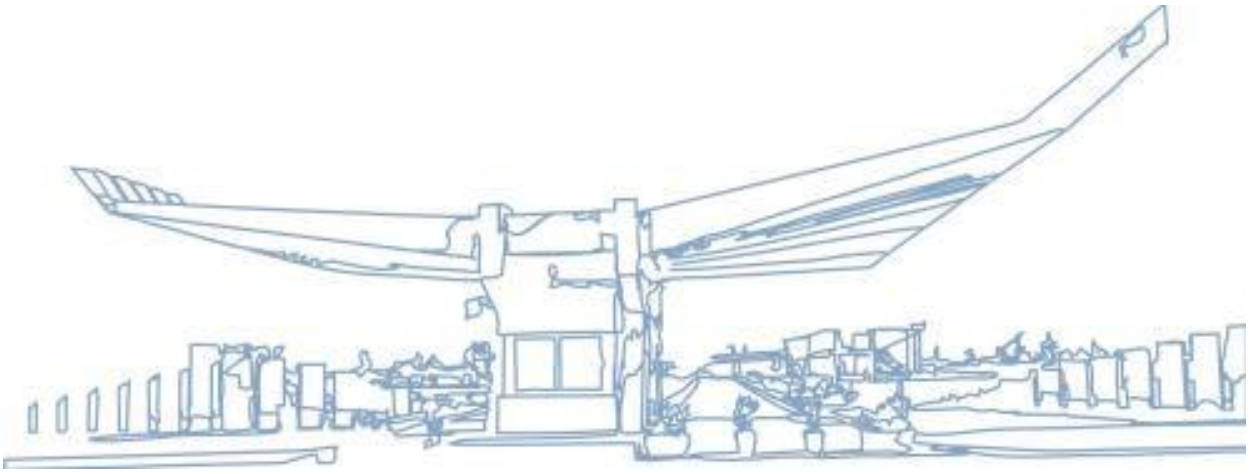
Machine Learning Project

Classifying Violent Crimes and Predicting Area Danger with Machine Learning



Group 1:

Borian Llukaçaj, Engjëll Abazaj, Ermin Lilaj, Indrit Ferati, Joel Bitri, Kristi Samara

EPOKA University

Department of Computer Engineering

Software Engineering

1. Introduction

In recent years, urban safety has become an important issue as cities face challenges in managing crime and ensuring public security. The rise in criminal activities has prompted law enforcement agencies and researchers to leverage data-driven approaches to understand and reduce crime patterns. The Los Angeles Crime Data from 2020 to Present dataset, provided by the Los Angeles Police Department (LAPD), offers a rich dataset containing detailed records of crime incidents, including time, location, and crime type. This dataset is particularly appropriate for applying machine learning techniques to predict crime outcomes and inform policing strategies.

In this study, we focus on two key tasks: a classification task to determine whether a crime is violent and a regression task to assess the dangerousness of specific areas. The classification task involves distinguishing violent crimes, such as assault or robbery, from non-violent crimes, such as theft or vandalism, using a binary classification framework. The regression task aims to predict a continuous danger score for different areas based on factors like crime frequency and severity, enabling a quantitative measure of safety.

Recent analyses of the Los Angeles crime dataset (2020–2023) reveal trends that are valuable for both violent crime classification and area danger score prediction. Vehicle theft is the most common crime, and identity theft has seen a recent surge, often spiking around payment cycles. Crime is concentrated in specific areas, with the Central and 77th Street divisions recording the highest number of incidents, making them critical for spatial risk modeling (Nguyenphantuan, 2023). Temporal patterns indicate higher crime rates on weekends, particularly Fridays and Saturdays, which should be factored into time-based predictions (Pangarego, 2023). Additionally, demographic data shows that victims are most often aged 20–40, with variations across racial groups depending on the district, offering potential features for model input (Nguyenphantuan, 2023).

A study from the World Journal of Advanced Research and Reviews (2025) notes that traditional crime analysis relies heavily on historical data and manual reporting, limiting its predictive power and real-time responsiveness. In contrast, it proposes a system that uses machine learning models like Random Forest Classifier to improve accuracy and enable proactive crime prevention (Sankul, Aruva, Kankal, Arrapogula, & Khan, 2025).

Building on these findings, we will employ a comprehensive set of machine learning and deep learning models, including SVM, kNN with Random Forest Hybrid, XGBoost Classifier, CatBoosting, Decision Tree Classifier, CNN, MLP, ExtraTreesClassifier, AdaBoost Classifier, LSTM, Logistic Regression Classifier and kNN Classifier for the classification task, as well as Random Forest, Hist GradientBoosting, XGBoost, LightGBM and Logistic Regression with Random Forest Hybrid for the regression task. By evaluating these models, we aim to identify the most effective approaches for classifying violent crimes and predicting area dangerousness, contributing to data-driven solutions for enhancing public safety in Los Angeles.

2. Methodology

Initially we downloaded the Los Angeles Crime Data from 2020 to Present dataset from the data catalog of data.gov and it provided us with detailed records of criminal incidents reported by LAPD. As of its latest update on May 3, 2025, the dataset contains 28 features and 1,005,200 rows, capturing information such as crime type, date, time, location, and victim demographics. The dataset was preprocessed to handle missing values, encode categorical variables, and normalize numerical features. The processed dataset was then split into temporal samples, where the data from 2020-2022 was used for training and the data for 2023-2024 for testing the models. Since not enough samples for 2025 were provided at the time of preprocessing to achieve meaningful results, we decided not to take them into consideration.

We generated classification reports for each classification model and kept track of MSE for each regression model tested under various parameter configurations, documenting the results in configuration tables. The best-performing results for each model were compared to identify the most effective model for each particular task.

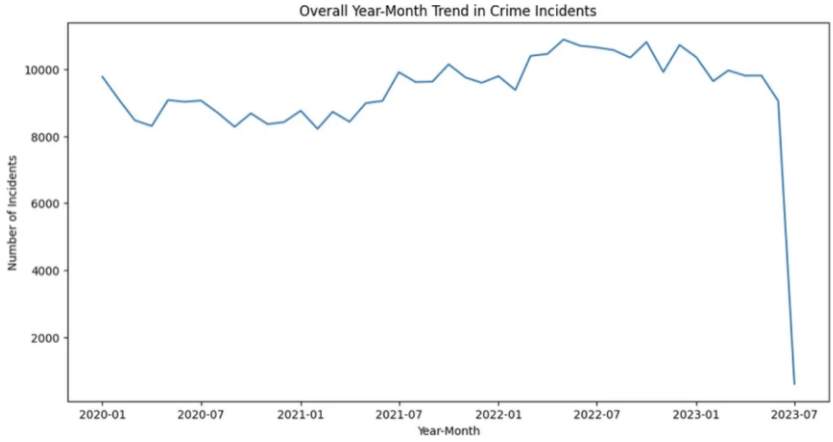


Figure 1: Overall Year-Month Trend in Crime Incidents 2020-2023 (Pangarego, 2023)

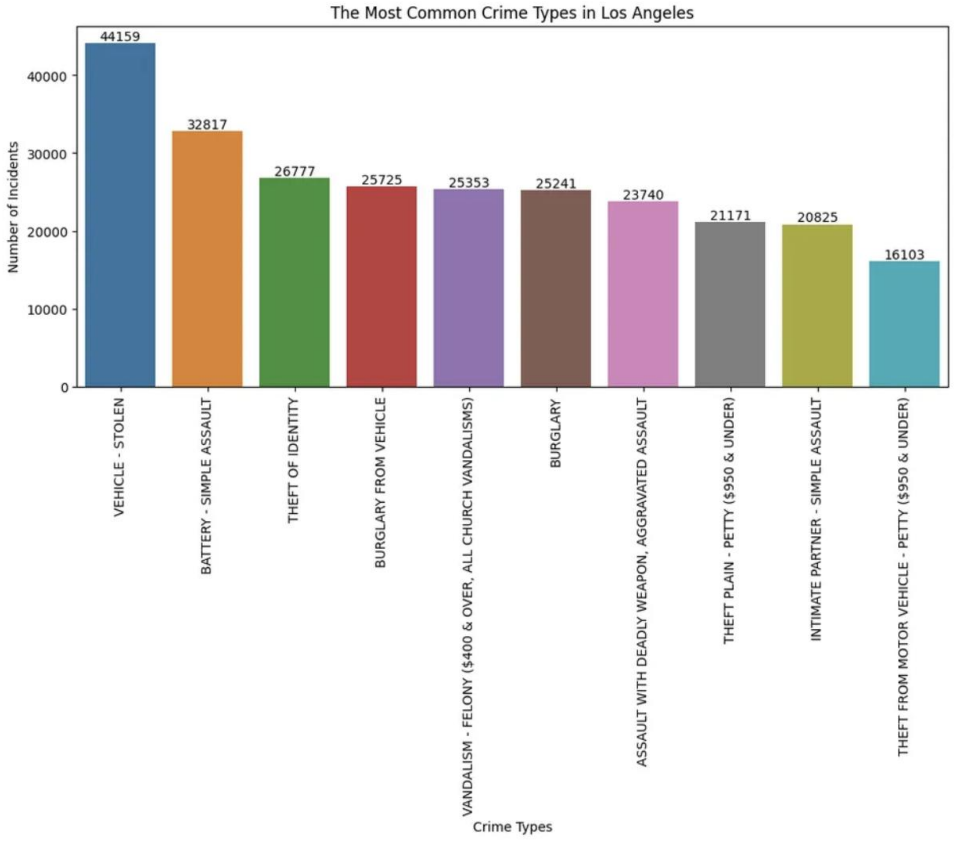


Figure 2: The Most Common Crime Types in Los Angeles (Pangarego, 2023)

3. Data Preprocessing and Normalization

Since the dataset had a large number of records and columns, we automated preprocessing using Python:

1. ***Dataset Loading and Basic Checks:***

We first loaded the dataset and verified its structure, dimensions, and a few initial rows. Summary statistics were also printed for latitude and longitude to assess location data.

2. ***Column Cleanup and Dropping Irrelevant Features:***

A number of columns such as identifiers, descriptions, and redundant crime codes were dropped (e.g., DR_NO, Date Rptd, Status, LOCATION, etc.) to reduce noise and focus on useful features.

3. ***Handling Invalid Coordinates:***

Rows with invalid LAT or LON values (equal to 0) were filtered out, and the number of dropped records was logged. This ensured spatial data quality.

4. ***Handling Missing Values:***

Victim age values of 0 were replaced with NaN and then filled with the median age. Categorical columns such as Vict Sex and Vict Descent were filled with the placeholder "Unknown" where missing.

5. ***Date and Time Formatting:***

The DATE OCC column was parsed into proper datetime format. Rows with invalid or unparseable dates were dropped. The hour was extracted from the TIME OCC field to support temporal analysis.

6. ***Feature Engineering:***

New features were created such as the day of the week from the date and a binary indicator Is Violent for specific violent crime codes. A composite key (district and month) was used to aggregate crime counts and averages.

7. ***Aggregation:***

The dataset was grouped by reporting district and month, computing averages of coordinates, victim age, and time, along with total crime counts and violent crimes per district-month.

8. ***Creating the Target Variable:***

A binary target variable was created where the data was labeled as 0 for non-violent crime and 1 for violent crime.

9. ***Encoding Categorical Features:***

Categorical features like Rpt Dist No and Premis Cd were label-encoded. The mappings were saved as a separate .csv file for reference and reproducibility.

10. ***Feature Scaling:***

Numerical features were standardized and normalized using StandardScaler to ensure all features had zero mean and unit variance before model training.

11. ***Exporting the Processed Data:***

The final dataset was saved as a CSV file (preprocessed_crime_data.csv) and downloaded, along with the encoding file (label_encodings.csv) for model usage.

DR_NO	Date Rptd	DATE OCC	TIME OCC	AREA	AREA NAME	Rpt Dist No	Part 1-2	Crm Cd	Crm Cd Desc	Mocodes	Vict Age	Vict Sex	Vict Descr	Premis Cd	Premis Des	Weapon Us	Weapon Dc	Status	Status Des	Crm Cd 1	Crm Cd 2	Crm Cd 3	Crm Cd 4	LOCATION	Cross Stree	LAT	LON
1.9E+08	3/1/2020 0:00	3/1/2020 0:00	2130	7	Wilshire	784	1	510	VEHICLE - STOLEN		0	M	O	101	STREET			AA	Adult Arres	510	998			1900 S	LONGWOOD	34.0375	-118.351
2E+08	2/9/2020 0:00	2/8/2020 0:00	1800	1	Central	182	1	330	BURGLARY FROM \	1822 1402	47	M	O	128	BUS STOP/LAYOVER (ALSO QUER			IC	Invest Cont	330	998			1000 S	FLOWER	34.0444	-118.263
2E+08	11/11/2020 0:00	11/4/2020 0:00	1700	3	Southwest	356	1	480	BIKE - STOLEN	0344 1251	19	X	X	502	MULTI-UNIT DWELLING (APARTME			IC	Invest Cont	480				1400 W	37TH	34.021	-118.3
2.01E+08	5/10/2023 0:00	3/10/2020 0:00	2037	9	Van Nuys	964	1	343	SHOPLIFTING-GRA	0325 1501	19	M	O	405	CLOTHING STORE			IC	Invest Cont	343				14000	RIVERSIDE	34.1576	-118.439
2E+08	9/9/2020 0:00	9/9/2020 0:00	630	4	Hollenbeck	413	1	510	VEHICLE - STOLEN		0			101	STREET			IC	Invest Cont	510				200 E	AVENUE 28	34.082	-118.213
2E+08	5/3/2020 0:00	5/2/2020 0:00	1800	2	Rampart	245	1	510	VEHICLE - STOLEN		0			101	STREET			IC	Invest Cont	510				2500 W	4TH	34.0642	-118.277
2E+08	7/7/2020 0:00	7/7/2020 0:00	1340	2	Rampart	265	1	648	ARSON	0329 1402	0	X	X	101	STREET			IC	Invest Cont	648	998			JAMES M	W ALVARADO	34.0536	-118.279
2.01E+08	3/27/2020 0:00	3/27/2020 0:00	1210	13	Newton	1333	1	510	VEHICLE - STOLEN		0			101	STREET			IC	Invest Cont	510				3200 S	SAN PEDRO	34.017	-118.264
2.01E+08	7/31/2020 0:00	7/30/2020 0:00	2030	11	Northeast	1161	1	510	VEHICLE - STOLEN		0			101	STREET			AA	Adult Arres	510				KENMORE	FOUNTAIN	34.0953	-118.297
2E+08	12/4/2020 0:00	12/3/2020 0:00	2300	1	Central	105	1	510	VEHICLE - STOLEN		0			101	STREET			IC	Invest Cont	510				400	SOLANO	34.071	-118.23
2.01E+08	6/26/2020 0:00	6/25/2020 0:00	2200	12	77th Street	1259	1	510	VEHICLE - STOLEN		0			101	STREET			IC	Invest Cont	510				FLORENCE	WADSWOF	33.9747	-118.259
2E+08	3/2/2020 0:00	3/1/2020 0:00	1430	4	Hollenbeck	407	1	310	BURGLARY	0344 1607	27	M	W	221	PUBLIC STORAGE			IC	Invest Cont	310				4500	HUNTINGTON D	34.0881	-118.188
2E+08	7/6/2020 0:00	7/4/2020 0:00	100	1	Central	157	1	510	VEHICLE - STOLEN		0			101	STREET			IC	Invest Cont	510				6TH	CROCKER	34.0423	-118.245
2.01E+08	7/20/2020 0:00	7/20/2020 0:00	700	9	Van Nuys	937	1	510	VEHICLE - STOLEN		0			101	STREET			IC	Invest Cont	510				BESSEMER	RANCHITO	34.1812	-118.436
2E+08	3/19/2020 0:00	3/19/2020 0:00	130	2	Rampart	236	1	510	VEHICLE - STOLEN		0			101	STREET			IC	Invest Cont	510				1900	MIRAMAR	34.0632	-118.269
2E+08	6/1/2020 0:00	5/30/2020 0:00	30	1	Central	153	1	310	BURGLARY	1609 0344	0	M	W	201	JEWELRY STORE			AA	Adult Arres	310				600 S	BROADWAY	34.0467	-118.252
2.01E+08	2/8/2020 0:00	2/7/2020 0:00	2000	9	Van Nuys	939	1	510	VEHICLE - STOLEN		0			101	STREET			AO	Adult Othe	510				AETNA	ATOLL	34.1803	-118.42
2E+08	9/26/2020 0:00	9/25/2020 0:00	1900	2	Rampart	237	1	510	VEHICLE - STOLEN		0			101	STREET			IC	Invest Cont	510				1600 W	COURT	34.0668	-118.263
2E+08	3/7/2020 0:00	3/6/2020 0:00	2200	3	Southwest	317	1	510	VEHICLE - STOLEN		0			101	STREET			IC	Invest Cont	510				VERMONT	24TH	34.0344	-118.292
2.01E+08	2/12/2020 0:00	2/7/2020 0:00	1615	6	Hollywood	646	2	805	PIMPING	1300 1402	23	F	H	101	STREET			AA	Adult Arres	805	998			HOLLYWO	MC CADDE	34.1016	-118.337
2.01E+08	12/7/2020 0:00	##########	2255	5	Harbor	541	1	510	VEHICLE - STOLEN		0			101	STREET			IC	Invest Cont	510				1400	BRETT	33.7553	-118.306
2.01E+08	3/10/2020 0:00	2/22/2020 0:00	30	5	Harbor	511	1	510	VEHICLE - STOLEN		0			101	STREET			AA	Adult Arres	510				1600 W	PACIFIC COAS	33.7926	-118.304
2.01E+08	6/11/2020 0:00	6/8/2020 0:00	2000	11	Northeast	1118	1	310	BURGLARY	0344 1607	0	M	O	203	OTHER BUSINESS			IC	Invest Cont	310				100	AVENUE 56	34.1089	-118.194
2.01E+08	7/14/2020 0:00	7/13/2020 0:00	2000	10	West Valley	1043	1	330	BURGLARY FROM \	0344 1602	41	M	W	108	PARKING LOT			AA	Adult Arres	330	998			18600	HATTERAS	34.1774	-118.539
2.01E+08	6/8/2020 0:00	6/8/2020 0:00	330	5	Harbor	587	1	510	VEHICLE - STOLEN		0			101	STREET			IC	Invest Cont	510				2900 S	KERCKHOFF	33.7179	-118.29
2E+08	6/24/2020 0:00	6/24/2020 0:00	700	4	Hollenbeck	413	1	510	VEHICLE - STOLEN		0			101	STREET			IC	Invest Cont	510				300 E	AVENUE 31	34.0809	-118.213
2.01E+08	9/24/2020 0:00	9/24/2020 0:00	1710	8	West LA	898	1	510	VEHICLE - STOLEN		0			101	STREET			IC	Invest Cont	510				ROBERTSC	GIBSON	34.0375	-118.389
2.01E+08	6/29/2020 0:00	6/29/2020 0:00	730	12	77th Street	1263	1	510	VEHICLE - STOLEN		0			101	STREET			IC	Invest Cont	510				81ST	DENKER	33.9655	-118.305
2E+08	5/14/2020 0:00	5/12/2020 0:00	2100	2	Rampart	266	1	510	VEHICLE - STOLEN		0			101	STREET			IC	Invest Cont	510				700 S	UNION	34.0531	-118.274
2.01E+08	9/27/2020 0:00	9/22/2020 0:00	1430	7	Wilshire	702	1	510	VEHICLE - STOLEN		0			101	STREET			IC	Invest Cont	510				800 N	FULLER	34.0853	-118.35
2.01E+08	2/14/2020 0:00	2/13/2020 0:00	2230	9	Van Nuys	914	2	806	PANDERING	0908 1212	0	F	X	101	STREET			AO	Adult Othe	806	998			SEPULVED,	BASSETT	34.1958	-118.466
2E+08	12/11/2020 0:00	##########	2000	2	Rampart	271	1	510	VEHICLE - STOLEN		0			101	STREET			IC	Invest Cont	510				900 S	ALVARADO	34.0529	-118.282
2.01E+08	11/27/2020 0:00	##########	950	6	Hollywood	642	1	510	VEHICLE - STOLEN		0			101	STREET			IC	Invest Cont	510				ORANGE	SUNSET	34.0981	-118.358
2E+08	8/30/2020 0:00	8/29/2020 0:00	2050	3	Southwest	333	1	510	VEHICLE - STOLEN		0			101	STREET			AO	Adult Othe	510				JEFFERSOI	HARCOUR	34.0278	-118.338
2.01E+08	1/1/2020 0:00	1/1/2020 0:00	2100	5	Harbor	526	2	946	OTHER MISCELLAN	1501 0906	0	X	X	101	STREET			AA	Adult Arres	946				G	ISLAND	33.7783	-118.266
2.01E+08	4/25/2020 0:00	4/21/2020 0:00	1400	8	West LA	858	1	510	VEHICLE - STOLEN		0			101	STREET			IC	Invest Cont	510				8800 W	PICO	34.0548	-118.384
2.01E+08	6/17/2020 0:00	6/16/2020 0:00	1600	8	West LA	829	1	510	VEHICLE - STOLEN		0			101	STREET			IC	Invest Cont	510				1300	WOODRUFF	34.0633	-118.429
2.01E+08	12/26/2020 0:00	##########	332	11	Northeast	1169	1	510	VEHICLE - STOLEN		0			101	STREET			IC	Invest Cont	510				200 S	AVENUE 51	34.1038	-118.199
2.01E+08	6/25/2020 0:00	6/25/2020 0:00	230	5	Harbor	563	1	510	VEHICLE - STOLEN		0			108	PARKING LOT			IC	Invest Cont	510				14TH	PACIFIC	33.7315	-118.288
2.01E+08	3/22/2020 0:00	3/21/2020 0:00	2000	11	Northeast	1137	1	510	VEHICLE - STOLEN		0			101	STREET			IC	Invest Cont	510				700 N	AVENUE 50	34.1109	-118.202
2E+08	10/19/2020 0:00	##########	1800	1	Central	159	1	510	VEHICLE - STOLEN		0			108	PARKING LOT			IC	Invest Cont	510				1800	CONWAY	34.0366	-118.233

Figure 3: Original Dataset

	Rpt Dist No	LAT	LON	TIME OCC	Day of Week	Is Violent	Vict Age	Premis Cd	Crime Count	DATE OCC	Target
2	0	1.0640688	-1.70974	-0.55105	-0.180741403	-0.58894	-0.71143	21	0.9619996963944251	1/6/2020	1
3	0	1.0701052	-1.71018	-0.37879	0.39344934445341584	0.3796930	-0.4127	66	1.3366051252429192	2/12/2020	1
4	0	1.0574760	-1.7082	0.1633812	0.39344934445341584	-0.26606	-0.13163	119	0.21278883869743673	3/12/2020	0
5	0	1.0577368	-1.72284	0.2876299	-0.37546696	-0.91181	-0.10102	163	0.5873942675459309	4/19/2020	0
6	0	1.0596961	-1.73098	-1.66485	0.565706569	-0.26606	0.3482523	119	0.4375520960065333	5/6/2020	0
7	0	1.0603501	-1.71793	-0.6299	-0.240368903	1.6711990	-0.18577	101	0.8121575248550275	6/1/2020	1
8	0	1.0502489	-1.73032	0.1219650	0.058504742	0.3796930	0.7950359	0	0.21278883869743673	7/16/2020	0
9	0	1.0615029	-1.72271	0.3497542	-0.20945094	1.0254460	-0.10712	0	0.6623153533156297	8/2/2020	1
10	0	1.0669232	-1.72225	0.7908371	-0.008484178	-0.26606	0.8912662	7	0.3626310102368344	9/20/2020	0
11	0	1.0526579	-1.7289	0.9917058	0.5944161058900669	-0.58894	-0.15836	119	0.21278883869743673	10/20/2020	0
12	0	1.0872228	-1.7051	-1.5761	-0.611384463	-0.58894	-1.06943	120	-0.536422019	11/11/2020	0
13	0	1.0545606	-1.69946	0.2578102	-1.021356656	0.3796930	0.1310467	0	0.7372364390853285	12/24/2020	1
14	0	1.0551209	-1.70916	-0.88386	0.565706569	0.3796930	-0.40784	119	0.4375520960065333	1/11/2021	0
15	0	1.0505795	-1.71777	0.1840893	-0.27643986	-0.58894	-0.11381	119	0.21278883869743673	2/19/2021	0
16	0	1.0516742	-1.71001	0.2228045	0.043941933	0.056817	-0.49152	119	0.5873942675459309	3/7/2021	0
17	0	1.0512999	-1.73488	0.2643332	0.8205037125062995	-0.58894	-0.42789	119	0.062946667	4/17/2021	0
18	0	1.0549725	-1.7039	0.6810841	1.1303274697211372	-0.26606	-0.38111	102	0.21278883869743673	5/12/2021	0
19	0	1.0433052	-1.70941	1.2816194	0.862371788	0.056817	-1.20976	49	0.21278883869743673	6/13/2021	0
20	0	1.0614911	-1.69917	0.4256840	0.4381086247726713	0.056817	-0.20588	20	0.8870786106247263	7/1/2021	1
21	0	1.0618500	-1.71842	-0.45786	0.7953828673267179	-0.26606	0.3726917	163	1.1118418679338227	8/20/2021	1
22	0	1.0586839	-1.7122	0.074632	-0.640094	-0.26606	0.1955057	0	0.4375520960065333	9/13/2021	0
23	0	1.0570391	-1.73135	-0.66753	0.8958662480450438	-0.58894	0.1935964	139	0.062946667	10/8/2021	0

Figure 4: Dataset After Preprocessing

4. Classification Results

This section explains how each model works, and the classification report parameters are shown in each respective table.

1. Support Vector Machines (SVM)

Support Vector Machine (SVM) is a robust classification model known for its ability to handle high-dimensional data and define clear decision boundaries. The core strength of SVM lies in its use of kernel functions, which project data into higher-dimensional spaces where it becomes easier to separate classes that are not linearly separable in the original space. Common kernel types include linear, polynomial, radial basis function (RBF), and sigmoid, each offering varying capacities for capturing relationships between features. The choice of kernel can greatly influence the model's performance depending on the complexity and structure of the dataset.

In the context of predicting crime categories, the SVM model was tested with all four aforementioned kernels. The linear and polynomial kernels both achieved 84% overall accuracy but completely failed to detect the minority class (class 1), with zero recall and F1-score, suggesting an inability to capture complex class boundaries likely due to class imbalance. The RBF kernel, while maintaining the same overall accuracy, significantly improved minority class detection with a precision of 0.51 and an F1-score of 0.26, making it the best-performing kernel for this task. Conversely, the sigmoid kernel underperformed across the board, yielding only 65% accuracy and failing to generalize effectively. These results underline the RBF kernel's superior handling of non-linear relationships but also highlight the need for additional strategies—like resampling, feature engineering, or ensemble methods—to address class imbalance and enhance fairness in classification outcomes.

SVM	**1**	**2**	**3**	**4**
kernel	**linear**	**sigmoid**	**rbf**	**poly**
gamma	**scale**	**scale**	**scale**	**scale**
C	**1**	**1**	**1**	**1**
Class Weight	**balanced**	**balanced**	**balanced**	**balanced**
precision	**0.84,** **0**	**0.82,** **0.09**	**0.86,** **0.51**	**0.84,** **0**

recall	**1.00, 0**	**0.75, 0.13**	**0.97, 0.18**	**1.00, 0**
f1-score	**0.91, 0**	**0.78, 0.10**	**0.91, 0.26**	**0.91, 0**
accuracy	**0.84 (decision boundary out of bounds)**	**0.65**	**0.84**	**0.84 (overfitting)**

Table 1: SVM Results

2. kNN with Random Forest Hybrid

In the crime classification study, a hybrid ensemble model was developed by combining K-Nearest Neighbors (KNN) and Random Forest classifiers to leverage the strengths of both algorithms. The ensemble utilized both hard and soft voting strategies to aggregate predictions, where hard voting selects the majority vote from individual models, and soft voting averages predicted probabilities for a more nuanced decision. Three different configurations of the ensemble were evaluated, each varying key parameters such as the number of KNN neighbors, the number of trees in the Random Forest, and the voting mechanism used. The goal was to assess how these configurations impacted the model's robustness, adaptability, and ability to handle class imbalance.

Across the three configurations, the ensemble model showed progressive improvement, particularly in handling the minority class. Model 1, using hard voting with 5 KNN neighbors and 100 Random Forest estimators, achieved an accuracy of 86.57% but struggled with the minority class, reaching only a 0.34 F1-score. Model 2 increased KNN neighbors to 7 and Random Forest estimators to 200, slightly improving the minority class F1-score to 0.47 and overall accuracy to 87.69%. However, the most significant improvement came with Model 3, which switched to soft voting while returning to 5 neighbors and 100 estimators. This configuration achieved the highest overall accuracy of 88.13% and considerably boosted minority class performance, with a precision of 0.65, recall of 0.53, and an F1-score of 0.58. These results highlight how soft voting can effectively mitigate class imbalance, making Model 3 the most balanced and practical choice for real-world crime classification tasks. Further optimization, like tuning class weights or incorporating additional diverse classifiers, could enhance performance even more.

Hybrid Model	1	2	3
KNN (n_neighbors)	5	7	5
RF (n_estimators)	100	200	100
Voting Type	hard	hard	soft
Precision	0.87, 0.74	0.89, 0.72	0.91, 0.65
Recall	0.99, 0.22	0.97, 0.35	0.95, 0.53
F1-Score	0.93, 0.34	0.93, 0.47	0.93, 0.58
Accuracy	0.86574	0.87686	0.88130

Table 2: kNN with Random Forest Hybrid Results

3. XGBoost Classifier

XGBoost (Extreme Gradient Boosting) is a powerful ensemble algorithm that builds a strong classifier by combining multiple decision trees using gradient boosting. It trains models in sequence, each correcting the errors of the previous one by optimizing a loss function. XGBoost incorporates regularization and system-level optimizations for speed and performance. Key parameters include max_depth (controls tree complexity), eta (learning rate, lower values improve generalization), subsample (fraction of training samples per tree), and colsample_bytree (fraction of features used per tree). These were tuned across 10 models to classify violent crimes using time-split data (2020–2022 for training, 2023–2024 for testing).

Model 8, with max_depth=8, eta=0.01, subsample=0.8, and colsample_bytree=0.8, achieved the best performance:

- Test accuracy: 0.8976
- F1-score for Class 1 (minority): 0.658
- Precision: 0.691, Recall: 0.628

This configuration balanced precision and recall for minority class detection, showing strong generalization and robustness to overfitting. XGBoost consistently delivered high performance across all splits, with macro-average F1-scores above 0.79, making it well-suited for imbalanced classification tasks.

Model	Objective	Eval Metric	Max Depth	Eta	Subsample	Colsample_bytree	Gamma	Lambda	Accuracy	F1 Score (Class 1)	Precision (Class 1)	Recall (Class 1)
Model 1	binary:logistic	logloss	3	0.1	0.8	0.8	0	1	0.892342	0.646978	0.666839	0.628265
Model 2	binary:logistic	logloss	4	0.05	0.9	0.9	0	1	0.895639	0.660685	0.674898	0.647059
Model 3	binary:logistic	logloss	5	0.3	0.7	0.7	0	1	0.889353	0.644852	0.65005	0.639736
Model 4	binary:logistic	logloss	6	0.2	1	1	0	1	0.895524	0.662791	0.671934	0.653893
Model 5	binary:logistic	logloss	3	0.15	0.85	0.85	0	1	0.893071	0.650025	0.668645	0.632414
Model 6	binary:logistic	logloss	4	0.25	0.95	0.75	0	1	0.892534	0.650025	0.665134	0.635587
Model 7	binary:logistic	logloss	7	0.05	0.6	0.6	0	1	0.895792	0.651634	0.685814	0.620698
Model 8	binary:logistic	logloss	8	0.01	0.8	0.8	0	1	0.897593	0.658225	0.691481	0.628021
Model 9	binary:logistic	logloss	6	0.2	0.9	0.8	0.1	1	0.893377	0.653117	0.667601	0.639248
Model 10	binary:logistic	logloss	5	0.3	1	1	0.2	1.5	0.893607	0.657284	0.665001	0.649744

Table 3: XGBoost Classifier Results

4. CatBoost Classifier

CatBoost (Categorical Boosting) is a gradient boosting algorithm that excels in handling categorical features natively and demonstrates strong performance on imbalanced datasets. It builds an ensemble of decision trees, optimizing a loss function while applying advanced techniques like ordered boosting and class weighting. Key parameters include iterations (number of trees), learning_rate (controls each tree's contribution), depth (tree complexity), and l2_leaf_reg (regularization strength). Four distinct hyperparameter configurations were tested to classify violent crimes.

Model 10, with iterations=400, learning_rate=0.1, depth=8, and l2_leaf_reg=5, delivered the best performance:

- Test accuracy: 0.8982
- F1-score for Class 1 (minority): 0.658

This setup balanced depth and regularization effectively while leveraging class weighting to enhance recall. CatBoost consistently achieved high macro-averaged and weighted precision and recall scores. Its minority class detection was reliable and closely matched the performance of the top XGBoost model, making it a strong contender in scenarios involving imbalanced data and categorical variables.

Model	Iterations	Learning Rate	Depth	L2 Leaf Reg	Accuracy	F1 Score (Class 1)	Precision (Class 1)	Recall (Class 1)
1	300	0.1	4	3	0.883911	0.57416	0.677056	0.498413
2	500	0.05	6	5	0.890694	0.608886	0.694836	0.54186
3	400	0.15	3	2	0.884984	0.581975	0.677807	0.509885
4	600	0.08	5	4	0.891461	0.612691	0.696734	0.546742
5	350	0.2	4	6	0.89445	0.636484	0.693015	0.588479
6	450	0.07	7	1	0.892764	0.622198	0.696283	0.562363
7	500	0.03	6	3	0.852445	0.205858	0.664447	0.121796
8	550	0.09	5	4	0.895409	0.638016	0.698722	0.587015
9	300	0.25	3	2	0.888165	0.610726	0.673433	0.558701
10	400	0.1	8	5	0.898168	0.658439	0.695546	0.625092

Table 4: CatBoost Classifier Results

5. *Decision Tree Classifier*

Eight different configurations of the Decision Tree were evaluated, varying the criterion ('gini' or 'entropy') and the minimum impurity decrease (0.0, 0.005, 0.01, 0.02). Below is a summary of the performance metrics derived from classification reports

The evaluation of eight different Decision Tree classifiers revealed consistently high overall accuracy, ranging between 83% and 84%. While this may initially seem promising, a closer inspection of the individual class performance metrics shows a significant imbalance in predictive power between the two classes. The dataset itself is imbalanced, with class 0 (the majority class) making up the bulk of instances, and class 1 (the minority class) forming a much smaller portion. As a result, the classifiers tend to favor class 0, and this is reflected in the metrics.

In conclusion, while all models achieve similar overall accuracy, their effectiveness in identifying the minority class varies widely depending on their parameter configuration. Models with low min_impurity_decrease (0.0) allow more flexible tree growth, leading to better recall for underrepresented classes. Conversely, increasing this parameter to 0.02 restricts the model's ability to split, hurting recall and making the model less sensitive to minority class patterns. Therefore, for tasks requiring reliable detection of minority class instances, less restrictive decision tree settings are preferable, despite the slight risk of overfitting.

Model	Criterion	Min Impurity Decrease
Decision Tree 1	**gini**	**0.0**
Decision Tree 2	**gini**	**0.005**
Decision Tree 3	**gini**	**0.01**
Decision Tree 4	**gini**	**0.02**
Decision Tree 5	**entropy**	**0.0**
Decision Tree 6	**entropy**	**0.005**
Decision Tree 7	**entropy**	**0.01**
Decision Tree 8	**entropy**	**0.02**

Table 5: Decision Tree Impurity

Model	Accuracy	Precision (Class 0)	Recall (Class 0)	F1-score (Class 0)	Precision (Class 1)	Recall (Class 1)	F1-score (Class 1)	Macro Avg F1	Weighted Avg F1
Decision Tree 1	0.84	0.93	0.88	0.9	0.5	0.65	0.57	0.73	0.85
Decision Tree 2	0.84	0.89	0.93	0.91	0.5	0.38	0.43	0.67	0.83
Decision Tree 3	0.83	0.89	0.91	0.9	0.46	0.39	0.42	0.66	0.82
Decision Tree 4	0.83	0.87	0.93	0.9	0.45	0.29	0.35	0.63	0.81
Decision Tree 5	0.84	0.93	0.88	0.9	0.52	0.68	0.59	0.75	0.85
Decision Tree 6	0.84	0.88	0.93	0.91	0.51	0.37	0.43	0.67	0.83
Decision Tree 7	0.83	0.89	0.91	0.9	0.46	0.39	0.42	0.66	0.82
Decision Tree 8	0.83	0.87	0.93	0.9	0.45	0.29	0.35	0.63	0.81

Table 6: Decision Treet Classifier Results

6. Convolutional Neural Network (CNN)

Convolutional Neural Networks (CNNs) are a class of deep learning models particularly effective for analyzing spatial and visual data. While traditionally used for image processing, they are also suitable for structured datasets where local patterns are meaningful. This report evaluates the performance of various CNN architectures in a binary classification task on a crime-related dataset.

Architecture 1:

- Input layer

- 1D Convolutional layer with 32 filters, kernel size 3

- Customizable activation (e.g., ReLU, Swish)

- MaxPooling layer

- Flattened output passed to a Dense layer

- Output activation: customizable (e.g., Sigmoid, Softmax, Tanh)

Architecture 2:

- Input layer

- 1D Convolutional layer with 64 filters

- MaxPooling and Dropout layer (rate = 0.3)

- Flattened output to Dense layer

- Customizable activation functions

Architecture 3:

- Input layer

- Multiple Conv1D layers with 64 and 128 filters

- BatchNormalization layers

- MaxPooling layer

- Dense output layer after flattening

- Customizable activations

The results highlight the influence of architecture complexity and activation functions on performance. Architecture 1 with softmax output and ReLU hidden activation achieved the highest accuracy (88.2%). Models using tanh in the output layer performed poorly, especially for the majority class.

Architecture	Output Activation	Hidden Activation	Accuracy	F1-Score (Weighted Avg)
1	Sigmoid	relu	0.88118964	0.87
2	Sigmoid	relu	0.87969493	0.87
3	Sigmoid	relu	0.87287291	0.87
1	Sigmoid	swish	0.87934999	0.87
2	Sigmoid	swish	0.87858347	0.87
3	Sigmoid	swish	0.87544075	0.87
1	Softmax	relu	0.88226276	0.87
2	Softmax	relu	0.88161122	0.87
3	Softmax	relu	0.86524605	0.86
1	Softmax	swish	0.87739537	0.87

2	softmax	swish	0.87843017	0.87
3	softmax	swish	0.87551740	0.87
1	tanh	relu	0.56415760	0.62
2	tanh	relu	0.84297869	0.77
3	tanh	relu	0.39475701	0.44
1	tanh	swish	0.18622566	0.11
2	tanh	swish	0.51697838	0.58
3	tanh	swish	0.82431397	0.82

Table 7: CNN Results

7. Multilayer Perceptron (MLP)

A Multilayer Perceptron (MLP) Classifier is a type of feedforward neural network used for supervised classification tasks. It consists of an input layer, one or more hidden layers with non-linear activation functions (like ReLU or tanh), and an output layer that uses sigmoid or softmax for binary or multi-class classification, respectively. The MLP learns through forward propagation, where input data flows through the network to produce predictions, and backpropagation, where errors from the output are used to update weights via gradient descent with optimizers like SGD or Adam. This allows the MLP to learn complex, non-linear patterns in the data.

To determine the best activation function for the MLP, we evaluated ReLU, Identity, Tanh, and Logistic using a dataset split into the first three years for training and the last two for testing. We tested various architectures, including a single hidden layer with 100 neurons, three hidden layers with 200 neurons each, and two hidden layers with 200 neurons each, recording the best results. ReLU consistently outperformed the other functions across all setups. A final test with three hidden layers (300 neurons in the first two, 200 in the last) where a accuracy was reached, but there was also a higher computational cost reached as well. Ultimately, we identified ReLU with a single hidden layer of 300 neurons as the optimal architecture, balancing accuracy and computational efficiency.

MLP Classifier

Model ID	Activation	Solver	Neurons	Accuracy	Precision	Recall	F1 Score
1	logistic	adam	200	0.876782	0.59	0.71	0.65
2	logistic	adam	300	0.880193	0.61	0.66	0.63
3	logistic	adam	800	0.879657	0.61	0.66	0.63
4	identity	adam	200	0.878507	0.6	0.68	0.64
5	identity	adam	300	0.877165	0.59	0.69	0.65
6	identity	adam	800	0.876552	0.59	0.68	0.64
7	Tahn	adam	200	0.875977	0.59	0.7	0.64
8	Tahn	adam	300	0.876591	0.56	0.69	0.64
9	Tahn	adam	800	0.878009	0.6	0.67	0.63
10	Relu	adam	200	0.888587	0.63	0.71	0.67
11	Relu	adam	300	0.890235	0.64	0.73	0.68
12	Relu	adam	800	0.888511	0.63	0.7	0.66

Table 8: MLP Results

8. ExtraTrees Classifier

The Extra Trees Classifier (Extremely Randomized Trees) is an ensemble learning method that builds multiple decision trees with added randomness to improve generalization and reduce overfitting. Unlike Random Forests, which find the best feature splits, Extra Trees randomly select both features and split points, then choose the best among these random splits. Each tree contributes a class prediction, and the final output is determined by majority voting. This approach speeds up training, performs well on high-dimensional data, and is less prone to overfitting, though it may be less interpretable and can underfit due to its high randomness.

To determine the optimal configuration for the ExtraTreesClassifier, we evaluated various setups using a dataset split into the first three years for training and the last two for testing. We tested combinations of 50 to 300 trees, max features settings like ‘log2’, ‘sqrt’, and 0.8, with min samples split ranging from 2 to 10, min samples leaf from 1 to 5, and max depths from 10 to 20 or None, recording the best outcomes. The configuration with 100 trees, max features at 0.8, and no maximum depth achieved the highest performance, yielding an accuracy of 89.01% and an F1-score of 0.67. A further test with 300 trees and the same max features setting improved the recall to 0.84, but the computational cost was prohibitive. Ultimately, we identified 100 trees with max features of 0.8 and no depth limit as the optimal setup, balancing accuracy and efficiency.

ExtraTreesClassifier

Model ID	Estimators	max_features	Min Samples split	Min samples leaf	Max depth	Accuracy	Precision	Recall	F1 Score
1	200	log2	5	3	5	0.836312	0.49	0.7	0.57
2	100	log2	5	3	15	0.842979	0.53	0.68	0.59
3	50	log2	5	3	10	0.696497	0.5	0.73	0.61
4	100	log2	10	5	None	0.872183	0.57	0.77	0.65
5	300	log2	5	3	10	0.736279	0.46	0.84	0.61
6	100	sqrt	2	1	10	0.842185	0.56	0.72	0.68
7	100	sqrt	5	3	20	0.780163	0.44	0.73	0.59
8	200	log2	10	5	10	0.853442	0.53	0.69	0.6
9	100	0.8	5	3	None	0.898015	0.68	0.65	0.67
10	300	0.8	10	5	10	0.733673	0.45	0.84	0.6

Table 9: ExtraTrees Classifier Results

9. AdaBoost Classifier

AdaBoost (Adaptive Boosting) is an ensemble algorithm that builds a strong classifier by combining multiple weak learners, typically decision trees. It iteratively trains models, increasing the weight of misclassified instances to focus on errors. Predictions are aggregated via weighted voting for classification. The notebook's key parameters are n_estimators (number of trees), learning_rate (each tree's contribution, lower values enhance robustness), and max_depth (tree complexity, higher risks overfitting). These are tuned across 10 models to classify violent crimes.

Model 8, with 180 estimators, a 0.15 learning rate, and max depth of 4, achieves the highest test accuracy of 0.8516. It excels at classifying non-violent crimes but has low recall for violent crimes. The classifier is quite sensitive to disbalance, thus some results of precision being 0. However, it still manages to predict a good result.

Model	n_estimato	learning_ra	max_depth	Test_Accuracy	Precision_0	Recall_0	F1_0	Precision_1	Recall_1	F1_1	Macro_Precision	Macro_Recall	Macro_F1	Weighted_Precision	Weighted_Recall	Weighted_F1
1	50	1	1	0.842979	0.842979	1	0.9148	0	0	0	0.421489	0.5	0.4574	0.710613	0.842979	0.771157
2	100	0.5	2	0.847999	0.861346	0.976949	0.915513	0.557205	0.155724	0.243419	0.709276	0.566337	0.57947	0.813589	0.847999	0.80998
3	150	0.1	3	0.848728	0.851758	0.993453	0.917165	0.671233	0.07176	0.129658	0.761495	0.532606	0.52341	0.823412	0.848728	0.79351
4	200	0.05	4	0.848574	0.851051	0.994408	0.917161	0.686224	0.065658	0.119849	0.768638	0.530033	0.51851	0.825169	0.848574	0.791966
5	75	0.3	2	0.842979	0.842979	1	0.9148	0	0	0	0.421489	0.5	0.4574	0.710613	0.842979	0.771157
6	120	0.2	3	0.849187	0.853425	0.991362	0.917236	0.649446	0.085917	0.151757	0.751436	0.538639	0.5345	0.821396	0.849187	0.79704
7	80	0.8	1	0.842979	0.842979	1	0.9148	0	0	0	0.421489	0.5	0.4574	0.710613	0.842979	0.771157
8	180	0.15	4	0.85164	0.856968	0.989088	0.918301	0.660057	0.113742	0.194045	0.758513	0.551415	0.55617	0.826049	0.85164	0.804577
9	60	0.4	2	0.842979	0.842979	1	0.9148	0	0	0	0.421489	0.5	0.4574	0.710613	0.842979	0.771157
10	140	0.25	3	0.851027	0.855924	0.989907	0.918053	0.66055	0.105443	0.181856	0.758237	0.547675	0.54996	0.825246	0.851027	0.802454

Table 10:AdaBoost Classifier Results

10. Long-short Term Memory (LSTM) Classifier

Long Short-Term Memory (LSTM) is a recurrent neural network tailored for sequential data, leveraging memory cells to capture long-term dependencies while addressing vanishing gradient challenges. It processes tabular crime data, formatted as sequences, to classify crimes as violent or non-violent. Key parameters include hidden_size (number of hidden units, defining model capacity), num_layers (stacked LSTM layers, increasing complexity), learning_rate (optimization step size, balancing convergence speed and stability), batch_size (samples per

training iteration, influencing training efficiency), dropout (regularization to prevent overfitting), and weight_decay (L2 regularization to manage model complexity). These are tuned across 10 configurations to maximize classification accuracy.

Model 1, with 64 hidden units, 2 layers, and a 0.001 learning rate, achieves the highest accuracy of 0.8249, tied with Model 2. The model's well-balanced architecture, with a moderate hidden size and multi-layered design, ensures efficient learning and robust generalization. Its carefully chosen learning rate supports stable and effective training, making it a strong performer for the classification task. Model 1's impressive accuracy and solid performance position it as a highly capable solution, with potential for even greater refinement through techniques like class weighting to further enhance its versatility.

Model	Hidden_Size	Num_Layers	Learning_Rate	Batch_Size	Dropout	Weight_Decay	Accuracy	Precision_0	Recall_0	F1_0	Precision_1	Recall_1	F1_1	Macro_Precision	Macro_Recall	Macro_F1	Weighted_	Weighted_Recall	Weighted_F1
1	64	2	0.001	64	0.3	0	0.824894	0.82847	0.990652	0.902331	0.677316	0.087315	0.154688	0.752893	0.538983	0.52851	0.800734	0.824894	0.765143
2	128	1	0.0005	32	0.2	0	0.824894	0.828877	0.989911	0.902265	0.668693	0.09061	0.159594	0.748785	0.54026	0.530929	0.799484	0.824894	0.765989
3	256	2	0.0001	128	0.4	0	0.819302	0.8203	0.997131	0.900113	0.686869	0.028007	0.053819	0.753584	0.512569	0.476966	0.795816	0.819302	0.744822
4	64	3	0.002	64	0.3	0.001	0.816505	0.816505	1	0.898985	0	0	0	0.408253	0.5	0.449492	0.666681	0.816505	0.734026
5	32	1	0.005	16	0.1	0	0.816505	0.816553	0.999907	0.898976	0.5	0.000412	0.000823	0.658277	0.50016	0.4499	0.758467	0.816505	0.73417
6	128	2	0.0002	256	0.5	0	0.818319	0.819829	0.996483	0.899566	0.62	0.025535	0.049051	0.719915	0.511009	0.474308	0.783162	0.818319	0.743501
7	512	1	0.0001	64	0.3	0	0.823761	0.831819	0.982877	0.901061	0.603004	0.115733	0.194195	0.717412	0.549305	0.547628	0.789833	0.823761	0.771355
8	64	2	0.001	32	0.2	0.0001	0.816505	0.816505	1	0.898985	0	0	0	0.408253	0.5	0.449492	0.666681	0.816505	0.734026
9	256	3	0.0005	128	0.4	0	0.821644	0.845612	0.956127	0.89748	0.533465	0.223229	0.31475	0.689538	0.589678	0.606115	0.788335	0.821644	0.790553
10	128	1	0.002	64	0.3	0	0.817337	0.817762	0.998889	0.899296	0.657143	0.009473	0.018676	0.737452	0.504181	0.458986	0.788289	0.817337	0.737707

Table 11: LSTM Classifier Results

11. Logistic Regression Classifier

To assess the likelihood of a crime being violent, we developed a classification model using five configurations of Logistic Regression, each with different regularization techniques and hyperparameters. The dataset included various spatial and temporal features such as district number, location coordinates, time of occurrence, and premise codes. After preprocessing and scaling, we trained the models on data from 2021 to 2023 and evaluated them on 2024 data. The tested configurations included L1, L2, and ElasticNet penalties, with various C values to control regularization strength, ensuring both robustness and generalizability.

All five logistic models achieved similar accuracy (around 0.749), but differences emerged in recall and F1-score. The best-performing configuration was L2 regularization with C=10 or 0.1, both yielding the highest F1 score of 0.34158 and recall of 0.8186, meaning the model successfully identified over 81% of actual violent crimes. However, precision remained low (~0.215), suggesting that while the model captures most violent incidents, it also mislabels many non-violent ones.

model	penalty	c	solver	acc	f1-score
1	l2	1	lbfgs	0.74963	0.34142
2	l2	10	lbfgs	0.74982	0.34158
3	l2	0.1	lbfgs	0.74981	0.34155
4	l1	1	liblinear	0.74926	0.34109
5	elesticnet	1	saga	0.74945	0.34125

Table 12: Logistic Regression Classifier Results

12. kNN Classifier

The K-Nearest Neighbors (KNN) classifier is a simple, non-parametric algorithm that classifies data points based on the majority class of their k nearest neighbors in the feature space. It calculates the distance between the test point and all training points, identifies the k closest data points, and assigns the most common class label among them to the test point. KNN is effective for spatial-temporal crime classification as it leverages proximity relationships in the feature space without assuming a specific data distribution.

The model evaluation revealed that the optimal k value was 25, achieving an accuracy of 84.0% and a specificity of approximately 95%, indicating strong performance in correctly identifying non-crime events. However, the recall for the minority class was relatively low at 35%, suggesting that the model struggled to detect actual crime cases effectively, despite its high precision of 61%.

model	n	acc	f1-score
1	25	0.84008	0.44694
2	19	0.83714	0.45096
3	15	0.836	0.45559
4	11	0.83389	0.46075
5	9	0.8317	0.46531
6	7	0.82777	0.46515
7	5	0.82013	0.45687
8	3	0.81061	0.45616
9	2	0.78703	0.45218

Table 13: kNN Classifier Results

5. Regression Results

1. Random Forest Regressor

To support strategic planning and optimize resource allocation in crime prevention, a predictive model was developed to forecast total crime counts for each geographical zone (Rpt Dist No). Unlike traditional classification tasks that focus on labeling individual incidents, this problem was approached as a regression task, aiming to predict aggregate crime levels for 2024 using historical data from 2020 to 2023. A Random Forest Regressor was selected for its robustness in handling non-linear relationships and its effectiveness with structured datasets. The model was trained on monthly aggregated crime counts per zone, utilizing temporal features such as year and month, alongside spatial identifiers like the zone number. This setup enabled the model to capture both spatial and temporal patterns in crime distribution across the city.

The Random Forest Regressor demonstrated strong predictive capability, achieving a low mean squared error (MSE) of 1.07 on the 2024 test data, indicating accurate tracking of actual

crime totals across most zones. Notably, the model identified zone 992 as the most crime-prone area in 2024, forecasting approximately 91 crimes for the year. These insights provide valuable guidance for law enforcement agencies in prioritizing surveillance and intervention strategies. Despite the model's strong performance, further improvements could be made by integrating socio-demographic and environmental variables to refine predictions. Additionally, evolving the model into a full-fledged time series forecasting framework might better capture seasonal patterns and dynamic crime trends. Overall, the Random Forest–based approach proved effective for high-level crime forecasting, offering a solid data-driven foundation for proactive crime mitigation planning.

Model	n_estimators	max_depth	Predicted Crime Count	MSE
1	100	5	29.75005	1.067199
2	200	6	86.16193	1.029573
3	150	7	84.22557	0.982184
4	100	8	83.8957	0.966816
5	120	4	16.23323	1.07593
6	180	9	83.23774	0.944612
7	160	6	85.75667	1.030184
8	130	5	30.32586	1.065247
9	110	7	83.62072	0.983548
10	140	6	85.64563	1.02938

Table 14: Random Forest Regressor Results

2. Hist GradientBoosting Regressor

HistGradientBoosting Regressor is a powerful gradient boosting algorithm designed for scalable and efficient regression tasks. It discretizes continuous features into histograms to accelerate training, making it well-suited for large datasets. The model iteratively builds shallow decision trees to minimize prediction error. Key parameters include max_iter (number of boosting iterations), learning_rate (controls model updates), and max_depth (limits tree complexity). Ten parameter combinations were evaluated to forecast zone-level crime counts for the year 2024.

Model 3, with max_iter=300, learning_rate=0.01, and max_depth=6, produced the best results:

- Most dangerous predicted zone: 992
- Predicted crime count: 33.01
- Mean Squared Error (MSE): 1.03

This configuration effectively captured spatial crime patterns by balancing depth and learning precision. While the overall R^2 score remained moderate (~0.32), the model excelled at providing interpretable and actionable insights, particularly in identifying high-risk zones for proactive

planning.

Model	Max Iter	Learning Rate	Max Depth	Top Zone Predicted (Rpt Dist No)	Predicted Crime Count	MSE
1	100	0.1	3	989	30.592858	1.039997
2	200	0.05	4	989	38.567744	1.045511
3	300	0.01	6	992	33.011724	1.025054
4	150	0.15	3	991	38.622426	1.04553
5	120	0.1	5	992	43.439997	1.058323
6	250	0.05	7	989	44.23285	1.067813
7	180	0.08	4	991	41.760491	1.051946
8	300	0.1	6	991	43.661748	1.068149
9	200	0.02	8	989	37.407431	1.040924
10	100	0.2	2	990	32.154313	1.051424

Table 15: Hist GradientBoosting Regressor Results

3. XGBoost Regressor

The XGBoost Regressor is a high-performance implementation of gradient boosting for regression tasks, building an ensemble of decision trees sequentially to predict continuous target values. It starts with an initial prediction (e.g., the mean) and iteratively adds trees that learn to correct the residual errors of the current model. Each new tree refines the prediction in an additive manner. XGBoost enhances this process with optimizations like regularization, parallel processing, and efficient handling of missing data, making it a fast, flexible, and accurate choice for real-world regression problems.

To determine the optimal configuration for the XGBoost Regressor, we evaluated multiple configurations using a dataset split into the first three years for training and the last two for testing. We tested combinations of 100 to 500 estimators, tree methods including ‘hist’ and ‘auto’, max depths from 3 to 10, and learning rates ranging from 0.01 to 0.2, recording the best outcomes based on MSE. The configuration with 300 estimators, ‘hist’ tree method, max depth of 7, and a learning rate of 0.1 (Model 2) achieved the lowest MSE of 0.9874, identifying district 990 as the most dangerous zone with a predicted crime count of 70.9798. Configurations with higher numbers of estimators, like Model 3 and 8 (500 estimators), resulted in a worse MSE of 0.9277 and 0.988, respectively. And also in increased computational demand. Ultimately, we identified 300 estimators with a learning rate of 0.1, max depth of 7, and ‘hist’ method as the optimal setup, balancing predictive accuracy and computational efficiency.

Model ID	Estimators	Tree Method	Max depth	Learning rate	Most dangerous zone	Predicted crime count	MSE
1	100	hist	3	0.01	980	14.4563	1.0967
2	300	hist	7	0.1	990	70.9798	0.9874
3	500	hist	3	0.05	992	76.7349	0.9927
4	200	auto	5	0.05	991	90.6151	0.9891
5	100	hist	10	0.2	991	102.747	1.026
6	400	hist	5	0.05	990	94.1408	0.9886
7	200	hist	3	0.05	993	63.0808	1.0215
8	500	hist	7	0.01	991	89.0717	0.988
9	300	auto	5	0.1	992	98.0046	0.9923
10	250	hist	6	0.07	992	99.9561	0.9914

Table 16: XGBoost Regressor Results

4. LightGBM

LightGBM is a high-performance gradient boosting framework for regression tasks, like predicting monthly crime counts per zone, using histogram-based tree building for efficiency. It processes tabular data (e.g., Year, Month, Rpt Dist No) to model complex patterns. Key parameters include: *n_estimators*: Number of trees, controlling model capacity; *learning_rate*: Gradient step size, balancing speed and stability; *max_depth:* Limits tree depth, controlling complexity; *num_leaves*: Number of leaves per tree, enabling detailed splits; *min_child_samples:* Minimum leaf samples, preventing overfitting; *subsample:* Data fraction per iteration, enhancing robustness; *colsample_bytree:* Feature fraction per tree, increasing diversity; *reg_alpha:* L1 regularization, reducing overfitting; *reg_lambda:* L2 regularization (assumed for reg_beta), managing complexity.

Tuned across 10 configurations to minimize MSE, Model 7 (500 trees, 0.005 learning rate, max_depth 8, 60 leaves, 10 min_child_samples, 0.6 subsample, 0.6 colsample_bytree, 0.4 reg_alpha, 0.4 reg_lambda) achieves the best MSE of 0.968675. Its balanced parameters ensure robust learning, predicting zone 992 as the most dangerous (61.190231 crimes). Model 7's low MSE and alignment with expected high-crime zones (e.g., 989-991) make it highly effective, with potential for further refinement via feature engineering.

Model	n_estimators	learning_rate	max_depth	num_leaves	min_child_samples	subsample	colsample_bytree	reg_alpha	reg_beta	Rpt Dist No	Predicted Crime Count	MSE
1	100	0.1	5	31	20	0.8	0.8	0	0	991	81.080523	0.988204
2	200	0.05	7	50	10	0.9	0.7	0.1	0.1	989	85.259763	0.97944
3	300	0.01	3	20	30	0.6	0.6	0.5	0.5	989	43.513258	1.05754
4	150	0.2	10	70	15	0.7	0.9	0	0.2	992	102.282902	1.002054
5	400	0.03	6	40	25	0.8	0.8	0.3	0.3	992	85.852243	0.98373
6	250	0.08	4	25	20	0.9	0.7	0.2	0	990	80.29609	0.99263
7	500	0.005	8	60	10	0.6	0.6	0.4	0.4	992	61.190231	0.968675
8	100	0.15	5	35	30	0.7	0.8	0.1	0.1	990	84.489756	0.987348
9	350	0.02	7	45	15	0.8	0.9	0	0.5	991	88.902639	0.988123
10	200	0.1	6	30	20	0.9	0.7	0.3	0.2	992	85.916723	0.983116

Table 17: LSTM Regressor Results

5. Linear Regression with Random Forest

The hybrid model combines **Linear Regression** and **Random Forest Regressor** within a VotingRegressor to leverage both linear interpretability and non-linear modeling capacity. Linear models such as LinearRegression, Ridge, and Lasso provide a baseline understanding of how

individual features influence crime count in a continuous and interpretable manner. In contrast, Random Forest captures complex, non-linear patterns and interactions through an ensemble of decision trees. The ensemble aggregates predictions from both models using weighted averaging, enabling a balanced trade-off between bias and variance. Feature scaling was applied specifically to linear models within each pipeline to ensure proper handling of feature magnitudes without affecting the tree-based regressor.

Across five hybrid configurations trained on 2021–2023 data and evaluated on 2024, the best-performing model was the 2nd one, which combined LinearRegression with a RandomForestRegressor (500 trees, depth=20, min leaf=2) and weights of 0.3 | 0.7 (Linear | Forest). It achieved the lowest MSE (0.934), indicating strong predictive accuracy on unseen data.

model	LinType	RF_n	RF_depth	RF_minLeaf	MSE
1	Lin. Regression	300	None	1	0.9448
2	Lin. Regression	500	20	2	0.934
3	Ridge	400	None	1	0.9376
4	Lasso	600	30	1	0.966
5	Lin. Regression	1000	15	1	0.936

Table 18: Linear Regression with Random Forest Results

6. Findings

This project addressed two core goals: classifying violent crimes and predicting area danger levels in Los Angeles using LAPD Crime Data (2020–2024). A broad spectrum of models was applied—ranging from traditional algorithms like SVM, Decision Trees, and Logistic Regression, MLP to ensemble methods such as Random Forest, AdaBoost, XGBoost, and CatBoost, as well as deep learning architectures like CNN and LSTM.

In the classification task, ensemble and deep learning models outperformed simpler approaches. Notably, XGBoost, ExtraTrees and CatBoost achieved the highest accuracies (approaching 90%) and maintained a strong balance between precision and recall for violent crime (minority class) detection. A hybrid kNN + Random Forest model using soft voting also yielded competitive results, with an F1-score of 0.58 for violent crimes. Deep learning models like CNN, particularly those using ReLU and softmax, delivered robust classification results. In contrast, basic models like kNN, SVM, and Logistic Regression exhibited difficulty in detecting minority classes.

For the regression task—focused on predicting crime counts to assess area danger—multiple techniques were evaluated. While the Random Forest Regressor and LightGBM consistently delivered strong performance with MSE values between 0.93 and 1.07, the Linear Regression and Random Forest Hybrid gave the best result. Hist GradientBoosting and XGBoost

Regressor performed slightly worse, but still managed to identify the correct area with the highest crime rates for the most part.

Overall, the study highlights the effectiveness of ensemble learning and deep learning for both classification and regression in crime analytics, where CatBoost and XGBoost Classifiers were the best for the classification task and Linear Regression with Random Forest Hybrid was the best for the regression task.

Model Name	Best Performing Model Number	Highest Accuracy
SVM	3	0.84
kNN + Random Forest Hybrid	3	0.88130
XGBoost Classifier	8	0.897593
CatBoost Classifier	10	0.898168
Decision Tree Classifier	1,2,5,6	0.84
CNN	Architecture 1, softmax, relu	0.88226276
MLP	11	0.890235
ExtraTrees Classifier	9	0.898015
AdaBoost Classifier	8	0.856968
LSTM Classifier	1,2	0.824894
Logistic Regression Classifier	2	0.74982
kNN Classifier	1	0.84008

Table 19: Classification Results Summary Table

Model Name	Best Performing Model Number	Best MSE
Random Forest Regressor	6	0.944612
Hist GradientBoosting Regressor	3	1.025054
XGBoost Regressor	7	1.0254
LightGBM	7	0.968675
Linear Regression + Random Forest	2	0.934

Table 20: Regression Results Summary Table

7. References

Nguyenphantuan. (2023, March 11). *Exploratory Data Analysis: Los Angeles Crime (2020–2023)*. Retrieved from Medium: https://medium.com/@ptuan5/exploratory-data-analysis-los-angeles-crime-2020-2023-5adab44973c9

Pangarego, R. (2023, October 17). *Case Study: LAPD Crime Data from 2020 to Oct 2023 Analysis using Python*. Retrieved from Medium: https://medium.com/@rpangarego/case-study-lapd-crime-data-from-2020-to-oct-2023-analysis-using-python-1d5d6dbf58f8

Sankul, R., Aruva, T. R., Kankal, S. V., Arrapogula, G., & Khan, S. (2025, February 4). Crime data analysis and prediction of arrest using machine learning. Hyderabad, Telangana, India.