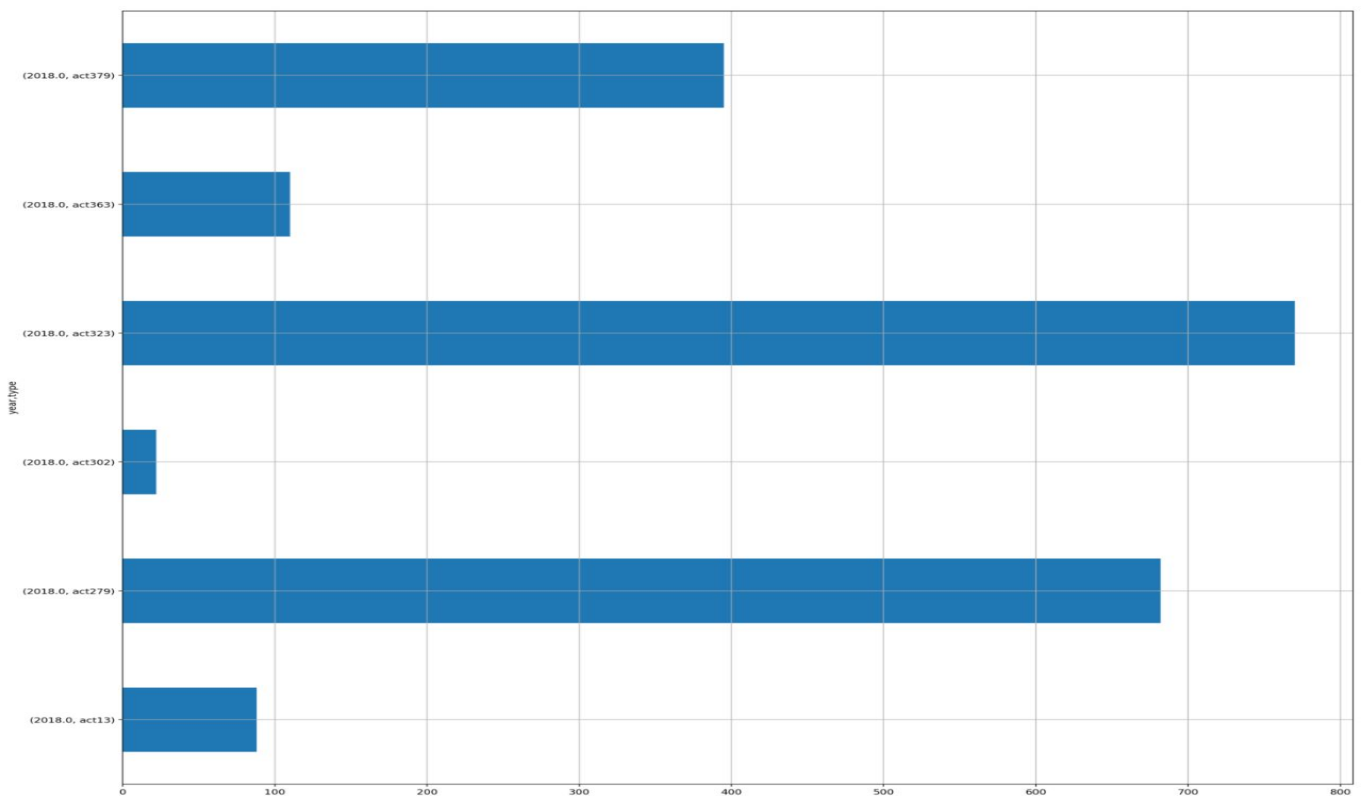


Figure 6.4: Relationship Between Year And Crime Category



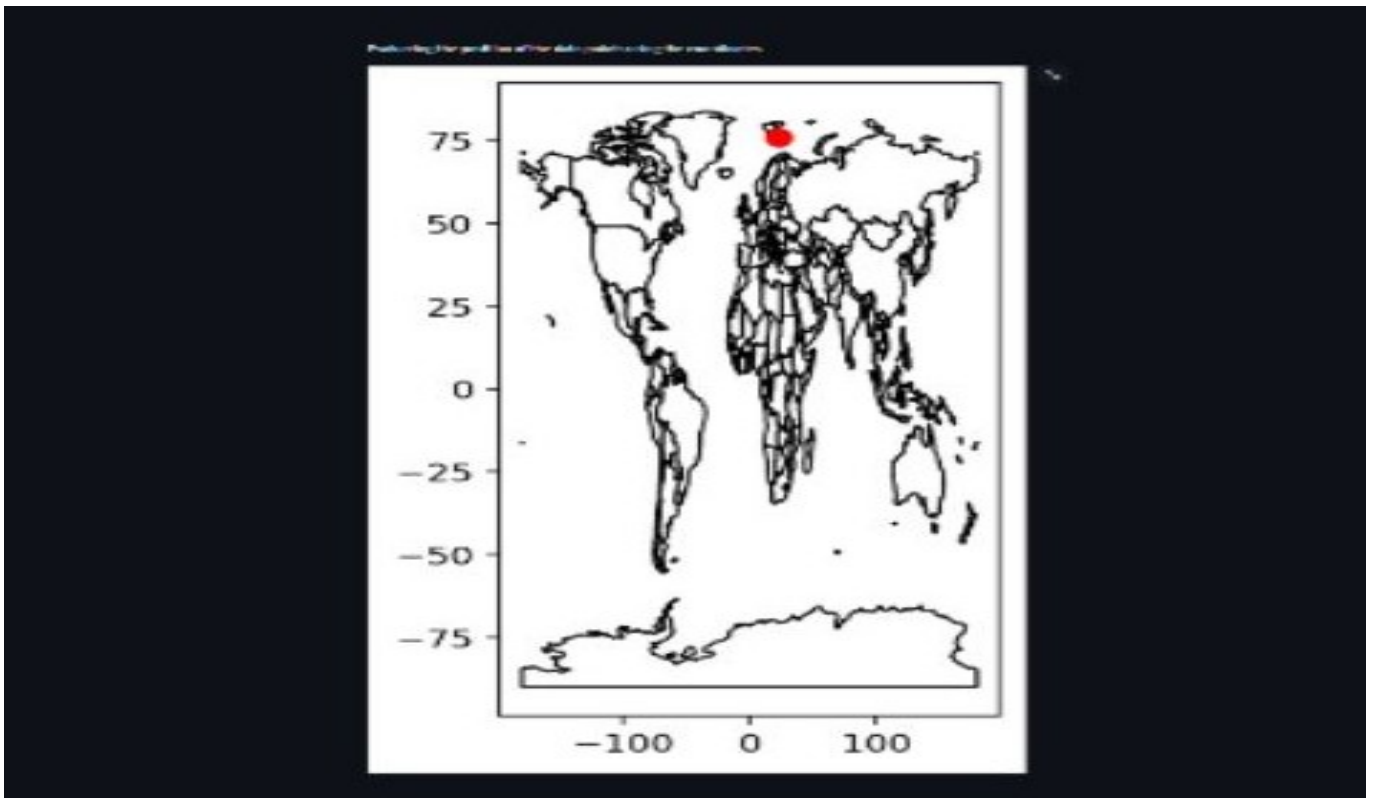


Figure 6.5: **Position of Data Points Using Coordinates**

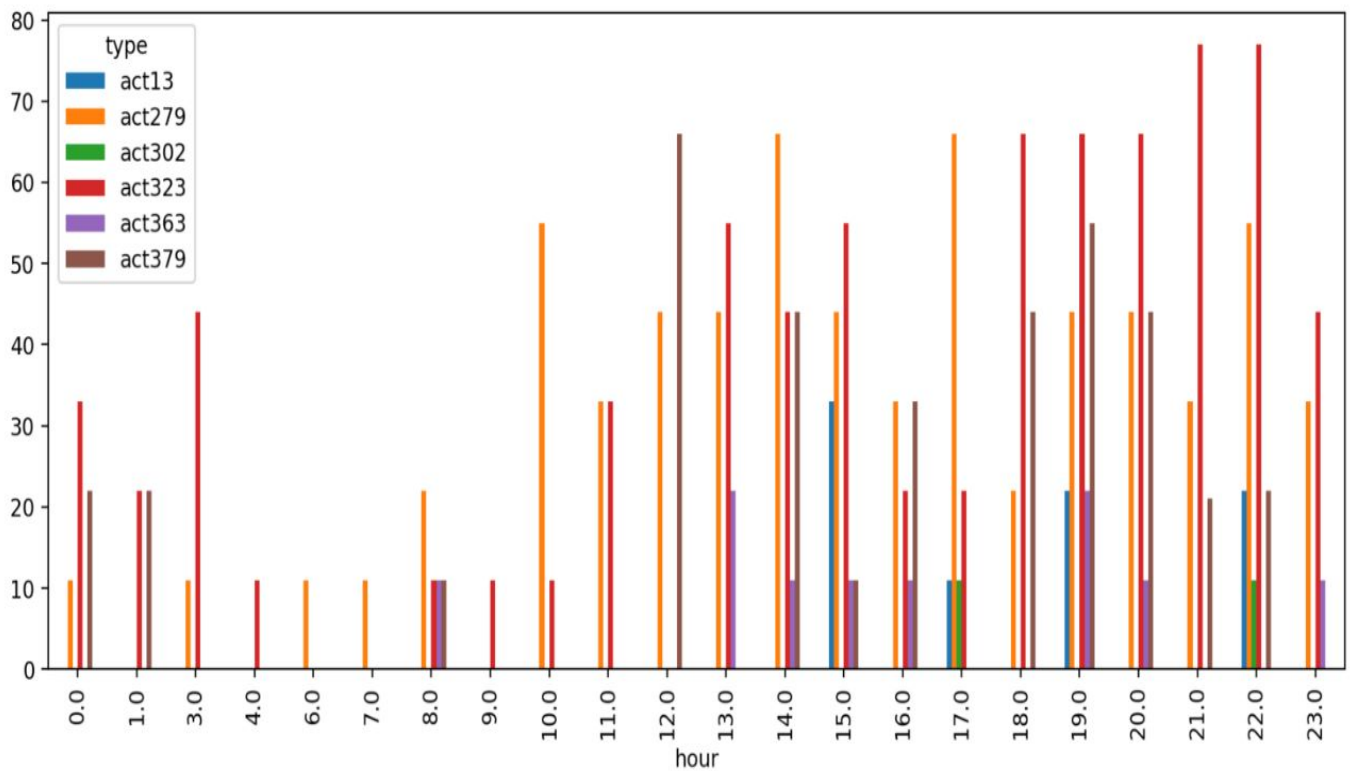


Figure 6.6: **Relationship between Day, Year, and Crime**

## Chapter 7

# CONCLUSION AND FUTURE ENHANCEMENTS

### 7.1 Conclusion

The current focus of this project, which is the forecasting of the crime that a specific offender is most likely to do, Able to forecast, as a future focus, the approximate amount of time that the offender will spend committing the crime. In addition to this, one can make an educated guess as to the location of the criminal activity. Going to determine how accurate frequently occurring item sets and prediction based on various test sets are. Therefore, the system will automatically learn the shifting patterns in criminal behavior by inspecting the patterns of criminal behavior. Additionally, the elements that contribute to crime shift throughout time. By reorganizing the data on crime, Need to locate new factors that are contributors to criminal behavior. Given that are only taking into account a select few parameters, it is impossible to reach complete accuracy. Need to identify more criminal attributes so that can get more accurate outcomes from our predictions. Our software is able to forecast the type of criminal act that will be committed by a specific offender. These models can be used to send high or low crime alerts to police officers. One interesting observation is the negligible impact of weather-related attributes on algorithm predictions even they seemed relevant based on feature selection techniques and in contrast with earlier studies. The system development intended to make the best use of the available data. On the one hand, the system could use the high-quality crime database from the Chilean police. On the other hand, the system could obtain context data from freely available databases, such as Open Street Maps.

The townships in which the system provided useful high efficacy risk surfaces, the integrating module always provided better performance than the individual modules. Therefore, the integrating procedure was successful in capturing various read-

ings of the data by the use of different algorithms. Crimes are serious threats to human society, safety, and sustainable development and are thus meant to be controlled. Investigation authorities often demand computational predictions and predictive systems that improve crime analytics to further enhance the safety and security of cities and help to prevent crimes. The predictions, and applied various machine learning and deep learning algorithms to compare their performances in predicting crime data. These models can be used to send high or low crime alerts to police officers. One interesting observation is the negligible impact of weather-related attributes on algorithm predictions even they seemed relevant based on feature selection techniques and in contrast with earlier studies. We presented a prediction method for spatio-temporal information on crimes based on publicly available city data with experiments on specific area. We fuse the crime records with information about schools, libraries, police stations, and 311 service calls using network analytic approach to identify observations that improve the quality of prediction. After all the features of data are identified, regression-based prediction can be applied. Crime patterns are also identified they are not static. We have identified the crime areas and which type of crime occurred very frequently in which place. Experimental results reveal that the proposed approach in the current study performs well on the collected dataset with the extra characteristics of better flexibility and adaptability to the targeted criminal group's changing interests and strategies. The proposed method can identify the textual malicious contents on social media with high accuracy and less computational complexity.

## **7.2 Future Enhancements**

The new features in the future method adapts satisfactorily to the changing data sets with updates according to the expected strategy of the concerned criminal group. To limit the scope, only the textual part of social media content is considered, and images and videos are not taken into account. Future work can be undertaken considering the images and videos from social media by suitably extracting features and identification of criminal content. The application of other spatial analysis approaches for the detection of crime dense regions and for modeling and forecasting crime trends on such regions. The perform an extended experimental evaluation on an wider urban territory, to assess the results obtained in the case study reported here. Can apply such an approach for spatio-temporal prediction of other kind of

events, different than crimes. features extraction algorithm, abbreviated as FEA, is the newly proposed algorithm for extracting features from textual posts. The proposed FEA extracts a feature vector of eight Boolean values per textual content in six steps. The algorithm does not need pre-processing of data and reduces the time and space complexity of the features extraction process. The proposed FEA reduces the computational overhead of text classification task by avoiding the pre-processing of text being not needed. It is very useful for investigators to solve the crime rate. Based on the fuzzy clustering technique the crime prone areas are identified it takes a less time. This project presented a general algorithm for Spatio Temporal Crime Prediction in urban areas, that takes advantages from the partitioning of the whole analyzed area by detecting crime dense regions. Different Python libraries were applied including Keras with Tensor Flow, Sk Learn, Pandas, Numpy, Seaborn, Scipy, and many others to generate the results in future.

The Future research will assess the impact of changing grid sizes on prediction accuracy. Third, the robustness and generality of the findings of this paper needs to be tested in other study areas. Nonetheless, the findings of this research have proven to be useful in a recent hotspot crime prevention experiment by the local police department at the study size. The future extension of the proposed work, the application of more machine learning classification models proves to increase accuracy in crime prediction and will enhance the overall performance. It helps in providing a better study for the future improvement by taking the income information into consideration for neighborhoods places in order to foresee if any relationship between the income levels of a particular in the neighborhood places and their crime rate. Feature selection techniques are mainly used to identify the best subsets from N number of predictors. There are various feature subset selection techniques. The smart policing technology can predict crime type and CRS using text-based crime event data. A real-time GUI-based application platform is implemented and applied to the predictive models. The developed GUI-based system can predict crime type and CRS in real-time for new input data. In addition, the developed system provides an intuitive GUI, allowing field personnel to use the system efficiently. The results are also shown and discussed in each section. The Firstly, the predictive accuracy is discussed based on different algorithms. crime particulars are thoroughly discussed in the exploratory data analysis section, and finally, crime forecasting and future crime trends are shown through the ARIMA model.

## **Chapter 8**

# **INDUSTRY DETAILS**

### **8.1 Industry name**

CORIZO : Empowering Tomorrow's Leaders

#### **8.1.1 Duration of Internship (From Date - To Date)**

05-Jan-2023 - 05-May-2023

#### **8.1.2 Duration of Internship in months**

4 Months

#### **8.1.3 Industry Address**

D-247/4A Procapitus Business Park, B-12, D Block, Sector 63, Noida, Uttar Pradesh  
201301 - Virtual Internship

## 8.2 Internship offer letter



**CORIZO**

Empowering Tomorrow's Leader

### Internship Offer Letter

**20th December 2022**

**Dear Siva Prasad Reddy Bandi ,**

With reference to your application, we are pleased to offer you an internship with **Corizo Edutech**.

We take this opportunity in wishing you the very best in your new employment as well as advising you that our offer letter is on the following terms and conditions:

**1. Period of Service: Four(4) Months** of your employment will be probationary.

You shall, for the purpose of your employment with us, sign this offer letter for submission and approval of the management.

**2. Designation:** You shall be employed as a **Data Science Intern**.

**3. Remuneration:** You will not be eligible for remuneration at the end of the program.

**Internship Start Date: 05/01/2023**

**Internship End Date: 05/05/2023**

Your responsibilities will include those for which you are engaged, as well as any other duties given to you by your mentor from time to time. By accepting this offer you agree to perform all responsibilities assigned to you with due care and diligence and in compliance with the management norms and clauses.

By accepting this offer letter of Employment, you acknowledge that you will keep all this information strictly confidential and refrain from using it for your own purposes, that is, disclosing it to anyone outside of the company.

By accepting this offer letter, you agree that throughout your internship, you will observe all policies and practices governing the conduct of our business and employees. This letter sets forth the complete offer we are extending to you and supersedes and replaces any prior inconsistent statements or discussions. **Official communication either within the company or outside the company should be through the official Email of the HR or support only.**

Corizo, D-247, B-12, Sector-63, Noida

supportlms@corizo.in

corizo.in

Figure 8.1: Siva Prasad Reddy Bandi Internship Offer Letter



## Internship Offer Letter

**20th December 2022**

**Dear THUMMALAPALLI NIKHIL CHOWDARY ,**

With reference to your application, we are pleased to offer you an internship with **Corizo Edutech**.

We take this opportunity in wishing you the very best in your new employment as well as advising you that our offer letter is on the following terms and conditions:

**1. Period of Service: Four(4) Months** of your employment will be probationary. You shall, for the purpose of your employment with us, sign this offer letter for submission and approval of the management.

**2. Designation:** You shall be employed as a **Web Development Intern**.

**3. Remuneration:** You will not be eligible for remuneration at the end of the program.

**Internship Start Date: 05/01/2023**

**Internship End Date: 05/05/2023**

Your responsibilities will include those for which you are engaged, as well as any other duties given to you by your mentor from time to time. By accepting this offer you agree to perform all responsibilities assigned to you with due care and diligence and in compliance with the management norms and clauses.

By accepting this offer letter of Employment, you acknowledge that you will keep all this information strictly confidential and refrain from using it for your own purposes, that is, disclosing it to anyone outside of the company.

By accepting this offer letter, you agree that throughout your internship, you will observe all policies and practices governing the conduct of our business and employees. This letter sets forth the complete offer we are extending to you and supersedes and replaces any prior inconsistent statements or discussions. **Official communication either within the company or outside the company should be through the official Email of the HR or support only.**

Figure 8.2: T Nikhil Internship Offer Letter





## Internship Offer Letter

20th December 2022

Dear K SAI KIRAN ,

With reference to your application, we are pleased to offer you an internship with **Corizo Edutech**.

We take this opportunity in wishing you the very best in your new employment as well as advising you that our offer letter is on the following terms and conditions:

**1. Period of Service: Four(4)** Months of your employment will be probationary. You shall, for the purpose of your employment with us, sign this offer letter for submission and approval of the management.

**2. Designation:** You shall be employed as a **Web Development Intern**.

**3. Remuneration:** You will not be eligible for remuneration at the end of the program.

**Internship Start Date: 05/01/2023**

**Internship End Date: 05/05/2023**

Your responsibilities will include those for which you are engaged, as well as any other duties given to you by your mentor from time to time. By accepting this offer you agree to perform all responsibilities assigned to you with due care and diligence and in compliance with the management norms and clauses.

By accepting this offer letter of Employment, you acknowledge that you will keep all this information strictly confidential and refrain from using it for your own purposes, that is, disclosing it to anyone outside of the company.

By accepting this offer letter, you agree that throughout your internship, you will observe all policies and practices governing the conduct of our business and employees. This letter sets forth the complete offer we are extending to you and supersedes and replaces any prior inconsistent statements or discussions. **Official communication either within the company or outside the company should be through the official Email of the HR or support only.**

Figure 8.3: K Sai Kiran Internship Offer Letter



### 8.3 Project Commencement Form



#### Project Commencement Form

Name of the Industry: Corizo

Address: Noida, Uttar Pradesh

Team Details:

S.No	ID No	Student Name	Degree & Branch
1.	14477	B. SIVA PRASAD REDDY	B.Tech/CSE
2.	14506	T. NIKHIL	
3.	15321	K. SAI KIRAN	

Date of reporting for project work: 05/01/2023

Name of the Industry Supervisor : Hemant Ingle

Department : Computer Science and Engineering

Designation : HR Manager

Contact Number : 9582716637

Email ID : hemant@corizo.in

Name of the Internal Supervisor : Dr. K. Seetha Lakshmi

Contact No. : 9840017225

Email ID : seethalakshmik@veltech.edu.in

Tentative Project Title / Project domain: Location Based Crime Type Prediction Using PyTorch Non-Linear Deep-2-layer Model with Streamlit WebApp/Machine learning and Web development

Brief Project/task description:

Figure 8.4: Project Commencement Form

## 8.4 Internship Completion Certificate



**CORIZO**

Empowering Tomorrow's  
Leaders

20th April 2023,

With reference to our company, we are pleased to inform you that, following students are doing internship with **Corizo Edutech**.

**Period of Service: Four(4) Months**

**Internship Start Date: 05/01/2023**

**Internship End Date: 05/05/2023**

The following students are currently undergoing internship in Corizo Edutech from 05/01/2023 and Internship will be completed by 05/05/2023. So, Internship certificate will be issued after completion of Internship only(**5th May 2023**).

Student Name	VTU NO	REG NO
Siva Prasad Reddy Bandi	14477	19UECS0104
T. Nikhil	14506	19UECS0986
k. Sai Kiran	15321	19UECS0402

Thank you,

Regards,

**VP - Human Resources, Corizo Edutech**

  
**Hemant Ingle**  
**HR Manager**

Figure 8.5: Internship Completion Certificate

## Chapter 9

# PLAGIARISM REPORT



### PLAGIARISM SCAN REPORT

**Date** April 08, 2023

**Exclude URL:** NO



Unique Content **90%**

Plagiarized Content **10%**

Paraphrased Plagiarism **0**

Word Count 3999

Records Found 37

#### CONTENT CHECKED FOR PLAGIARISM:

Crime reduction is becoming one of the most important social issues in enormous metropolitan areas because it influences people's concerns about their personal safety, the development of children, and individuals socioeconomic standing. Crimes are influenced by organizations and other places occurred frequently in a society. This is due to the fact that larger cities have higher crime rates than smaller cities. The crime rate forecasting method is a method that determines the crime rate by using a variety of algorithms and basing their findings on previous information.

Figure 9.1: Plagiarism Report

# Chapter 10

## SOURCE CODE & POSTER PRESENTATION

### 10.1 Source Code

```
1 import pandas as pd
2 import numpy as np
3 import os
4 data_frame_main = pd.read_csv("data.csv")
5 dataset=pd.read_csv('data.csv')
6 data_frame_main['timestamp'] = pd.to_datetime(data_frame_main['timestamp'], errors='coerce')
7 data_frame_main['timestamp'] = pd.to_datetime(data_frame_main['timestamp'], format = '%d/%m/%Y %H:%M:%S')
8 column_1 = data_frame_main.iloc[1:,0]
9 db=pd.DataFrame({"year": column_1.dt.year,
10                 "month": column_1.dt.month,
11                 "day": column_1.dt.day,
12                 "hour": column_1.dt.hour,
13                 "dayofyear": column_1.dt.dayofyear,
14                 "week": column_1.dt.week,
15                 "weekofyear": column_1.dt.weekofyear,
16                 "dayofweek": column_1.dt.dayofweek,
17                 "weekday": column_1.dt.weekday,
18                 "quarter": column_1.dt.quarter})
19 dataset1=dataset.drop('timestamp',axis=1)
20 data1=pd.concat([db,dataset1],axis=1)
21 data1.dropna(inplace=True)
22 file = "main_data.csv"
23 if file in os.listdir(os.getcwd()):
24     print("Already Present")
25 else:
26     data1.to_csv(file,index=False)
27     print("File saved")
28 x = data_frame_main['year']
29 y = data_frame_main['act279']
30 print(inputs_train.shape)
31 print(inputs_test.shape)
32 print(targets_train.shape)
33 print(targets_test.shape)
34 class LogisticRegression(nn.Module):
```

```

35     def __init__(self, no_of_features):
36         super(LogisticRegression, self).__init__()
37         self.linear1 = nn.Linear(no_of_features, 10)
38         self.relu = nn.ReLU()
39         self.linear2 = nn.Linear(10, 6)
40     def forward(self, targets_train):
41         out = self.linear1(targets_train)
42         out = self.relu(out)
43         out = self.linear2(out)
44         return out
45 model = LogisticRegression(no_of_features=no_of_features)
46 loss_fn = nn.MSELoss()
47 optimizer = torch.optim.Adam(model.parameters(), lr=0.0001)
48 loss_hist = []
49 acc_hist = []
50 for epoch in range(num_epochs):
51     targets_preds = model(inputs_train)
52     loss = loss_fn(targets_preds, targets_train)
53     loss.backward()
54     optimizer.step()
55     optimizer.zero_grad()
56     if (epoch+1)%10 == 0:
57         with torch.no_grad():
58             targets_preds = model(inputs_test)
59             targets_preds_cls = targets_preds.round()
60             acc = targets_preds_cls.eq(targets_test).sum() / float(inputs_test.shape[0])
61             loss_hist.append(loss.item())
62             acc_hist.append(acc)
63             print(f'Epoch: {epoch+1}, Loss: {loss.item():.4f}, ACC: {acc}')
64             torch.save(model.state_dict(), 'model/final_crime_lats.pth')
65             plt.plot(loss_hist)
66 plt.show()
67 plt.plot(acc_hist)
68 plt.show()
69 plt.plot(loss_hist, label="Loss")
70 plt.plot(acc_hist, label="Accuracy")
71 plt.legend()
72 plt.show()
73 x = df['year']
74 y = df['act279']
75 plt.plot(x, y)
76 plt.show()
77 plt.plot(df['year'])
78 plt.show()
79 def create_gdf(df):
80     gdf = df.copy()
81     gdf['Coordinates'] = list(zip(gdf.latitude, gdf.longitude))
82     gdf.Coordinates = gdf.Coordinates.apply(Point)
83     gdf = gpd.GeoDataFrame(
84         gdf, geometry='Coordinates', crs={'init': 'epsg:4326'})

```

```

85     return gdf
86 geo_df = create_gdf(df)
87 plt.show()
88 df_sub = df[['act379', 'act13', 'act279', 'act323', 'act363', 'act302']]
89 df_sub.head()
90 df_sub['type'] = df_sub.idxmax(1)
91 df_sub.head()
92 type_column = df_sub.type
93 df = df.join(type_column)
94 df.head()
95 data = df.groupby('weekday').count().iloc[:, 0]
96 data = data.reindex([1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0])
97 plt.figure(figsize=(10, 5))
98 with sns.axes_style("whitegrid"):
99     ax = sns.barplot(
100         data.index, (data.values / data.values.sum()) * 100,
101         orient='v',
102         palette=cm.ScalarMappable(cmap='Reds').to_rgba(data.values))
103     xy=(data.median(), 0.004),
104     xytext=(200, 0.005),
105     arrowprops=dict(arrowstyle='->', color=col[1], shrinkB=10))
106 data = df.groupby(['hour', 'day', 'type'], as_index=False).count().iloc[:, :4]
107 data.rename(columns={'Dates': 'Incidents'}, inplace=True)
108 data = data.groupby(['hour', 'type'], as_index=False).mean()
109 data = data.loc[data['type'].isin(['act379', 'act13', 'act279', 'act323', 'act363', 'act302'])]
110 fig, ax = plt.subplots(figsize=(14, 4))
111 ax = sns.lineplot(x='hour', y='day', data=data, hue='type')
112 year = c.number_input(label="Enter Year", value=2022, min_value=1900, max_value=3000)
113 date_in = a.number_input(label="Enter Date", value=29, max_value=31, min_value=1)
114 month_in = b.number_input(label="Enter Month", value=7, max_value=12, min_value=1)
115 min_in = b.number_input(label="Minutes", value=15, max_value=59, min_value=0)
116 hour_in = a.number_input(label="Hour", value=23, max_value=23, min_value=0)
117 sec_in = c.number_input(label="Seconds", value=0, max_value=59, min_value=0)
118 latitude = c.number_input(label="Latitude", value=22.720992, format="%f")
119 longitude = d.number_input(label="Longitude", value=75.876083, format="%f")
120 enter = st.button(label="Predict Crime")
121 if enter:
122     if len(str(month_in)) == 1:
123         month_in = "0" + str(month_in)
124     if len(str(hour_in)) == 1:
125         hour_in = "0" + str(hour_in)
126     if len(str(sec_in)) == 1:
127         sec_in = "0" + str(sec_in)
128     main_date = '-'.join([str(year), str(month_in), str(date_in)])
129     main_time = ':'.join([str(hour_in), str(min_in), str(sec_in)])
130     date_time = " ".join([main_date, main_time])
131     sr = pd.Series([date_time])
132     sr = pd.to_datetime(sr)
133     print(sr)
134     print(type(sr.dt.month))

```

```

135     main_ins = {
136         "month": sr.dt.month.tolist()[0],
137         "day": sr.dt.day.tolist()[0],
138         "hour": sr.dt.hour.tolist()[0],
139         "dayofyear": sr.dt.dayofyear.tolist()[0],
140         "week": sr.dt.week.tolist()[0],
141         "weekofyear": sr.dt.weekofyear.tolist()[0],
142         "dayofweek": sr.dt.dayofweek.tolist()[0],
143         "weekday": sr.dt.weekday.tolist()[0],
144         "quarter": sr.dt.quarter.tolist()[0],
145     }
146     inputs = list(main_ins.values())
147     inputs.append(latitude)
148     inputs.append(longitude)
149     inputs_arr = np.array(inputs)
150     labels = ['Robbery', 'Gambling', 'Accident', 'Violence', 'Murder', 'Kidnapping']
151     vals, dicts = predict(inputs_arr)
152     # print(list(vals.values()))
153     explode = (0.2, 0.2, 0.2, 0.2, 0.2, 0.2)
154     plt.pie(vals, labels=labels, explode=explode, shadow=True, autopct='%1.2f%')
155     plt.savefig('images/predict_graph.png')
156     st.markdown("> Probablity of types of crimes happening on {} at {}".format(main_date, main_time))
157     st.image('images/predict_graph.png')
158     st.markdown("> Acts of Crime")
159     st.write(dicts)
160 st.markdown("***")
161 st.markdown("Evaluating the position of the data points using the coordinates")
162 df = pd.read_csv("main_data.csv")
163 def create_gdf(df):
164     gdf = df.copy()
165     gdf['Coordinates'] = list(zip(gdf.latitude, gdf.longitude))
166     gdf.Coordinates = gdf.Coordinates.apply(Point)
167     gdf = gpd.GeoDataFrame(
168         gdf, geometry='Coordinates', crs={'init': 'epsg:4326'})
169     return gdf
170 with sns.axes_style("whitegrid"):
171     ax = sns.barplot(
172         data.index, (data.values / data.values.sum()) * 100,
173         orient='v',
174         palette=cm.ScalarMappable(cmap='Reds').to_rgba(data.values))
175 plt.title('Incidents per Weekday', fontdict={'fontsize': 16})
176 plt.xlabel('Weekday')
177 plt.ylabel('Incidents (%)')
178 st.pyplot()
179 st.markdown("***")
180 st.markdown("distribution of number of incidents per day")
181 col = sns.color_palette()
182 plt.figure(figsize=(10, 6))
183 data = df.groupby('day').count().iloc[:, 0]
184 plt.axvline(x=data.median(), ymax=0.95, linestyle='--', color=col[1])

```


```

185 plt.annotate(
186     'Median: ' + str(data.median()),
187     xy=(data.median(), 0.004),
188     xytext=(200, 0.005),
189     arrowprops=dict(arrowstyle='->', color=col[1], shrinkB=10))
190 plt.title('Distribution of number of incidents per day', fontdict={'fontsize': 16})
191 plt.xlabel('Incidents')
192 plt.ylabel('Density')
193 plt.legend().remove()
194 st.pyplot()
195 st.markdown("***")
196 st.markdown("Crime on hourly basis plotted yearly")
197 data = df.groupby(['hour', 'day', 'type'], as_index=False).count().iloc[:, :4]
198 data.rename(columns={'Dates': 'Incidents'}, inplace=True)
199 data = data.groupby(['hour', 'type'], as_index=False).mean()
200 data = data.loc[data['type'].isin(['act379', 'act13', 'act279', 'act323', 'act363', 'act302'])]
201 sns.set_style("whitegrid")
202 fig, ax = plt.subplots(figsize=(14, 4))
203 ax = sns.lineplot(x='hour', y='day', data=data, hue='type')
204 ax.legend(loc='upper center', bbox_to_anchor=(0.5, 1.15), ncol=6)
205 plt.suptitle('Average number of incidents per hour')
206 fig.tight_layout(rect=[0, 0.03, 1, 0.95])
207 st.pyplot()
208 def softmax(x):
209     return (np.exp(x)/np.exp(x).sum())
210 class LogisticRegression(nn.Module):
211     def __init__(self, no_of_features):
212         super(LogisticRegression, self).__init__()
213         self.linear1 = nn.Linear(no_of_features, 10)
214         self.relu = nn.ReLU()
215         self.linear2 = nn.Linear(10, 6)
216     def forward(self, targets_train):
217         out = self.linear1(targets_train)
218         out = self.relu(out)
219         out = self.linear2(out)
220         # out = self.relu(out)
221         return out
222 model = LogisticRegression(no_of_features=11)
223 model.load_state_dict(torch.load('model/final_crime_lats.pth'))
224 def predict(arr):
225     main_tensor = torch.from_numpy(arr.astype(np.float32))
226     preds_all = model(main_tensor)
227     softmax_pred = softmax(preds_all.detach().numpy())*100
228     final_dict = {'act379': round(softmax_pred[0], 2), 'act13': round(softmax_pred[1], 2), 'act279': round(
229         softmax_pred[2], 2), 'act323': round(softmax_pred[3], 2), 'act363': round(softmax_pred[4], 2), '
230         act302': round(softmax_pred[5], 2)}
229     print(final_dict)
230     return softmax_pred, final_dict

```



## 10.2 Poster Presentation



**Vel Tech**  
Vellore Institute of Technology  
Rajaraman Dr. Sagunthala  
Vellore Institute of Technology  
Chennai-600 155, India

### Location Based Crime Type Prediction Using PyTorch Non-Linear Deep-2-layer Model With Streamlit WebApp

Department of Computer Science & Engineering  
School of Computing  
1156CS701 – MAJOR PROJECT  
WINTER SEMESTER 2022-2023

#### ABSTRACT

- Finding both spatial and temporal criminal hotspots is the primary emphasis of this work. It examines several datasets based on crimes that have occurred in the actual world.
- After that, it elucidates how it implemented an algorithm in order to develop intriguing common patterns for crime hotspots. In addition, the research demonstrates how the utilized logistic regression to make predictions for possible sorts of criminal activity.
- This research presents an analytical study that combines findings of crime's dataset with its demographics information in order to capture the aspects that can affect the safety of neighborhoods.
- The results of implementing this method might be utilized to increase people's awareness of the dangerous regions and to assist agencies in predicting future crimes in a certain location within a specific time frame. Both outcomes would be beneficial.

<Student 1.vtu14477/B.Siva Prasad>  
 <Student 2.vtu14506/T.Nikhil>  
 <Student 3.vtu15321/K.Sai Kiran>  
 <Student 1.7989949804>  
 <Student 2.8919981303>  
 <Student 3.7032414235>  
 <Student 1.vtu14477@veltech.edu.in>  
 <Student 2.vtu14506@veltech.edu.in>  
 <Student 3.vtu15321@veltech.edu.in>

#### INTRODUCTION

- Crime reduction is becoming one of the most important social issues in enormous metropolitan areas because it influences people's concerns about their personal safety, the development of children, and individuals' socioeconomic standing. This is due to the fact that larger cities have higher crime rates than smaller cities.
- The crime rate forecasting method is a method that determines the crime rate by using a variety of algorithms and basing their findings on previous information. Because of the nature of our work, that are required to travel to a large number of locations on a daily basis.
- Google Maps may provide us one, two, or even more routes to reach our destination; nonetheless, the almost always take the shortest route, even though this indicates that do not fully understand the path condition.
- This research introduces the creation and implementation of a plan based on historical crime data and evaluates the crime rate in previous places at separate moments. Because of the uncertainty over whether or not it is actually secure, they are forced to deal with a number of unfavorable scenarios.
- Using a collection of information taken from the real world, the purpose of this research is to identify both temporal and spatial hotspots for criminal activity.

#### RESULTS

- This project addresses the challenge of efficiently separating harmful insects from crops. Current methods, which rely on manual labor, are time-consuming and require significant manpower.
- In addition to this, one can make an educated guess as to the location of the criminal activity. Going to determine how accurate frequently occurring item sets and prediction based on various test sets are.
- The proposed solution utilizes a convolutional neural network (CNN) model to accurately identify and classify insects on crops. The use of this technology has the potential to greatly increase efficiency and reduce the negative impact of insects on crop yield, providing a more sustainable and cost-effective solution for farmers.

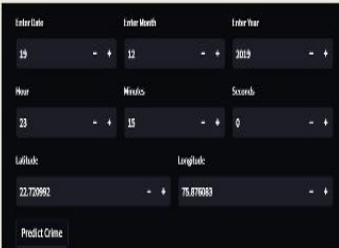


Table 1. Label in 20pt Calibri.




Figure 1. Label in 20pt Calibri.




Figure 2. Label in 20pt Calibri.

#### METHODOLOGIES

- Data Acquisition and Pre-Processing. We will be using a csv dataset with 18 columns and 2100 rows from Kaggle on crime prediction. Data preprocessing changes the data into a format that can be processed in machine learning, and other data science tasks more quickly and efficiently.
- Making The Model. The algorithms used for our model are binary classification with logistic regression. Logistic regression is used in the classifier construction process. Compared to linear regression, logistic regression is more sophisticated.
- Making A Prediction Function and Creating A Web App. The last stage is to develop a prediction function that will accept input as location and time and determine the crimes that can take place in the given area at the given time.

#### CONCLUSIONS

- In addition to the current focus of our study, which is the forecasting of the crime that a specific offender is most likely to do. Able to forecast, as a future focus, the approximate amount of time that the offender will spend committing the crime.
- Therefore, the system will automatically learn the shifting patterns in criminal behavior by inspecting the patterns of criminal behavior. Additionally, the elements that contribute to crime shift throughout time.



Chart 1. Label in 20pt Calibri.

#### ACKNOWLEDGEMENT

1. Project Supervisor: Dr. K. Seethalakshmi/Associate Professor
2. Project supervisor Contact No: +919840017225
3. Project supervisor Mail ID: seethalakshmik@veltech.edu.in

Figure 10.1: Poster Presentation

# References

- [1] Charlie Catlett, Eugenio Cesario, Domenico Talia, Andrea Vinci (2018). A DataDriven Approach for Spatio-Temporal Crime Predictions in Smart Cities, Vol 7, Issue 2018.
- [2] Imarn Shafi, Sadia Din, Zahid Hussain, Imran Asheaf And Gyu Sang Choi(2021),Adaptable Reduced-Complexity Approach Based on State Vector Machine for Identification of Criminal Activists on Social Media, Digital Object Identifier 10.1109/ACCESS.2021.3094532.
- [3] Lavanya Elluri, Varun Mandalapu, Nirmalya Roy (2019), Developing Machine Learning Based Predictive Models for Smart Policing, IEEE International Conference on Smart Computing (SMARTCOMP)978-1-7281-1689-1
- [4] .M. Patel, Ripal Patel, “Enhance Algorithm To Predict A Crime Using Data Mining” Journal of Emerging Technologies and Innovative Research, Vol. 5, Issue 04, April 2017.
- [5] Myung-Sun Baek, Wonjoo Park, Jaehong Park, Kwang-Ho Jang, Yong-Tae Lee (2021). Smart Policing Technique With Crime Type and Risk Score Prediction Based on Machine Learning for Early Awareness of Risk Situation, IEEE Access (Volume: 9), Digital Object Identifier 10.1109/ACCESS.2021.3112682.
- [6] R. Iqbal (2021) was focused on An experimental study of classification algorithms for crime prediction, Indian J. of Sci. and Technol., vol.6,no.3,pp.4219-4225..
- [7] Sai Tarlekar, Rucha Bhosle, Elysia D’souza, Sana Sheikh (2021). Geographical Crime Rate Prediction System, IEEE India Council International Subsections Conference (INDISCON) 978-1-6654-3833-9/21.
- [8] H. Chen, W. Chung, J. J. Xu, G. Wang, Y. Qin, and M. Chau, ”Crime data mining: a general framework and some examples,” IEEE Computer, vol. 37, no. 4, pp. 50-56, Apr. 2021.
- [9] Saroj Kumar Dash, Ilya Safro, Ravisutha Sakrepatna Srinivasamurthy (2018). Spatio-temporal prediction of crimes using network analytic approach,IEEE International Conference on Big Data (Big Data) 978-1-5386-5035-6

- [10] Umair Muneer Butt, Sukumar Letchmunan, Fadratul Hafinaz Hassan, Mubashir Ali, Anees Baqir and Hafiz Husnain Raza Sherazi, “Spatio-Temporal Crime Hotspot Detection and Prediction: A Systematic Literature Review”, IEEE Transactions on September 2020.
- [11] Vrushali Pednekar (2018) “Crime Rate Prediction using KNN”, International Journal on Recent and Innovation Trends in Computing and Communication, Vol 6, Issue 1.
- [12] Wajiha Safat, Sohail Asghar, Saira Andleeb Gillani (2021). Empirical Analysis for Crime Prediction and Forecasting Using Machine Learning and Deep Learning Techniques, IEEE Access (Volume: 9), Digital Object Identifier 10.1109/ACCESS.2021.3078117.
- [13] Xu Zhang 1,2, Lin Liu2,3, Luzi Xiaa2 , Jiakai Ji4 (2020), Comparison of Machine Learning Algorithms for Predicting Crime Hotspots, Digital Object Identifier 10.1109/ACCESS.2020.302842