Crime Rate Prediction Using Machine Learning

By

Chee Man Hang

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF COMPUTER SCIENCE (HONOURS)

Faculty of Information and Communication Technology

(Kampar Campus)

May 2022

UNIVERSITI TUNKU ABDUL RAHMAN

REPORT STATUS DECLARATION FORM

Title: Crime Rate Prediction Using Machine Learning

Academic Session: May 2022

I _____CHEE MAN HANG_____

(CAPITAL LETTER)

declare that I allow this Final Year Project Report to be kept in

Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.

Verified by,

_____ _____

(Author's signature) (Supervisor's signature)

Address:

9,Persiaran Bandar Bary Tambun

14,Desa Tambun Indah

31400,Ipoh Perak

DR CHANG JING JING

Supervisor's name

Date: __7/9/2022_ **Date**: _____9/9/2022_____

FACULTY/INSTITUTE* OF INFORMATION AND COMMUNICATION TECHNOLOGY

UNIVERSITI TUNKU ABDUL RAHMAN

Date: __7/9/2022___

SUBMISSION OF FINAL YEAR PROJECT /DISSERTATION/THESIS

It is hereby certified that _____***Chee Man Hang***_____ (ID No: __***18ACB03448***__) has completed this final year project/ dissertation/ thesis* entitled "_Crime Rate Prediction Using Machine Learning_" under the supervision of _ Dr Chang Jing Jing(Supervisor) from the Department of __ Computer Science___, Faculty/Institute* of _Information and Communication Technology.

I understand that University will upload softcopy of my final year project / dissertation/ thesis* in pdf format into UTAR Institutional Repository, which may be made accessible to UTAR community and public.

Yours truly,

_____

(*Chee Man Hang*)

*Delete whichever not applicable

DECLARATION OF ORIGINALITY

I declare that this report entitled “Crime Rate Prediction Using Machine Learning” is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature : _____

Name : ___CHEE MAN HANG_____

Date : _____8/9/2022_____

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my Final Year Project Supervisor, Chang Jing Jing for her patience and guidance in helping me complete this Crime rate prediction project.

ABSTRACT

As crime is a plague to society, every country has been actively trying to come up with solutions to reduce crimes. From things like campaigns to raise money for low-income household, crime watches, more frequent patrols, etc. However, even with these measures crime rates still remains at an all-time high. Therefore, with the implementation of this crime rate prediction system, the police can employ predictive policing whereby they can patrol the areas with a higher chance of crimes. With this, they can make a more informed decision on the areas to patrol. To develop this system, I used the San Francisco crime dataset. With this I have employed Feature engineering to aid the system in getting higher accuracies. I have also employed various ensemble learning methods such as XGBoost classifier, Decision tree, and Random Forest Classifier. After which I performed hyperparameter tuning with RandomSearchCV to aid in increasing the accuracies of the prediction of the system. One additional model was also used which was the SARIMAX model which was used to forecast future crime statistics for each Police District.

Table of Contents

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

SVM	Support Vector Machine
ANN	Artificial Neural Network
MAE	Mean Absolute Error
R^2	Coefficient of Determination
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
U.S.	United States
GLM	Generalized Linear Models
OLS	Ordinary-Least-Squares
L1	Lasso Regression
L2	Ridge Regression

Chapter 1

Introduction

1.1 Brief overview of crime and its effect on a country

Crime, like any other definition of word, is not always so simple to define as it may mean differently for different person. A typical understanding of the word 'crime', according to Britannica, can be defined as an act that is socially harmful or dangerous that is usually prohibited and punishable under criminal law [12]. Crime has been known to be a prevalent social problem that has affected the quality of life and the economic growth of every country.

Now to truly understand the effects of crime has on society, let us dive into why it is a social problem. Firstly, the effects it could have on a city is that it creates chaos which in turn disrupts the natural order of society. As crime naturally goes against social conventions, it disrupts many everyday activities from running a business, going shopping or even just walking outside. Another effect crime has on society is that it impedes collaboration and trust in a community. As with higher crime rates the trust toward law enforcement will be affected. Seeing how the law enforcement that was supposed to maintain the peace has failed to do their job, the people's willingness to collaborate will decrease not only towards law enforcement but also others in their community [14].

Moving on to economic losses, let us take our neighbouring country, Indonesia. It is much like Malaysia and has an abundance of natural resources as well as human resources which should have accelerated the pace of their economy, and yet it was found that the number of crimes may have limited the economic growth. The growth of Indonesia's economy is usually attributed to the consumption of goods which is directly influenced by the ability of income sources of households. Other than that, it is also found that foreign investments also aid in the economic growth of the country as it increases the production capacity of the country by reducing the basic costs and variable costs of the industrial sector which in turn increases the purchasing power of the people

thus aiding in the increase of the consumption. Though if crime were to increase it would give investors a bad perspective thus causing fewer investments to be made in the country. Thus, Kusuma, Hariyani, and Wahyu found that when the number of criminal acts increases it would reduce the Gross regional domestic product (GRDP) of the country [7].

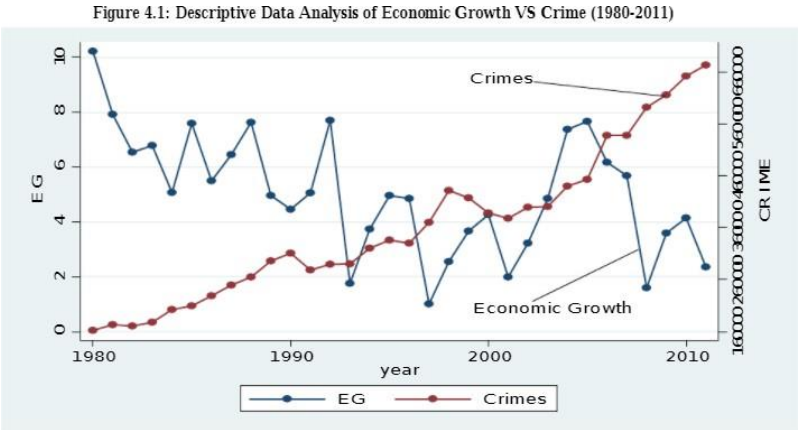


Figure 1.1: Descriptive Data Analysis of Economic Growth Vs Crime (1980-2011)

Another example of crime affecting the economy is a study done on Pakistan which is also another developing country. Pakistan's economy much like Indonesia could benefit from foreign investors but due to the high crime rates, it may have deterred some investors from investing in Pakistan. Thus, according to the figure shown below, between the years 1980 to 2011 Ahmad, Ali, and Ahmad (2014) found that the economic growth fluctuates through the years, but the trend is that as crime increases economic growth decreases [1].

Reasons to Use Machine Learning in Crime rate Prediction

Machine Learning has been gaining a lot of traction these past few years from being used to forecast future business investments to being used in medicine. Over the years more and more papers have been popping up on using machine learning in order to predict crime rate. A paper that did an overview on other papers regarding crime rate predictions found that a variety of methods from Support Vector Machine (SVM) that were used for hotspots prediction, Fuzzy Theory, which was used to increase the prediction efficiency, Artificial Neural Network (ANN) that was used to predict geo-temporal variation of disorder, etc [10].

The reason why this is important for the police and the civilians is it could potentially aid in reducing crime and increasing the safety in our country. An instance of this is that in the United States, Pearsall found that every New Year's Eve there would be an increase in random gunfire. Hence, by using the data they have gathered over the years, the police managed to anticipate the location, time, and nature of future incidents. Thus, with this data gathered the police were putto locations they were able to reduce the cases of random gunfire by 47 percent and increase the number of weapons seized by 246 percent whilst saving the police department around 15,000 USD in personnel costs that day [3].

1.2 Problem Statement and motivation

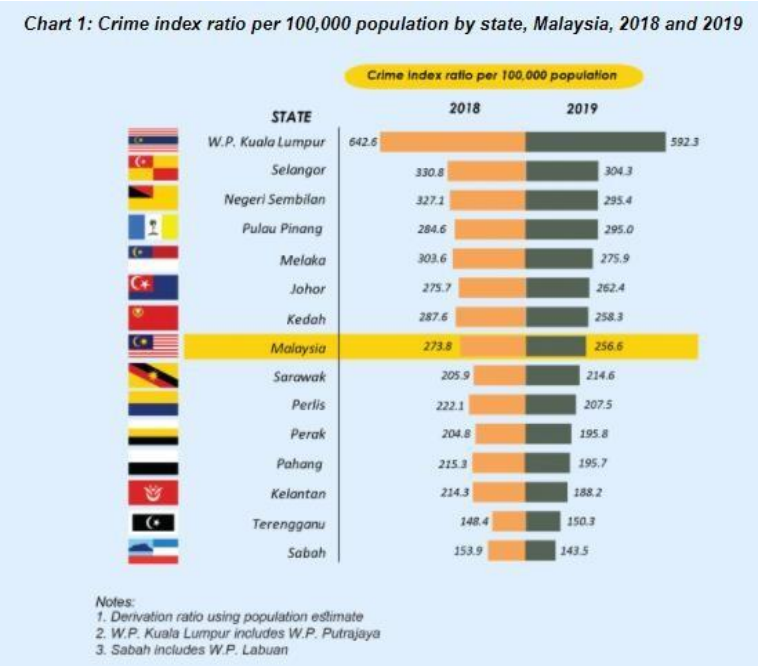


Figure 1.3.1 Crime index in the Year 2020

Crime rate affects a large amount of people annually in Malaysia. In 2019 crime rate was shown to have affected an average of 256.6 in a 100,000 people and while there has been a minor decrease compared to the year before this rate is still relatively high [5]. Hence most Malaysians are afraid to be alone outside or bring out their valuables with them. So, through the implementation of this system the police can pre-emptively patrol the highest risk areas, effectively reducing the crime rate and catch more criminals.

The next problem statement is that with the help of this system we could increase the effectiveness of predictive policing. As to analyse all the data manually would take up a large amount of time and effort making it extremely tedious for the police. With this the police could inherently reduce what could be days of works to minutes.

1.3: Project Scope and Project Objectives

Project Scope

The scope of this project is to develop a framework that is able to aid the police in predictive policing through predicting the location, category and time of a future crime with a decent amount of accuracy. The proposed tool will enable users to characterize and analyse crime data to find the actionable patterns and future trends. It should also be able to take in large amounts of data to aid the police in analysing the large amounts of data effectively reducing the time needed for police to go through the data.

Project objectives:

-To produce a system that is able predict areas that will have higher crime rates.
-To explore and enhance classification algorithms to predict future crime category based on previous crime trends.
-Create a web-based system to allow for easy access to the application

1.4 Impact, significance, and contribution

The benefit of implementing this project is that it can help the police in analysis of crime hotspots and where crime could potentially occur. This project could potentially aid in increasing the effectiveness of the crime force in our country if implemented correctly. This project will produce a kind of hotspot mapping for the police to allow them to predict areas and time of the crime with crime type.

Other than that, with the implementation of this system we could significantly improve the safety of the city. If the police were to follow the hotspot mapping of the crime and pre-emptively patrol those area this would aid in reducing the crime rates of the city.

Another impact this project could have is that it will reduce personnel cost as the police no longer have to be deployed to areas without crime hence allowing more criminals to be caught with less waste in personnel costs.

The last impact is that it could help the police analyse large amounts of data is a very short amount of time thus making more time for them to do other things such as patrolling, solving more pressing cases, etc.

1.5 Report Organization

In chapter 1 the background of crime, problem statements, Project Scope, Project Objectives and impact this project would have. In Chapter 2, I reviewed the literature reviews regarding crime rate prediction models and spatial mapping of crime. Chapter 4 describes the methodology, technology used, System designs of the project. Chapter 5 shows the implementation and the results of the projects for all 3 models. Chapter 6 shows how the Web application is used and the results it can produce. Chapter 7 wraps up the whole project with a conclusion, future work and the novelties and contribution of this project.

Chapter 2

Literature Review

2.1 Crime rate prediction in the urban environment using social factors

2.1.1 Brief overview

Ingilevich and Ivanov [15] introduced a crime rate prediction system that uses several predictors to determine crime rates such as the number of schools, police stations, churches, malls, number of buildings, shops that sell alcohol, bars, and the population of that area. In this paper, 3 models that were used are Linear regression, logistic regression, and gradient boosting with a dataset.The dataset contains information of the date of crime, coordinates, and descriptions of the crimethat were split into three-part for clustering to remove noise from their dataset which are banditry which group records such as assaults and street shootings, massacre for things such asbeatings in apartment and fights in the street and lastly robbery for cases such as theft or theftfrom cars.

Figure 2.1.1: Number of clusters with crime type

Type of crime	Number of clusters	Number of points in clusters
banditry	30	3347
massacre	33	86200
robbery	14	3133

The paper then used the 3 models which they compared to one another and evaluated the models with the use of Mean Absolute Error (MAE) and the coefficient of determination(R^2) and used cross-validation to determine the average values of the metrics of MAE and R^2.

Type of crime	Method	MAE (Cross-validation)	R^2 (Cross-validation)	MAE of prediction for grid
banditry	Linear regression	17	0.9	18
	Logistic regression	216	0	156
	Gradient boosting	17	0.9	31
massacre	Linear regression	231	0.9	110
	Logistic regression	5021	0	3516
	Gradient boosting	56	0.9	264
robbery	Linear regression	28	0.9	364
	Logistic regression	189	0.3	5861
	Gradient boosting	34	0.9	470

Figure 2.1.2: Evaluation of the predictions

2.1.2 Strengths

For the strength of this paper, they used a method known as DBSCAN (Density-based spatial clustering of applications with noise). The advantage of using this algorithm is that it finds arbitrarily shaped clusters. They done this by using a python package known as sckit-learn and through this they managed to obtain a set of separated individual clusters for each type of crime.The resulting clusters allowed them to further study the spatial pattern in more detail. A visualization of the clusters is as seen below.

Figure 2.1.3: The clusters of different crime types

Another strength of the paper is that they state how other papers pay little attention to the selection of factors that affect the crime rate. In this paper, they state how they used feature selection techniques known as the chi-squared test that have helped increase the accuracy of their predictions and helped avoid overfitting of their models. The chi-squared test is used to select the features that will aid in easing of computation and make interpreting the data easier.Through this, they reduced the features to the population number, bars, churches, and schoolseach with a weight

coefficient as follows 0.6, 0.3,0.1,0.2 with the bars being the highest decisive factor of criminal activity.

2.1.3 Weaknesses

For the weakness of this paper, it was found that the linear regression models have produced a negative value as observed from the table below for each of the factors,

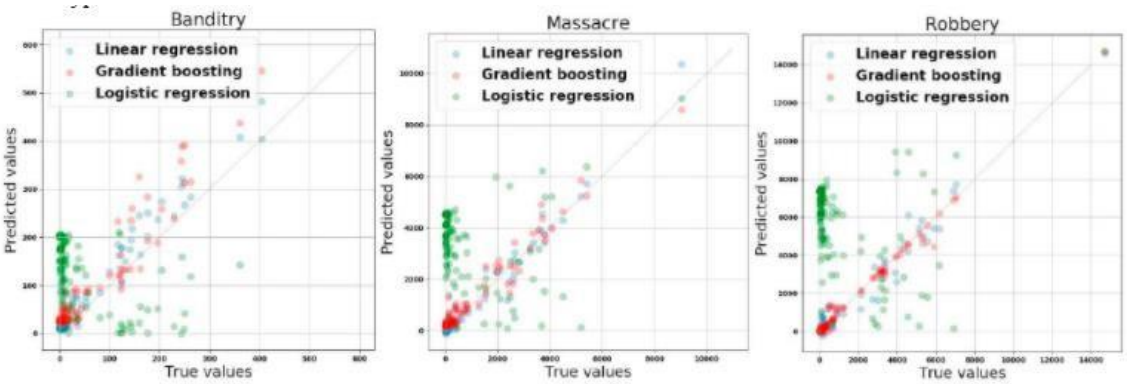


Figure 2.1.4: The Predicted number of crimes based on types and the comparison of models

This is theoretically impossible as a crime cannot be negative as you cannot theoretically "unmurder" a man.

2.1.4 Recommendation

The way they resolved the problem above is that they used other models they had other modelsthat they were using to compare the results. Hence, when the linear regression model did not perform to their satisfaction, they still had others they could use for the study in this case the gradient boosting model. In this case the Gradient boosting model was deemed the best as whencompared with logistic regression that had a high MAE and does not have negative values likethe linear regression model. Therefore, it was used to visualize the results of the crime rate prediction.

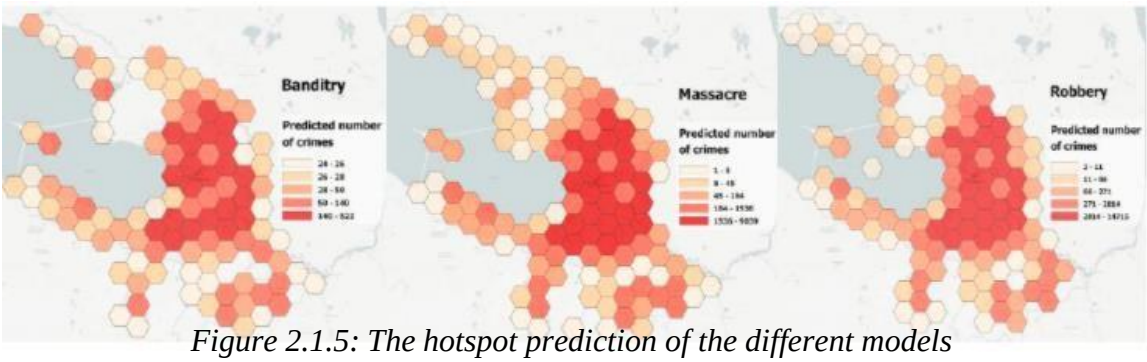


Figure 2.1.5: The hotspot prediction of the different models

2.2 Predicting Spatial Crime Occurrences through an Efficient Ensemble-Learning Model

2.2.1 Brief overview

Lamari, et. Al[17] presented an efficient machine learning framework that can predict spatial crime occurrences across the United States census block groups. The paper used real-world datasets reported from across 11 cities in U.S. and was tested based on 6.4 percent of the total block groups of the United States. This paper used predictors from, demographics, socioeconomics, and environmental and spatial data in urban areas as seen below.

Social Structural Variables	Relationship to Crime
Concentrated Disadvantage	Positive
Unemployment	Unclear, possibly positive
Family Disruption	Positive
Residential Instability	Positive
Racial/Ethnic Heterogeneity	Positive
Segregation	Positive
Income Inequality	Positive
Immigration	Unclear
Gender (Male)	Positive
Age (Younger)	Positive

Figure 2.2.1: The Direct and Indirect effect of variables on urban crime

However, while they did consider rural areas, they found the predictors were not very reliable. Hence, the papers predictions were only for urban and suburban areas.

Table 2. Social disorganization variables effects on rural crime [66,74].

Structural Variables	Relationship to Crime
Poverty, Income, Income Inequality	No relationship or Inverse
Unemployment	Unclear, possibly positive
Family Disruption	Unclear, possibly no relationship or even inverse
Residential Instability	Unclear
Racial/Ethnic Heterogeneity	Unclear

Figure 2.2.2: The Direct and Indirect effect of variables on rural crime

The features that were selected are as follows

Themes	Number of Attributes	Mean Absolute Correlation (%)	Mean Feature Importance (%)
Poverty	14	23.57	0.59
Residential instability	4	19.89	0.75
Housing and commuting	14	19.18	0.65
Income	4	18.4	0.68
Population	4	16.95	1.26
Family disruption	10	16.79	0.69
Unemployment	8	11.16	0.66
Gender	2	9.29	0.71
Climate	60	8.99	0.31
Education	36	8.73	0.54
Socio-economic indicators	5	8.67	0.12
Age	10	7.45	0.64
Law enforcement	4	7.37	0.65
Ethnic heterogeneity	12	5.17	0.61
Land area	1	4.47	3.61

Figure 2.2.3: The predictors and their importance

The paper then states they decided on 3 different machine learning families of algorithm whichare generalized linear models, deep learning, and ensemble learning that were made from a comparative study. The paper also focused on different predictive modelling families which include Ensemble learning, deep learning, and Generalized Linear Models (GLMs) to help predict crime.

2.2.2 Strengths

The strength of their approach is that they managed to get an accuracy of 59 percent to 64 percent. They stated that while this papers performance may seem moderate in comparison to other papers that can reach accuracies of around 97 percent through the use of aggregated datafrom data such as city, county and state and also past crimes. However, unlike other papers they can predict at higher resolution at the United States census block group level. The U.S. Census Block Group level is a local area that usually contains from 600 to 3000 people with amedian area of 1.3 km^2. The reason they use this as a unit of analysis to study neighbourhood effects is, they found that it usually aligns with the residents' perception of their neighbourhood. The paper also stated that by rather than predicting the exact crime count by predicting if an observation would lie within a category of the figure below it would increase to accuracy to around 75 percent for the total count of crime. When the crimes are predicted individually, they got an accuracy of 77 percent for

motor vehicle thefts, 77% for vandalism acts, 77% for violent acts, 77% for motor vehicle thefts, and 73% for property crimes.

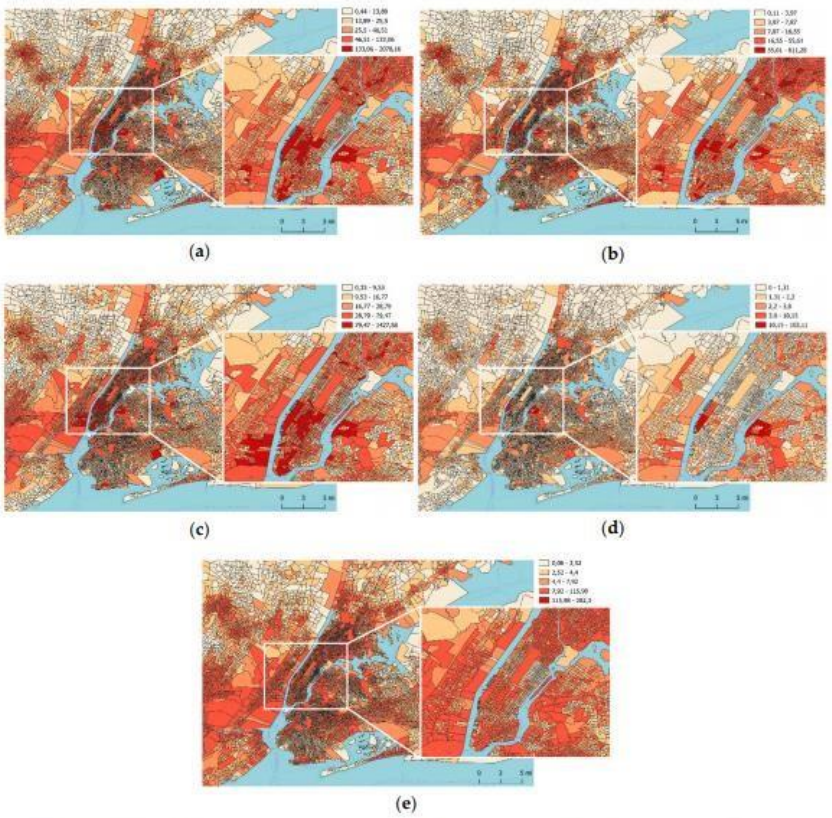


Figure 2.2.4: Map visualization of crime rate and the type of crime type

Another strength they have stated is they did not use past crimes as a predictor unlike other papers. Hence, they believe that it would be difficult to compare it across various location as past crime data only tend to be available in major urban areas as they found that databases onlytend to be defined at either an aggregated level (city, country) or at the local level (a detailed grid for only a city)

2.2.3 Weaknesses

For the limitations it was found that the accuracy of the paper could be higher. As other papers managed to reach an accuracy of around 90 percent. Other than that, it was found that there was

a huge overestimation of crime rate when the model compared with actual total crime ratefrom NIBRS crime data using violent crime cases in 17 states where they used data from 2018 and 2019. From the figure below you can see there a few states with very huge overestimationin predicted crime rate and actual crime rate.

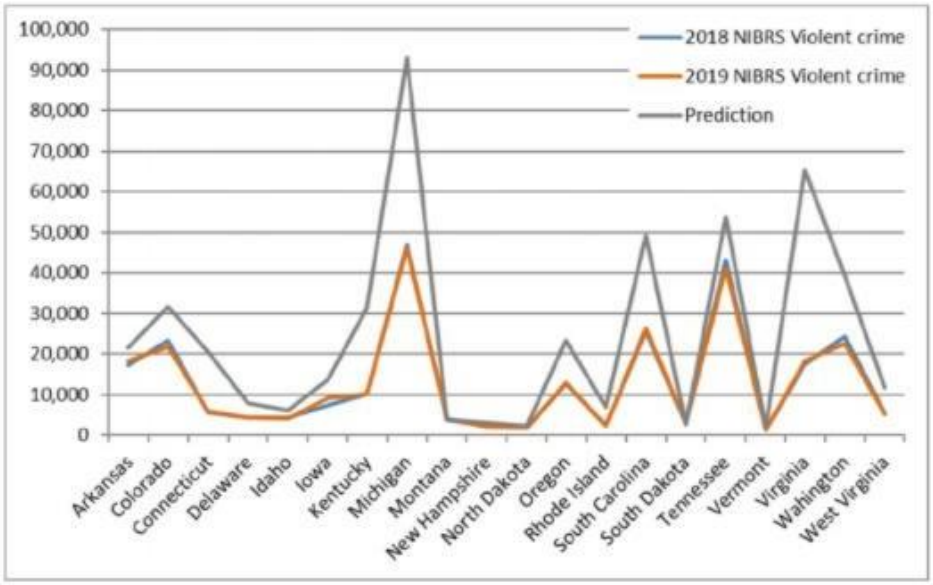


Figure 2.2.5: The Comparison of the predicted crime occurrences against the NIBRS data at the state level.

2.2.4 Recommendation

A recommendation for the limitations is that to improve the accuracy the paper could add additional type of features to improve the analysis. They could add points of interest or where people normally frequent such as malls, churches, schools, or places near streetlights. This could potentially help increase accuracy and aid in the overestimation of the crime count in thesystem.

2.3 Crime prediction through urban metrics and statistical learning

2.3.1 Brief Overview

Alves et al. [9] used a machine learning ensemble-based algorithm in this case a random forest regressor to predict the quantity of crime and influence of the urban indicators on the crime. The paper mainly focuses on one type of crime which is homicide and uses urban indicators from the Brazilian cities to predict the crime. For the analysis of the paper, they used the number of homicides at the city level as a crime indicator which they found as a reliable indicator as it is always reported and can usually be found from the Department of Informaticsof the Brazilian Public Health System - DATASUS. They then selected 10 urban indicators which they took from a national census conducted in 2000 which is Gross Domestic Product, illiteracy, family income, elderly population, child labour, female population, male population,population, sanitation, and unemployment. They stated that the urban indicators were used to train the model as a way to prevent underfitting or overfitting in their model. They also investigated how the accuracy changes as the time-lag would increase from a year to a decadeas you can see with the figure below.

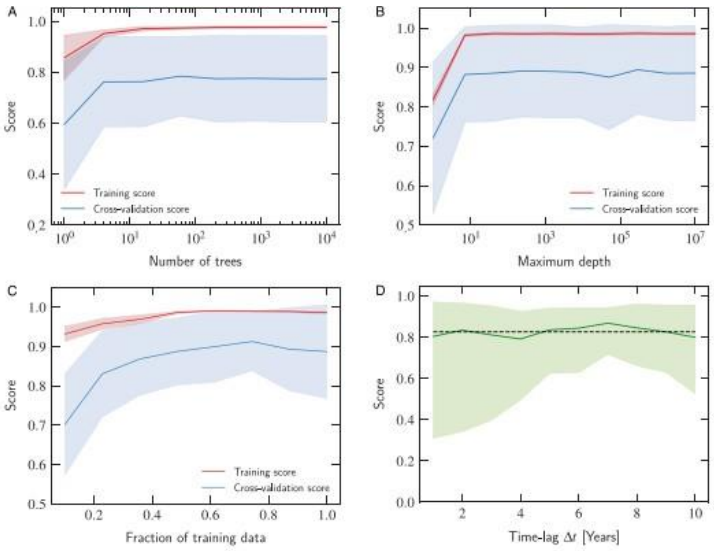


Figure 2.3.1: Validation and learning curves for the random forest regressor

2.3.2 Strengths

The strength of the paper is their use of the random forest algorithm as they found that with it after properly training it by splitting the data into 80 percent of training data and 20 percent testing data, they managed achieve high accuracies up to 97 percent while still being easy to interpret. They stated how this is very high compared to other papers as that used the same datathat only managed to predict homicides with an accuracy rating of around 38 percent. The firstfigure shows the empirical data versus the random forest prediction that is used for the realization of the algorithm. As the dataset was randomly split different runs gave them different results and scores. The second figure shows the probability distribution that was worked out using kernel density estimation method. From this they observed how the R^2 for 100 different splits were mostly concentrated around the peak of 80%.

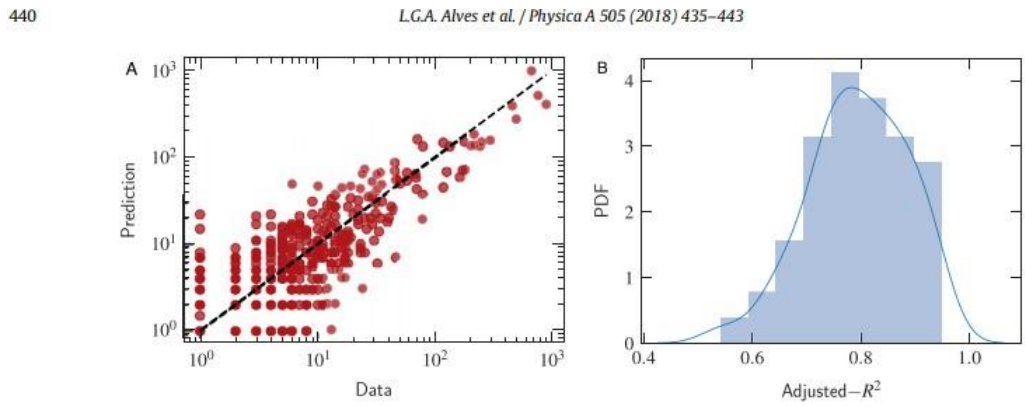


Figure 2.3.2: Data vs prediction of the models

Among other things they found that the random forest algorithm required little data preparation when performing regression. Another strength they state is how their approach is non-parametric, so it makes no assumption about the data. The paper also showed how even when the dataset is slightly changed the features still remained stable unlike simple linear models. The simple linear models are through ordinary-least-squares (OLS) linear regressions and that the predictors are assumed to be error free, constant variance, linearity, a lack of multicollinearity and normal residual distribution. Thus, when modelling crime several of these assumptions are often not satisfied which in turn causes misconceptions when the conclusions about factors affecting crimes are made.

2.3.3 Weaknesses and Limitations

For the limitation of this paper, it was found that they only used one model which was the random forest model. This is a problem as a comparison could not properly be made other thancomparing their model with models from another paper. Other than that, a random forest algorithm takes a lot of time for training this is due to the fact it combines several decision trees in order to determine the class. It also requires a lot of computational power and resources to build the numerous amounts of trees to combine their outputs. (Great Learning Team ,2020)

2.3.4 Recommendations

For the recommendation paper could include more algorithms for comparisons or use other algorithms. This is as algorithms such as gradient boosting algorithm can also provide predictive accuracy that is also very high while not needing as much computational power or time to train. An example of this is the gradient boosting algorithm which also like random forest algorithm usually provides high predictive accuracy while also having a lot of flexibility that can be optimized on loss function and allows for hyperparameter tuning option to make the function fit very flexible. The paper could also use other algorithms like Decision tree as itwas also found that it has less training time than random forest. As when we increase the number of trees in the future the random forest would require a lot more time to train. [2]

Chapter 3

System Methodology/Approach

3.1 Use Case Diagram

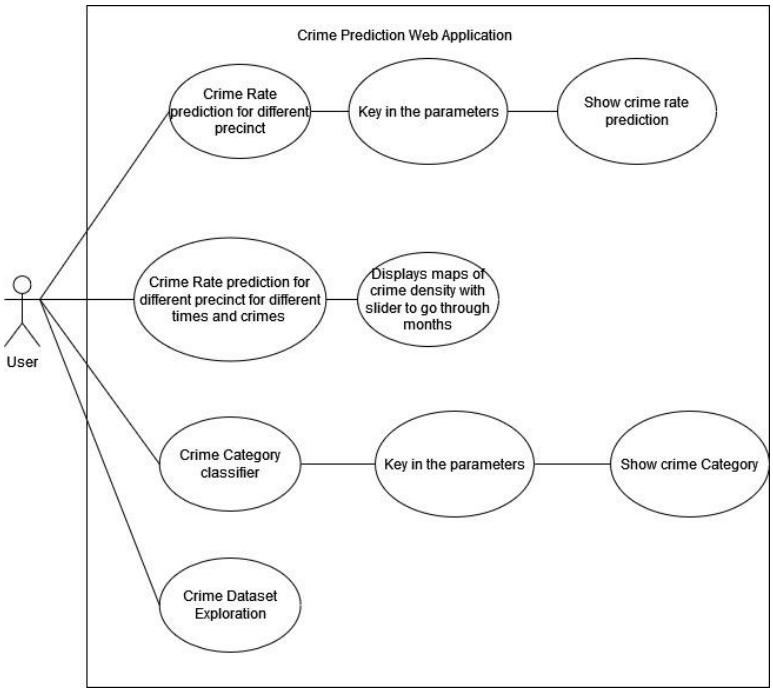


Figure 3.1.1: Use Case of Crime Prediction system

Figure 3.1.1 illustrates a use case diagram of how the users will interact with the web application. The users can choose from 4 different pages that they can use to help in the prediction of crime. For the first page, the user can explore the dataset used to aid in the prediction of this crime system. The user can access a histogram and a pie chart of the most common crimes. The user can also access a map that shows the crime rate of the hour, month, and year. For the second page, the user can key in the parameters of the police district, date, and hour zone. This will then output a prediction on whether the crime rate will be low, medium, or high. For the third page, the user can access multiple maps and a slider for each of the months that will show the predicted crime rate for each district. For the last page, the user can use a crime rate classification page where the users can input the parameters including a longitude and latitude parameter accompanied by a map to allow the users to be sure if the longitude and latitude inputted are correct.

3.2 Activity Diagram

3.2.1 Crime Category Classifier

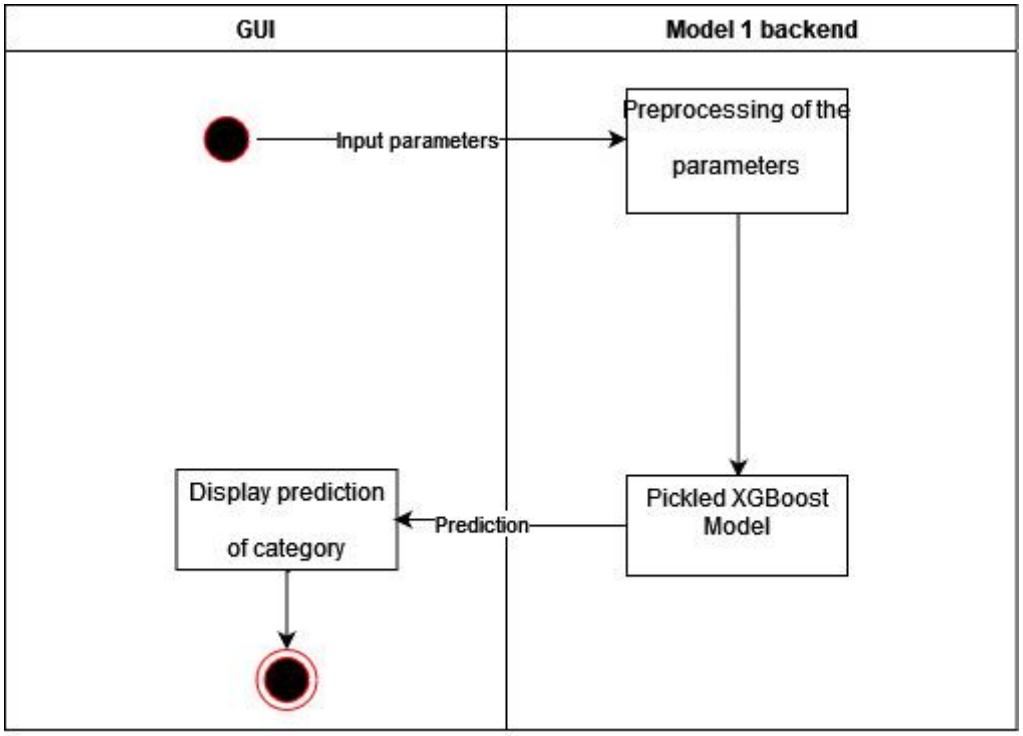


Figure 3.2.1: Activity diagram of Crime Classification module

The user can input the parameters of the place they would like to predict such as the police district, longitude, latitude, hour of the day, etc. The back end of the model will then proceed to perform feature extraction of the values inputted. For example, the longitude and latitude will be used to form 3 variances of the rotated cartesian coordinates and polar coordinates such as radian and angle. After that, the variables will then be passed on to the pickled XGBoost model that has been trained. The web app will then proceed to display the predicted category of the 4 crimes which are ‘LARCENY/THEFT’, ‘VEHICLE THEFT’, ‘DRUG/NARCOTIC’, and ‘VANDALISM’.

3.2.2 Crime Rate Classifier

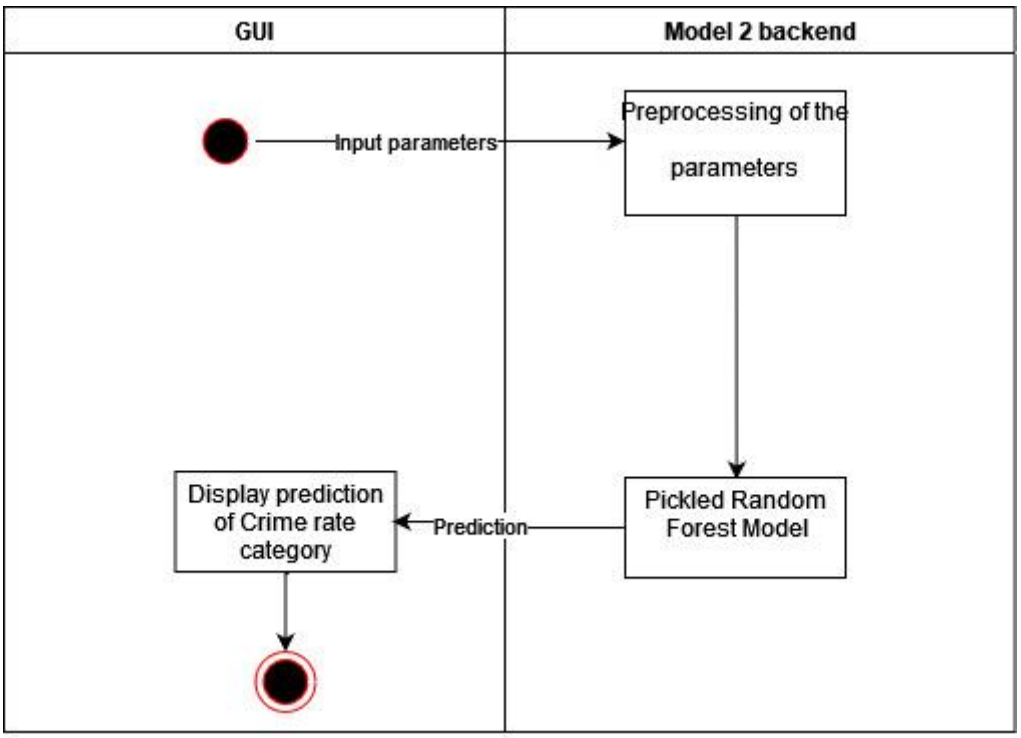


Figure 3.2.2: Activity diagram of Crime rate Classification module

The user can input the parameters of the place they would like to predict such as the police district, hour zone of the day, etc. The back end of the model will then proceed to perform feature extraction of the values inputted. After that, the variables will then be passed on to the pickled Random Forest model that has been trained and had the appropriate hyperparameter tuning. The web app will then proceed to display the predicted category of the 3 crime rates which are 'High Crime Rate', 'Medium Crime Rate', and 'Low Crime Rate'.

3.2.3 Crime Map Density

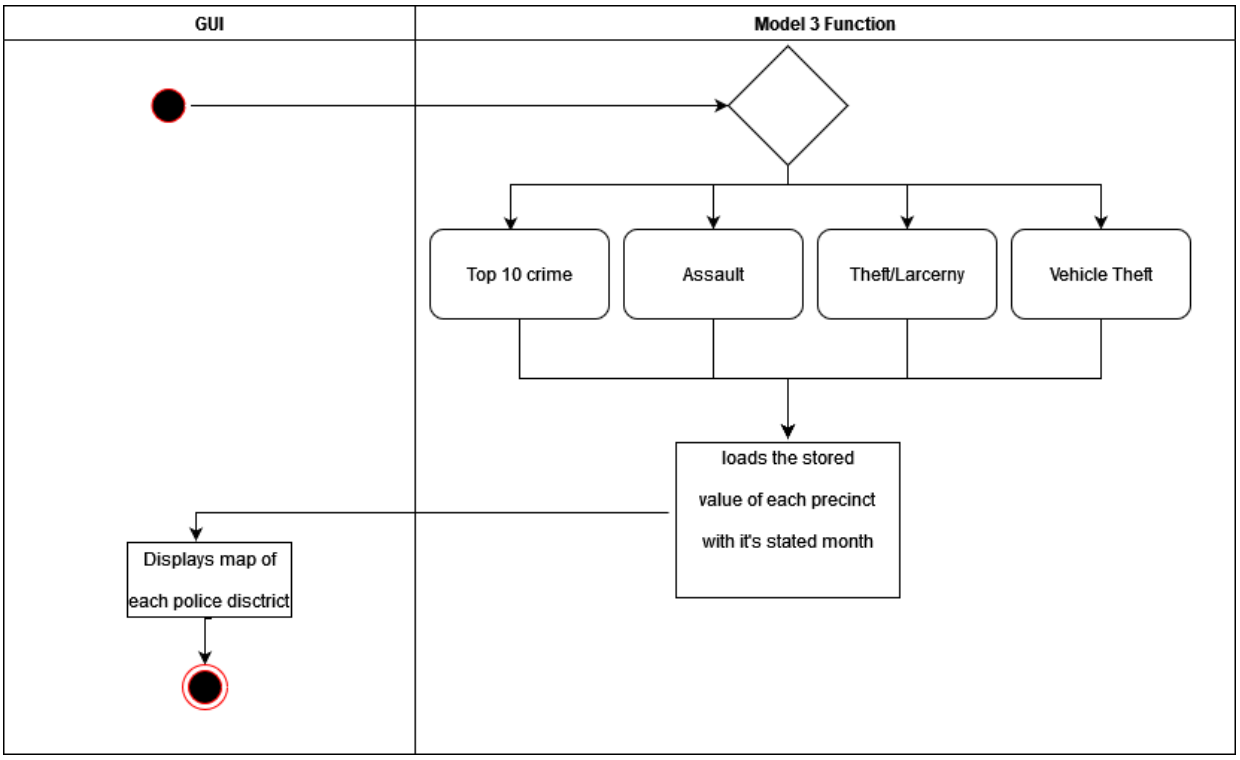


Figure 3.2.3: Activity diagram of Crime district density module

For this page, the user is able to choose from 4 different maps which were produced through the use of the Seasonal Auto-Regressive Integrated Moving Average with exogenous factors(SARIMAX) which are “Combined top 10 crime category of San Francisco”, “Assault”, “Theft/Larceny”, and “Vehicle Theft”. The user can use the slider to navigate through the months they want. The map will then display the density of the crime and when hovered over the police district will display the number of predicted crimes in said police district.

3.2.4 Exploratory Crime Dataset Analysis

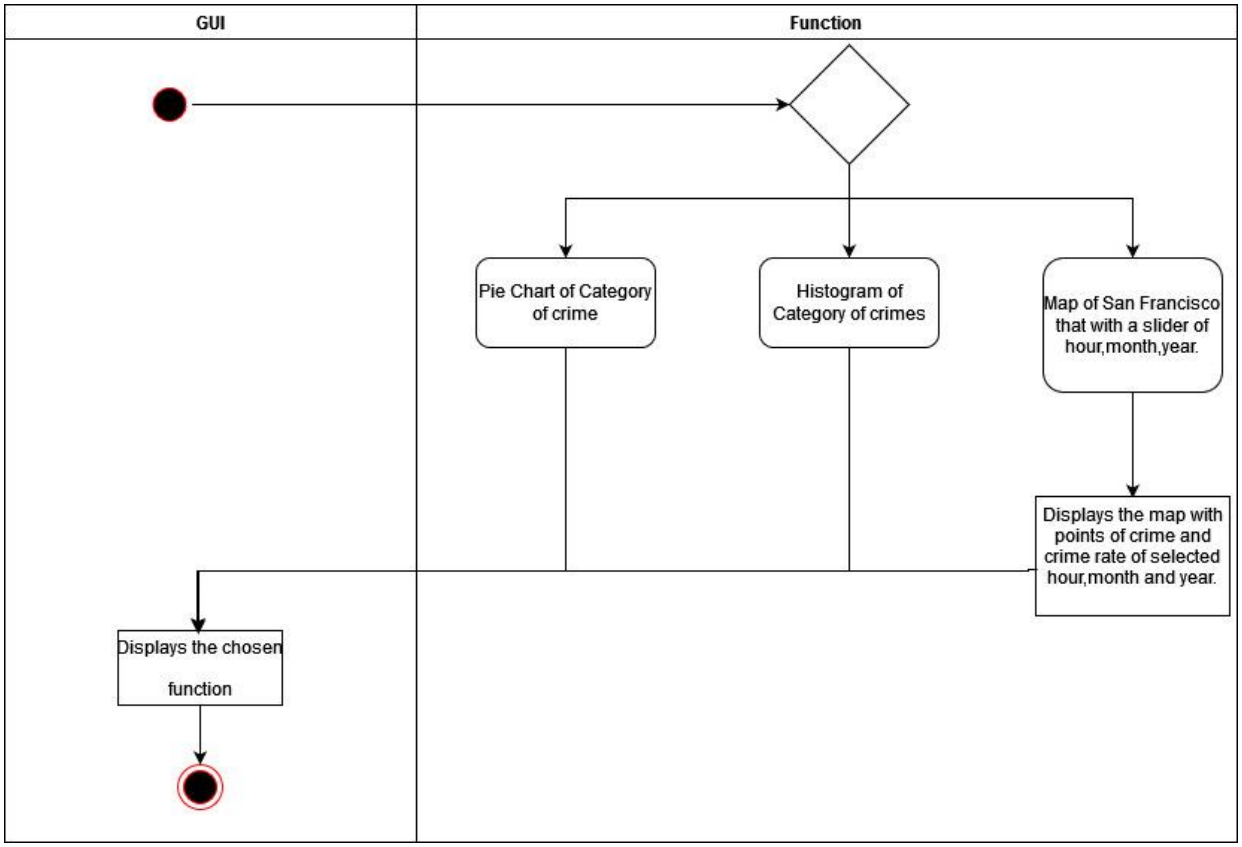


Figure 3.2.4: Activity diagram of Exploratory dataset Analysis module

This page allows the user to explore the dataset. The user can visualize the category of crimes in a pie chart or a histogram. The user can also use the slider provided for crime density in San Francisco by using the sliders of the hour, month, and year. This will provide a map that will display a map with points of crime and the total crime rate of the selected hour, month, and year.

3.3 Block Diagram

3.3.1 Category Classification Model Block Diagram

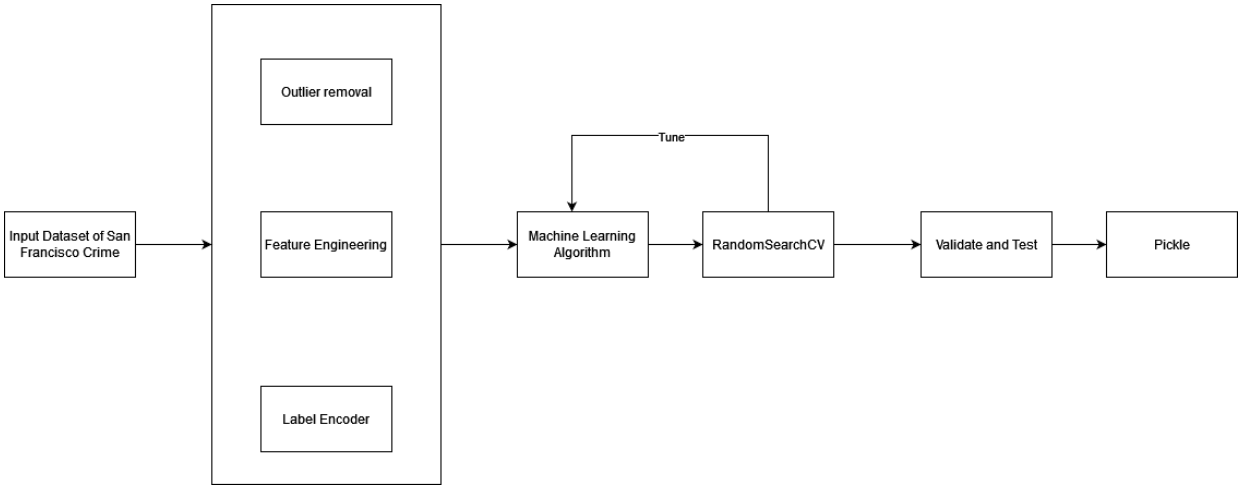


Figure 3.3.1: Block Diagram of category classification model

Figure 3.3.1 shows a block diagram of the model training process of the category classification model. Firstly, the data outliers are first removed to ensure the predictions are more accurate. Next, the data undergoes feature engineering whereby new features are produced from the existing data to simplify and speed up the data transformation to aid in improving the accuracy of the predictions. Lastly, the data then undergoes Label encoding to transform all categorical data into numerical values. The data will then be used to train a model to best predict its category of crime. The model will then be tuned using RandomSearchcv to increase its accuracy of the model. Lastly, all the models will be tested and validated using 5 cross-fold validation to ensure the model is not overfitting. The model will then be pickled and then be deployed in the web app.

3.3.2 Crime Rate Classification Model Block Diagram

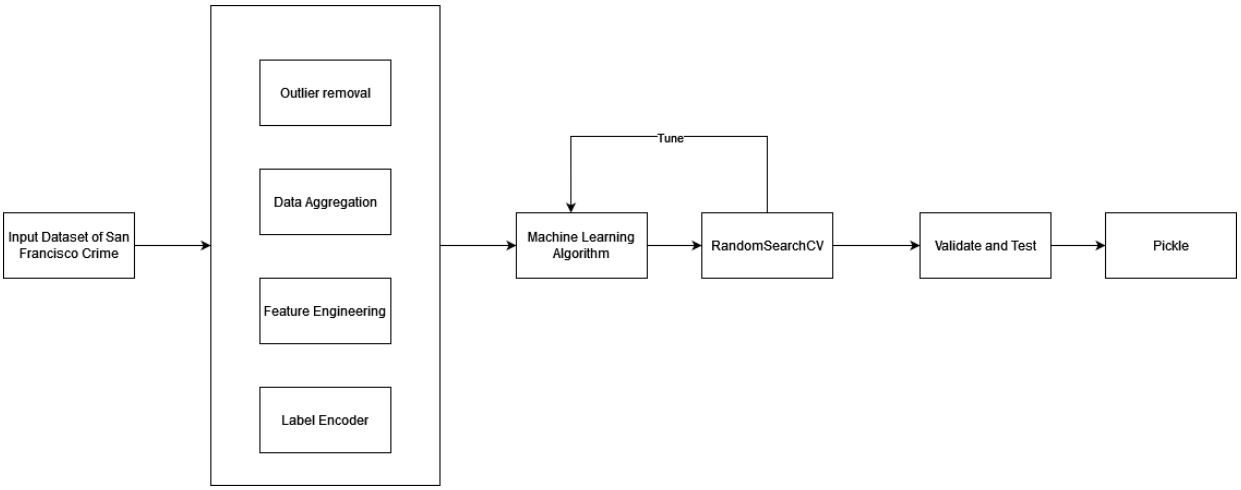


Figure 3.3.2: Block Diagram of crime rate classification model

Figure 3.3.2 shows a block diagram of the model training process of the crime rate classification model. Firstly, the data outliers are first removed to ensure the predictions are more accurate. After that, the data undergoes Data Aggregation to create a new column in which category counts. this is done to create a new column for the model to predict which will be the alarm type which will either be high crime rate, medium crime rate, or low crime rate. Next, the data undergoes feature engineering whereby new features are produced from the existing data to simplify and speed up the data transformation to aid in improving the accuracy of the predictions. Lastly, the data then undergoes Label encoding to transform all categorical data into numerical values. The data will then be used to train a model to best predict its category of crime. The model will then be tuned using RandomSearchcv to increase its accuracy of the model. Lastly, all the models will be tested and validated using 5 cross-fold validation to ensure the model is not overfitting. The model will then be pickled and then be deployed in the web app.

3.3.3 Crime Forecast Block Diagram

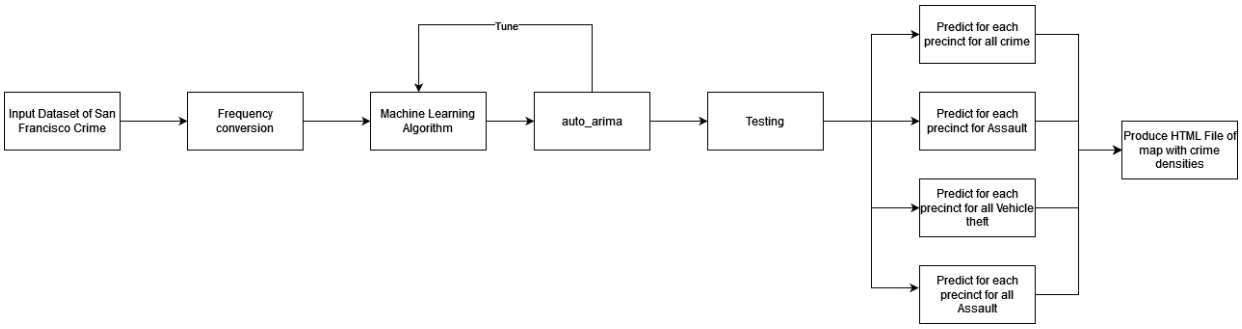


Figure 3.3.2: Block Diagram of crime density model

Figure 3.3.2 shows a block diagram of the model training process of the crime density model using SARIMAX. Firstly, I group all the data for every month and produce a data frame consisting of the total crime rate of each month. I then remove the last month as the last month's count is significantly lower than the other months. I then begin by using a base SARIMAX model to train and test the dataset. I then use auto_arima to get the best model parameters. After that, I tested the model that has the lowest MAE. The best model will then be used to predict the future crime statistic for each precinct and type of crime to get a crime density map which will be output as an HTML file. This will then be put into the web app to allow users to explore the crime rate of each subsequent month out of the dataset.

3.4 System Architecture Diagram

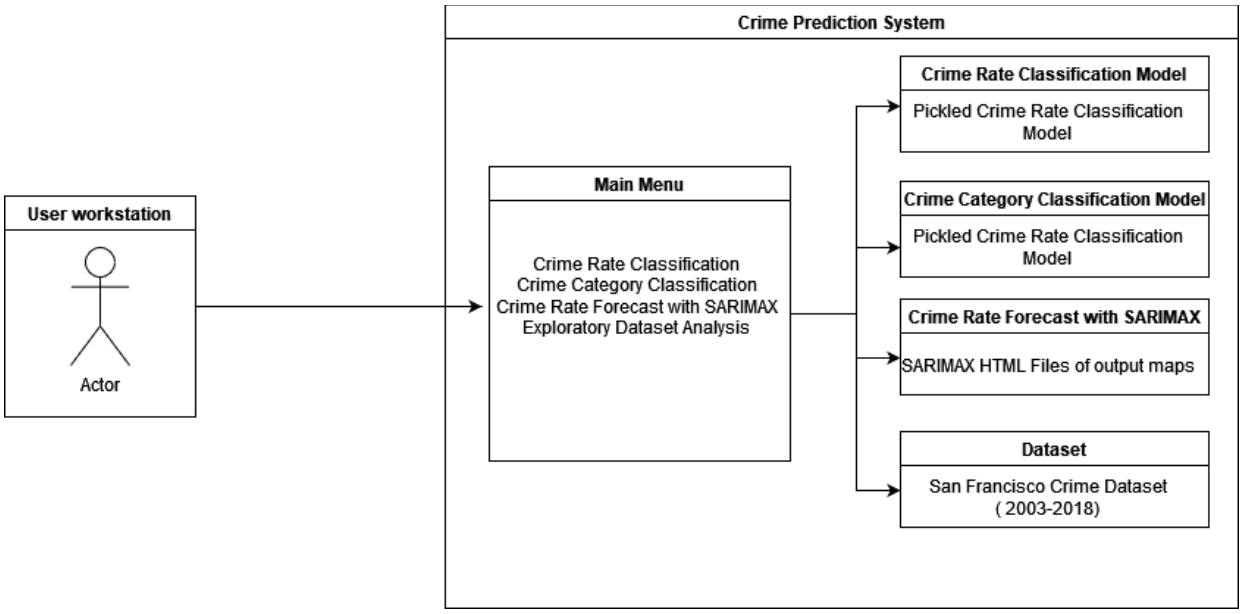


Figure 3.4.1 System Architecture diagram of Web Application

Figure 4.3.1 shows the system Architecture Diagram of the web application. The main menu represents the 4 pages the user can access which are Crime Rate Classification, Crime Category Classification, Crime Rate forecast with SARIMAX, and the Exploratory Dataset Analysis. The Crime Rate Classification allows the users to predict either “High Crime Rate”, “Medium Crime Rate” or “Low Crime Rate” given the police district quarter of the day and date. The Crime Category Classification much like the Crime Rate classification allows the user to predict the top 4 crimes given the location, date, police district, street type, and block number. For the Crime Rate forecast with SARIMAX, the user can access the forecasted crime types on 4 different maps which will show the user the predicted number of crimes and also which police district with a slider to go through the months. For the last page, the users can explore 2 different charts which are the pie chart and a histogram of all crime categories. The user is also able to visualise the crime across the years on a map. The map allows them to visualise the crime rate and also the crime density in different spots in the city given the hour, month, and year.

Chapter 4

Methods/Technologies Involved

4.1 Methodology

4.1.1 Classification Pipeline for Model 1

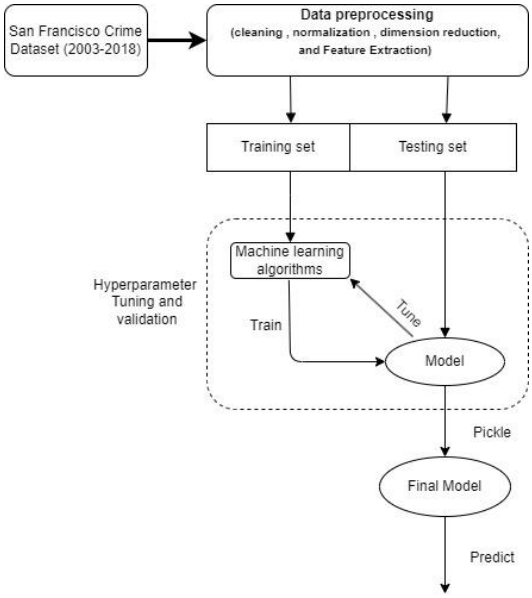


Figure 4.1.1: Classification pipeline of model 1

In this project, the crime rate classification system is developed using XGBoostRandom Forest and the Decision tree Algorithm. Firstly, the San Francisco Crime Rate Dataset is first input into the system where it will be pre-processed by doing dimension reduction based on feature importance and normalization. The data set will then go through feature extraction whereby values will be extracted After that the data set will be split into 8:2 for the training set and testing set respectfully. I will then pass it into the XGBoost, Random Forest, and Decision tree. I will then apply 5 – fold cross-validation to the models to detect overfitting. The models will then be tuned with RandomSearchCV to generate combinations of hyperparameters that would lead to models with higher accuracies. For testing, the best estimator for each model is then used and the process is repeated using the precision, recall, and f1 to calculate how well the model generalizes the data. The model will then be pickled so that it can be used in the web application whereby the user can input parameters such as the date, time, longitude, and latitude in order to predict the most probable crime.

4.1.2 Classification Pipeline for Model 2

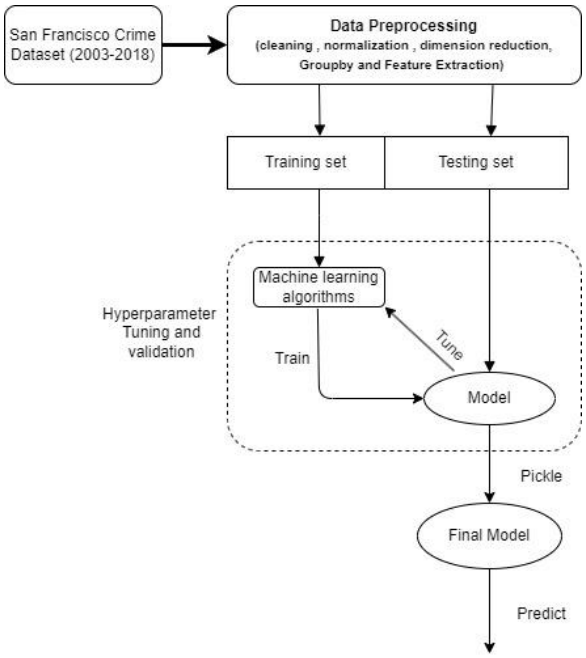


Figure 3.1.2: Classification pipeline of model 2

In this project, the crime rate classification system is developed using XGBoostRandom Forest and the Decision tree Algorithm. Firstly, the San Francisco Crime Rate Dataset is first input into the system where it will be pre-processed by removing wrong values of the longitude and latitude. The data set will then go through feature engineering by decomposing the Date and time feature. After that data aggregation is performed whereby to get the crime count per the time. The goal of this is to predict the crime count. After that, the data set will be split into 8:2 for the training set and testing set respectively. I will then pass it into the XGBoost, Random Forest, and Decision tree. The models will then be tuned with RandomSearchCV to generate combinations of hyperparameters that would lead to models with higher accuracies. For testing, the best estimator for each model is then used and the process is repeated using the Unweighted Average Recall, precision, recall, and f1 to calculate how well the model generalizes the data. The model will then be pickled so that it can be used in the web application whereby the user can input parameter such as the date, hour_zone, and police district to predict the most probable crime rate in the police district at the time.

4.1.3 ARIMA Pipeline for Model 3

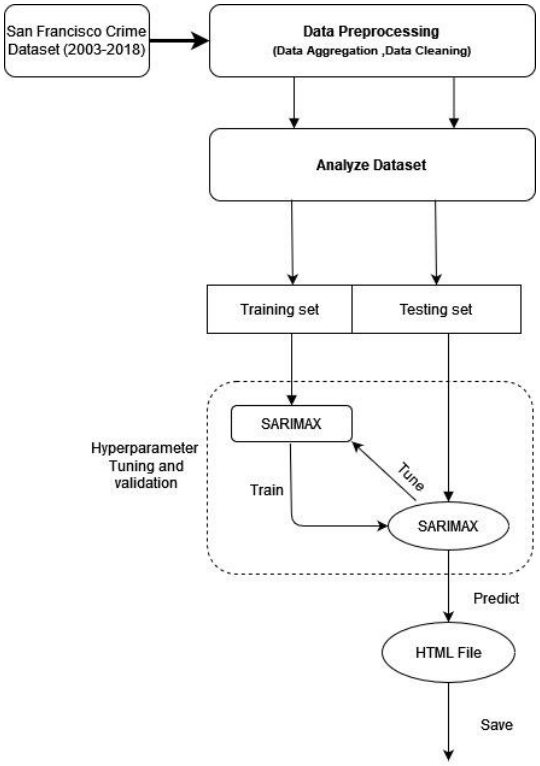


Figure 4.1.3: ARIMA pipeline of model 3

In this model, the San Francisco crime dataset will be processed with the resample('MS').sum() function that will group all the dates of every month and produce a data frame consisting of the total crime rate of each month. This will then be used to analyse the dataset where I will get the ADF Statistic the P-value and the critical value through the use of the Dickey-Fuller test. This is done to see if the time series dataset is stationary or not. The autocorrelation is then used to check for seasonality based on the lag shown. After that, the SARIMAX model is then trained and tested to see how well it predicts the crime rate. Hyperparameter tuning is then performed using the auto_arima function which helps decide the best pdq PDQ values. This chooses the one with the largest log-likelihood and lowest AIC. The best model is then used to predict crime for each precinct and different crime types. This will then be output to an html file using the chloropleth Mapbox function to visualise the crime density.

4.2 Technology Used

4.2.1 Laptop Specifications

Central Processing Unit (CPU)	- AMD Ryzen 7 5800H Processor 3.2GHz, 16M Cache, 8 Cores
Graphics Processing Unit (GPU(s))	NVIDIA GeForce RTX3060 6GB GDDR6
Random Access Memory (RAM)	- 8GB
Drives	- SSD 500GB
OperatingSystem	- Microsoft Windows 11 Home 64-bit

Table 3.2.1: Specifications of Laptop

4.2.2 Software

1. Visual studio code

Visual studio code provides several useful extensions that can be downloaded. It includes auto-indentation, easy to use shortcuts, syntax highlighting and a very clean user interface.

2. Python 3.9.4

Figure 4.3.1: Download Page for Python 3.9.4

We use python as it has an extensive selection of library and frame works for machinelearning from Sckit-learn, Numpy ,etc

Extension used from Visual Studio Code:

Figure 4.3.2: Extension Download from extension tab of Visual Studio

This is a python extension that can be downloaded from the extension tab. Using it in visual studio code allows me use to autocomplete and Intellisense while coding in python.

3. Streamlit

Streamlit is an open-source python framework for data scientist to build their web apps for machine learning models quickly and easily. Streamlit much like flask allows the users to create an app for data scientist to explore their models without having to master back-end engineering tasks. This allows for users to create an interactive dashboard extremely quickly as the learning curve is not very steep.

4.3 User requirements

- User should be able to input parameter of the place you would like to predict and get predictions for crime category
- User should be able analyse the crime rate density in police precinct of the city based on crime category.
- User should be able to analyse hotspot based on hour zone of the day and see the crime rate

4.4 Non-Functional Requirements

- The system shall be able to make predictions within 1 second

4.5 System performance

The system allows its users to observe the crime density in different locations of the city. One of the things the system should have a decent accuracy around 65% make sure the area the police are going to patrol will not be empty which may waste the time of the police personnel.

The system should also be able to properly show density of the crime rate to allow police to know where to be placed easily.

4.6 Verification plans

To ensure the system can produce the required outputs. The inputs are as seen below.

1) Input parameter of location, time and date of crime.

The verification will be done based on the one scenario and explained as follow:

a) Detection Scene

Procedure Number	P1
Test Name	Crime Category Classification
Applicable Requirements	Have an Internet connection
Purpose/Scope	To produce a crime prediction system to show crime category
Precautions	The records put in should not have null values
Equipment/Facilities	Laptop
Acceptance Criteria	The system displays the category of the crime based on parameters input
Procedures	1.Input the parameters of the crime location, time, date, street type and block number.
Troubleshooting	Repeat the procedure

Table:4.6.1 Verification plan for Crime category prediction

Procedure Number	P2
Test Name	Crime alarm Classification
Applicable Requirements	Have an Internet connection
Purpose/Scope	To produce a crime prediction system to show crime rate of precinct during certain times of day
Precautions	The parameters put in should not have null values
Equipment/Facilities	Laptop
Acceptance Criteria	The system displays crime rate
Procedures	1.Input the parameters of the crime location, time, date, street type and block number. 2. The crime rate category will then be displayed.
Troubleshooting	Repeat the procedure

Table:4.6.2 Verification plan for Crime rate prediction

Procedure Number	P3
Test Name	Crime Category Classification
Applicable Requirements	Have an Internet connection
Purpose/Scope	To produce a crime prediction map to show predicted crime density based on the month and category of crime
Equipment/Facilities	Laptop
Acceptance Criteria	The system displays the density of the crime on the map
Procedures	1.Users can scroll through the months to see the crime densities of the crime based on the month selected 2.Hover over each precinct to see predicted crime rate.
Troubleshooting	Repeat the procedure

Table:4.6.3 Verification plan for SARIMAX

4.7 System design

4.7.1 Crime Category Prediction

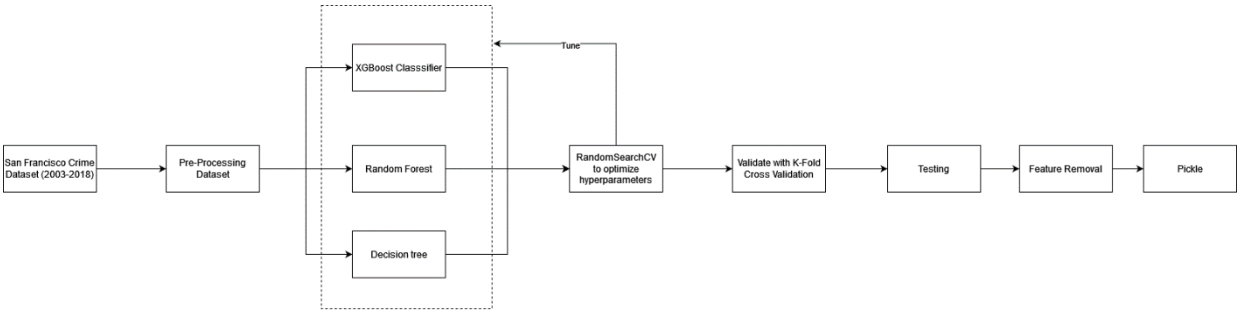


Figure 4.7.1 System diagram Crime Category Prediction

Dataset Used

To begin the implementation for the crime rate prediction. I used the San Francisco Crime Dataset that was made publicly available from San Francisco government website and includes a total of 2,129,524 instances. Each of the instances comes with 34 variables. However, I chose 9 which were listed in the website which are Date, Time, Category, DayOfWeek, PdDistrict, Address, X , Y.

Pre-processing

For the pre-processing stage I drop the rows where the X and Y coordinates for x=-120.5, y= 90 as it all has different police districts. I then perform feature engineering where I extracted temporal features from date and then Spatial features from address, X, and Y.

Variables	Feature Engineering
X, Y	Rot30_X, Rot30_Y, Rot45_X, Rot45_Y, Rot60_X, Rot60_Y, Radius, and Angle
Date	Month, Year, Day
Time	Hour
Address	BlockNo, StreetType
DayOfWeek	Weekend
Month	Season
Hour	Hour_Zone

Table:4.7.1 Feature Engineering of Variables in dataset

I then performed label encoding on some columns that used to be categorical data to allow the model to use it more efficiently. I then proceed to split the dataset into x- for features and y- for output label.X and y are then split into 2 sets which is training set and testing sets which will uses a random sampling ratio of 8: 2. The testing set will allow me to perform an unbiased evaluation of the model after the hyperparameter tuning is done. This would improve the confidence in the model's ability to generalize. The Y features will be Category. As there are too many categories to predict reliably, I decided to predict the top 3 which are "LARCENY/THEFT", "ASSAULT"," VEHICLE THEFT" and "DRUG/NARCOTIC". The reason why I did not choose "OTHER OFFENSES" and "NON-CRIMINAL" was that it included things like 'TRAFFIC VIOLATION ARREST' or 'FRAUDULENT GAME OR TRICK', 'OBTAINING MONEY OR PROPERTY' 'TRAFFIC ACCIDENT' or 'MENTAL DISTURBED' all of which did not seem to have any correlation to one another. After dropping all the other rows, I had 889,008 rows remaining.

Model Training

For the Model training stage, according to the literature review I implemented 2 models which are Random Forest,and XGBoost.

Random Forest was chosen as it reduces overfitting in decision trees and helps improve accuracy by combining the trees. It also works well with both categorical and continuous values. Like gradient boosting it also handles missing values that are in the dataset. It does not need normalizing of data as it uses a rule-based approach [8].

XGBoost is considered a better version of the Gradient Boosting model as it is a more regularised version of it. This is as it has a built in L1 (Lasso Regression) and L2 (Ridge Regression) which helps prevent overfitting. It also able to utilise the GPU hence making XGBoost faster to train compared to other models [11].

Decision tree algorithm much like random forest also does not require you to normalize the data and can also be implemented even without scaling the data. It also does not require you toimpute the data for missing values. Its pre-processing step also requires less time and also less code and analysis to compared to the other algorithms above [4].

To evaluate the initial models, I decided to use Log loss and accuracy to check how well they perform initially without hypermeter tuning. The way log loss works is how close the prediction probability is to the model. The smaller the log loss value the closer it is to the actual value. I will then use these values to compare with the tuned models to gauge the increase in performance [6].

Tuning

For the tuning stage I used the RandomSearchCV to generate a combination of hyperparameters that would lead to a higher accuracy. Unlike with GridSearchCV it does not go through every single set of possible values. It is also faster as you can set the amount of iteration it will go through on the parameters you set rather than having it go through each and every one which can take upwards to 24 hours. The way I tuned the parameter is by setting the scoring to log loss and choosing the one with the smallest log loss value.

Validation

For the validation stage to detect for overfitting, the models use the 5-fold cross – validation with the scoring set to accuracy. This will return a classification accuracy score for each fold in the training ser. The mean k-score is calculated and compared to the accuracy score of the models before cross-validation. This would allow me to find out the differences in the accuracy scores and verify the accuracy of the models on multiple and different subsets of data.This would also help me find out how well the models chosen can help generalize the data.

Other than that, cross-validated estimates of each instance in the input matrix are generated to compute the confusion matrix for each model. This is helpful as it can help calculate the precision, recall and F1 score for each model.

Testing

Lastly for the testing stage the best estimator for each model is then used and the process is repeated using the precision, recall, and F1-Score to calculate how well the model generalizes the data.

Feature Removal

As there are too many features to be implemented, I will use the built-in ensemble method to find the importance of each feature. Any Feature found to have less than 1 percent of importance will be removed. After that I will use the accuracy, log loss, precision, recall, and F1-Score. To check how the model performed before and after the feature removal

Pickling

The best model is then pickled and then put into drop box in order to be deployed to Streamlit cloud.

4.7.2 Crime Rate Classification

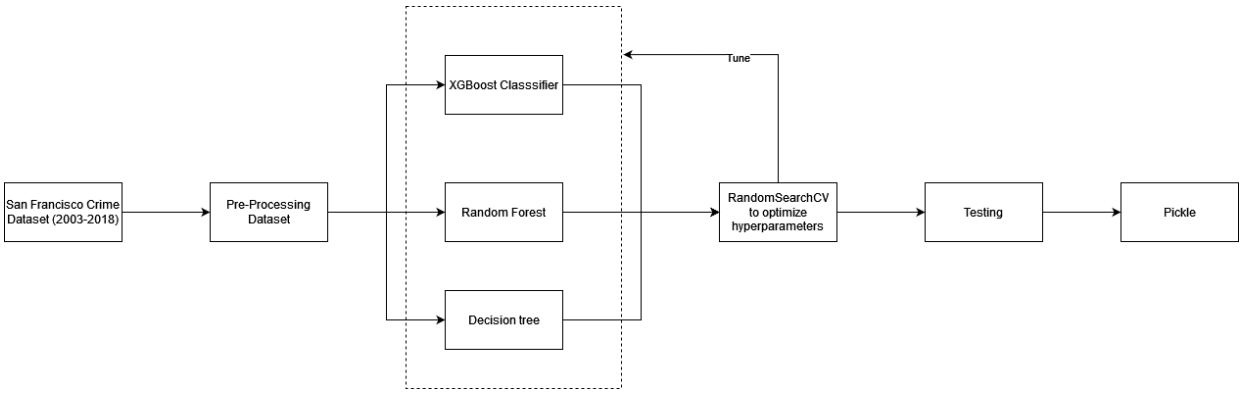


Figure 4.7.2 System diagram Crime Rate Classification

Dataset Used

To begin the implementation for the crime rate prediction. I used the San Francisco Crime Dataset that was made publicly available from San Francisco government website and includes a total of 2,129,524 instances. Each of the instances comes with 34 variables.

Pre-processing

For the pre-processing stage I drop the rows where the X and Y coordinates for x=-120.5, y= 90 as it all has different police districts. I then perform feature engineering where I extracted temporal features from date and time.

Variables	Feature Engineering
Date	Month, Year, Day
Time	Hour
Month	Season
Hour	Hour_Zone

Table:4.7.2 Feature Engineering of Variables in dataset

I then perform data aggregation using the above variables which are “Year”, “Month”, “Day”, “DayofWeek”, “Hour_Zone”, “PdDistrict”, and “Season”. I then aggregates all categories to make a count that will be used for the Y. The count will then be split into 3 categories to be predicted which is “high crime rate”, “medium crime rate” and “low crime rate”. This is Done by checking the 0.75 percentile of each end to find out what count should be for “High Crime Rate” and “Low Crime Rate”.

I will then perform label encoding on some columns that used to be categorical data to allow the model to use it more efficiently. I will then proceed to split the dataset into x- for features and y- for output label.X and y are then split into 2 sets which is training set and testing sets which will uses a random sampling ratio of 8: 2. The testing set will allow me to perform an unbiased evaluation of the model after the hyperparameter tuning is done. This would improve the confidence in the model's ability to generalize. The Y features will be crime rate category derived from the crime count, I decided to use the top 10 categories of crime.

Model Training

For the Model training stage, according to the literature review I implemented 3 models which are Random Forest, Decision Tree, and XGBoost Classifier.

To evaluate the initial models, I decided to use Log loss,unweighted average recall and accuracy to check how well they perform initially without hypermeter tuning. The reason why I used unweighted average recall works is to ensure that the rate classes are also represented which are the "Low Crime Rate" and "High Crime Rate".

Tuning

For the tuning stage I used the RandomSearchCV to generate a combination of hyperparameters that would lead to a higher accuracy. Unlike with GridSearchCV it does not go through every single set of possible values. It is also faster as you can set the amount of iteration it will go through on the parameters you set rather than having it go through each and every one which can take upwards to 24 hours. The way I tuned the parameter is by setting the scoring to log loss and choosing the one with the smallest log loss value.

Testing

Lastly for the testing stage the best estimator for each model is then used and the process is repeated using the Log Loss,precision, recall, UAR, and F1-Score to calculate how well the model generalizes the data.

Pickling

The best model is then pickled and then put into dropbox in order to be deployed to Streamlit cloud.

4.7.3 Crime Rate Prediction With SARIMAX

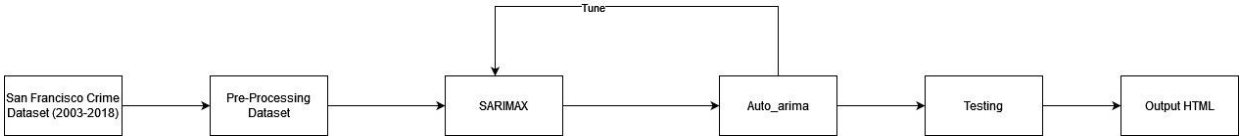


Figure 4.7.3 System diagram Crime Rate Prediction With SARIMAX

Dataset Used

To begin the implementation for the crime rate prediction. I used the San Francisco Crime Dataset that was made publicly available from San Francisco government website and includes a total of 2,129,524 instances. Each of the instances comes with 34 variables.

Pre-processing

I first chose the top 4 crimes in the dataset before using the groupby function in order to use the resample('MS'). sum() function to group all the months together to create a count of crime for each month. This is as the resample function does not work on dataframes that are not DatetimeIndex. After that I remove the last month of the of the dataset as the count there is significantly lower than all the other months. After that I will perform the Dickey-Fuller test to get the ADF Statistic, the P-value, and the critical value. This test is done in order to check if the dataset is stationary or not. After that an autocorrelation test is done to check for the seasonality of the dataset based on the lag that is most prominent.

Model training and tuning of parameters

For the Model training stage, I use the SARIMAX model. For the seasonality I chose 12 as that was the lag that was observed from the auto correlation test. This model is then trained and tested. After that to ensure that I chose the best hyper parameters I use the auto_arima function which helps decide the best pdq PDQ values. This chooses the one with the largest log likelihood and lowest AIC. The best model is then used to predict crime for each precinct and different crime types.

Creating the map and predicting

To create the map plot I first used the geopandas .readfile function to read in a geojson with the geometry of each police district. I then create a function that would read the dataset and make a datetime index for each police district with it's own set of frequency counts. I will then use the .predict function from SARIMAX with the optimal parameters to predict up to a year of crime out of the dataset. I then use the output to plot a chloropleth map that will display the density of crime. This is repeated 3 more times for 3 other category of crimes which are "Theft/Larcerny" , "Vehicle Theft" , and "Assault".

4.7 Timeline

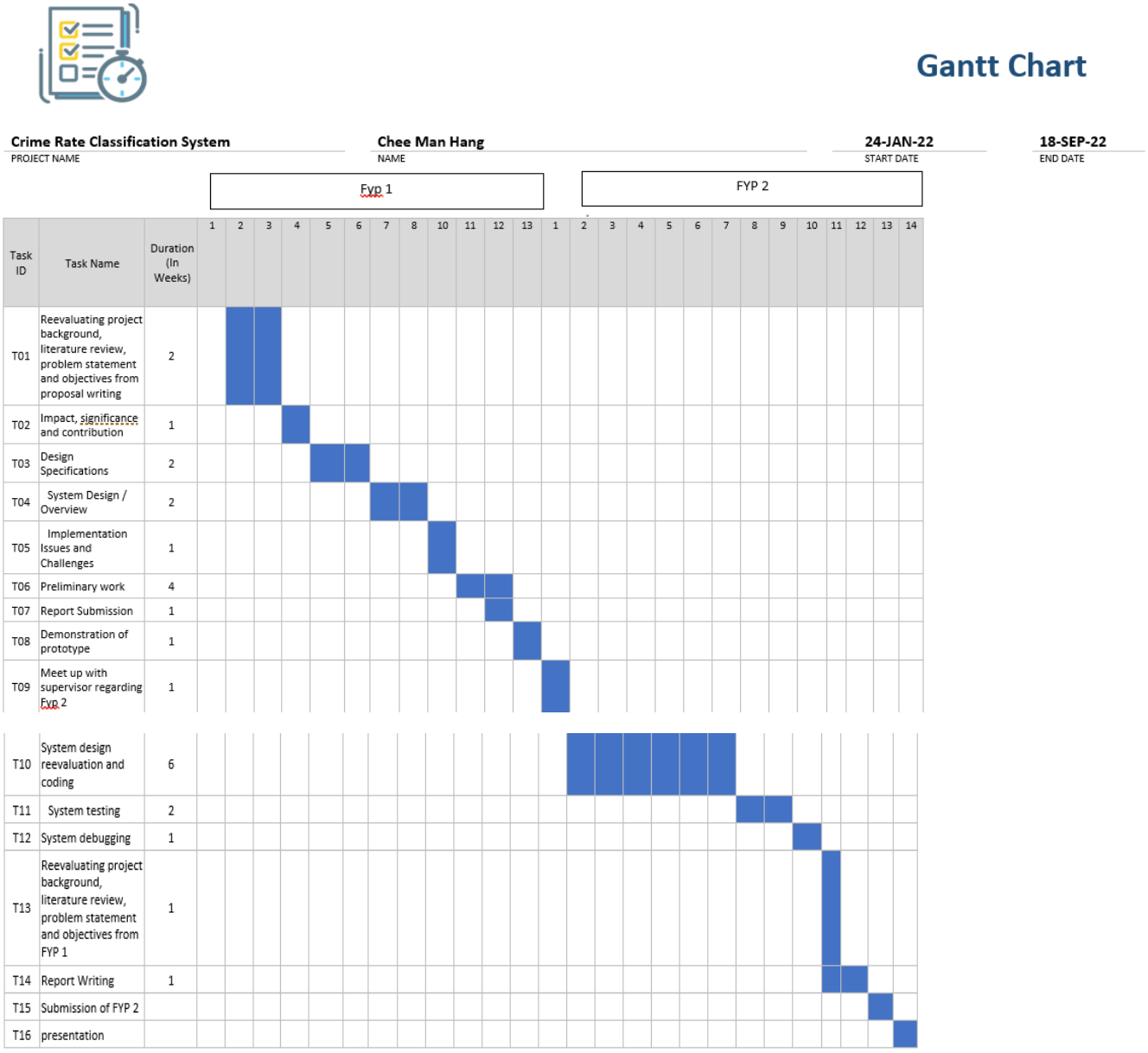


Figure 4.7.1 Gantt chart of FYP 1 and FYP 2 Schedule

CHAPTER 5

System Implementation and Evaluation

5.1 System Implementation for Model 1 Crime Category Classification

Dataset Used

The Crime dataset includes a total of 2,129,525 instances. Each of the instances comes with 35 variables.However I kept only 8 as per the website I took them from which are Date, Time, Category, DayOfWeek, PdDistrict, Address, X ,and Y. I then visualised the crime rate density based on the areas of San Francisco city.

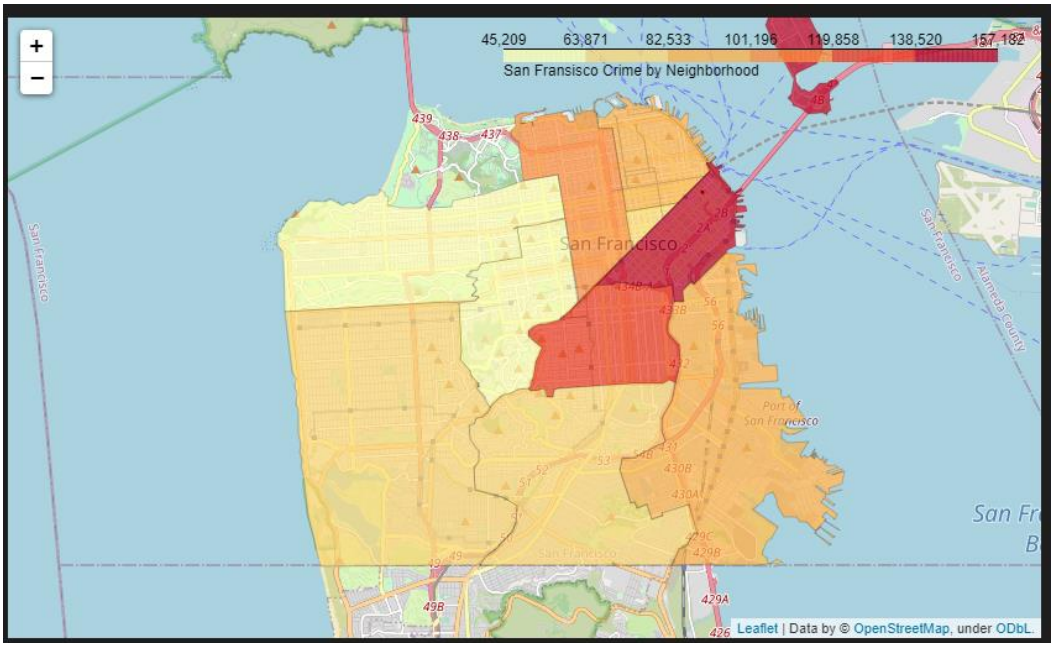


Figure 5.1.2 Density of crime by neighbourhood

Pre-processing

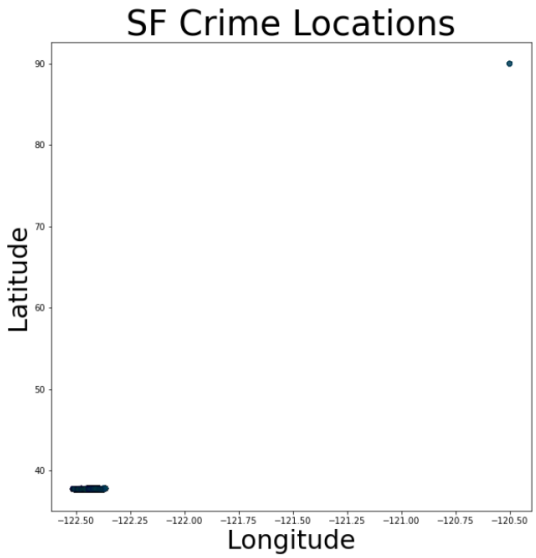
For the pre-processing stage I first dropped rows of data that have I found had wrong values which was rows with the X and Y coordinates for x=-120.5, y= 90.

```
print(crime['Y'].min())
print(crime['Y'].max())
print(crime['X'].min())
print(crime['X'].max())
crime['Y'].replace(to_replace= crime['Y'].max() ,value=np.nan, inplace=True)
crime['X'].replace(to_replace= crime['X'].max() ,value=np.nan, inplace=True)
crime = crime.dropna()

crime.isnull().sum()

2]  ✓ 3.9s
```

Figure 5.1.2 Coding of removing rows of outlier points



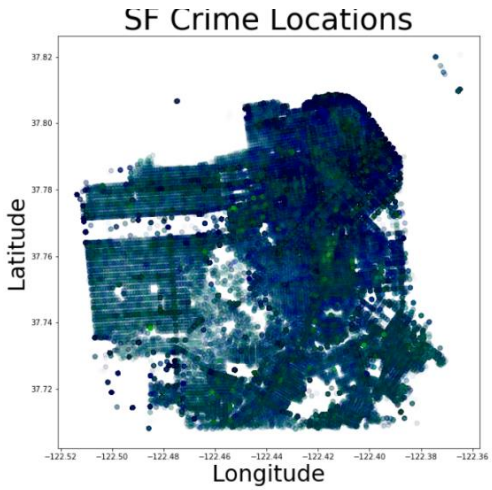


Figure 5.1.3 and 5.1.4 Before and After removing Outlier from dataset presented in a plot.

As there are too many categories to predict reliably, I decided to predict the top 33 which are “LARCENY/THEFT”, “ASSAULT”, “VEHICLE THEFT” and “DRUG/NARCOTIC”. The reason why I did not choose “OTHER OFFENSES” and “NON-CRIMINAL” was that it included things like ‘TRAFFIC VIOLATION

ARREST' or 'FRAUDULENT GAME OR TRICK', 'OBTAINING MONEY OR PROPERTY' 'TRAFFIC ACCIDENT' or 'MENTAL DISTURBED' all of which did not seem to have any correlation to one another. After dropping all the other rows, I had 889,008 rows remaining.

I then performed some feature engineering to extract features the feature engineering are as seen below.

Variables	Feature Engineering
X, Y	Rot30_X, Rot30_Y, Rot45_X, Rot45_Y, Rot60_X, Rot60_Y, Radius, and Angle
Date	Month, Year, Day
Time	Hour
Address	BlockNo, StreetType
DayOfWeek	Weekend
Month	Season
Hour	Hour_Zone

Table 5.1.1 Feature engineering of dataset for spatial and temporal variables

```
# Transform the Date into a python datetime object.
crime[["Month", "Day", "Year"]] = crime["Date"].str.split("/", expand = True)
crime[["Hour", "Minute"]] = crime["Time"].str.split(":", expand = True)
```

Figure 5.1.2 Coding of Date and Time Feature engineering e

For the temporal features I extracted Month, Year, Day from the Date Column. From the Time Column I extracted the hour feature which was then used to derive the Hour_zone feature as seen below.

```python
# Hour Zone 0 - Past midnight, 1 - morning, 2 - afternoon, 3 - After work hours, 4 - Late night
def get_hour_zone(hour):
    if hour >= 0 and hour < 6:
        return 0
    elif hour >= 6 and hour < 12:
        return 1
    elif hour >= 12 and hour < 18:
        return 2
    elif hour >= 18 and hour < 22:
        return 3
    elif hour < 0 or hour >= 22:
        return 4

train["Hour_Zone"] = train["Hour"].map(get_hour_zone)
```

Figure 5.1.3 Code of hour zone

```python
crime['Season']=(crime['Month']%12 + 3)//3
```

Figure 5.1.4 Coding of season Feature engineering

From the month column I derived the Season feature.

```python
# Weekday = 0, Weekend = 1
days = {'Monday':0 ,'Tuesday':0 ,'Wednesday':0 ,'Thursday':0 ,'Friday':0, 'Saturday':1 ,'Sunday':1}

crime['Weekend'] = crime['DayOfWeek'].replace(days).astype('int')
```

[34] ✓ 0.4s Python

Figure 5.1.5 Coding of Weekend Feature engineering

From the DayOfWeek Column I derived the Weekend column.

```python
import re


def find_streets(address):
    street_types = ['AV', 'ST', 'CT', 'PZ', 'LN', 'DR', 'PL', 'HY',
                    'FY', 'WY', 'TR', 'RD', 'BL', 'WAY', 'CR', 'AL', 'I-80',
                    'RW', 'WK','EL CAMINO DEL MAR']
    street_pattern = '|'.join(street_types)
    streets = re.findall(street_pattern, address)
    if len(streets) == 0:
        return 'OTHER'
    elif len(streets) == 1:
        return streets[0]
    else:
        return 'INT'

crime['StreetType'] = crime['Address'].map(find_streets)
```

✓ 2.2s Python

Figure 5.1.6 Coding of Street Type Feature engineering

For the Spatial Features I extracted the features from the address column. From the Address column I extracted the street type and Block Number Feature from the address.For the Street type I would take “800 Block of BRYANT ST” and categorise it to‘ST’.

```
#Augmentation
cos_30 = math.cos(math.radians(30))
sin_30 = math.sin(math.radians(30))
cos_45 = math.cos(math.radians(45))
sin_45 = math.sin(math.radians(45))
cos_60 = math.cos(math.radians(60))
sin_60 = math.sin(math.radians(60))


crime["Rot30_X"] = crime['X'] * cos_30 - crime['Y'] * sin_30
crime["Rot30_Y"] = crime['X'] * sin_30 + crime['Y'] * cos_30
crime["Rot45_X"] = crime['X'] * cos_45 - crime['Y'] * sin_45
crime["Rot45_Y"] = crime['X'] * sin_45 + crime['Y'] * cos_45
crime["Rot60_X"] = crime['X'] * cos_60 - crime['Y'] * sin_60
crime["Rot60_Y"] = crime['X'] * sin_60 + crime['Y'] * cos_60
crime["Radius"] = np.sqrt(crime['X'] ** 2 + crime['Y'] ** 2)
crime["Angle"] = np.arctan2(crime['X'], crime['Y'])
```

Figure 5.1.7 Coding of X and Y Feature engineering

From the X and Y column I extracted "Rot30_X", "Rot30_Y", "Rot45_X", “Rot45_Y", "Rot60_X", "Rot60_Y”, "Radius", and "Angle"."Rot30_X", "Rot30_Y", "Rot45_X", “Rot45_Y", "Rot60_X",and "Rot60_Y” features are three variants of rotated cartesian coordinates where they are calculated through the use of x = xCos + ySin and y = yCos – xSin .

I then performed Feature Encoding with the use of LabelEncoder for PdDistrict, DayOfWeek, StreetType. This is done so that XGBoost can be used as it is not able to work with these columns very well and this makes the training data more expressive and useful.

I then proceeded to split the dataset into x- for features and y- for output label. X and y are then split into 2 sets which is training set and testing sets which will use a random sampling ratio of 8: 2. The testing set will allow me to perform an unbiased evaluation of the model after the hyperparameter tuning is done. This would improve the confidence in the model’s ability to generalize.

Model Training (Performance Analysis)

For the model training I have chosen 2 models for this which are XGBoost and Random Forest.

Firstly, the XGBoost and Random Forest models performed decently on the given dataset. The training and testing accuracy for the 2 models were over 63%. The testing accuracy for the 2 models were also quite similar. The Decision Tree Classifier seemed to performed the worse as the accuracy could not go over 59 percent.

Model / Evaluation Set	XGBoost Classifier	Random Forest Classifier	Decision Tree Classifier
Training Set accuracy (Untuned)	64.04%	66.23%	56.54%
Test Set accuracy (Untuned)	63.22%	63.82%	56.58%
Training Set accuracy (Tuned)	80.95%	70.75%	59.37%
Test Set accuracy (Tuned)	66.73%	64.87%	59.32%

Table 5.1.2 Accuracy of Crime category Classification model

The log loss is for XGBoost and Random Forest models were as it is not over one. The Decision Tree Model however is over one signifying it is not able to classify the categories of crime very well.

Model / Evaluation Set	XGBoost Classifier	Random Forest Classifier	Decision Tree Classifier
Training Set Log Loss (Untuned)	0.89009	0.84295	1.09490
Test Set Log Loss (Untuned)	0.91010	0.89744	1.09349
Training Set Log Loss (Tuned)	0.56343	0.72254	1.01211
Test Set Log Loss (Tuned)	0.83913	0.86652	1.01345

Table 5.1.3 Log Loss of Crime category Classification model

Regarding for XGBoost and Random Forest the confusion matrix the category for "ASSAULT" and "VEHICLE THEFT" seem to have a lot of false negatives as they keep classifying themselves as the first column which "LARCERNY/THEFT". However in the case of "LARCENY/THEFT" and "DRUG/NARCOTIC" it was able to classify those categories quite well.

Model / Evaluation Set	XGBoost Classifier	Random Forest Classifier
Confusion Matrix Train Set	[[362240 8838 3877 7402] [45175 76563 3084 8806] [30699 3403 65171 1697] [16674 4516 1284 71777]]	[[351540 13609 8534 8674] [63926 52999 6056 10647] [50330 6282 41984 2374] [26351 8558 2659 56683]]
Confusion Matrix Test Set	[[82966 5717 4353 2553] [17094 11061 1974 3278] [12801 1962 9837 643] [6002 2155 627 14779]]	[[84355 5013 3742 2479] [18270 9976 1969 3192] [14792 2009 7846 596] [7216 2475 703 13169]]

Model / Evaluation Set	Decision Tree Classifier
Confusion Matrix Train Set	[[343541 6430 16304 16082] [95865 12259 9271 16233] [74211 4214 19605 2940] [38251 5804 3382 46814]]
Confusion Matrix Test Set	[[85976 1628 3973 4012] [24044 3021 2276 4066] [18669 1028 4842 704] [9620 1432 880 11631]]

Table 5.1.4 Confusion matrix of Crime category Classification model

The Decision Tree Classifier seems to perform the worst here as they kept classifying more than half of “ASSAULT” and “VEHICLE THEFT” Records as “LARCERNY/THEFT” and “DRUG/NARCOTIC” also misclassifies half of it’s categories.

For the XGBoost classifier the training and test set is decent as it managed to achieve a 67 percent for both macro average and the weighted average. However as seen below the models were not able to classify 1 “ASSAULT’ and “Vehicle Theft” very well as the recall and f1 scores are not very high for both the training and testing set.

	precision	recall	f1-score	support
0	0.80	0.95	0.87	382357
1	0.82	0.57	0.67	133628
2	0.89	0.65	0.75	100970
3	0.80	0.76	0.78	94251
accuracy			0.81	711206
macro avg	0.83	0.73	0.77	711206
weighted avg	0.81	0.81	0.80	711206

Figure 5.1.8 Classification report of XGBooost Train set

	precision	recall	f1-score	support
0	0.70	0.87	0.77	95589
1	0.53	0.33	0.41	33407
2	0.59	0.39	0.47	25243
3	0.70	0.63	0.66	23563
accuracy			0.67	177802
macro avg	0.63	0.55	0.58	177802
weighted avg	0.65	0.67	0.65	177802

Figure 5.1.9 Classification report of XGBoost test set

For the Random Forest Classifier the training and test set is worse than the XGBoost as it only managed a score of around 62 percent for both macro average.Much like XGboost as it was not able to classify 1 “ASSAULT’ and 2 “VEHICLE THEFT” very well as the recall and f1 scores of “ASSAULT” are very low at around 0.40 and 0.49 for the training set and 0.31 and 0.38 for the testing set. The recall and f1 scores of “VEHICLE THEFT” while being better than “ASSAULT” is also very low too at around 0.42 and 0.52 for the training set and 0.31 and 0.40 for the testing set.

	precision	recall	f1-score	support
0	0.71	0.92	0.80	382357
1	0.65	0.40	0.49	133628
2	0.71	0.42	0.52	100970
3	0.72	0.60	0.66	94251
accuracy			0.71	711206
macro avg	0.70	0.58	0.62	711206
weighted avg	0.70	0.71	0.69	711206

Figure 5.1.10 Classification report of Random Forest Train Set

	precision	recall	f1-score	support
0	0.68	0.88	0.77	95589
1	0.51	0.30	0.38	33407
2	0.55	0.31	0.40	25243
3	0.68	0.56	0.61	23563
accuracy			0.65	177802
macro avg	0.60	0.51	0.54	177802
weighted avg	0.63	0.65	0.62	177802

Figure 5.1.11 Classification report of Random Forest Test Set

For Decision Tree Classifier it performed way worse than the other 2 models as it managed a macro average of 42 for for both macro average and the weighted average. Much like the models above as it was not able to classify 1 “ASSAULT’ and 2 “VEHICLE THEFT” very well as the recall and f1 scores of “ASSAULT” are extremely low at around 0.09 and 0.15 for both the training set and testing set. The recall and f1 scores of “VEHICLE THEFT” while being better than “ASSAULT” is also very low too at around 0.19 and 0.26 for the training set and testing set. Out of the 3 models This model too is unable to classify “DRUG/NARCOTIC” well either as the F1 score managed is only 0.53.

	precision	recall	f1-score	support
0	0.62	0.90	0.74	382357
1	0.43	0.09	0.15	133628
2	0.40	0.19	0.26	100970
3	0.57	0.50	0.53	94251
accuracy			0.59	711206
macro avg	0.51	0.42	0.42	711206
weighted avg	0.55	0.59	0.53	711206

Figure 5.1.12 Classification report of Decision Tree Train Set

	precision	recall	f1-score	support
0	0.62	0.90	0.74	95589
1	0.42	0.09	0.15	33407
2	0.40	0.19	0.26	25243
3	0.57	0.49	0.53	23563
accuracy			0.59	177802
macro avg	0.51	0.42	0.42	177802
weighted avg	0.55	0.59	0.53	177802

Figure 5.1.13 Classification report of Decision Tree Test Set

Model Validation and Testing

To ensure that there was not over fitting 5-fold cross-validation was performed to with the scoring set to accuracy as seen below the models were clearly not overfitting.

```
Decision Tree k-scores:        [0.5952391  0.59483553 0.59409734 0.59473007 0.59304279]
XGB k-scores:                  [0.66363662 0.6640842  0.66210867 0.66267813 0.65964806]
Random Forest k-scores:        [0.64443695 0.64813943 0.64533433 0.64550305 0.64193868]
```

```
Mean k-scores for Decision Tree: 0.5944
Mean k-scores for XGBoost: 0.6624
Mean k-scores for Random Forest : 0.6451
```

Figure 5.1.14 Cross fold validation scores and mean

Model Selection and Feature Removal

After that we move on to model selection based on the results above, the XGBoost model is chosen as it had the highest accuracy lowest log lost and the best precision, recall and f1 score out of the 3 models. In order to prevent overfitting and to make it easier to implement the web application onto streamlit , feature removal is done through the use of the .feature_importance_ method this measures the mean decrease in the Gini information which is available to ensemble methods such as the random forest and XGBoost. Hence any features that had less than 1 percent importance was dropped.

```
importances = pd.DataFrame({'feature': X_train.columns,'importance': np.round(xgb.feature_importances_, 5)})
importances = importances.sort_values('importance',ascending=False).set_index('feature')
```

feature	importance
Rot45_Y	0.09802
Rot30_Y	0.08533
Rot60_Y	0.08335
Radius	0.07773
Angle	0.07434
Rot30_X	0.06987
Rot60_X	0.06257
PdDistrict	0.06135
Y	0.05952
Rot45_X	0.04542
Year	0.04319
BlockNo	0.04113
X	0.03986
Hour	0.03182
StreetType	0.02907
Day	0.02250
DayOfWeek	0.02231
Month	0.02185
Season	0.02132
Hour_Zone	0.00556
Weekend	0.00390

Figure 5.1.15 Coding for implementation of feature importance function and results

The model was then retrained with the same hyperparameters and are once again reevaluated. As seen below whilst the testing set accuracy slightly lower the confusion matrix for the category of 'ASSAULT', and 'VEHICLE THEFT' is slightly better. The Log Loss is also slightly better as it is lower.

Model / Evaluation Set	XGBoost Classifier after Feature Removal	XGBoost Classifier before Feature Removal
Training Set accuracy (Tuned)	81.03%	80.95%
Test Set accuracy (Tuned)	66.68%	66.73%
Training Set Log Loss (Tuned)	0.5614423166082795	0.563439527404093
Test Set Log Loss (Tuned)	0.8391786154815627	0.8391343221926233

Table 5.1.5 Comparison of before and after feature removal

Model / Evaluation Set	XGBoost Classifier after Feature Removal	XGBoost Classifier before Feature Removal
Confusion Matrix Train Set	[[362022 8926 4011 7398] [45160 76708 3014 8746] [30487 3288 65512 1683] [16580 4390 1208 72073]]	[[362240 8838 3877 7402] [45175 76563 3084 8806] [30699 3403 65171 1697] [16674 4516 1284 71777]]
Confusion Matrix Test Set	[[82980 5692 4347 2570] [17151 10994 1953 3309] [12878 1944 9773 648] [6032 2151 577 14803]]	[[82966 5717 4353 2553] [17094 11061 1974 3278] [12801 1962 9837 643] [6002 2155 627 14779]]

Table 5.1.4 Confusion matrix of before and after feature removal

Pickling

```
import pickle
data = {"model": xgb}
with open('xgboostforcrimeclassification.pkl', 'wb') as file:
    pickle.dump(data, file)

import joblib

joblib.dump(xy_scaler, 'xy_scaler.save')
```

Figure 5.1.16 Coding for Pickling best model

The model and scaler are then saved and put into Dropbox so that it can be deployed in the Streamlit.

Web Application development for Crime Classification Model

```
pic = pd.read_pickle('https://www.dropbox.com/s/5sqt4o3x329j6m3/xgboostforcrimeclassification.pkl?dl=1')

xgb = pic["model"]
xy_scaler = joblib.load("xy_scaler.save")
```

Figure 5.1.17 Coding for importing best model to Streamlit from dropbox

The code snippet above shows how the Model and standard scaler is loaded into the web application to allow for predictions based on the parameters inputted by the user.

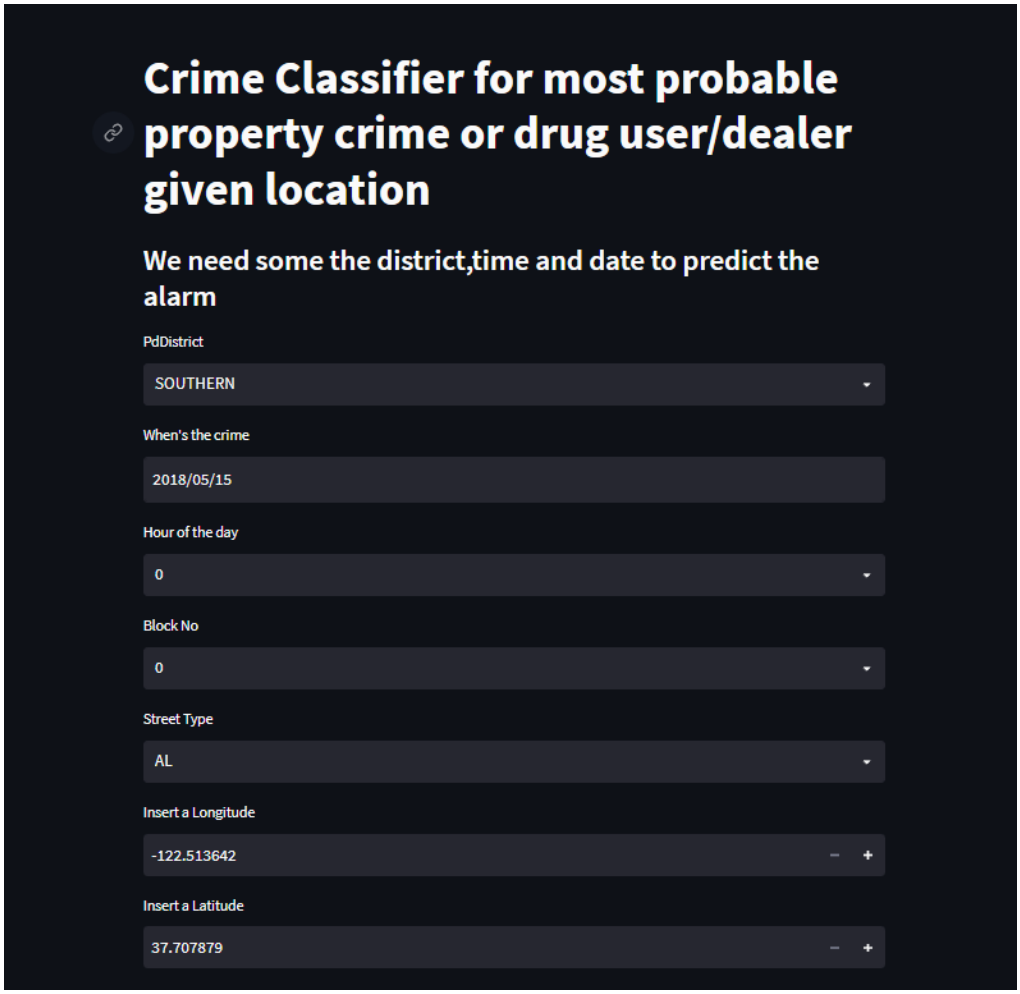


Figure 5.1.18 GUI of crime category classification model

```
def show_classifier_page():
    st.title("Crime Classifier for most probable property crime or drug user/dealer given location")

    st.write("""### We need some the district,time and date to predict the alarm""")

    PdDistrict = ('SOUTHERN', 'MISSION', 'NORTHERN', 'CENTRAL', 'BAYVIEW',
                  'INGLESIDE', 'TENDERLOIN',  'TARAVAL', 'PARK', 'RICHMOND' )
    Pdoptions = list(range(len(PdDistrict)))
    Pdvalue = st.selectbox("PdDistrict", Pdoptions, format_func=lambda x: PdDistrict[x])

    crimed= st.date_input("When's the crime",datetime.date(2018, 5, 15),max_value=datetime.date(2019, 12, 31),min_value=datetime.date(2004, 1, 1))

    Hour = ( 0 ,1 ,2 ,3 ,4 ,5 ,6 ,7 ,8 ,9 ,10 ,11 , 12 ,13 ,14 ,15 ,16 ,17 ,18 ,19 ,20 ,21 ,22 ,23 )
    Hours = st.selectbox("Hour of the day", Hour)

    Blockno = (0 ,100 ,200 ,300 ,400 ,500 ,600 ,700 ,800 ,900 ,1000 ,1100 ,1200 ,1300 ,1400 ,1500 ,1600 ,1700 ,1800 ,1900 ,2000 ,2100 ,2200 ,2300 ,2400 ,
               2500 ,2600 ,2700 ,2800 ,2900 ,3000 ,3100 ,3200 ,3300 ,3400 ,3500 ,3600 ,3700 ,3800 ,3900 ,4000 ,4100 ,4200 ,4300 ,4400 ,4500 ,4600 ,4700 ,
               4800 ,4900 ,5000 ,5100 ,5200 ,5300 ,5400 ,5500 ,5600 ,5700 ,5800 ,5900 ,6000 ,6100 ,6200 ,6300 ,6400 ,6500 ,6600 ,6700 ,6800 ,6900 ,7000 ,
               7100 ,7200 ,7300 ,7400 ,7500 ,7600 ,7700 ,7800 ,7900 ,8000 ,8100 ,8200 ,8300 ,8400)
    blockoptions = list(range(len(Blockno)))
    blockvalue =st.selectbox("Block No", blockoptions, format_func=lambda x: Blockno[x])

    Street=('AL', 'AV', 'BL', 'CR', 'CT', 'DR', 'EL CAMINO DEL MAR', 'HY', 'I-80',
            'INT', 'LN', 'OTHER', 'PL', 'PZ', 'RD', 'ST', 'TR', 'WAY', 'WK', 'WY')
    streetoptions = list(range(len(Street)))
    Streetvalue =st.selectbox("Street Type", streetoptions, format_func=lambda x: Street[x])

    Longitude = st.number_input('Insert a Longitude',max_value=-122.364937,min_value=-122.513642,step=0.001,format="%.6f")
    Latitude = st.number_input('Insert a Latitude',max_value=37.8206208380702,min_value=37.7078790224135,step=0.001,format="%.6f")
```

Figure 5.1.19 coding implementation parameter input of crime category classification model

The code snippet above shows the coding of users can input parameters.

```
Longitude = st.number_input('Insert a Longitude',max_value=-122.364937,min_value=-122.513642,step=0.001,format="%.6f")
Latitude = st.number_input('Insert a Latitude',max_value=37.8206208380702,min_value=37.7078790224135,step=0.001,format="%.6f")


df = pd.DataFrame(np.array([[Latitude,Longitude]]),
columns=['lat', 'lon'])
st.map(df)
```

Figure 5.1.20 coding implementation parameter input for longitude and latitude to show map

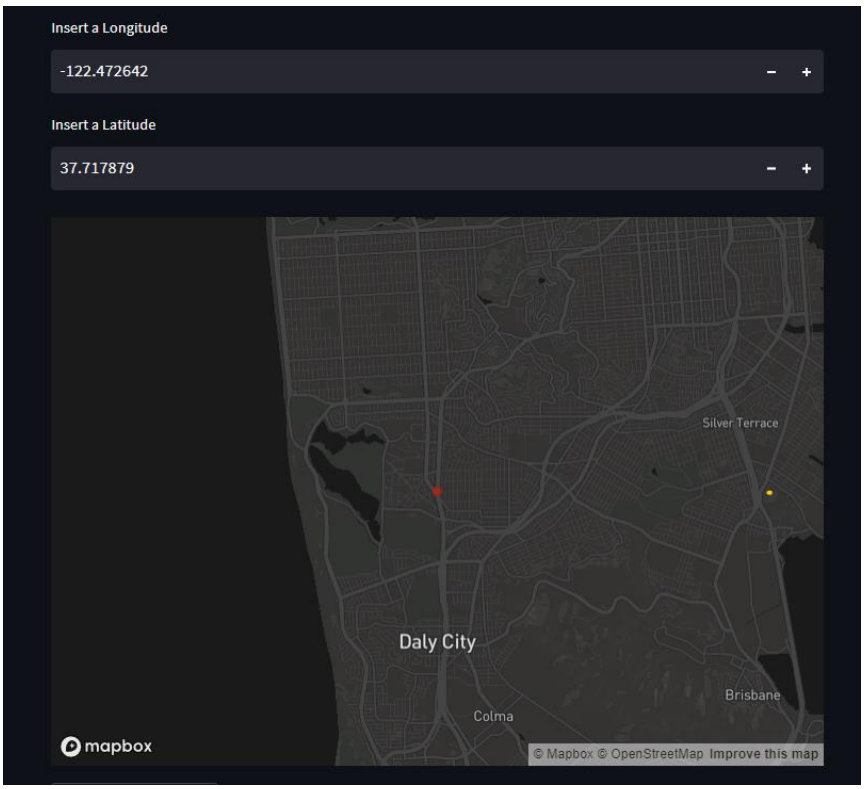


Figure 5.1.21 showing map from entering longitude and latitude

The code snippet above shows the coding of users can input parameters of longitude and latitude and how it will display a map based on the longitude and latitude that the user inputs.

```
dayofweek=crimed.weekday()
scrimed = crimed.strftime('%Y-%m-%d')
dt = parse(scrimed)
year=dt.year
month=dt.month
day=dt.day
season = (dt.month%12 + 3)//3

coor = np.array([[Longitude,Latitude]])
coor[:, [0, 1]]=xy_scaler.transform(coor[:, [0, 1]])
xcor=coor[:, [0]]
ycor=coor[:,[1]]
xcoordinate=xcor[0,0]
ycoordinate=ycor[0,0]
cos_30 = math.cos(math.radians(30))
sin_30 = math.sin(math.radians(30))
cos_45 = math.cos(math.radians(45))
sin_45 = math.sin(math.radians(45))
cos_60 = math.cos(math.radians(60))
sin_60 = math.sin(math.radians(60))

Rot30_X = xcoordinate * cos_30 - ycoordinate * sin_30
Rot30_Y = xcoordinate * sin_30 + ycoordinate * cos_30
Rot45_X = xcoordinate * cos_45 - ycoordinate * sin_45
Rot45_Y = xcoordinate * sin_45 + ycoordinate * cos_45
Rot60_X = xcoordinate * cos_60 - ycoordinate * sin_60
Rot60_Y = xcoordinate * sin_60 + ycoordinate * cos_60
Radius = np.sqrt(xcoordinate ** 2 + ycoordinate ** 2)
Angle = np.arctan2(xcoordinate, ycoordinate)
```

Figure 5.1.22 Preprocessing of parameters that that have been input

After the user finishes inputting the parameters and presses the predict button. Preprocessing is done to match the one used to train the model as seen below.

```
#       X       Y       Month   Day     Year    Hour    StreetType      BlockNo Rot30_X Rot30_Y Rot45_X Rot45_Y Rot60_X Rot60_Y Radius  Angle
X = np.array([[dayofweek,Pdvalue,xcoordinate,ycoordinate,month,day,year,Hours,season,Streetvalue,blockvalue,Rot30_X,Rot30_Y,Rot45_X,Rot45_Y,Rot60_X,Rot60_Y,Radius,Angle]])
y_pred=xgb.predict(X)
pred=y_pred[0]
if pred == 0:
    st.write("The most likely crime here is LARCENY/THEFT ")
elif pred == 1:
    st.write("The most likely crime here is ASSAULT")
elif pred == 2:
    st.write("The most likely crime here is VEHICLE THEFT")
elif pred == 3:
    st.write("The most likely crime here is DRUG/NARCOTIC")
```

Figure 5.1.23 Implementation of parameters to pass through model for prediction

The values are then placed into a numpy array and is then passed to the classifier to receive the prediction whereby the classifier would show one of the 4 categories it was trained on.

Figure 5.1.24 Output after pressing predict Crime Type

5.2 System Implementation for Model 2 Crime Hotspot Classification

Dataset Used

The Crime dataset includes a total of 2,129,525 instances. The Dataset used is the same as the one above.

Pre-processing

As per the above model the outliers of X and y were once again removed. The top 10 crimes and dropped the others in an effort to allow the application to run faster and to ensure that only crimes were chosen as some categories in the dataset were not really crimes such as “OTHER OFFENSES” and “NON-CRIMINAL”. Feature engineering is then performed where temporal features are extracted from date and time as the model above.

Variables	Feature Engineering
Date	Month, Year, Day
Time	Hour
Month	Season
Hour	Hour_Zone

Table 5.2.1 Feature engineering of dataset for temporal variables

```
# Hour Zone 0 - Past midnight, 1 - morning, 2 - afternoon, 3 - After work hours
def get_hour_zone(hour):
    if hour >= 0 and hour < 6:
        return 0
    elif hour >= 6 and hour < 12:
        return 1
    elif hour >= 12 and hour < 18:
        return 2
    elif hour >= 18 and hour < 24:
        return 3

crime["Hour_Zone"] = crime["Hour"].map(get_hour_zone)
```

Figure 5.2.1 Code of hour zone

```
crime['Season']=(crime['Month']%12 + 3)//3
```

Figure 5.2.2 Implementation of season feature engineering

Data aggregation using the above variables which are “Year”, “Month”, “Day”, “DayofWeek”, “Hour_Zone”, “PdDistrict”, and “Season” where the data is aggregated on category whereby a count of all crime types that happen in that time period is counted

```
crimeG = crime.groupby(['Year','Month','Hour_Zone','Day','PdDistrict','DayOfWeek','Season'], as_index=False).agg({"Category":"count"})
crimeG = crimeG.sort_values(by=['PdDistrict'], ascending=False)
```

Figure 5.2.3 Implementation of Data Aggregation

The upper and lower bound of the Category count is then used to derive the categories to predict which are “High Crime Rate” and “Low Crime Rate”. “Low Crime Rate” is classified as when the crime counts were below 4 during the time. “Medium crime rate is classified” as when there were between 4 and 10 counts of crime. High Crime rate was classified when there were more than 10 counts of crime at the given time.

```
lower = np.mean(crimeG['Category'])-0.75*np.std(crimeG['Category'])
higher = np.mean(crimeG['Category'])+0.75*np.std(crimeG['Category'])
print(lower, higher)
```

```
2.6476564321180023 9.148411110946366
```

Figure 5.2.4 Upper and lower bound of category count

```python
def crime_rate_assign(x):
    if(x<=3):
        return 0
    elif(x>3 and x<=10):
        return 1
    else:
        return 2

crimeG['Alarm'] = crimeG['Category'].apply(crime_rate_assign)
```

Figure 5.2.5 Implementation of Alarm Variable

The dataset is then split into x- for features and y- for output label. X and y are then split into 2 sets which is training set and testing sets which will use a random sampling ratio of 8: 2. The testing set will allow me to perform an unbiased evaluation of the model after the hyperparameter tuning is done. This would improve the confidence in the model's ability to generalize.

Model Training (Performance Analysis)

For the model training I have chosen 3 models which are XGBoost, Random Forest and Decision Tree.

Firstly, the XGBoost , Random Forest ,and Decision Tree models performed decently on the given dataset. The training and testing accuracy for all the 3 models were over 60%. The testing accuracy for the 3 models were also quite similar. The one that performed best here is the XGBoost Classifier but only by 0.05% when compared with the Random Forest Classifier.

Model / **Evaluation Set**	**XGBoost Classifier**	**Random Forest Classifier**	**Decision Tree Classifier**
Training Set accuracy (Untuned)	68.72%	67.07%	62.01%
Test Set accuracy (Untuned)	66.57%	65.81%	61.88%
Training Set accuracy (Tuned)	68.92%	70.57%	64.25%
Test Set accuracy (Tuned)	66.66%	66.61%	64.00%

Table 5.2.2 Accuracy of Crime rate Classification model

The Log Loss were all decent as well as none of them went above 1.The one that perfomed the best is the best in this category is the XGBoost Classifier.

Model / Evaluation Set	XGBoost Classifier	Random Forest Classifier	Decision Tree Classifier
Training Set Log Loss (Untuned)	0.66441	0.70053	0.77210
Test Set Log Loss (Untuned)	0.70231	0.71823	0.77446
Training Set Log Loss (Tuned)	0.66267	0.62987	0.76686
Test Set Log Loss (Tuned)	0.70201	0.70484	0.77342

Table 5.2.3 Log Loss of Crime rate Classification model

Regarding the Confusion matrix all 3 models were not able to classify low and high crime rate well as they seem to classify it to medium crime rate quite frequently with Decision tree being the worst of them all. Though the XGBoost managed the best with it having the least amount of false positives in the high and low crime rate category.

Model / Evaluation Set	XGBoost Classifier	Random Forest Classifier	Decision Tree
Test Set Confusion Matrix (Untuned)	8987 5874 25 4054 16549 1149 47 3007 2664	8032 6833 21 3340 17522 890 36 3359 2323	7921 6948 17 4577 16608 567 125 3908 1685
Test Set Confusion Matrix (Tuned)	9024 5839 23 4058 16502 1192 48 2958 2712	8758 6105 23 3817 16899 1036 40 3119 2559	7762 7020 104 3631 16875 1246 77 3410 2231

Table 5.2.4 Confusion Matrix of Crime rate Classification model

Model / Evaluation Set	XGBoost Classifier	Random Forest Classifier	Decision Tree
Test Set UAR (Untuned)	0.6101	0.5837	0.5301
Test Set UAR (Tuned)	0.6130	0.6042	0.5730

Table 5.2.5 UAR of Crime rate Classification model

Classification report of test set for all 3 models

For XGBoost classifier the test set is decent as they all achieved a precision of over 65 percent for all categories. However, the recall for the High crime rate category seems to perform the worst at 0.48 even though the f1 score remains decent.

```
----------Classification Report-----------------------------------
              precision    recall  f1-score   support

           0       0.69      0.61      0.64     14886
           1       0.65      0.76      0.70     21752
           2       0.69      0.47      0.56      5718

    accuracy                           0.67     42356
   macro avg       0.68      0.61      0.64     42356
weighted avg       0.67      0.67      0.66     42356
```

Figure 5.2.6 Classification report of XGBoost

For Random Forest while the precision of low and high crime rate category is slightly higher compared to XGBoost the recall is worst in recall for High and low Crime Rate.

```
----------Classification Report-----------------------------------
              precision    recall  f1-score   support

           0       0.69      0.59      0.64     14886
           1       0.65      0.78      0.71     21752
           2       0.71      0.45      0.55      5718

    accuracy                           0.67     42356
   macro avg       0.68      0.60      0.63     42356
weighted avg       0.67      0.67      0.66     42356
```

Figure 5.2.7 Classification report of Random Forest

For the Decision tree which performed the worst out of the 3 categories as there has been a decrease in all precision recall and f1 score. This shows that it is unable to classify the categories as well as the other models.

```
----------Classification Report-----------------------------------
              precision    recall  f1-score   support

           0       0.68      0.53      0.59     14886
           1       0.62      0.78      0.69     21752
           2       0.65      0.42      0.51      5718

    accuracy                           0.64     42356
   macro avg       0.65      0.57      0.60     42356
weighted avg       0.65      0.64      0.63     42356
```

Figure 5.2.8 Classification report of Decision tree

Pickling

Much like before I pickled the best model in this case XGBoost as it had the highest UAR, lowest Log Loss and the least false positives in the confusion matrix.

Web Application development for Crime Classification Model

```
pic = pd.read_pickle('https://www.dropbox.com/s/642r89udaoedcz7/alarmmodelxgboost.pkl?dl=1')

xgb = pic["model"]
```

Figure 5.2.9 Importing the Pickled Model

The code snippet above shows how the Model is loaded into the web application to allow for predictions based on the parameters inputted by the user.

```
def show_alarm_page():
    st.title("Crime Alarm For each police District")

    st.write("""### We need some the district,time and date to predict the alarm""")

    PdDistrict = ('SOUTHERN',
                  'MISSION',
                  'NORTHERN',
                  'CENTRAL',
                  'BAYVIEW',
                  'INGLESIDE',
                  'TENDERLOIN',
                  'TARAVAL',
                  'PARK',
                  'RICHMOND'
    )
    Hour_zone = ( "12:00AM-5:59AM","6:00AM-11:59AM","12:00PM-5:59PM","6:00PM-11:59PM"
            )
    Pdoptions = list(range(len(PdDistrict)))
    Pdvalue = st.selectbox("PdDistrict", Pdoptions, format_func=lambda x: PdDistrict[x])

    crimed= st.date_input("When's the crime",datetime.date(2019, 7, 6))

    houroptions = list(range(len(Hour_zone)))
    hourvalue =st.selectbox("Hour_zone", houroptions, format_func=lambda x: Hour_zone[x])
```

Figure 5.2.10 Implementation of parameters for webapp for Crime Rate Classification model

The code snippet below shows the coding of the parameters and how to setup the parameter options for the users.

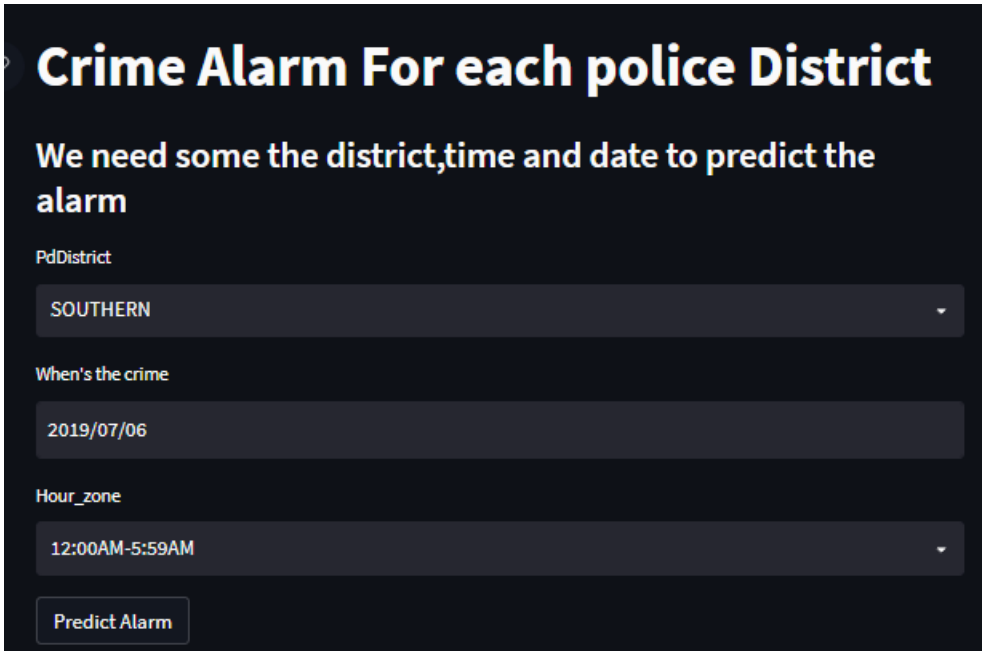


Figure 5.2.11 GUI of Webapp for crime rate Classification model

```
ok = st.button("Predict Alarm")
if ok:
    dayofweek=crimed.weekday()
    scrimed = crimed.strftime('%Y-%m-%d')
    dt = parse(scrimed)
    year=dt.year
    month=dt.month
    day=dt.day
    season = (dt.month%12 + 3)//3
    X = np.array([[year,month,day,dayofweek,hourvalue,Pdvalue,season]])
    alarm=xgb.predict(X)
    pred=alarm[0]
    if pred == 0:
        st.write("Low Crime at this precinct and time (Predicted less than 3 occurences of crimes)")
    elif pred == 1:
        st.write("Medium Crime Rate at this precinct and time(Predicted less than 10 occurences of crimes but more than 3 occurences of crime )")
    elif pred == 2:
        st.write("High Crime at this precinct and time(Predicted more than 10 occurences of crimes)")
```

Figure 5.2.12 Pre-processing of Input parameters

After the user finishes inputting the parameters and presses the predict button. Preprocessing is done to match the one used to train the model as seen below.

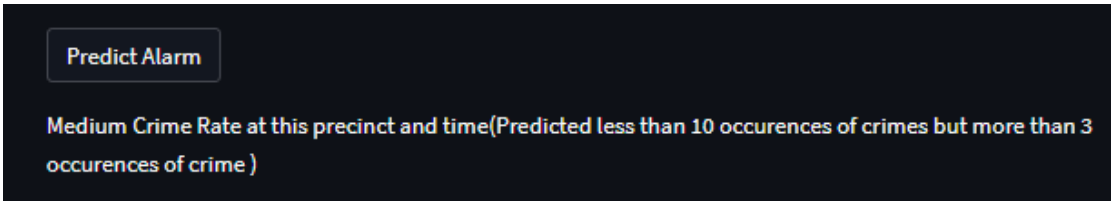


Figure 5.2.13 Output after pressing the Predict button

5.3 Time Series Forecasting Model SARIMAX

Dataset Used

The Crime dataset includes a total of 2,129,525 instances. The Dataset used is the same as the one above.

Preprocessing

```
def into_timeseries(df):

    # transform crime dataframe ii timeseries
    df = df.groupby('Date').agg(count=('PdDistrict','count')).reset_index()
    df = df.set_index('Date')
    df.index = pd.to_datetime(df.index)
    return df
```

```
crime_ts = into_timeseries(crime)
```

Figure 5.3.1 Transforming dataset to time series

The figure above shows how to change the dataset into a time series whereby a count would be made based on the police district which is the first step needed to make a time series dataset. Next the resample('MS'.sum() function is used to group up the dates into months with a count.

```
[ ] monthly = crime_ts.resample('MS').sum()
```

```
[ ] monthly
```

	count
Date	
2003-01-01	5475
2003-02-01	5203
2003-03-01	5851
2003-04-01	5831
2003-05-01	5470
...	...
2018-01-01	5691
2018-02-01	4576
2018-03-01	4995
2018-04-01	4687
2018-05-01	1533

185 rows × 1 columns

Figure 5.3.2 Implementation of frequency conversion after transforming to timeseries.

After that the last month of the dataset is removed as it is lower than all the other months as seen above. Next the Dickey-Fuller test is done to get the ADF Statistics, , the P-value and the critical value. The auto correlation test is also performed to find the seasonal component.

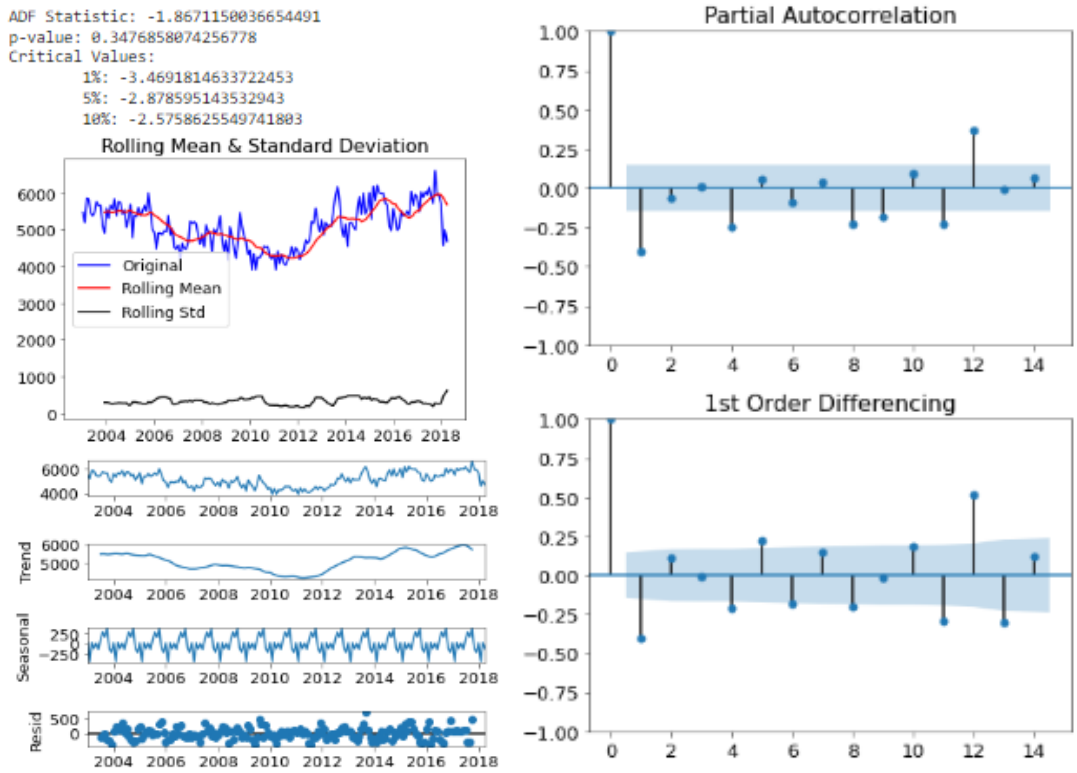


Figure 5.3.3 Results of Dickey-Fuller Test and Auto correlation test.

Model Training and tuning of parameters

For the Model training stage, I use the SARIMAX model. For the seasonality I chose 12 as that was the lag that was observed from the auto correlation test. This model is then trained and tested.

```
model = SARIMAX(train['count'],order=(1,1,1),seasonal_order=(1, 0, 0, 12)) # enforce_invertibility=False
results = model.fit()
results.summary()
start = len(train)
end = len(train) + len(test) - 1
# typ= 'levels' to return the differenced values to the original units
preds = results.predict(start=start,end=end,typ='levels').rename('SARIMAX (1,1,1)(1,0,0,12) Predictions')
```

```
[140] test['count'].plot(figsize=(12,6),legend=True,alpha=0.5,label='Testing Data')
      preds.plot(legend=True,alpha=1)
      plt.grid(True)
      plt.xlabel('Date')
      plt.title('San Francisco Crime Rate');
```

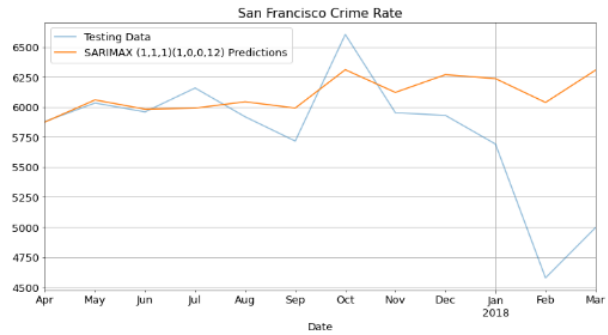


Figure 5.3.4 Plot of actual vs Model Prediction of untuned SARIMAX

After that to ensure that I chose the best hyper parameters I use the auto_arima function which helps decide the best pdq PDQ values. This chooses the one with the largest log likelihood and lowest AIC. Below shows the new plot with the tuned Hyper parameters.

```
[143] model = SARIMAX(train['count'],order=(1,1,1),seasonal_order=(2, 0, 0, 12)) # enforce_invertibility=False
      results = model.fit()
      results.summary()
      start = len(train)
      end = len(train) + len(test) - 1
      # typ= 'levels' to return the differenced values to the original units
      preds = results.predict(start=start,end=end,typ='levels').rename('SARIMAX (0,1,1)(2,0,0,12) Predictions')
```

```
test['count'].plot(figsize=(12,6),legend=True,alpha=0.5,label='Testing Data')
preds.plot(legend=True,alpha=1)
plt.grid(True)
plt.xlabel('Date')
plt.title('San Francisco Crime Rate');
```

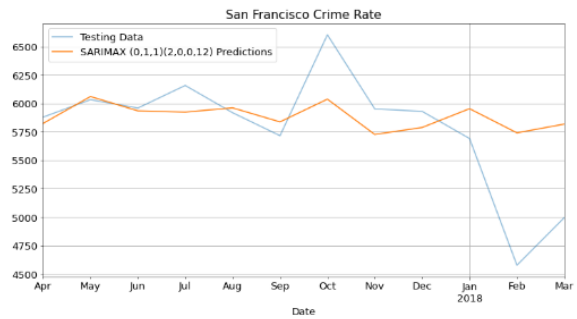


Figure 5.3.5 Plot of actual vs Model Prediction of Tuned SARIMAX

Model / Evaluation Set	SARIMAX(order=(1,1,1),seasonal_order=(1, 0, 0, 12) (Untuned)	SARIMAX(order=(1,1,1),seasonal_order=(2, 0, 0, 12))(Tuned)
Root-mean-square deviation	612.766	462.868

Table 5.3.6 Root Mean Square Error of Both Models

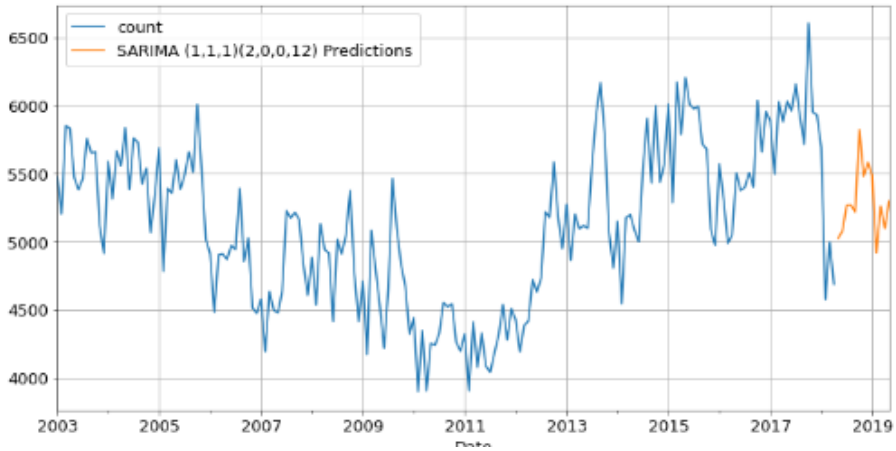


Figure 5.3.7 Plot forecast of Tuned SARIMAX

The best model is then used to predict crime for out of the dataset. Then it is then used for each precinct and different crime types.

Creating the map and predicting

Firstly, to create a map to predict crime statistics for each month I first downloaded a geojson file from https://data.sfgov.org/Public-Safety/Current-Police-Districts/wkhw-cjsf to get the geometry of each police district. After that I used a function that would choose all rows that has the police district and put it into a dataframe. This then groups up all the unique police district and dates which will then be made into a frequency count like above. The model is then trained and used to predict the next year of monthly statistics for the police districts. The predictions will be stored in a

dataframe with the police district it is predicting. The output is then concatenated to change the 10 data frames to 1.

```
def District_arima(district):

    District_data = crime.loc[(crime.PdDistrict == district)]
    district_ts = into_timeseries(District_data)
    district_ts = district_ts.resample('MS').sum()
    district_ts = district_ts[:-1]

    start = len(district_ts)
    end = len(district_ts) + 12

    model = SARIMAX(district_ts['count'],order=(1,1,1),seasonal_order=(2, 0, 0, 12)) # enforce_invertibility=False
    results = model.fit()
    forecasted_values = results.predict(start=start,end=end,typ='levels').rename('prediction')
    forecasted_values=forecasted_values.to_frame()
    output = pd.DataFrame(forecasted_values)
    output['PdDistrict']=district


    return output
```

```
district_arimas = [District_arima(x) for x in list(crime.PdDistrict.unique())]
```

```
# concatenate output dataframes
output = pd.concat(district_arimas)
```

Figure 5.3.8 Implementation of prediction for each Police District

This concatenated output is then used in the chloropleth_mapbox function to form a map with the statistics with a monthly slider.

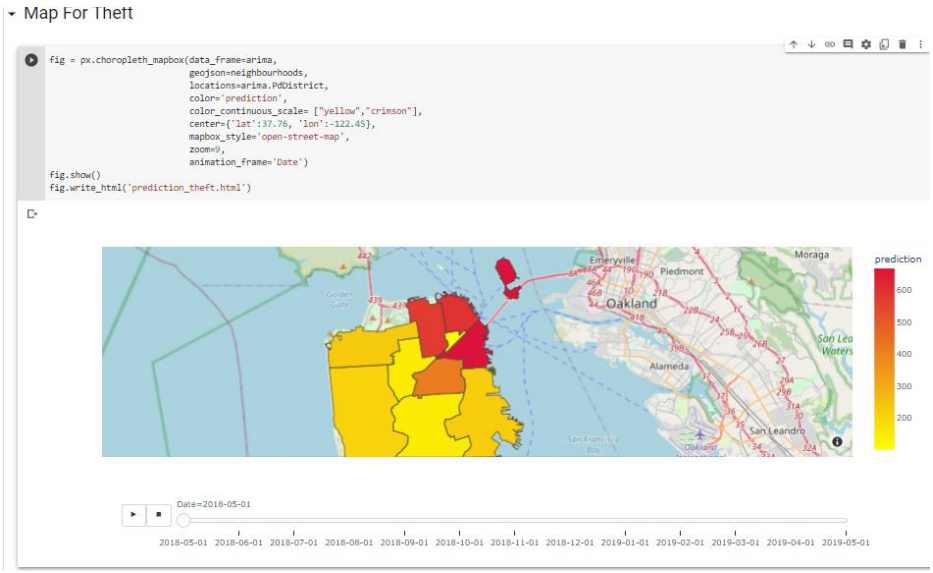


Figure 5.3.9 Implementation of prediction for each Police District onto a map

This is then repeated 3 more times for 3 other categories of crimes which are “Theft”, “Assault” and “Vehicle Theft”. These outputs are then saved as a html which will then be used in the webapp.

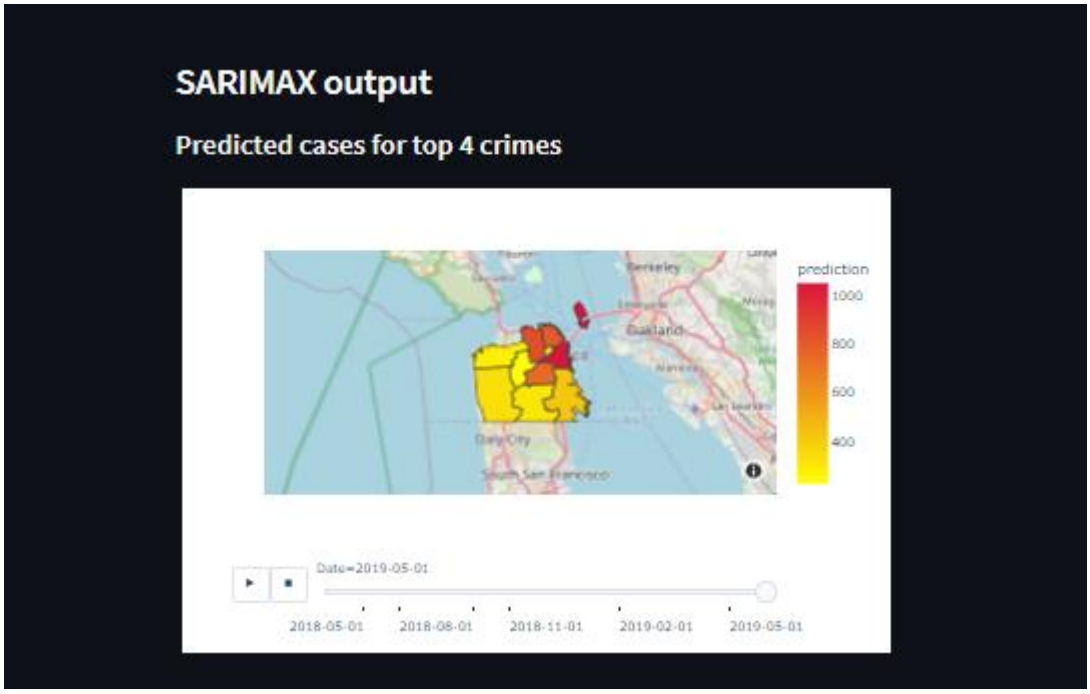


Figure 5.3.10 Map of predicted in the web app

5.4 Exploratory Dataset Analysis

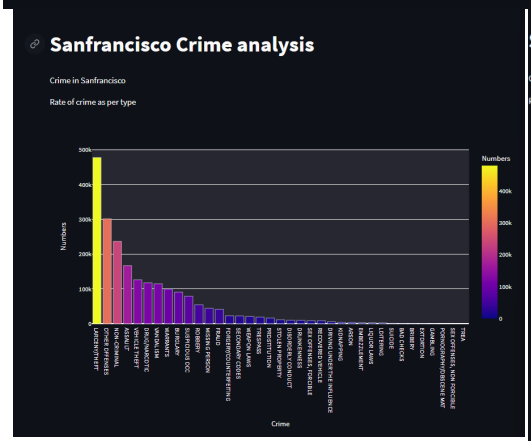
```
def show_explore_page():
    st.title("Sanfrancisco Crime analysis ")
    st.markdown('Crime in Sanfrancisco')
    st.sidebar.title('Sanfrancisco Crime analysis')
    st.sidebar.subheader('Crime in Sanfrancisco')
    @st.cache(persist=True)
    def load_data():
      data = pd.read_csv('https://www.dropbox.com/s/ug5mkq9ija3mmpn/train.csv?dl=1')
      data['Dates']=pd.to_datetime(data['Date'] + ' ' + data['Time'])
      return data

    data = load_data()
    data =data[['Dates','Category','Descript','DayOfWeek','PdDistrict','Resolution','Address','X','Y']]
    data.columns = ['Dates','Category','Descript','DayOfWeek','PdDistrict','Resolution','Address','longitude','latitude']
```

Figure 5.4.1 Loading the dataset and renaming

For the implementation of the Dataset Analysis page the dataset is first loaded from the dropbox into the application. The Date and Time Column is then concatenated to datetime to be used in the application. The columns for X and Y are also renamed to be used for the map later as in order to use the Streamlit map function. This is as the .map function for Streamlit requires the columns to be named longitude and latitude.

```
st.sidebar.markdown('Type of crimes and their stats')
select = st.sidebar.selectbox('visualization type',['Histogram','Pie chart'])
crime_count = data['Category'].value_counts()
crime_count = pd.DataFrame({'Crime':crime_count.index, 'Numbers': crime_count.values})
st.markdown("Rate of crime as per type")
if select == "Histogram":
    fig = px.bar(crime_count,x="Crime",y="Numbers",color="Numbers",height =600,width=900)
    st.plotly_chart(fig)
else:
    fig = px.pie(crime_count,values="Numbers",names="Crime")
    st.plotly_chart(fig)
```



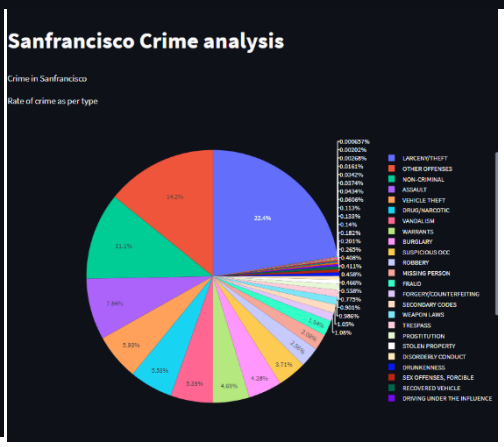
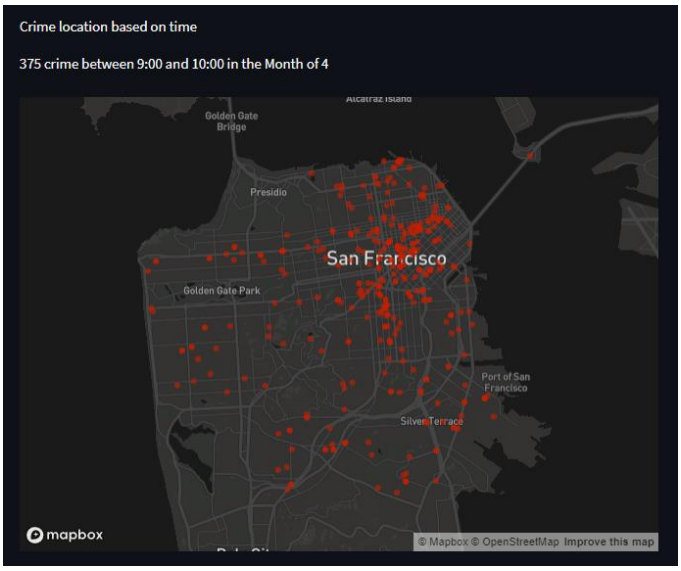
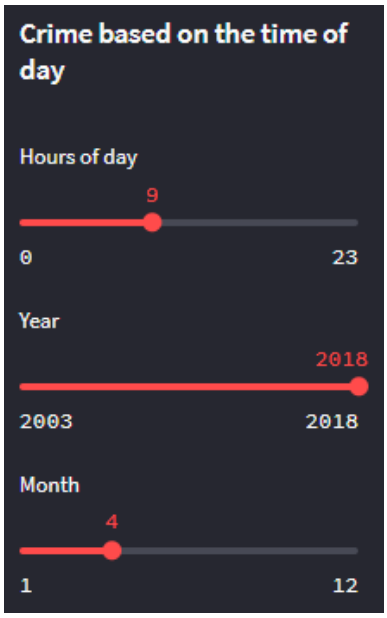


Figure 5.4.2,5.4.3,5.4.4 Implementation of the category function for pie chart and histogram

The figures above show the implementation of the visualizations for the categories. The category column is changed to value counts for it to be visualized as either a Histogram or a pie chart.

```
st.sidebar.subheader("Crime based on the time of day")
hour = st.sidebar.slider("Hours of day",0,23)
year = st.sidebar.slider("Year",2003,2018)
month = st.sidebar.slider("Month",1,12)
mod_dat = data[data['Dates'].dt.hour == hour]
mod_dat = mod_dat[data['Dates'].dt.year == year]
mod_dat = mod_dat[data['Dates'].dt.month == month]
if not st.sidebar.checkbox("Don't show map",False):
    st.markdown("Crime location based on time")
    st.markdown("%i crime between %i:00 and %i:00 in the Month of %i" % (len(mod_dat),hour,hour+1,month))
    st.map(mod_dat)
```





Figures 5.4.5,5.4.6 and 5.4.7 Visualization of map based on hour,year month.

The figures above show the implementation of the visualizations for the map based on the time inputted. The concatenated date and time column is then used to match the slider input by the user which are hour,year and month. This then outputs a map with all the points onto the map as seen in figure 5.4.6.

5.4 Implementation issue and challenges

During the implementation phase, the problems that the decision tree model may face during implementation is that any small change in the data could cause a large change in the structure of the decision tree which may cause large changes in the tree structure. This could be a problem as the end goal of the project is to allow more records to be added to aid in future classifications of crime rates. The time and space complexity of the decision tree is relatively higher compared to other models so this will take up more memory and more time for the mathematical calculation hence increasing the training time [4].

The last issue of implementation was when training XGBoost Classifier the training would take an obscene amount of time unless trained on GPU.As such when performing hyper parameter tuning it would take up to 7 hours just to complete 300 iterations in random search.

CHAPTER 6

SYSTEM EVALUATION AND DISCUSSION

6.1 System Testing

6.1.1 System Testing for Model 1 Crime Category Classification

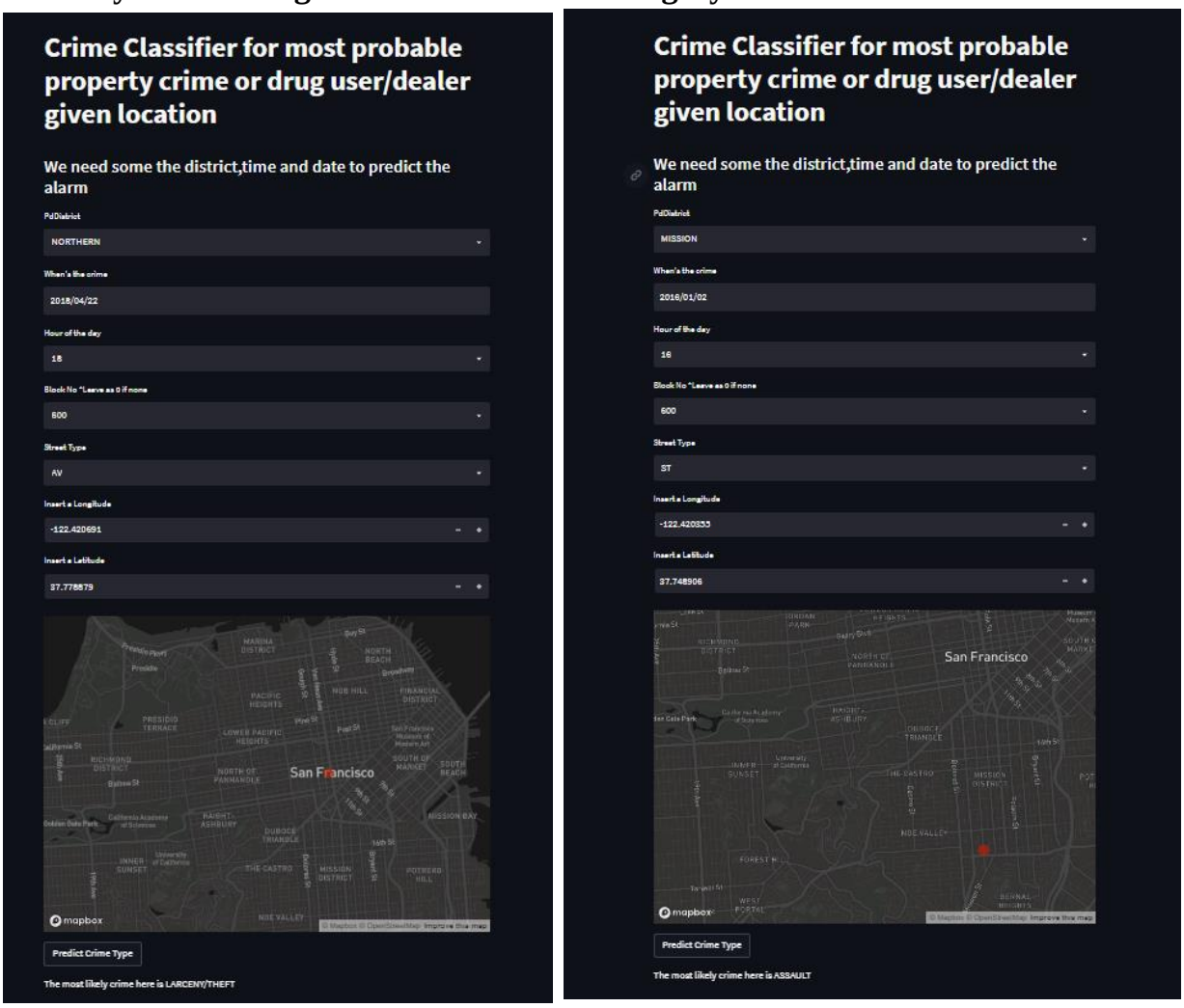


Figure 6.1.1.1 GUI of Crime Category classification model

User can key in parameters to make their prediction in this case is LARCERNY/THEFT or ASSAULT, VEHICLE THEFT or DRUGS/NARCOTICS

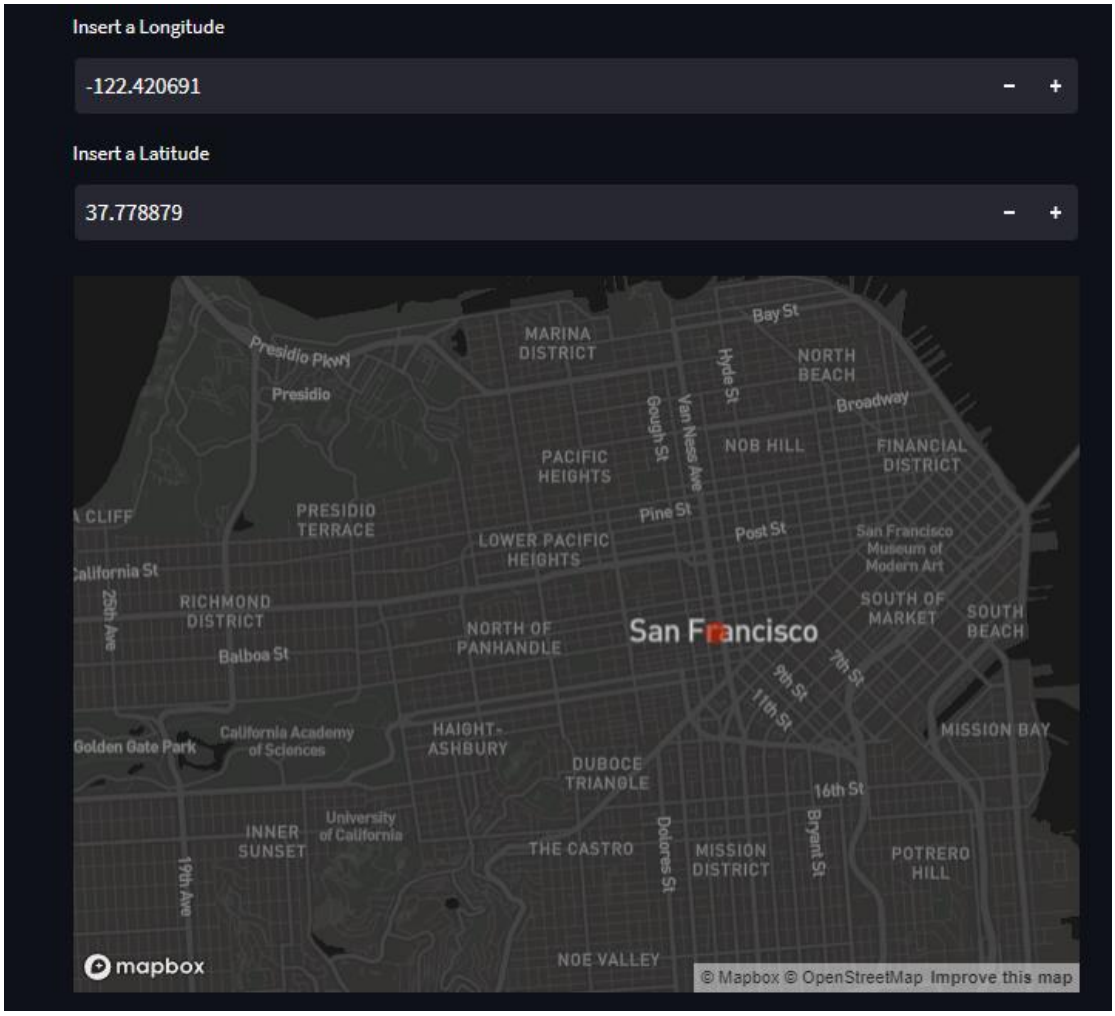
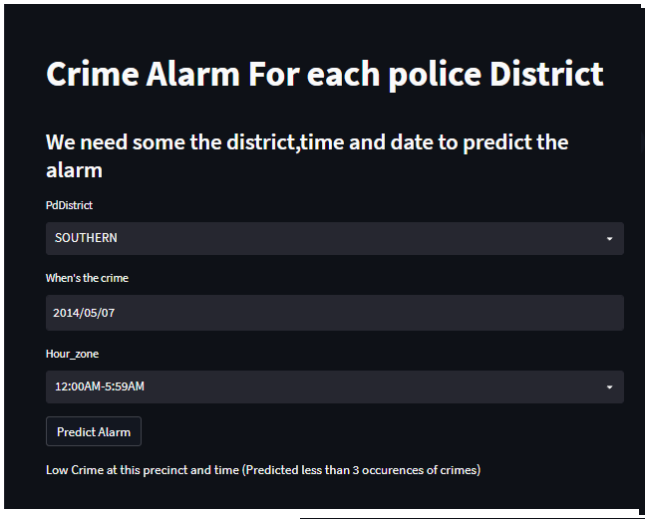
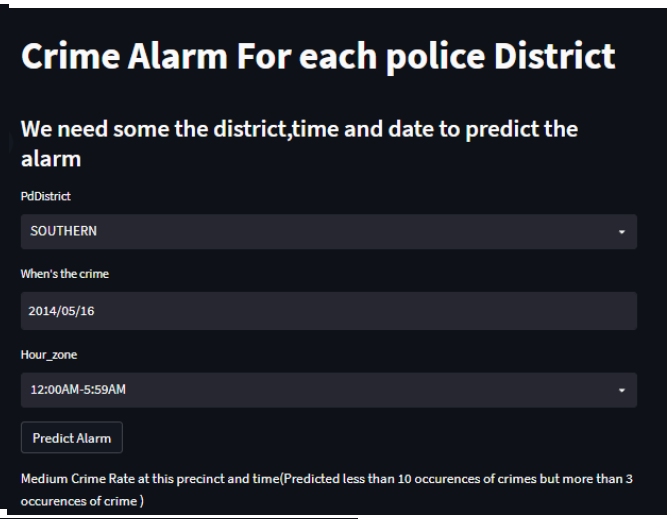


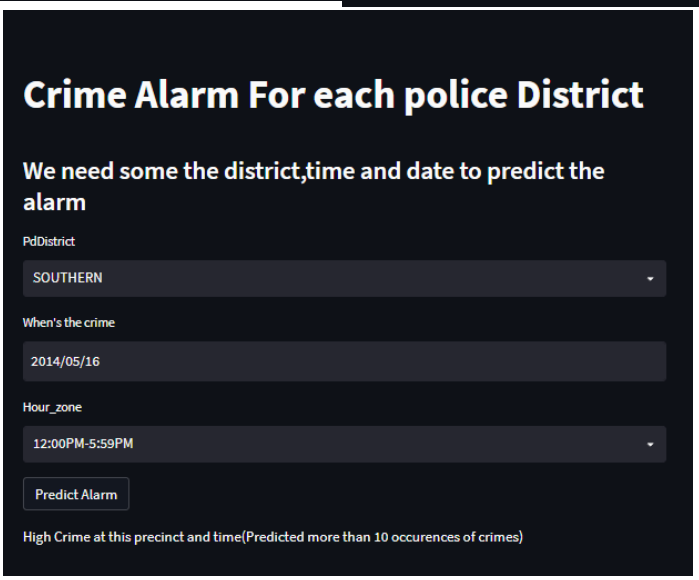
Figure 6.1.1.2 Map with point of inserted Longitude and Latitude

The user can enter the longitude and latitude and see if the points is correct on the map.

6.1.2 System Testing for Model 2 Crime Rate Classification







Figures 6.1.2.1, 6.1.2.2, and 6.1.2.3 Map with point of inserted Longitude and Latitude

The figures above show the user inputting the parameters for the crime rate prediction. The 3 figures above show how the system is able to predict 3 different types of crime rates.

6.1.3 System Testing for SARIMAX page

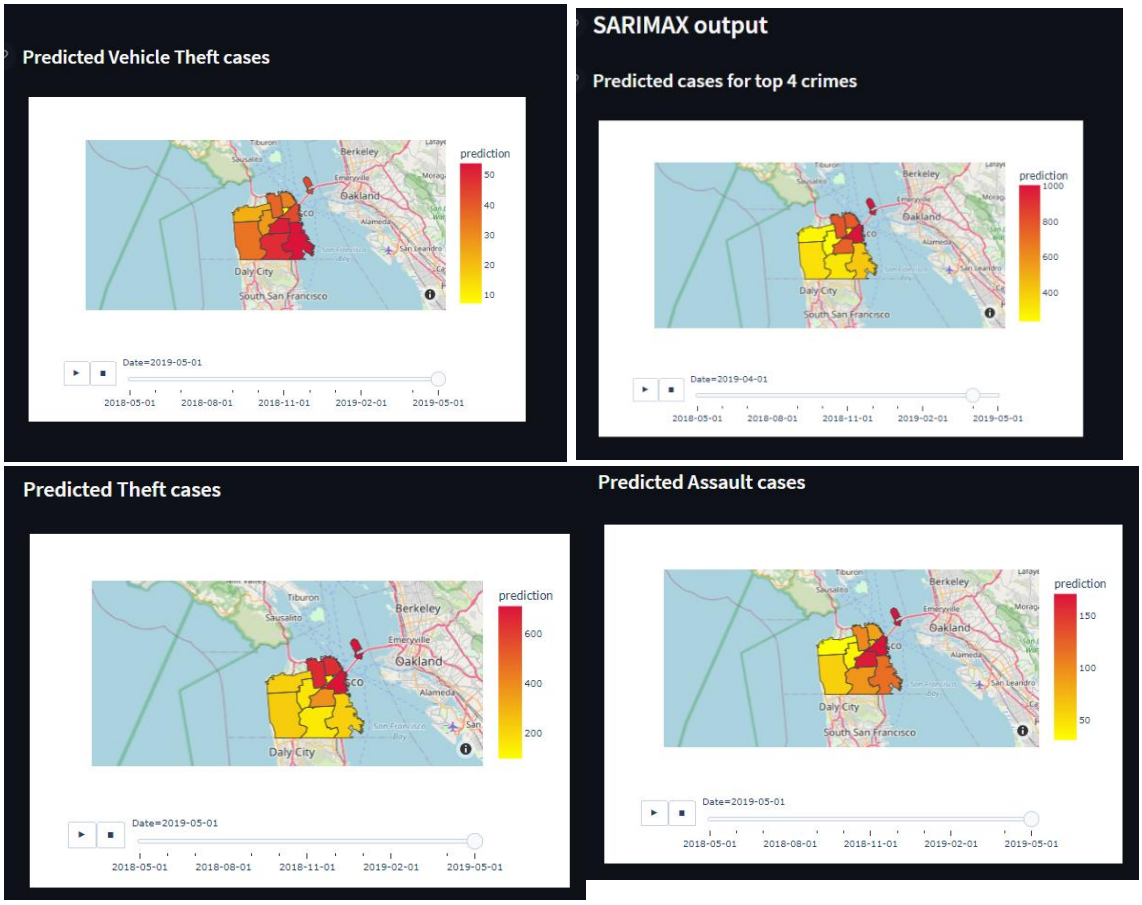


Figure 6.1.3.1, 6.1.3.2, 6.1.3.3, and 6.1.3.4 Map with of crime density for 4 different crimes

The above figures show the predicted crime statistics for each Police District. The user is able to choose from 4 maps each for different crime types as seen from the titles.

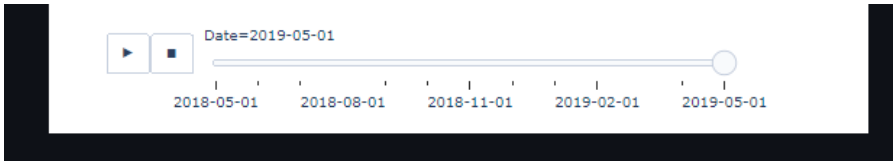


Figure 6.1.3.5 Slider of map with dates

The user can drag through to the months to see the predicted statistics for each month for each precinct

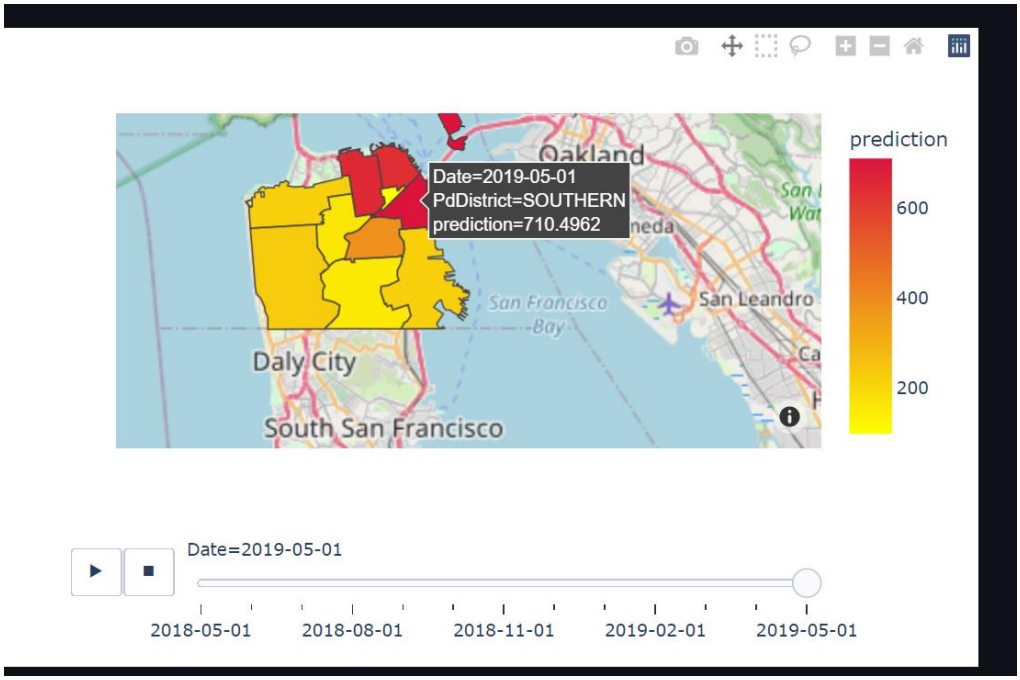


Figure 6.1.3.5 Map with police district with crime forecastt

The user can hover over the precinct and get the predicted crime rate of that month

6.1.4 System Testing for Exploratory dataset analysis page

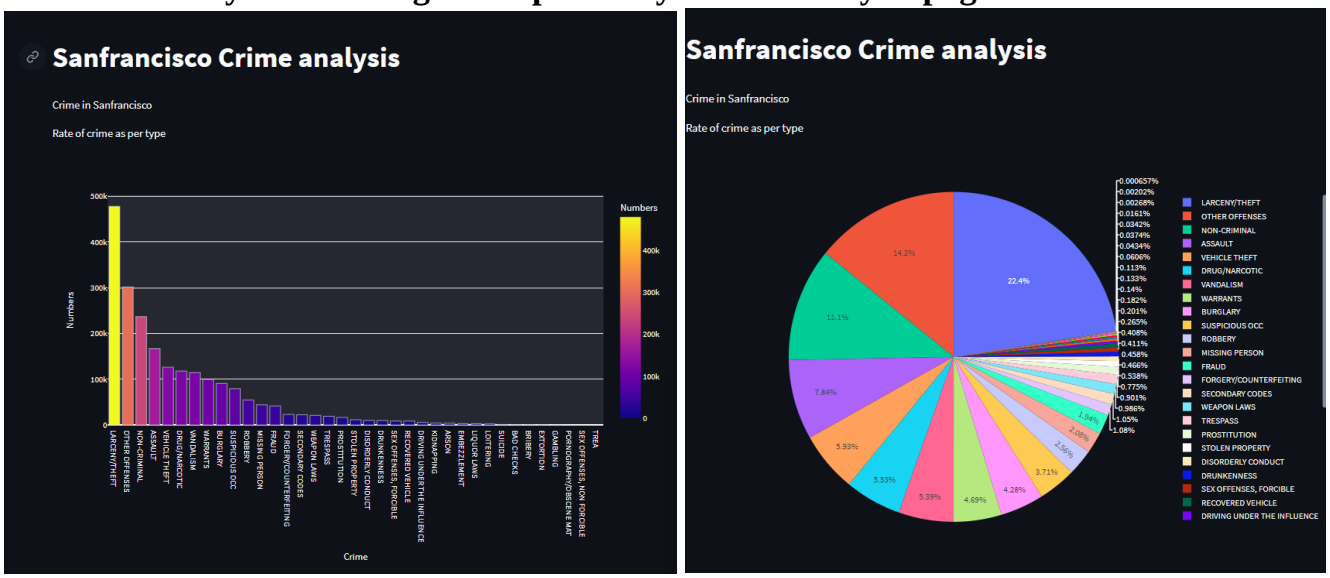


Figure 6.1.4.1 Piechart and Histogram of category for crime analysis

On this page, the user can choose between 2 visualization types for the category visualization in the dataset. This function allows the users to understand the most commited type of crime.

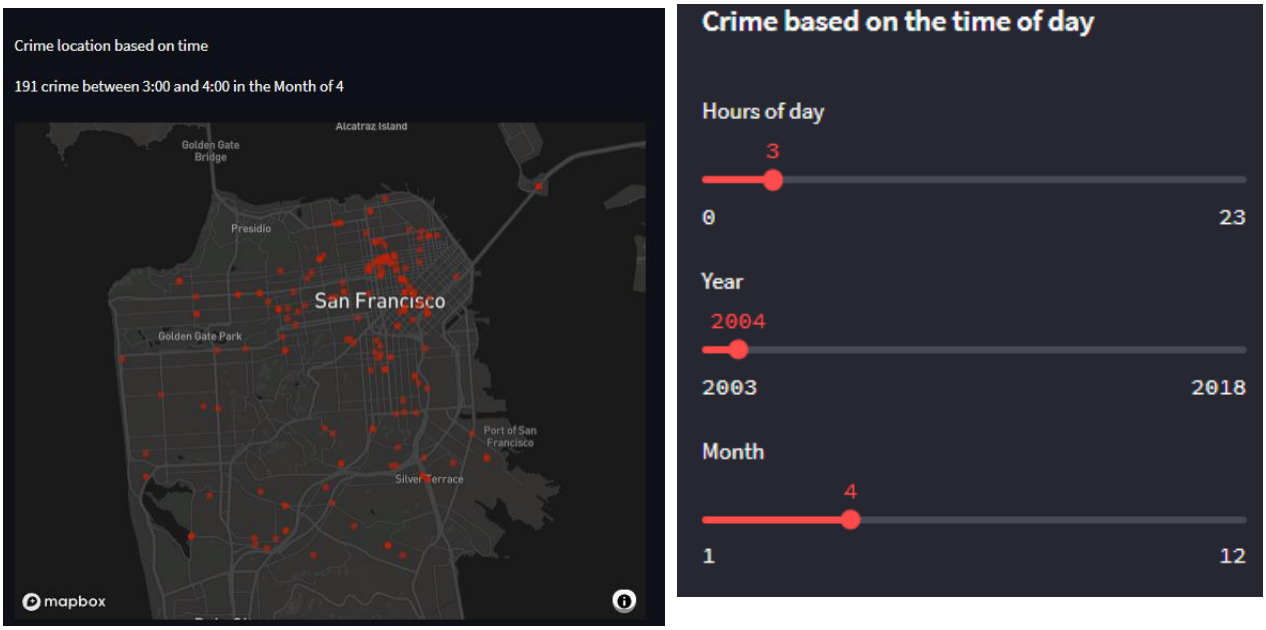


Figure 6.1.4.2,6.1.4.3 Map of points of crime given hour,year,month and slider for inputing parameters

The user can also enter the year, month and the hour of the day to explore the crime density based on the hour, month and year.

6.2 Objectives Evaluation

To produce a system that is able predict areas that will have higher crime rates.

This objective has been achieved through models 2 and 3. This system is able to allow users to predict the police districts that will have higher crime rates. Model 2 allows users to check the quarter of the day given the date, hour of day and precinct to predict the crime rate category. Model 3 however shows the monthly predicted crime statistic for the next year.

To explore and enhance classification algorithms to predict future crime category based on previous crime trends.

This objective has been achieved through the use of models 1 and 2 as they both tested more than 2 models each to achieving a testing accuracy of over 65 percent for both for multiclass classification.

Create a web-based system to allow for easy access to the application

This objective has achieved as the models has been deployed to Streamlit and can be accessed by anyone with the link.

Chapter 7

Conclusion

7.1 Project review

Thousands of crimes are committed daily even right now while you are reading this passage. Somewhere, someone is getting robbed, stabbed, etc. It is without a doubt that crime is seen as a plague to society everywhere. Hence the need to reduce crime rates has become a priority globally. The main objectives I hope to achieve with the completion of this project is to aid in increasing the effectiveness of predictive policing.

The aim of this project was to develop a web-based application for the users to find out crime density in different parts of the city. The project was also able to explore and enhance classification algorithms to predict future crime category based on previous crime trends. It also managed to predict future crime rates in different parts of the city.

7.2 Novelties and Contributions

This project proposed a crime category and crime rate classification model that was trained on 3 different models while a crime rate forecast model known as SARIMAX was used to predict crime rate of each police district onto a map. The novelty of this project is that it can predict crime rate and crime category. Besides that, the contribution of this project is its ease of use as it has been deployed onto a website and can be used to predict crime easily.

7.3 Future Work

While the objectives of the project have been met, there are still several aspects of the project that can be further refined. Firstly, the model 1 can be used to predict more categories of crime as the current model is only being used on 4 categories. For model 2 the police district could be made smaller by turning the map into a grid and using that instead of police district.

REFERENCES

[1] A. Ahmad, S. Ali, and N. Ahmad, "Crime and Economic Growth in Developing Countries: Evidence from Pakistan" researchgate, 2014. [Online]. Available: https://www.researchgate.net/profile/Sharafat-Ali/publication/275019421_Crime_and_Economic_Growth_in_Developing_Countries_Evidence_from_Pakistan/links/552e67070cf2acd38cb93de5/Crime-and-Economic-Growth-in-Developing-Countries-Evidence-from-Pakistan.pdf. [Accessed: 12-Apr-2022].

[2] A. Sharma, "Decision Tree vs. Random Forest - which algorithm should you use?," *Analytics Vidhya*, 12-May-2020. [Online]. Available: https://www.analyticsvidhya.com/blog/2020/05/decision-tree-vs-random-forest-algorithm/. [Accessed: 12-Apr-2022].

[3] B. Pearsall, "Predictive policing: The future of law enforcement?," National Institute of Justice, 2010. [Online]. Available: https://nij.ojp.gov/topics/articles/predictive-policing-future-law-enforcement. [Accessed: 15-Apr-2022].

[4] BotBark, "Top 6 advantages and disadvantages of Decision Tree Algorithm," *Bot Bark*, 08-Nov-2020. [Online]. Available: https://botbark.com/2019/12/19/top-6-advantages-and-disadvantages-of-decision-tree-algorithm/. [Accessed: 12-Apr-2022].

[5] Department of Statistics Malaysia, "Crime Statistics, Malaysia, 2020 ," *Department of Statistics Malaysia Official Portal*, 2020. [Online]. Available: https://www.dosm.gov.my/v1/index.php?r=column%2FcthemeByCat&cat=455&bul_id=UFZxV+npONEJqUU5pckJIbzlXeEJ1UT09&menu_id=U3VPMldoYUxzVzFaYmNkWXZteGduZz0+9. [Accessed: 12-Apr-2022].

[6] G. Dembla, "Intuition behind log-loss score," *Medium*, 03-Dec-2021. [Online]. Available: https://towardsdatascience.com/intuition-behind-log-loss-score-4e0c9979680a#:~:text=Log%2Dloss%20is%20indicative%20of,is%20the%20log%2Dloss%20val. [Accessed: 13-Apr-2022].

[7] H. Kusuma, H. F. Hariyani, and W. Hidayat, "The relationship between crime and economics growth in Indonesia: KNE Social Sciences," *KNE Publishing*, 2019. [Online]. Available: https://knepublishing.com/index.php/KnE-Social/article/view/4271/8772. [Accessed: 12-Apr-2022].

[8] J. Hoare, “What is a random forest?,” *displayr*, 09-Jun-2021. [Online]. Available: https://www.displayr.com/what-is-a-random-forest/#:~:text=Disadvantages%20of%20random%20forests,than%20a%20single%20decision%20tree. [Accessed: 13-Apr-2022].

[9] L. G. A. Alvesa, H. V. Ribeirob, and F. A. Rodriguesa, “Crime prediction through urban metrics and statistical learning,” *arxiv*, 2018. [Online]. Available: https://arxiv.org/pdf/1712.03834.pdf. [Accessed: 12-Apr-2022].

[10] N. H. M. Shamsuddin, N. A. Ali, and R. Alwee, “(PDF) an overview on crime prediction methods - researchgate,” *researchgate*, 2017. [Online]. Available: https://www.researchgate.net/publication/320650913_An_overview_on_crime_prediction_methods. [Accessed: 12-Apr-2022].

[11] N. Kumar, “Advantages of XGBoost algorithm in machine learning,” *blogspot*. [Online]. Available: http://theprofessionalspoint.blogspot.com/2019/03/advantages-of-xgboost-algorithm-in.html. [Accessed: 13-Apr-2022].

[12] T. J. Bernard, “Crime,” *Encyclopædia Britannica*, 2020. [Online]. Available: https://www.britannica.com/topic/crime-law. [Accessed: 13-Apr-2022].

[13] UC Business Analytics R Programming Guide, “Gradient Boosting Machines,” *UC Business Analytics R Programming Guide*. [Online]. Available: http://uc-r.github.io/gbm_regression#idea. [Accessed: 12-Apr-2022].

[14] UniversalClass, “The impact of crime on Community Development,” *universalclass*. [Online]. Available: https://www.universalclass.com/articles/business/the-impact-of-crime-on-community-development.htm?fbclid=IwAR3DSed6PoRDp42snZNCGP8KYHydPT2tAYC42LTduQvvub1y0csu_7nC-DA. [Accessed: 12-Apr-2022].

[15] V. Ingilevich and S. Ivanov, “Crime rate prediction in the urban environment using social factors,” *researchgate*, 2018. [Online]. Available: https://www.researchgate.net/profile/Varvara-Ingilevich/publication/327901578_Crime_rate_prediction_in_the_urban_environment_using_social_factors/links/5dd848d8458515dc2f4589ce/Crime-rate-prediction-in-the-urban-environment-using-social-factors.pdf. [Accessed: 12-Apr-2022].

[16] V. Kurama, “Gradient boosting for classification,” *Paperspace Blog*, 2021. [Online]. Available: https://blog.paperspace.com/gradient-boosting-for-

classification/#:~:text=Let%20us%20look%20at%20some,be%20time%20and%20memory%20exhaustive. [Accessed: 12-Apr-2022].

[17] Y. Lamari, B. Freskura, A. Abdessamad, and S. Eichberg, “Predicting Spatial Crime Occurrences through an Efficient Ensemble-Learning Model,” *researchgate*, 2020. [Online]. Available: https://www.researchgate.net/publication/345159839_Predicting_Spatial_Crime_Occurrences_through_an_Efficient_Ensemble-Learning_Model. [Accessed: 12-Apr-2022].

Appendix

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y4S1	Study week no.: 2
Student Name & ID: Chee Man Hang 18ACB03448	
Supervisor: Chang Jing Jing	
Project Title: Crime Rate Prediction Using Machine Learning	

1. WORK DONE [Please write the details of the work done in the last fortnight.] - Reevaluating project's scope and objectives - Rewriting problem statement and background information
2. WORK TO BE DONE - Choosing a more recent dataset to develop this system on
3. PROBLEMS ENCOUNTERED none
4. SELF EVALUATION OF THE PROGRESS Decent

Supervisor's signature

Student's signature

(Project II)

Trimester, Year: Y4S1	Study week no.: 4
Student Name & ID: Chee Man Hang 18ACB03448	
Supervisor: Chang Jing Jing	
Project Title: Crime Rate Prediction Using Machine Learning	

1. WORK DONE [Please write the details of the work done in the last fortnight.] - Found a more recent dataset to develop this system on - Redrawing the system design diagrams and choosing models to train on
2. WORK TO BE DONE -Creating a new model
3. PROBLEMS ENCOUNTERED None
4. SELF EVALUATION OF THE PROGRESS Progress is decent

Supervisor's signature

Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y4S1	Study week no.: 6
Student Name & ID: Chee Man Hang 18ACB03448	
Supervisor: Chang Jing Jing	
Project Title: Crime Rate Prediction Using Machine Learning	

1. WORK DONE [Please write the details of the work done in the last fortnight.] -Created a new model for crime rate prediction classification -Added Decision tree model to the mix
2. WORK TO BE DONE -Hyper parameter tuning and finding what else to use to evaluate models -Use SARIMAX for extra crime rate prediciton
3. PROBLEMS ENCOUNTERED None
4. SELF EVALUATION OF THE PROGRESS Slow

Supervisor's signature

Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y4S1	Study week no.: 8
Student Name & ID: Chee Man Hang 18ACB03448	
Supervisor: Chang Jing Jing	
Project Title: Crime Rate Prediction Using Machine Learning	

1. WORK DONE [Please write the details of the work done in the last fortnight.] - Finished with model 2 - Made a model with sarimax also managed to use it for different police districts .
2. WORK TO BE DONE - Finish reports
3. PROBLEMS ENCOUNTERED None
4. SELF EVALUATION OF THE PROGRESS Slow

Supervisor's signature

Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y4S1	Study week no.: 10
Student Name & ID: Chee Man Hang 18ACB03448	
Supervisor: Chang Jing Jing	
Project Title: Crime Rate Prediction Using Machine Learning	

1. WORK DONE [Please write the details of the work done in the last fortnight.] -Rewrite chapter 4 and 5
2. WORK TO BE DONE - Finish report
3. PROBLEMS ENCOUNTERED None
4. SELF EVALUATION OF THE PROGRESS Slow

Supervisor's signature

Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

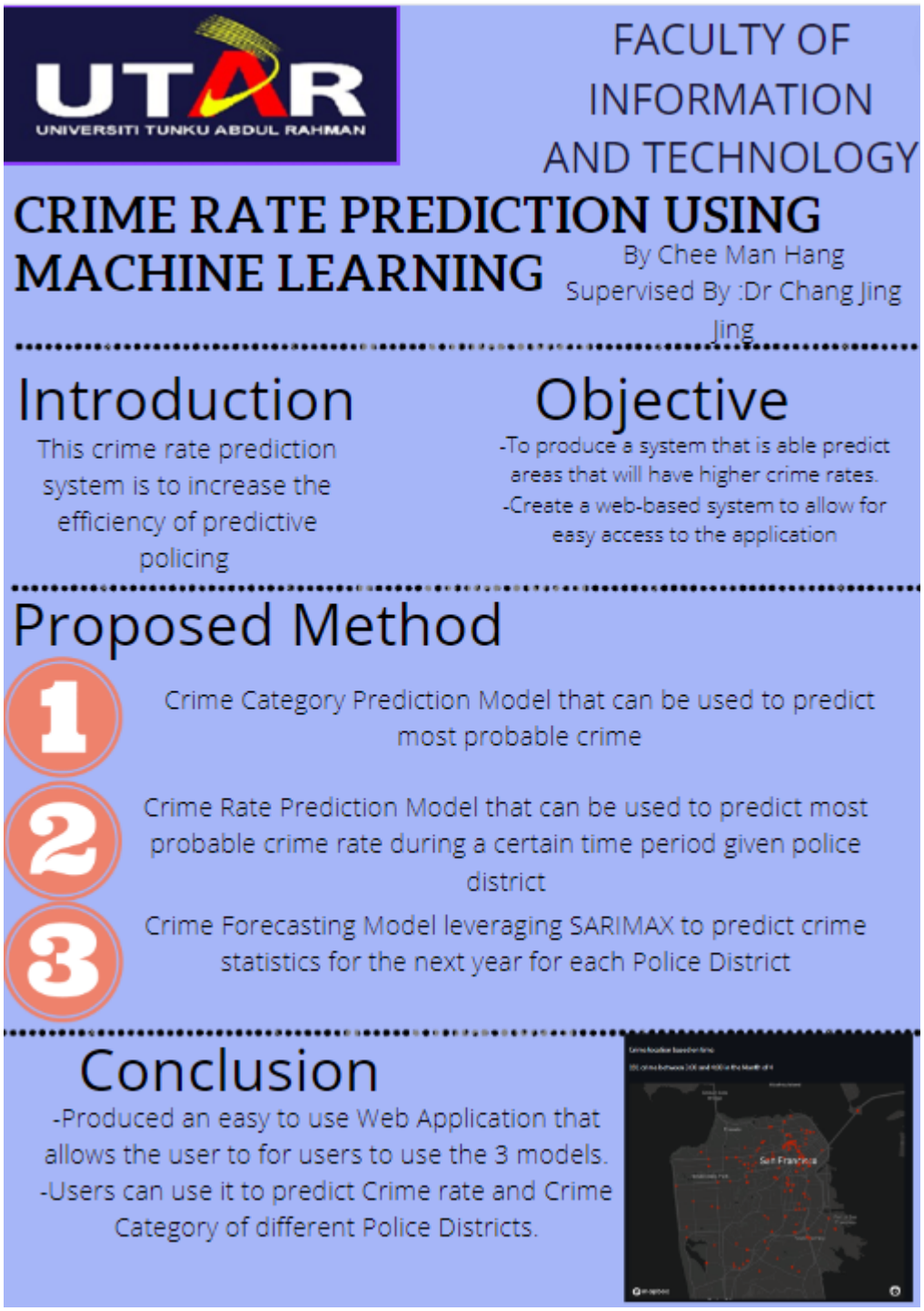
Trimester, Year: Y4S1	Study week no.: 12
Student Name & ID: Chee Man Hang 18ACB03448	
Supervisor: Chang Jing Jing	
Project Title: Crime Rate Prediction Using Machine Learning	

1. WORK DONE [Please write the details of the work done in the last fortnight.] -writing chapter 6 and 7
2. WORK TO BE DONE
3. PROBLEMS ENCOUNTERED None
4. SELF EVALUATION OF THE PROGRESS Slow

Supervisor's signature

Student's signature

POSTER


UTAR
UNIVERSITI TUNKU ABDUL RAHMAN
FACULTY OF INFORMATION AND TECHNOLOGY
CRIME RATE PREDICTION USING MACHINE LEARNING
By Chee Man Hang
Supervised By :Dr Chang Jing Jing
Introduction
This crime rate prediction system is to increase the efficiency of predictive policing
Objective
-To produce a system that is able predict areas that will have higher crime rates.
-Create a web-based system to allow for easy access to the application
Proposed Method
1
Crime Category Prediction Model that can be used to predict most probable crime
2
Crime Rate Prediction Model that can be used to predict most probable crime rate during a certain time period given police district
3
Crime Forecasting Model leveraging SARIMAX to predict crime statistics for the next year for each Police District
Conclusion
-Produced an easy to use Web Application that allows the user to for users to use the 3 models.
-Users can use it to predict Crime rate and Crime Category of different Police Districts.

PLAGIARISM CHECK RESULT

18ACB03448_FYP2

ORIGINALITY REPORT

2%	%	2%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Varvara Ingilevich, Sergey Ivanov. "Crime rate prediction in the urban environment using social factors", Procedia Computer Science, 2018 Publication	1%
2	Luiz G.A. Alves, Haroldo V. Ribeiro, Francisco A. Rodrigues. "Crime prediction through urban metrics and statistical learning", Physica A: Statistical Mechanics and its Applications, 2018 Publication	<1%
3	Dhanush M, Hasifa A S, Merin Meleet. "Crime Prediction and Forecasting using Voting Classifier", 2021 Fourth International Conference on Electrical, Computer and Communication Technologies (ICECCT), 2021 Publication	<1%
4	Rima Hajjo, Dima A. Sabbah, Sanaa K. Bardaweel, Alexander Tropsha. "Identification of Tumor-Specific MRI Biomarkers Using Machine Learning (ML)", Diagnostics, 2021 Publication	<1%

5	Md. Martuza Ahamad, Sakifa Aktar, Md. Rashed-Al-Mahfuz, Shahadat Uddin et al. "A machine learning model to identify early stage symptoms of SARS-Cov-2 infected patients", Expert Systems with Applications, 2020 Publication	<1 %
6	Jiajun Bu, Xin Shen, Bin Xu, Chun Chen, Xiaofei He, Deng Cai. "Improving Collaborative Recommendation via User-Item Subgroups", IEEE Transactions on Knowledge and Data Engineering, 2016 Publication	<1 %
7	FARMAN HASSAN, Auliya Ur Rahman, Ali Javed, Ali Alhazmi, Majed Alhazmi. "CNN-CardioAssistant: Deep Convolutional Neural Network and Recursive Feature Elimination Method for Heart Disease Detection", Research Square Platform LLC, 2022 Publication	<1 %
8	Hanae Aoulad Ali, Chrayah Mohamed, Bouzidi Abdelhamid, Nabil Ourdani, Taha El Alami. "Chapter 5 A Novel Hybrid Classification Approach for Predict Performance Student in E-learning", Springer Science and Business Media LLC, 2023 Publication	<1 %

9	Mohammad Hassan Fathollahzadeh, Paulo Cesar Tabares-Velasco. "Electric demand minimization of existing district chiller plants with rigid or flexible thermal demand", Applied Energy, 2021 Publication	<1 %
10	Shen Khang Teoh, Vooi Voon Yap, Humaira Nisar. "Fast Regression Convolutional Neural Network for Visual Crowd Counting", 2021 International Conference on Computer & Information Sciences (ICCOINS), 2021 Publication	<1 %
11	Talha Ahmed Khan, Asif Mehmood, Javier Jose Diaz Rivera, Wang-Cheol Song. "Machine Learning Approach for Automatic Configuration and Management of 5G Platforms", 2019 20th Asia-Pacific Network Operations and Management Symposium (APNOMS), 2019 Publication	<1 %
12	Allen, Phillip E.. "CMOS Analog Circuit Design", Oxford University Press Publication	<1 %
13	Giorgio Visani, Enrico Bagli, Federico Chesani, Alessandro Poluzzi, Davide Capuzzo. "Statistical stability indices for LIME: Obtaining reliable explanations for machine learning	<1 %

9	Mohammad Hassan Fathollahzadeh, Paulo Cesar Tabares-Velasco. "Electric demand minimization of existing district chiller plants with rigid or flexible thermal demand", Applied Energy, 2021 Publication	<1 %
10	Shen Khang Teoh, Vooi Voon Yap, Humaira Nisar. "Fast Regression Convolutional Neural Network for Visual Crowd Counting", 2021 International Conference on Computer & Information Sciences (ICCOINS), 2021 Publication	<1 %
11	Talha Ahmed Khan, Asif Mehmood, Javier Jose Diaz Rivera, Wang-Cheol Song. "Machine Learning Approach for Automatic Configuration and Management of 5G Platforms", 2019 20th Asia-Pacific Network Operations and Management Symposium (APNOMS), 2019 Publication	<1 %
12	Allen, Phillip E.. "CMOS Analog Circuit Design", Oxford University Press Publication	<1 %
13	Giorgio Visani, Enrico Bagli, Federico Chesani, Alessandro Poluzzi, Davide Capuzzo. "Statistical stability indices for LIME: Obtaining reliable explanations for machine learning	<1 %

models", Journal of the Operational Research Society, 2021
Publication

14	Sikha Bagui. "An Approach to Mining Crime Patterns", International Journal of Data Warehousing and Mining, 2006 Publication	<1 %
15	Wilf R. LaLonde. "A smalltalk window system based on constraints", Conference proceedings on Object-oriented programming systems languages and applications - OOPSLA 88 OOPSLA 88, 1988 Publication	<1 %

Exclude quotes	On	Exclude matches	Off
Exclude bibliography	On		

Universiti Tunku Abdul Rahman			
Form Title : Supervisor's Comments on Originality Report Generated by Turnitin for Submission of Final Year Project Report (for Undergraduate Programmes)			
Form Number: FM-IAD-005	Rev No.: 0	Effective Date: 01/10/2013	Page No.: 1of 1

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Full Name(s) of Candidate(s)	CHEE MAN HANG
ID Number(s)	18ACB03448
Programme / Course	BACHELOR OF COMPUTER SCIENCE (HONOURS)
Title of Final Year Project	Crime Rate Prediction Using Machine Learning

Similarity	**Supervisor's Comments (Compulsory if parameters of originality exceeds the limits approved by UTAR)**
Overall similarity index:___2_____ % **Similarity by source** Internet Sources: _____N/A_____% Publications: _____2_____% Student Papers:_____N/A_____ %	The similarity index is low.
Number of individual sources listed of more than 3% similarity: _0_____	

Parameters of originality required and limits approved by UTAR are as Follows:

(i) Overall similarity index is 20% and below, and

(ii) Matching of individual sources listed must be less than 3% each, and

(iii) Matching texts in continuous block must not exceed 8 words

Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.

Note Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.

Signature of Supervisor	Signature of Co-Supervisor
Name: __Dr Chang Jing Jing_____	Name: _____
Date: ____8 Sep 2022_____	Date: _____

UNIVERSITI TUNKU ABDUL RAHMAN

FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY (KAMPAR CAMPUS)

CHECKLIST FOR FYP2 THESIS SUBMISSION

Student ID	18ACB03448
Student Name	Chee Man Hang
Supervisor Name	Dr Chang Jing Jing

TICK (√)	DOCUMENT ITEMS Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
	Front Plastic Cover (for hardcopy)
√	Title Page
√	Signed Report Status Declaration Form
√	Signed FYP Thesis Submission Form
√	Signed form of the Declaration of Originality
√	Acknowledgement
√	Abstract
√	Table of Contents
√	List of Figures (if applicable)
√	List of Tables (if applicable)
	List of Symbols (if applicable)
√	List of Abbreviations (if applicable)
√	Chapters / Content
√	Bibliography (or References)
√	All references in bibliography are cited in the thesis, especially in the chapter of literature review
	Appendices (if applicable)
√	Weekly Log
√	Poster
√	Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)
√	I agree 5 marks will be deducted due to incorrect format, declare wrongly the ticked of these items, and/or any dispute happening for these items in this report.

*Include this form (checklist) in the thesis (Bind together as the last page)

I, the author, have checked and confirmed all the items listed in the table are included in my report.

_____

(Signature of Student)

Date:8/9/2022