Crime Prediction Model using Three Classification Techniques: Random Forest, Logistic Regression, and LightGBM

Abdulrahman Alsubayhin, Muhammad Sher Ramzan, Bander Alzahrani
King Abdulaziz University, Jeddah, Kingdom of Saudi Arabia

***Abstract*—Predicting the likelihood of a crime occurring is difficult, but machine learning can be used to develop models that can do so. Random forest, logistic regression, and LightGBM are three well-known classification methods that can be applied to crime prediction. Random forest is an ensemble learning algorithm that predicts by combining multiple decision trees. It is an effective method for classification tasks, and it is frequently employed for crime prediction because it handles imbalanced datasets well. Logistic regression is a linear model that can be used to predict the probability of a binary outcome, such as the occurrence of a crime. It is a relatively straightforward technique that can be effective for crime prediction if the features are carefully chosen. LightGBM is a gradient-boosting decision tree algorithm with a reputation for speed and precision. It is a relatively new algorithm, but because it can achieve high accuracy on even small datasets, it has rapidly gained popularity for crime prediction. The experimental results show that the LightGBM performs best for binary classification, followed by Random Forest and Logistic Regression.**

Keywords*—Crime prediction; random forest; logistic regression; LightGBM

I. Introduction

In a culture where crime is low, it is always disturbing to see the number of crimes rising [1]. Crime is a social issue that hinders the economic growth of a nation. Crime has always existed, and violent crime is the greatest threat to society [2]. Population growth and urbanization have dramatically increased criminal activity [3], particularly in urban areas [4].

In recent years, crime prediction has acquired popularity because it enables investigation authorities to handle crimes computationally [5]. Better predictive algorithms that direct police patrols towards criminals are required [6]. Several research investigations have been conducted to predict crime categories, crime rates, and crime hotspots using crime datasets from various regions, such as South Korea and the United States [7]. Additionally, using the Canada dataset, various prototype projects are expanded to identify crime-related geographic locations, such as residential and commercial areas [8].

Crime threatens us and society, necessitating serious consideration if we expect to reduce its onset or consequences.

Daily, data officers working alongside law enforcement authorities throughout the United States record hundreds of crimes. Numerous cities in the United States have signed the Open Data initiative, making crime data and other categories of data accessible to the public. This initiative aims to increase citizen participation in decision-making by uncovering interesting and valuable facts using this data [9].

San Francisco is one of many cities that have joined this Open Data movement. The data scientists and engineers working with the San Francisco Police Department (SFPD) have documented over one hundred thousand criminal cases based on police complaints [10]. Using these historical data, numerous patterns can be uncovered. This would help us identify crimes that may occur in the future, allowing the municipal police to better protect the city's population [11].

Violent and nonviolent crimes are predicted and classified using random forest, logistic regression, and LightGBM. The primary objective of this paper is to propose a crime prediction model based on past criminal records.

Using three techniques, the proposed model evaluates accuracy, log loss, ROC AUC, precision, and recall evaluation matrices. The data is descriptively analyzed, and crime statistics' spatial and temporal distribution are visualized to identify potential patterns. The original dataset's features are extracted, and classification is carried out using random forest, logistic regression, and LightGBM techniques.

LightGBM has the highest performance for binary classification, followed by random forest and logistic regression, according to the experimental results. LightGBM has the best precision, accuracy, log loss, ROC AUC, and F1 score. It has the lowest recall, but this is not inherently a negative attribute. In this case, the dataset is imbalanced, as there are far more examples of class 0 than class 1. This means that avoiding false positives is essential to avoiding false negatives. LightGBM accomplishes this by emphasizing recall while maintaining high precision. Random forest has a lower accuracy, log loss, ROC AUC, precision, and F1 score than LightGBM but a higher recall. This indicates that random forest is superior at avoiding false negatives but less effective at predicting true positives. Logistic regression has the three models' lowest accuracy, log loss, ROC AUC, precision, and F1 score. It has the lowest recall as well. This indicates that logistic regression is the model with the worst performance for binary classification.

Overall, the model with the greatest performance for binary classification is LightGBM, followed by random forest and logistic regression.

A. Related Work

Due to the relationship between crime and society, predictions of future crime have been investigated extensively. These studies use machine learning algorithms to address these predictions. Using machine learning algorithms to predict spatial crime data has proven effective [12].

Accurate crime prediction is difficult but essential for preventing criminal behavior. Accurately estimating the crime rate, types, and hot areas based on historical patterns presents numerous computational challenges and opportunities [5].

Prediction analysis is dominated by crime prediction based on machine learning; however, few studies systematically compare machine learning methods. The ability of machine learning algorithms to process non-linear rational data has been validated in numerous disciplines, including crime prediction. It can process high-dimensional data with a faster training pace and extract the characteristics of the data [13].

Despite extensive research efforts, the literature lacks relative accuracy for crime prediction from large datasets for multiple locations, such as Los Angeles and Chicago datasets.

The authors of [14] employ the model to improve the effectiveness of criminal investigation systems. This model identifies crime patterns based on inferences gathered from the crime site and predicts the perpetrator's description of the suspect most likely responsible for the crime. This work has two primary elements: Analyzing and forecasting the perpetrator's identity. The crime analysis phase identifies the number of unsolved crimes and evaluates the impact of variables such as year, month, and weapon on those crimes. The prognosis phase estimates the perpetrators' characteristics, such as age, gender, and relationship to the victim. These hypotheses are based on the evidence gathered at the crime scene. The system predicts the perpetrator's physical characteristics using algorithms such as multilinear regression, K-neighbors classifier, and neural networks. It was trained and evaluated using the San Francisco Homicide dataset (1981-2014) and Python.

Yao et al. used the San Francisco Dataset; this paper is based on the random forest algorithm, which splits the study areas into four groups based on the hot spot distribution based on historical crime data: frequent hot areas, common hot areas, occasional hot areas, and non-hot areas; then, corresponding covariates from non-historical crime data are added to the prediction model to investigate changes in the result accuracy of crime prediction [15]. The data relies on actual data, and the experimental findings reveal that compared to the inference approach based solely on historical crime data, the model with covariates outperforms the model without covariates.

A preliminary analysis of the spatiotemporal crime patterns in San Francisco is attempted in this study [16]. They use spectral analysis to examine the temporal evolution of all crime categories, discovering that many exhibit a weekly or monthly pattern and other components. They demonstrate that spatial distribution has weekly patterns. These findings can be used to develop predictive models for policing and increase knowledge of crime dynamics.

II. Data Analysis

The model in the study is built using a Kaggle dataset [17]. The dataset (training set/data) has several properties, each with its own link. The Kaggle incidences of San Francisco crimes are included in the training dataset. The data spans the years January 2003 to May 2015. The collection covers nearly 12 years of San Francisco criminal reports. The collection contains categories of all crimes containing various crime types.

The original training dataset is arbitrarily mixed and divided into training and testing datasets of 80% and 20%, respectively, in the study. Any data imbalances relating to the "Primary Type" feature were corrected using a combination of oversampling (SMOTE) and random sampling; SMOTE stands for Synthetic Minority Over-sampling Technique. It is a data augmentation approach used in machine learning to deal with skewed datasets. SMOTE generates synthetic minority class samples by combining existing minority class samples. This balances class distribution and improves machine learning model performance on minority class predictions.

A. Features

Every entry in our data set pertains to a specific crime, and each data record includes the following characteristics:

- Dates - The date and time of the crime.
- Category - The type of crime. In the classification stage, we must forecast this target/label.
- Descript - A brief description of any relevant details of the crime.
- DayOfWeek - The weekday on which the offense happened.
- PdDistrict - The Police Department District to which the offense has been assigned.
- Resolution - How the crime was resolved (for example, by arresting or booking the culprit).
- Address - The crime scene's approximate street address.
- X - Longitude of a crime's location.
- Y - The latitude of a crime's site.

B. Preprocessing

We execute various preprocessing processes on our datasets to achieve better classification results before deploying any algorithms. These are some examples:

- Eliminating features like resolution, description, and address. The resolution and description of a crime are only known after the crime has occurred and have limited relevance in a practical, real-world scenario where one is attempting to predict what type of crime has occurred; hence, these were deleted. We deleted the address since we already knew the latitude and longitude; in that context, the address added little marginal value.

- The weekdays, police, and criminal categories were all indexed and replaced with numbers.
- The timestamp included the year, date, and time of each offense. This was broken down into five components: Year (2003-2015), Month (1-12), Date (1-31), Hour (0-23), and Minute (0-59).
- We used Python's get_dummies() function to turn categorical variables into dummy variables. Dummy variables are binary variables that show whether a given category exists. For example, if a category variable contains three types, the get_dummies() function will generate two dummy variables. One dummy variable indicates the presence of the first category, while the other indicates the presence of the second category. The lack of the first two dummy variables will implicitly reflect the third category.
- In machine learning, the get_dummies() function is a popular technique to deal with categorical variables. Many machine learning algorithms cannot deal with categorical data directly. Machine learning algorithms may train models using data that includes categorical variables by turning categorical variables into dummy variables. We also remove certain unnecessary elements, such as incidentNum and coordinate.

Feature inclusion is imperative to the predictive capabilities of any model, ensuring its ability to capture the complexity of crime patterns. Excluding specific demographic, economic, or environmental features may result in a less comprehensive understanding of the factors influencing criminal activities, leading to lead to oversimplified predictions or overlooking important contributing factors.

Following these preprocessing processes, we ran some out-of-the-box learning algorithms as part of our early exploratory stages.

C. Feature Engineering

The act of changing raw data into features more suited for machine learning algorithms is known as feature engineering. This can include some tasks, such as:

- Data cleaning entails removing errors, outliers, and missing values from the data.
- Feature selection: entails choosing the most essential features from the data.
- Feature extraction is the process of producing new features from current ones.
- Feature transformation: entails changing features to a different format, such as category or numerical values.

The purpose of feature engineering is to produce informative and predictive features. Informative features provide relevant information about the target variable. Predictive characteristics are those that can accurately anticipate the target variable. Feature engineering is a critical step in the machine learning process. We can improve the performance of our machine learning models by carefully engineering features.

D. Exploratory Data Analysis

The first dataset analysis found a major imbalance in the "Primary Type." This is evident in Fig. 1, which demonstrates that "larceny/theft," "other offenses," and "noncriminal" make up a significant portion of the total crimes committed in San Francisco. Since these offenses are more likely to occur, it is reasonable to propose allocating more police resources to combat them.

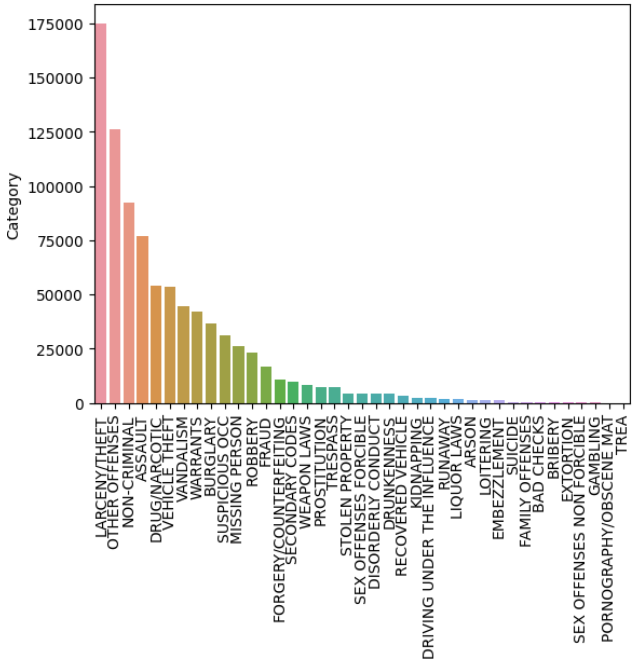

Fig. 1. Number of crimes.

Fig. 2 is a data visualization based on the PdDistricts; it displays the locations where crime occurs most frequently according to the district's name. Southern has the highest crime rate, while Richmond has the lowest crime rate. According to a crime map created by NeighborhoodScout, the Southern District has the highest crime rate in the United States, with 60.5 crimes per 1,000 residents. The Richmond neighborhood has the lowest crime rate, with 18.2 crimes per 1,000 residents.

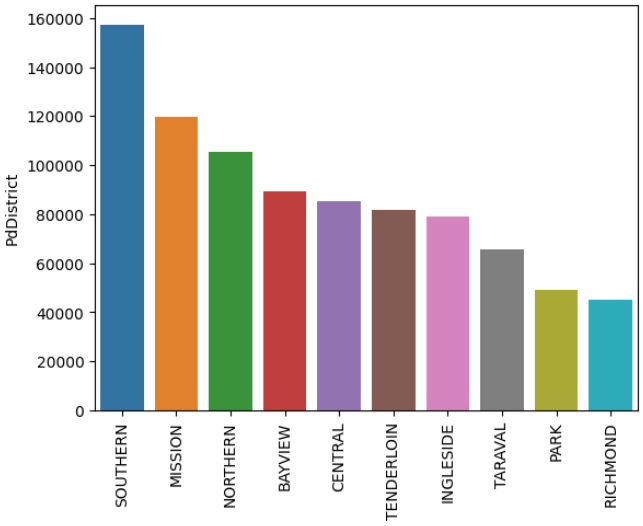

Fig. 2. The visualization for data depends on the PdDistricts.

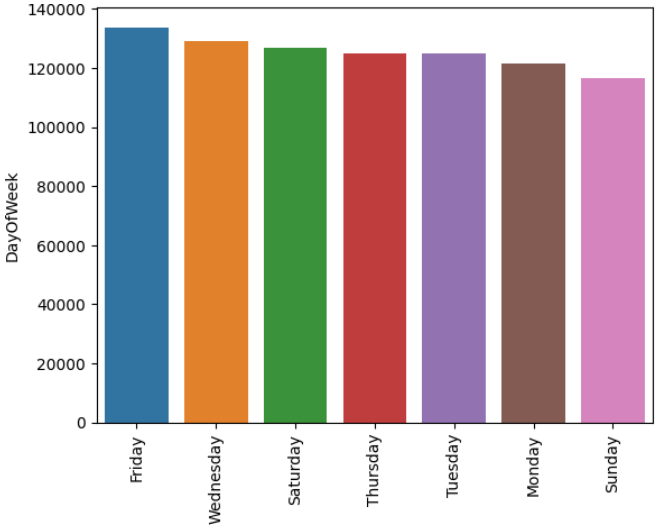

Fig. 3. The visualization for data depends on the day of the week.

Fig. 3 depicts a data visualization based on the day of the week. This pattern has several possible explanations. A possible explanation is that people are more likely to be out and about on Fridays, making them more susceptible to becoming victims of crime. A second possibility is that people are more likely to be intoxicated on Fridays, which can increase aggression and violence.

Regardless of the cause, it is evident that the daily crime rate varies significantly. Law enforcement officials and policymakers should consider this factor when devising strategies to reduce crime.

According to Fig. 4, the highest crime rates in San Francisco occur at 1, 2, 6, and 11 p.m. These times are typically when people are sleeping or are out and about in the early morning, as well as when people are leaving work or school or running errands. This increases their likelihood of being targeted by criminals.

When working with vast datasets, it is inevitable to encounter imbalances. Most machine learning algorithms tend to presume, by default, that the data they are working with is balanced [18]. Imbalances can cause problems when attempting to train a classification model. This presumption causes the trained models' outputs to be biased and skewed toward the majority class [18].

Fig. 5 depicts the most widespread types of crimes in descending order. For the past 13 years, theft has been the most frequent offense in San Francisco. As opposed to shoplifting or purse snatching, this form of theft does not involve force or violence. Additionally, prevalent in San Francisco are Assault, Burglary, and Vehicle Theft.

Fig. 6 displays intriguing year-based data and results. It shows the increase or decrease of the top ten offenses in San Francisco from 2003 to 2015.

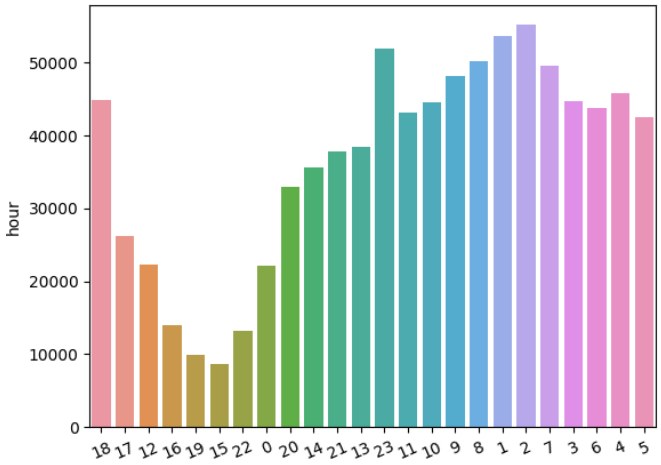

Fig. 4. The visualization of data depends on the hour.

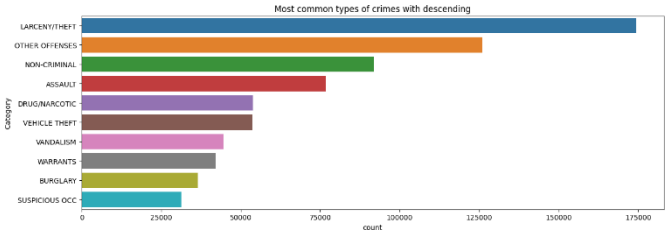

Fig. 5. The most common types of crimes in descending order.

1) Variable selection. In the San Francisco crime dataset, the dependent variable for prediction is "Category." Given the other variables in the dataset, the analysis attempts to predict the crime committed.

Resolution and description are irrelevant to the analysis because they are not numerical in character. "Resolution" is a categorical variable that denotes how the case was resolved, whereas "Description" is a text variable that provides a comprehensive description of the incident. The other variables are independent variables used to predict the dependent variable.

2) Variable transformation. A handful of variables are transformed to improve the characteristics of the dataset. In the San Francisco crime dataset, the "Date" variable is separated into four distinct variables:

- Year: The values for this variable range from 2003 to 2015 and denote the year in which the incident occurred.
- Month: This variable represents the month in which the incident occurred. This variable has values between 1 and 12.
- Day: This variable represents the day of the month the incident occurred. This variable's values range from 1 to 31.
- Hour: This variable specifies the time of day when the incident occurred. This variable's values range from 0 to 23.

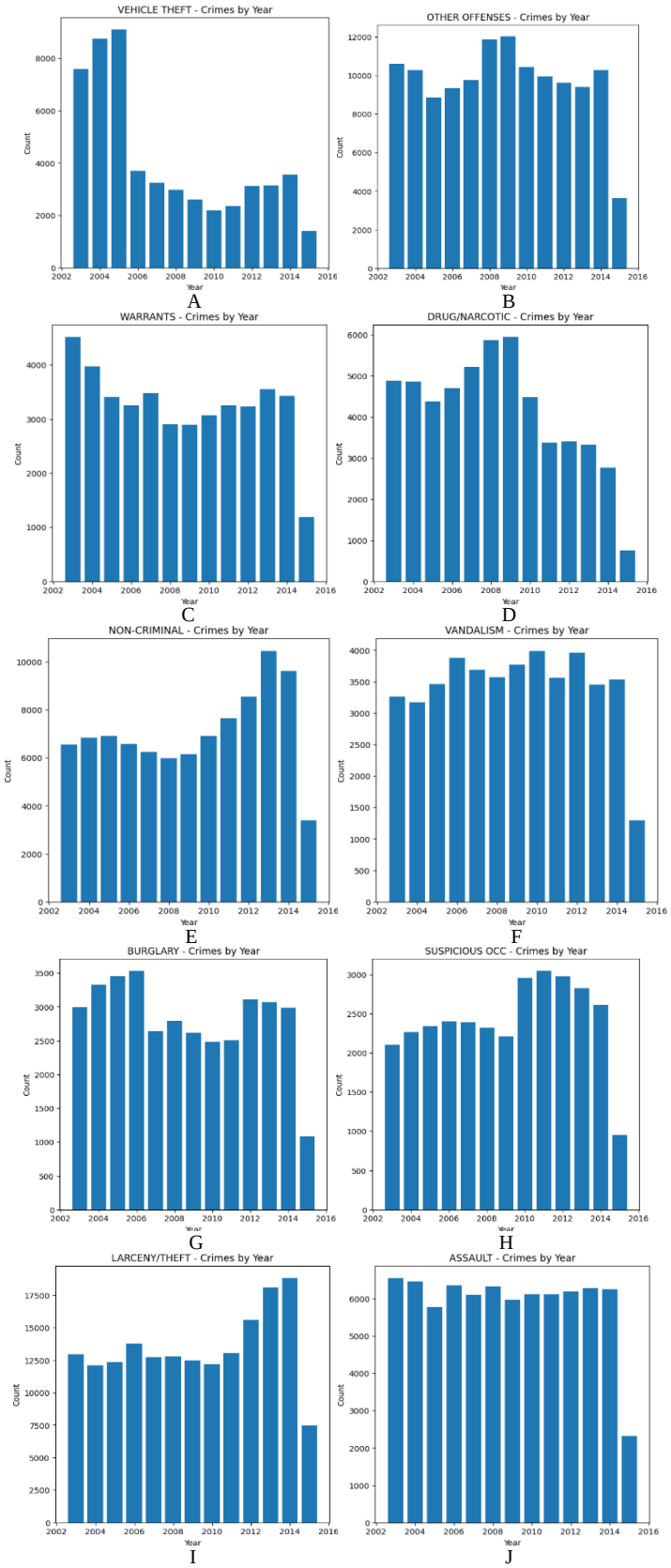

Fig. 6. Top ten crimes based on years (A) Vehicle theft, (B) Other offences, (C) Warrants, (D) Drug / narcotic, (E) Non-criminal, (F) Vandalism, (G) Burglary, (H) Suspicious OCC, (I) Larceny / theft, (J) Assault.

This makes the data more manageable and permits a more thorough analysis. For instance, we could use the "Year" variable to determine how crime rates have changed over time or the "Hour" variable to determine which hours of the day are most likely associated with criminal activity.

Note that "Date" is not the only variable that can be used to analyze crime data. Other significant variables include "PdDistrict," which indicates the police district where the incident occurred, and "Category," which indicates the category of crime committed. Combining these variables makes it possible to understand crime in San Francisco more deeply.

The "DayOfWeek" and "PdDistrict" variables are indexed and substituted with numbers in the San Francisco crime dataset. This makes the data more manageable and permits a more thorough analysis.

The index range for the "DayOfWeek" variable is 1 to 7, with 1 representing Monday and 7 representing Sunday. The "PdDistrict" variable has an index range of 1 to 10, where 1 represents the Northern District, and 10 represents the Southern District. This enables us to compare crime rates across days of the week and police districts with ease.

E. Model

The prediction model is based on Random forest, logistic regression, and light GBM techniques, briefly discussed below:

1) Random forest: Random forests are a widely used ensemble learning technique that constructs multiple classifiers on training data and integrates their outputs to make the most accurate predictions on test data. Consequently, the random forests algorithm is a variance-minimizing algorithm that employs randomness to avoid overfitting the training data when making split decisions.

Random forest is a supervised learning technique capable of managing classification and regression problems based on a single fundamental concept - the collective intelligence of a population. It employs many independent decision trees as an ensemble [19]. The model's overall prediction is the class with the most votes [19]. It conducts classifications by summing the classifications produced by each individual tree within the "forest," and the class with the most votes is the model's overall prediction.

A random forest classifier is an ensemble classifier that aggregates a family of classifiers h(xjθ1); h(xjθ2);::h(xjθk). Each family member, h(xjθ), is a classification tree, and k is the number of trees chosen from a model random vector.

Also, each θk is a randomly chosen parameter vector. If D(x; y) denotes the training dataset, each classification tree in the ensemble is built using a different subset Dθk(x; y) ⊂ D(x; y) of the training dataset. Thus, h(xjθk) is the kth classification tree, which uses a subset of features xθk ⊂ x to build a classification model. Each tree then works like regular decision trees: it partitions the data based on the value of a particular feature (selected randomly from the subset) until the data is fully partitioned or the maximum allowed depth is reached. The final output y is obtained by aggregating the results thus:

$$y = argmax_{p\in\{h(x_1)..h(x_k)\}}\left\{\sum_{j=1}^{k}(I(h(x|\theta_j)=p))\right\}$$

where:

- I denotes the indicator function.

2) Logistic regression: Logistic regression is a statistical model used to predict the probability of a binary outcome, such as whether a customer will click on a commercial, whether a loan applicant will default, or whether a patient has a disease. The binary outcome is first converted to a probability for logistic regression to function. The logistic function, a sigmoid function that accepts a real number as input and returns a number between 0 and 1, is used for this purpose. The logistic function is defined as follows:

logistic(x) = 1 / (1 + e(-x))

Where:

- x is the function's input.

After transforming the outcome into a probability, logistic regression employs a linear regression model to predict the likelihood. The independent variables are input into the linear regression model, which returns a predicted probability. The model's accuracy is then improved by comparing the predicted probability to the actual probability and updating the model accordingly.

Logistic regression is a highly effective technique for predicting binary outcomes. It is simple to comprehend and interpret and can be applied to various data types. Additionally, logistic regression is comparatively robust against outliers and absent data.

Detection of fraud, customer segmentation, risk analysis, and targeted marketing are some of the applications of logistic regression.

3) Light gradient boosting machine: LightGBM is a free and open-source distributed gradient-boosting framework for machine learning. It is intended to be quick, effective, and scalable. LightGBM is based on decision tree algorithms and builds models using gradient boosting.

LightGBM is a well-liked option for various machine-learning tasks, such as classification, regression, and ranking. It is ideally adapted for large-scale datasets and can be used to train highly accurate models.

LightGBM offers advantages; it is one of the quickest gradient-enhancing frameworks available. Several refinements, including tree pruning and histogram-based splitting, contribute to this. In terms of memory usage, LightGBM is also very efficient. This makes it an excellent option for training models with large datasets. LightGBM is intended for use with large datasets. This is accomplished via distributed training and a variety of other optimizations. In addition, LightGBM is capable of achieving high accuracy in a variety of machine-learning tasks. This is due to its use of decision tree and gradient enhancement algorithms.

$g(x) = f(x) + \beta * h(x)$

Where:

- $g(x)$ is the predicted value for x.
- $f(x)$: is the base learner.
- β is the learning rate.
- $h(x)$ is the gradient boosting step.

The base learner in LightGBM is a decision tree. The gradient boosting step is a technique that iteratively adds new decision trees to the model to improve the accuracy of the predictions.

The equation for LightGBM can be simplified as:

$g(x) = f(x) + \beta * (y - f(x))$

Where:

- y is the actual value for x.

This equation shows that the predicted value for x is a linear combination of the base learner and the gradient boosting step. The learning rate β controls the weight of the gradient boosting step.

III. RESULTS

Each of the three models was trained and presented with distinct parameter and feature selections in the preceding section. The data exploration section notes that both temporal and geographical characteristics are significant. For analysis, all three models are trained and evaluated using the Kaggle training dataset containing 878,049 records, and each model is divided into two sections with a ratio of 80:20. Consequently, 80% of the dataset was used to train the model. In contrast, 20% was used for testing it.

A. Random Forest

Random forest is an ensemble learning technique that integrates the predictions of multiple models to produce a final prediction. The individual models within random forests are decision trees. Each decision tree within a random forest is trained with a unique bootstrap sample of the training data. This means that each tree will observe a distinct subset of the data, thereby helping to prevent overfitting. The random forest also employs a technique known as feature randomness in addition to bootstrap sampling. This means that each decision tree can only consider a random subset of the features when making a split.

Accuracy score, log loss, confusion matrix, and ROC curve are all metrics used to evaluate the performance of classification models. However, they measure different aspects of the model's performance.

Some of the hyperparameters that can be tuned for a random forest classifier:

- n_estimators: This is the number of trees in the forest. The higher the number of trees, the more accurate the model will be, but it will also take longer to train.
- max_depth: This is the maximum depth of the trees in the forest. A higher depth will allow the model to make more complex decisions but can also lead to overfitting.
- min_samples split: This is the minimum number of samples required to split a node in the tree. A higher number of samples will make the model more conservative but can also lead to underfitting.
- min_samples leaf: This is the minimum number of samples required in a leaf node. A higher number of samples will make the model more conservative but can also lead to underfitting.
- random_state: This random number generator seed initializes the random forest algorithm. A higher value of random state will lead to more reproducible results but can also lead to overfitting. A lower value of random state will lead to less reproducible results but can also lead to better generalization.

random forest classifier with tuned hyperparameters random forest model = random forest classifier (n_estimator = 100, max_depth = 32, min samples split = 16, random state = 42)

random forest model fit (x train, y train)

#predict on the test set

Y pred = random forest model predict (x test)

The accuracy score is the most common metric for evaluating classification models. It is simply the percentage of instances that were correctly classified. For example, if a model correctly classifies 90 out of 100 instances, its accuracy score would be 0.90.

For Random Forest Accuracy = 0.4262

The accuracy score is generally the easiest metric to understand, but it can sometimes be misleading. Thus, log loss, confusion matrix, and ROC curve are all metrics used to evaluate the performance of classification models. However, they measure different aspects of the model's performance.

Log loss is a measure of the difference between the predicted probabilities of a model and the actual labels. It is a continuous measure, and it can be interpreted as the average amount of information lost when the predicted probabilities are used to represent the actual labels. A lower log loss indicates a better model and a log loss of 0 indicates a perfect model.

For Random Forest, the log loss = 1.74

A log loss of 1.74 is not a bad score, but it is not great. Getting better scores with a more complex model or with more training data is possible. However, getting worse scores with a more complex model or with more training data is also possible.

The log loss measures the difference between the predicted probabilities and the actual labels. A lower log loss indicates a better model. However, it is important to note that log loss is not the only measure of model performance. Other measures, such as accuracy and precision, can also be used to evaluate model performance.

The confusion matrix is a table that summarizes the performance of a classification model. It shows the number of instances correctly classified (true positives and true negatives) and the number of incorrectly classified (false positives and false negatives).

For Random Forest, the confusion matrix in Fig. 7.

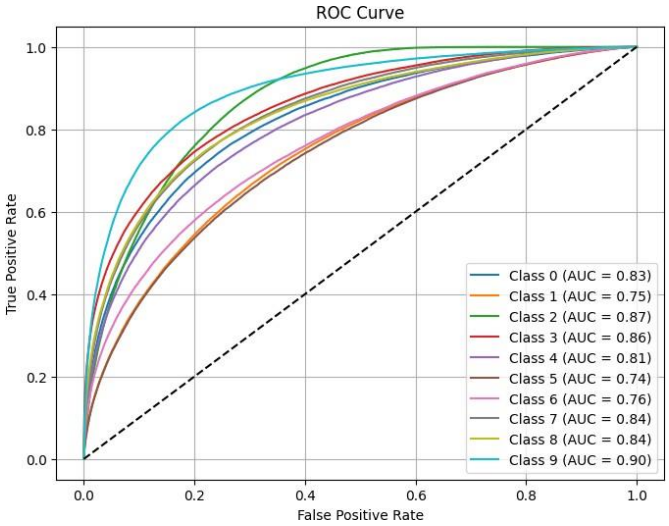

Fig. 7. Random forestROC curve.

ROC curve, or Receiver Operating Characteristic curve, is a graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied. The ROC curve plots the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. The TPR is also known as recall and is defined as the fraction of positive instances correctly identified as positive. The FPR is defined as the fraction of negative instances that are incorrectly identified as positive.

A perfect classifier would have a ROC curve that passes through the upper-left corner of the graph with a TPR of 1 and an FPR of 0. However, in practice, no classifier is perfect, and the ROC curve will typically be a curve that falls below the upper-left corner.

The Random Forest ROC curve in Fig. 8 shows an AUC of 0.90, which is a good score for a binary classification model. AUC stands for area under the curve, a measure of the model's ability to distinguish between the two classes. A higher AUC indicates a better model.

In the case of class 9, an AUC of 0.90 means that the model can correctly classify 90% of the instances in the test set. This is a good score; however, some factors can affect the AUC of a model, including the complexity of the model, the amount of training data, and the model's hyperparameters.

The accuracy score is generally the easiest metric to understand, but it can sometimes be misleading. Log loss is a more sensitive metric but is not as easy to interpret. The confusion matrix is a good way to get a detailed view of the model's performance, but it can be difficult to interpret for large datasets. The ROC curve is a good way to visualize the model's performance and compare different models.

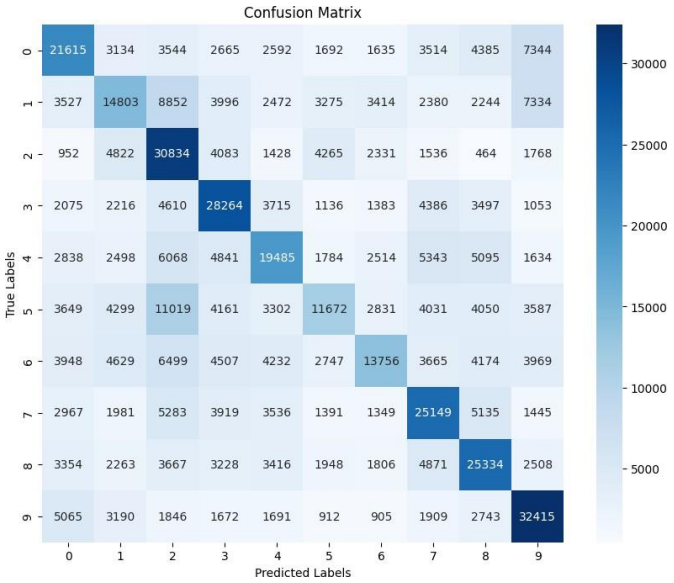

Fig. 8. Confusion matrix for random forest classifier.

Ultimately, the best way to evaluate a classification model is to use a combination of metrics. This will give a complete picture of the model's performance and help make better decisions about the model.

B. *Logistic Regression*

Logistic regression is a statistical model used to estimate the probability of a binary outcome. The result can either be "success" or "failure." In contrast to linear regression, logistic regression is used to predict probabilities rather than continuous values.

Logistic regression is a prominent classification model for binary problems. Additionally, the model is easy to comprehend and implement. However, logistic regression can be susceptible to overfitting; therefore, cross-validation must be used to evaluate the model's performance.

Some of the hyperparameters that can be tuned for a random forest classifier:

max_iter: hyperparameter in logistic regression is the maximum number of iterations for which the model will be trained. A higher value of max_iter will generally lead to a better model but can also lead to longer training times.

Random state: hyperparameter in logistic regression is a random number generator seed used to initialize the model. A higher value of random state will lead to more reproducible results but can also lead to overfitting. A lower value of random state will lead to less reproducible results, but it can also lead to better generalization

Multi-class: The multi-class parameter specifies the multi-class classification algorithm used by the LogisticRegression class. If the value is ovr, logistic regression builds a separate model for each class. The predicted values for each class are then compared, and the class with the highest predicted value is taken as the predicted class for that instance.

Logistic regression with tuned hyperparameter

Logistic regression Model = LogisticRegression (max_iter = 1000, random state = 42, mutliclass = 'ovr')

For logistic regression accuracy = 0.221: an accuracy of 0.221 is not a very good score. It means the model can only correctly classify 22.1% of the instances in the test set. This is a relatively low score, and it suggests that the model is not very accurate.

For logistic regression, the log loss = 2.11: a log loss of 2.11 is not a good score. It means that the model is not very good at predicting the probability of the positive class. A lower log loss indicates a better model.

For Logistic regression, the confusion matrix in Fig. 9.

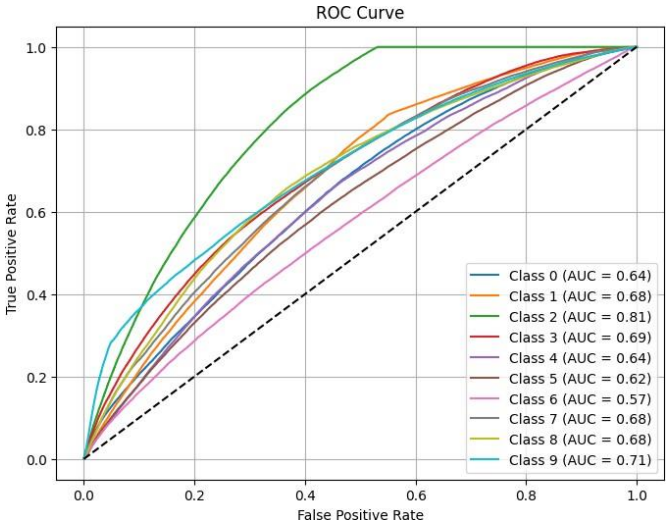

Fig. 9. Logistic regression ROC curve.

For the Logistic regression ROC curve in Fig. 10.

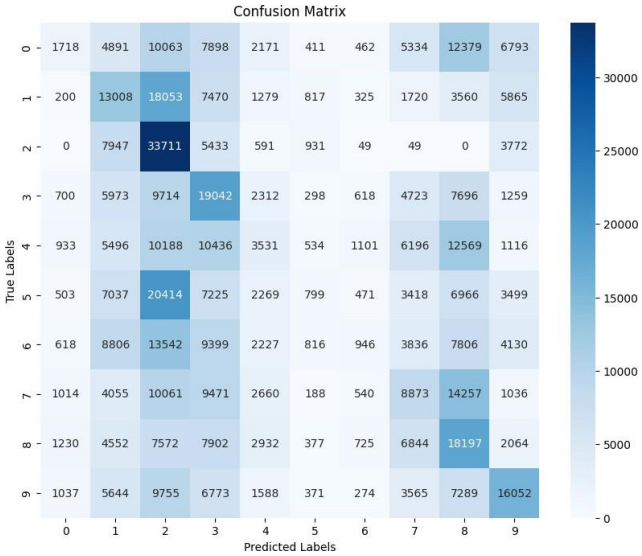

Fig. 10. Logistic Regression confusion matrix.

An AUC of 0.71 is a good score for a binary classification model. AUC stands for area under the curve, and it is a measure of the model's ability to distinguish between the two classes. A higher AUC indicates a better model.

In the case of class 9, an AUC of 0.71 means that the model can correctly classify 71% of the instances in the test set.

C. LightGBM

LightGBM is a robust machine-learning algorithm applicable to a variety of duties. It is quick, effective, and simple to use. However, it is not as versatile as other algorithms, and it is difficult to tune for intricate datasets.

Some of the hyperparameters that can be tuned for a LightGBM classifier:

Objective: This specifies the type of task the model tries to solve. For classification, the objective should be set to "multi-class."

Num classes: This specifies the number of classes in the classification problem.

Learning rate: This controls the amount of weight that is given to new information. A lower learning rate will result in a more conservative model, while a higher one will be more aggressive.

Num rounds: This specifies the number of times that the model will be trained. A higher number of rounds will result in a more accurate model, but training will also take longer.

LightGBM classifier with specific parameter lgb params =

'objectives': multi-class, 'num classes: 10,

'learning rate': 0 056,

'num round': 200,

For LightGBM Accuracy = 0.32: an accuracy of 0.32 is not a very good score for a LightGBM classifier. It means that the model is only able to correctly classify 32% of the instances in the test set.

For LightGBM, the log loss = 1.91: a log loss of 1.91 is not a good score for a LightGBM classifier. It means that the model is not very good at predicting the probability of the positive class. A lower log loss indicates a better model.

For LightGBM, the confusion matrix in the Fig. 11.

In the case of class 9, an AUC of 0.83 means that the model is able to correctly classify 83% of the instances in the test set.

Accuracy, log loss, precision, F1 score, and recall are all metrics used to assess machine learning models' performance for binary classification tasks.

Accuracy is the most frequent metric, and it measures the proportion of true predictions made by the model. However, accuracy can be deceiving if the dataset is imbalanced, i.e., there are significantly more instances of one class than the other.

Log loss is a metric that evaluates the model's average cross-entropy loss. Cross-entropy loss assesses the degree to which the model's predictions correspond to the actual labels. Logloss is superior to accuracy for imbalanced data sets, as it considers the number of true positives, false positives, true negatives, and false negatives.

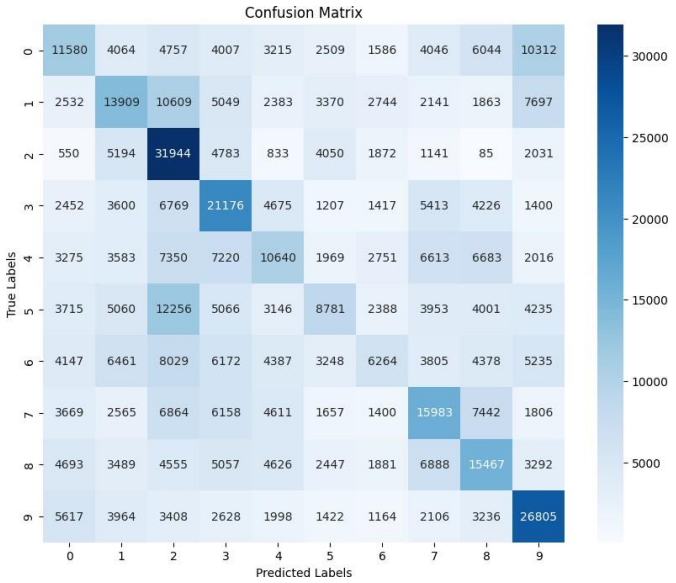

Fig. 11. LightGBM confusion matrix.

For the LightGBM ROC curve in Fig. 12.

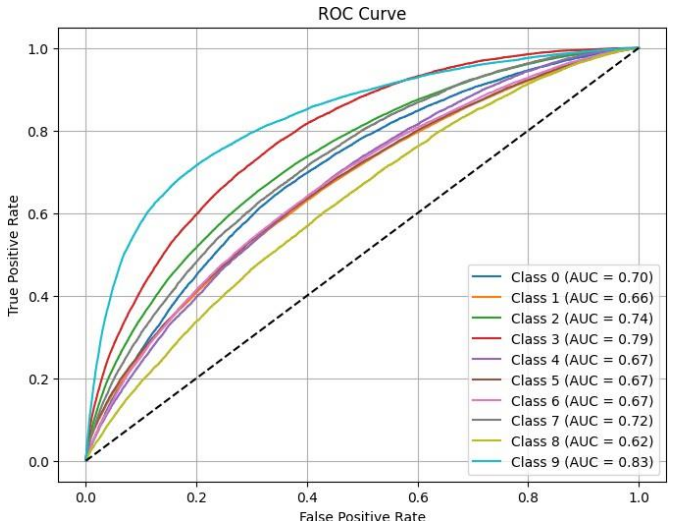

Fig. 12. LightGBM ROC curve.

Precision assesses the proportion of accurate positive predictions. If a model predicts that 100 patients have cancer and 90 of those patients actually have cancer, then the model's precision is 90%.

Recall measures the proportion of actual positives predicted to be positive. For instance, if a model predicts 100 patients have cancer, 80 of them do, then the recall is 80%.

The F1 score is an average of accuracy and recall. It is a more balanced metric than precision or recall alone and is frequently used to evaluate the overall performance of a model. The optimal metric to use depends on the particular application. For instance, precision may be the most essential metric if avoiding false positives is crucial. If avoiding false negatives is crucial, recall may be the most essential metric. Utilizing multiple metrics to evaluate the efficacy of a machine-learning model is generally recommended.

TABLE I. ACCURACY, LOG LOSS, PRECISION, F1 SCORE, AND RECALL FOR RANDOM FOREST CLASSIFIER, LOGISTIC REGRESSION, AND LIGHTGBM

	Techniques		
	Random Forest Classifier	***Logestic_Regression***	***LightGBM***
Accuracy	0.42704413339452346	0.22157908826678904	0.31082491968793025
logloss	1.74	2.11	1.9209639647523717
Precision	0.42	0.21	0.3013950978996477
F1 score	0.42	0.18	0.2959045359008416
Recall	0.43	0.22	0.31082491968793025

Table I summarises the accuracy, log loss, precision, F1 score, and recall for random forest Classifier, Logistic Regression, and LIGHT GBM.

IV. CONCLUSIONS

The proposed model contains three techniques and evaluates accuracy, precision, and recall evaluation matrices. The data is descriptively analyzed, and statistical crime distribution over space and time is visualized to help attain potential patterns. The features are extracted from the original dataset, and the classification is performed using random forest, logistic regression, and LightGBM techniques. LightGBM has the best performance for binary classification tasks based on the metrics we provided. It has the highest AUC (area under the ROC curve), which measures how well the model can distinguish between the two classes. LightGBM also has the highest precision and F1 score, which measures the accuracy of the model's predictions.

Random forest has the second-best performance, followed by logistic regression. Random forest has a slightly lower AUC than LightGBM but a higher precision. Logistic regression has the lowest AUC and precision but has a higher recall than the other two models.

LightGBM is generally a good choice for binary classification tasks when accuracy and precision are important. Random forest is a good choice when accuracy and recall are important. Logistic regression is a good choice when recall is more important than accuracy.

Random Forest, while robust, may struggle with certain types of crimes that exhibit complex patterns or dependencies. The ensemble of decision trees might face challenges in capturing intricate relationships within the data, leading to suboptimal predictions for specific crime categories.

Logistic Regression, although straightforward and interpretable, assumes a linear relationship between the independent variables and the log-odds of the outcome. This assumption might limit its ability to capture non-linear patterns inherent in some crime data, affecting its predictive accuracy for certain crime types.

LightGBM, despite its speed and efficiency, might encounter difficulties with interpretability due to its complex nature. The "black-box" aspect of gradient-boosting algorithms

can hinder understanding the rationale behind specific predictions, making it challenging to identify why certain types of crimes are predicted with higher or lower accuracy.

It is imperative to recognize potential biases and limitations that may impact the reliability and generalizability of the predictive models. Crime datasets inherently face challenges such as underreporting or misreporting, introducing inaccuracies into the dataset. Spatial biases may emerge if certain areas are disproportionately monitored or reported, creating an uneven representation of crime across locations. Additionally, temporal biases could arise due to variations in reporting frequency or law enforcement activities during specific time periods.

Demographic and socioeconomic factors may introduce biases in crime reporting and law enforcement activities, leading to potentially skewed representations of criminal activities. Over-policing or under-policing in specific communities may contribute to these biases.

Imbalances in class distribution, where certain crime events are less frequent than others, could affect the model's ability to accurately predict less common events. Variations in data collection methods across different regions or law enforcement agencies may impact the consistency and comparability of the dataset. Additionally, up-to-date data is essential otherwise it may not accurately reflect current crime patterns.

Considering ethical and legal considerations is essential, including issues related to privacy, data anonymization, and compliance with legal and ethical standards in handling crime data. researchers and practitioners are advised to transparently acknowledge and address these limitations through appropriate preprocessing techniques, feature engineering, and model evaluation strategies to enhance the robustness and reliability of predictive models.

Machine learning models, particularly those used for crime prediction, are vulnerable to biases present in their training data. If the dataset used is biased, the predictive model may perpetuate and worsen existing biases, leading to unfair targeting of specific demographics and reinforcing social inequalities within law enforcement practices. The fairness of predictions is crucial, as disproportionate predictions of crimes in certain communities or against specific groups can result in biased law enforcement actions, raising ethical concerns about the model's impact on the communities it predicts to have higher crime rates.

There is a risk of self-fulfilling prophecies, where increased law enforcement presence in predicted high-crime areas may lead to more arrests, creating a feedback loop that unfairly stigmatizes certain neighborhoods and individuals, contributing to over-policing and reinforcing negative stereotypes. Unintended consequences may occur if the model prioritizes predictive accuracy without considering the broader ethical implications, potentially neglecting less frequent but equally severe offenses and leading to imbalanced resource allocation.

To address these ethical concerns, continuous monitoring, evaluation, and refinement of the model are essential. Implementing fairness-aware algorithms, regularly auditing for biases, and involving diverse stakeholders in the development process can help mitigate ethical risks and ensure the responsible use of machine learning in crime prediction

Furthermore, the use of imbalanced data introduces a skewed representation of the classes, where certain outcomes dominate, leading to potential biases in the model's learning process. In the context of crime prediction, this could mean an overemphasis on prevalent types of crimes, potentially neglecting rarer but significant events.

Addressing imbalanced data is crucial for model robustness. Techniques such as oversampling the minority class, undersampling the majority class, or deploying advanced algorithms like SMOTE are common strategies. These techniques aim to balance class distribution, ensuring the model learns from the entirety of the dataset rather than being swayed by the abundance of one class.

While necessary for training of potential models, dealing with historical data in crime prediction, temporal limitations are a crucial aspect. Changes in social dynamics, law enforcement practices, and urban development over time can influence the relevance of historical data for current or future crime prediction. Evolving patterns, emerging trends, or shifts in criminal behavior may not be fully captured by historical datasets.

It is important to note that these are just one dataset’s results. The performance of the models may vary depending on the dataset. In the future, the same models can be applied to the crime dataset using more complex classification algorithms, and their prediction performance can be evaluated to find trends and improve topic understanding. Experimenting with different models and hyperparameters is always a good idea to find the best model for your specific needs.

REFERENCES

[1] S. Agarwal, L. Yadav, and M. K. Thakur, “Crime Prediction Based on Statistical Models,” Eleventh International Conference on Contemporary Computing (IC3), pp. 1–3, 2018.

[2] A. Falade, A. Azeta, A. Oni, and I. Odun-Ayo, “Systematic Literature Review of Crime Prediction and Data Mining,” Review of Computer Engineering Studies, pp. 56–63, 2019.

[3] X. Q. Zhang, “The Trends, Promises and Challenges of Urbanisation in the World,” Habitat International, vol. 54, 2015.

[4] S. Stebbins, “The Midwest is home to many of America’s most dangerous cities,” 10 2019. [Online]. Available: https://www.usatoday.com/story/money/2019/ 10/26/crime-rate-higher-us-dangerous-cities/40406541/

[5] W. Safat, S. Asghar, and S. Gilani, “Empirical Analysis for Crime Prediction and Forecasting Using Machine Learning and Deep Learning Techniques,” IEEE Access, pp. 1–1, 2021.

[6] P. Brantingham, M. Valasik, and G. Mohler, “Does Predictive Policing Lead to Biased Arrests? Results From a Randomized Controlled Trial,” Statistics and Public Policy, vol. 5, pp. 11–17, 2018.

[7] A. Stec and D. Klabjan, “Forecasting Crime with Deep Learning,” arXiv, 2018.

[8] J. Fitterer, T. Nelson, and F. Nathoo, “Predictive crime mapping,” Police Practice and Research, vol. 16, 2015.

[9] “Open Government.” [Online]. Available: https://data. gov/open-gov/

[10] Datasf, City, C. O. San, and Francisco, 2003. [Online]. Available: https://data.sfgov.org/Public-Safety/Police-Department-Incident-Reports-Historical-2003/tmnfyvry/about data

[11] I. Pradhan, K. Potika, M. Eirinaki, and P. Potikas, "Exploratory data analysis and crime prediction for smart cities," 23rd International Database Applications & Engineering Symposium, pp. 1–9, 2019.

[12] R. Mohammed and H. Abdulmohsin, "A study on predicting crime rates through machine learning and data mining using text," Journal of Intelligent Systems, vol. 32, 2023.

[13] X. Zhang, L. Liu, L. Xiao, and J. Ji, "Comparison of Machine Learning Algorithms for Predicting Crime Hotspots," IEEE Access, vol. 8, pp. 181 302–181 310, 2020.

[14] A. M. Shermila, A. B. Bellarmine, and N. Santiago, "Crime Data Analysis and Prediction of Perpetrator Identity Using Machine Learning Approach," 2nd International Conference on Trends in Electronics and Informatics (ICOEI), pp. 107–114, 2018.

[15] S. Yao et al., "Prediction of Crime Hotspots based on Spatial Factors of Random Forest," 15th International Conference on Computer Science & Education (ICCSE), pp. 811–815, 2020.

[16] L. Venturini and E. Baralis, "A spectral analysis of crimes in San Francisco," 2nd ACM SIGSPATIAL Workshop on Smart Cities and Urban Analytics, pp. 1–4, 2016.

[17] B. Kochar and R. Chhillar, "An Effective Data Warehousing System for RFID Using Novel Data Cleaning, Data Transformation and Loading Techniques," International Arab Journal of Information Technology, vol. 9, 2012.

[18] R. Addo Danquah, "Handling Imbalanced Data: A Case Study for Binary Classification Problems," figshare, 2020.

[19] T. Yiu, 2023. [Online]. Available: https://towardsdatascience.com/understanding-random- forest-58381e0602d2.