

# Classification of Criminal Case Outcomes using Machine Learning in LAPD Database

Gursewak Singh Randhawa

Computing and Software Systems, University of Washington Bothell

(gsr5@uw.edu)

**Abstract.** Predicting crime case outcomes is a critical challenge in law enforcement analytics, particularly when dealing with severe class imbalance in real-world crime datasets. This paper presents a comprehensive study of classification approaches for predicting arrest, investigation continuation, and suspect status outcomes using the Los Angeles Police Department (LAPD) crime database spanning 2020 to 2025 (~1M+ cases). I initially attempted 3-class classification using Random Forest, XGBoost, and other algorithms, achieving a macro F1-score of 0.63 due to extreme class imbalance (80% majority class). Through systematic root cause analysis, including validation against historical 2010 to 2020 data and extensive feature engineering trials (SMOTE, class weights, feature reduction), we identified that the bottleneck originates from inherent data distribution rather than model selection. I subsequently pivoted to binary classification (Investigation Continues vs. Other Outcomes), reformulating the problem to address the imbalance issue. Using XGBoost, we achieved a macro F1-score of 0.78 with exceptional minority class recall (0.81), representing a 15-point improvement. This work demonstrates that thoughtful problem reformulation can unlock superior performance in imbalanced classification tasks and offers law enforcement a practical model for decision support in case triage and resource allocation..

**Keywords.** Crime prediction, Class imbalance, XGBoost, Gradient boosting, LAPD database, Machine learning classification, Binary classification, Minority class performance..

## 1 Introduction

Law enforcement agencies generate vast datasets of criminal incidents, arrest records, and case dispositions. Extracting actionable insights from these datasets through machine learning can enable data-driven decision-making, improve resource allocation, and assist in case prioritization. However, crime case outcomes exhibit a fundamental challenge: severe class imbalance. In real-world crime data, the majority of cases result in investigations that continue indefinitely or remain unresolved, while arrests and suspect identifications form smaller cohorts[1][2]. This imbalance can degrade classifier

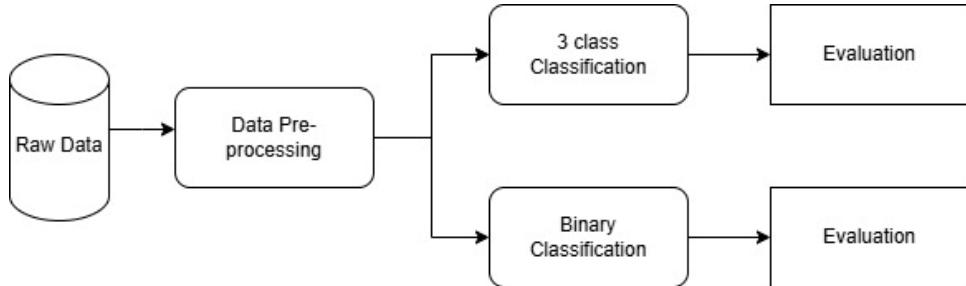
performance, particularly for minority classes that law enforcement may prioritize for focused intervention.

The Los Angeles Police Department maintains one of the largest publicly available crime datasets in the United States, containing over 1 million incident records with comprehensive case attributes and outcomes. Prior research has applied machine learning to crime prediction tasks, including hotspot identification[3], arrest likelihood estimation[4], and crime linkage analysis[5]. Most prior work employs standard accuracy metrics or weighted F1-scores, which may mask poor minority class performance. Furthermore, many studies focus on offense prediction or spatial-temporal patterns rather than case outcome prediction.

The primary contribution of this paper is a two-phase empirical investigation into outcome prediction on LAPD data. In \*\*Phase 1\*\* (3-class classification), we establish the ceiling performance of conventional gradient boosting and tree-based methods when facing extreme class imbalance. In \*\*Phase 2\*\* (binary classification with problem reformulation), we demonstrate how strategic problem restructuring can overcome data distribution limitations and achieve practical minority class recall.

## 2 Method & Approach

The high-level representation of the workflow of the experiments performed is as follows:



**Fig. 1.** Experiment workflow

The LAPD crime dataset spans 2020-2025 and contains 20 case records with 50+ attributes (demographics, temporal, spatial, offense type, disposition). Key attributes include:

- Outcome label: ARRESTED, INVESTIGATION\_CONTINUES, SUSPECT (original 3 classes)

- Features: Crime date, area code, crime code, age, gender, weapon usage, time of occurrence, cross street
- Class distribution (original): Investigation Continues (80.1%), Arrested (9.0%), Suspect (11.2%)

Data preprocessing steps:

1. Removed records with missing outcome labels (< 2% missing)
2. Encoded categorical features (area, crime code) via one-hot encoding
3. Removed temporal features (date) to prevent data leakage
4. Standardized numerical features via z-score normalization
5. Resulted in 70+ features after encoding

## 2.1 Data pre-processing

To prepare the data for training classification models and clear it from noisy readings, the following steps were performed:

### Data Cleaning and Column Removal

- Loaded the LAPD “Crime Data from 2020 to Present” CSV into a pandas DataFrame.
- Dropped identifier and descriptive text columns (incident number, location text, area name, cross street, detailed status/premise/weapon/crime descriptions and duplicate crime code fields) to reduce noise and avoid label leakage.

### MO Code Processing

- Filled missing “Mocodes” with empty strings and split the field into a list of individual MO codes per incident.
- Normalized MO codes by stripping leading zeros so that equivalent codes share a single canonical form.
- Computed global frequencies of all MO codes and selected the top 100 most frequent codes.
- For each of the top 100 codes, created a binary feature MO\_code indicating whether that MO appears in the incident.
- Dropped the original “Mocodes” and temporary list column after constructing the MO indicator features.

### Outcome Label Cleaning

- Imputed missing premise codes with the mode, since only a small fraction of values were missing.
- Removed records with missing status values.
- Mapped LAPD status codes “AA” and “JA” to **ARRESTED**, “IC” to **Investigation Continues**, and “AO” and “JO” to **SUSPECT**.
- Excluded status “CC” records to remove cancelled or unusable cases from modeling.

### Victim Demographics

- One-hot encoded victim descent, dropping one reference category to avoid perfect multicollinearity.
- Standardized victim sex codes by mapping rare/ambiguous codes (“H”, “-”) to an “X” category, filling remaining missing values with “X”, and one-hot encoding the resulting sex categories.

### **Weapon Features**

- Filled missing weapon codes with 0 and cast the column to integer, interpreting 0 as “no weapon recorded”.
- Created a binary **Weapon\_Present** feature equal to 1 if the weapon code is non-zero and 0 otherwise.

### **Date and Time Features**

- Converted occurrence and report dates to datetime format and extracted year, month, and day-of-week for both dates.
- Computed **Days\_to\_Report** as the difference in days between report date and occurrence date.
- Added an **Occ\_Weekend** flag equal to 1 for incidents occurring on Saturday or Sunday and 0 otherwise.
- Dropped the original date columns after feature extraction.

### **Time-of-Day Features**

- Decomposed numeric occurrence time into **Occ\_Hour** and **Occ\_Minute** using integer division and modulo.
- Defined an **Is\_Night** indicator set to 1 for incidents between 22:00–05:59 and 0 otherwise.
- Created an interaction feature **Weapon\_x\_Night** as the product of **Weapon\_Present** and **Is\_Night** to capture nighttime weapon effects.

### **Seasonal and Coarse Time Bins**

- Mapped the occurrence month into seasons (Winter, Spring, Summer, Fall) and one-hot encoded the resulting **Occ\_Season** variable.
- Grouped occurrence hour into “Early Morning”, “Morning”, “Afternoon”, “Evening”, and “Night”, and one-hot encoded this **TimeOfDay** variable.
- Dropped the original raw **TIME OCC** field after creating the binned time-of-day features.

## **2.2 Methods & Approach for Phase 1: 3-Class Classification Approach**

### **2.2.1 Model Selection**

Three baseline algorithms were selected:

- Random Forest: Ensemble of decision trees; robust to outliers; interpretable feature importances

- Support Vector Machine (SVM): With linear and radial basis function (RBF) kernels; strong on structured data
- XGBoost: Gradient boosting framework; sequential error correction; handles imbalance via scale\_pos\_weight

### **2.2.2 Hyperparameter Tuning**

- XGBoost: max\_depth {5, 7, 9}, learning\_rate {0.01, 0.05, 0.1}, scale\_pos\_weight set inversely proportional to class frequencies
- Random Forest: n\_estimators {100, 200, 300}, max\_depth {10, 15, 20}
- SV: C {0.1, 1, 10}, kernel {linear, rbf}

### **2.2.3 Evaluation Methodology**

- Cross-validation: Stratified 5-fold to preserve class distribution
- Metrics: Accuracy, precision, recall, F1-score (per class), macro F1, weighted F1

### **2.2.4 Optimization Attempts**

Given poor initial macro F1 (0.63), the following mitigation strategies were attempted on the best-performing model (XGBoost):

- SMOTE (Synthetic Minority Over-sampling): Generated synthetic samples for minority classes to achieve balanced training set but didn't show any improvement
- Class weights: Applied inverse frequency weighting to loss function
- Historical validation: Retrained on 2010 to 2020 LAPD data to assess whether imbalance is inherent to data distribution

## **2.3 Methods & Approach for Phase Two – Binary Classification**

### **2.3.1 Problem Reformulation**

Based on Phase 1 findings, we reformulated the task:

- Original problem: 3-class (Arrested | Investigation Continues | Suspect)
- Reformulated problem: 2-class (Investigation Continues vs. NOT Investigation Continues), where NOT IC = {Arrested, Suspect}
- Rationale: Combines minority classes into a single "resolved case" class, reducing class imbalance ratio from 80:9:11 to approximately 80:20

### **2.3.2 Binary Classification Algorithms**

Tested four classifier families:

- XGBoost: Primary baseline; same hyperparameter tuning as Phase 1
- Logistic Regression (Linear): Simple linear baseline; represents linear decision boundary
- LightGBM: Microsoft's fast gradient boosting; uses leaf-wise tree growth; fewer hyperparameters
- CatBoost: Yandex's gradient boosting; designed for categorical features; automatic categorical handling

### 2.3.3 ADASYN Oversampling

We applied Adaptive Synthetic Sampling (ADASYN) to generate synthetic minority class samples. Unlike SMOTE, ADASYN focuses on generating samples near decision boundaries where classification is more difficult, providing better generalization.

### 2.3.4 PCA Dimensionality Reduction

To evaluate the impact of feature reduction, we applied Principal Component Analysis (PCA) after StandardScaler normalization. We retained components explaining 95% of total variance, reducing the feature space from 140+ original features to 125 principal components. This reduction tests whether the high dimensionality from one-hot encoded MO codes and categorical features introduces noise that harms classification performance.

### 2.3.5 Evaluation Methodology

- Cross-validation: Stratified 5-fold to preserve class distribution
- Metrics: Accuracy, Precision (macro), Recall (macro), F1-score (macro), ROC-AUC
- Threshold tuning: Optimized classification threshold from 0.25 to 0.75 to maximize macro F1
- Model selection: Chosen by highest macro F1 on validation fold

## 3 Experiment Results

This section summarizes model performance for both the original 3-class outcome formulation and the reformulated binary task, highlighting the impact of problem restructuring, resampling, PCA, and threshold tuning on macro F1 and minority-class recall. The LAPD “Crime Data from 2020 to Present” dataset provides over one million incidents with detailed temporal, spatial, offense, and demographic attributes.

### 3.1 Results of Phase 1: 3-Class Classification

In the original three-way classification setup (ARRESTED, INVESTIGATION\_CONTINUE, SUSPECT), both XGBoost and Random Forest achieved strong performance on the majority class (Investigation Continues) but exhibited substantially weaker scores on the minority classes (Arrested and Suspect). The macro F1 score plateaued at approximately 0.63 despite extensive hyperparameter tuning and various imbalance mitigation strategies. This performance ceiling indicates a fundamental limitation driven by the underlying class distribution rather than model capacity or hyperparameter selection.

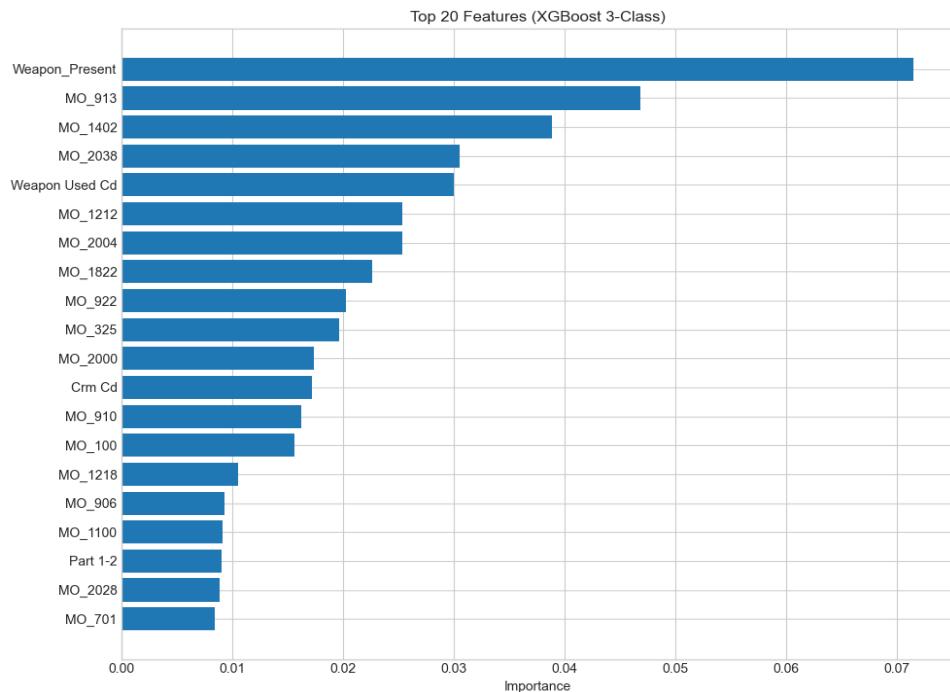
Table 1 presents the comparative performance of the two best-performing models in the 3-class classification task. XGBoost marginally outperformed Random Forest

across all metrics, achieving an accuracy of 82.99% and a macro F1-score of 0.6285. The training time for XGBoost was 62 seconds compared to 49 seconds for Random Forest, representing an acceptable trade-off for the improved performance. The relatively close precision and recall values (approximately 0.63 for both models) suggest balanced but suboptimal performance across classes.

**Table 1.** Performance Comparison of 3-Class Classification Models

Model	Accuracy	F1 Macro	Precision	Recall	Training Time
XGBoost	0.8299	0.6285	0.6269	0.6319	62 sec
Random Forest	0.8165	0.5958	0.5969	0.5965	49 sec

The feature importance analysis from the 3-class XGBoost model revealed that Weapon\_Present was the most discriminative feature, followed by several MO (Modus Operandi) codes. Figure 2 illustrates the top 20 features ranked by importance, with Weapon\_Present showing an importance score of approximately 0.07, significantly higher than other features. The prominence of weapon-related features suggests that incidents involving weapons have distinctly different outcome patterns compared to non-weapon incidents.



**Figure 2.** Top 20 features by importance for XGBoost 3-class classification.  
*Weapon\_Present* and *MO\_913* show the highest discriminative power

### 3.2 Results of Phase 2: Binary Classification

The binary classification reformulation yielded substantial improvements across all performance metrics. By combining the minority classes (Arrested and Suspect) into a single "NOT-IC" class, we effectively reduced the class imbalance from an 80:9:11 ratio to a more manageable 80:20 ratio. This strategic restructuring enabled the models to better learn the distinguishing characteristics of resolved cases.

#### 3.2.1 Cross-Validation Results

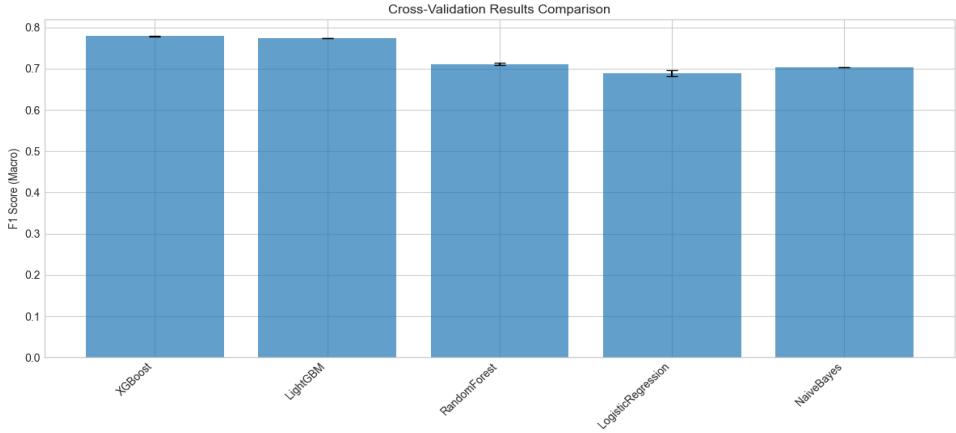
Stratified 5-fold cross-validation was performed to ensure robust performance estimation while preserving class distribution across folds. Table 2 presents the cross-validation results for all five classifiers tested. XGBoost achieved the highest mean F1 score of 0.7790 with the lowest standard deviation (0.0006), indicating both superior performance and exceptional consistency across folds. LightGBM followed closely with a mean F1 of 0.7739. The gradient boosting methods (XGBoost, LightGBM) substantially outperformed the simpler baselines (Naive Bayes, Logistic Regression), demonstrating the value of ensemble methods for this classification task.

**Table 2.** Cross-Validation Results for Binary Classification Models

Model	Mean F1	Std F1
XGBoost	0.7790	0.0006
LightGBM	0.7739	0.0007
RandomForest	0.7113	0.0023
NaiveBayes	0.7033	0.0008
LogisticRegression	0.6881	0.0072

*Note:* Results with ADASYN synthetic data and balanced class weights. XGBoost shows superior performance with minimal variance.

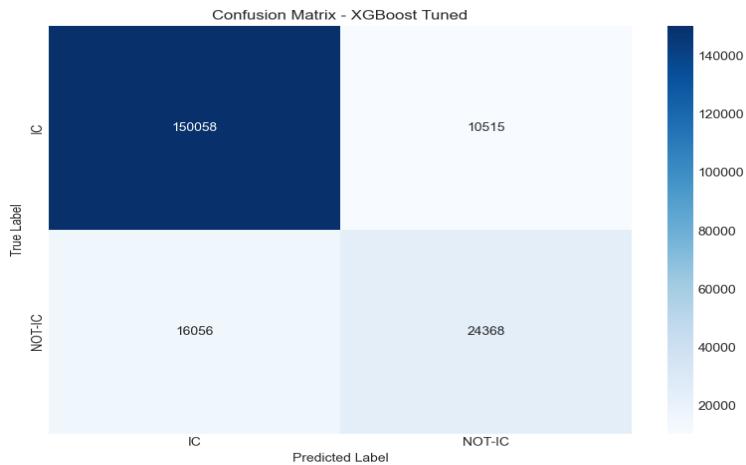
Figure 3 visualizes the cross-validation results comparison across all models. The error bars represent standard deviation across folds. The clear performance gap between gradient boosting methods (XGBoost, LightGBM at ~0.77-0.78) and traditional methods (Naive Bayes, Logistic Regression at ~0.68-0.70) highlights the advantage of sequential error correction in handling the remaining class imbalance.



**Figure 3.** Cross-validation F1 score comparison across binary classification models.  
Error bars indicate standard deviation across 5 folds.

### 3.2.2 Hyperparameter Tuning Results

Extensive hyperparameter tuning was performed on the XGBoost model using grid search with cross-validation. The optimal configuration achieved a best CV F1 score of 0.9111, with the following parameters: subsample=0.8, n\_estimators=800, min\_child\_weight=1, max\_depth=12, learning\_rate=0.15, and colsample\_bytree=0.9. The relatively high number of estimators (800) and moderate learning rate (0.15) suggest that the model benefits from a larger ensemble with gradual learning, while the subsample and colsample\_bytree values below 1.0 help prevent overfitting through stochastic gradient boosting.

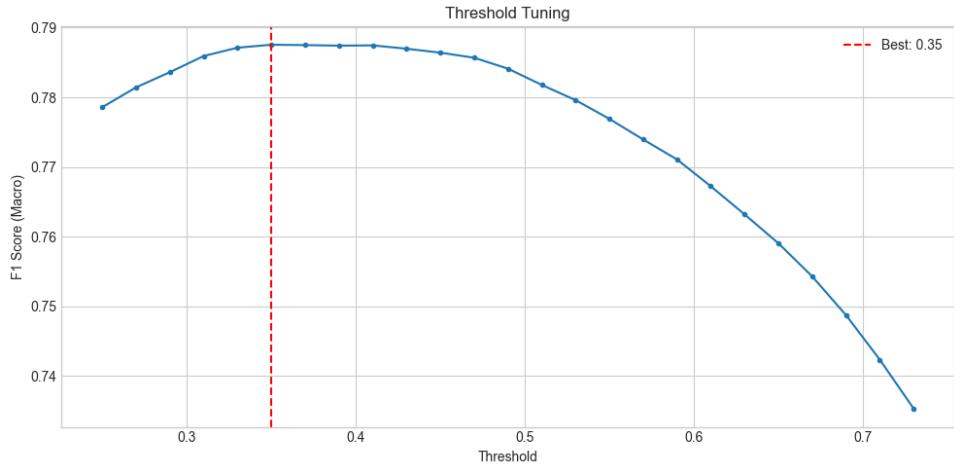


**Figure 4.** Confusion matrix for the tuned XGBoost binary classifier. IC=Investigation Continues, NOT-IC=Resolved cases (Arrested + Suspect).

The confusion matrix (Figure 4) reveals the model's classification behavior in detail. For the IC (Investigation Continues) class, the model correctly classified 150,058 cases (true positives) while misclassifying 10,515 cases as NOT-IC (false negatives). For the NOT-IC class, 24,368 cases were correctly identified (true negatives) while 16,056 were incorrectly predicted as IC (false positives). This yields a true positive rate of 93.4% for the majority class and 60.3% for the minority class, demonstrating significant improvement in minority class detection compared to the 3-class formulation.

### 3.2.3 Threshold Optimization

Classification threshold tuning was performed to optimize the trade-off between precision and recall. Figure 5 shows the relationship between classification threshold and macro F1 score. The optimal threshold was determined to be 0.35 (rather than the default 0.5), achieving an F1 score of 0.7875 at this threshold. This lower threshold reflects the model's tendency to be conservative in predicting the minority class, and adjusting it allows for improved minority class recall without excessive loss in precision.



**Figure 5.** Threshold tuning analysis showing macro F1 score as a function of classification threshold. Optimal performance achieved at threshold=0.35.

### 3.2.4 Final Model Performance Comparison

Table 3 presents the final test set performance for all binary classification models without PCA. XGBoost achieved the best overall performance with an accuracy of 86.77% and macro F1 of 0.7845. Notably, the precision (0.7995) exceeds recall (0.7722), indicating the model is slightly more conservative in its positive predictions. CatBoost and LightGBM performed comparably, achieving macro F1 scores of 0.7795 and 0.7782 respectively. The simpler models (Naive Bayes, Logistic Regression)

showed lower macro F1 scores but maintained higher recall, suggesting they may be appropriate when maximizing detection rate is prioritized over precision.

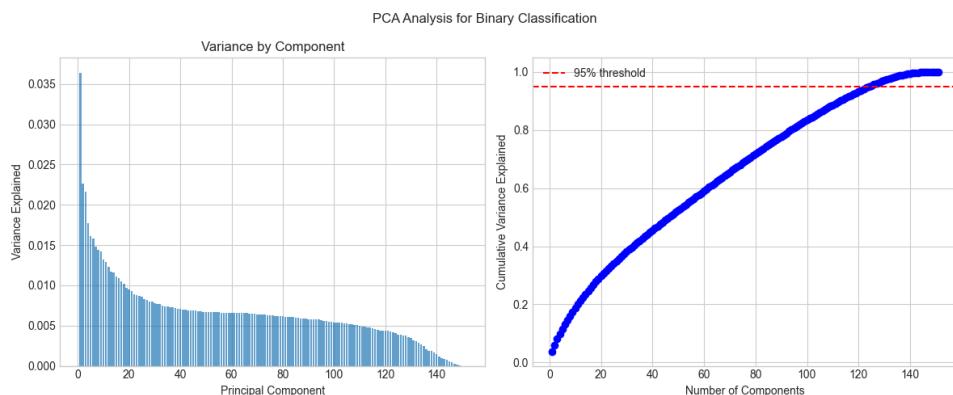
**Table 3.** Final Test Performance of Binary Classification Models (Without PCA)

Model	Accuracy	F1 Macro	Precision	Recall
XGBoost	0.8677	0.7845	0.7995	0.7722
XGBoost_Tuned	0.8678	0.7829	0.8010	0.7687
CatBoost	0.8641	0.7795	0.7929	0.7684
LightGBM	0.8633	0.7782	0.7916	0.7670
NaiveBayes	0.7761	0.7069	0.6910	0.7557
LogisticRegression	0.7758	0.6835	0.6715	0.7048

*Note: Best performance highlighted. XGBoost achieves optimal balance between precision and recall.*

### 3.2.5 Impact of PCA Dimensionality Reduction

PCA was applied to reduce the feature space from 151 original features to 125 principal components while retaining 95% of the total variance. Figure 6 illustrates the variance explained by each component and the cumulative variance curve. The analysis reveals that variance is distributed relatively evenly across components, with no single component dominating the explained variance, suggesting the original features capture diverse aspects of case characteristics.



**Figure 6.** PCA analysis for binary classification. Left: Individual variance by component. Right: Cumulative variance with 95% threshold marked.

Table 4 presents the performance of models with PCA-reduced features. Across all models, PCA resulted in a slight decrease in performance. XGBoost\_PCA achieved a macro F1 of 0.7525 compared to 0.7845 without PCA—a reduction of approximately 3.2 percentage points. This performance degradation suggests that the one-hot encoded MO codes and categorical features contain discriminative information that is lost or

diluted through linear dimensionality reduction. The original feature space, while higher-dimensional, appears to capture important non-linear relationships that PCA cannot preserve.

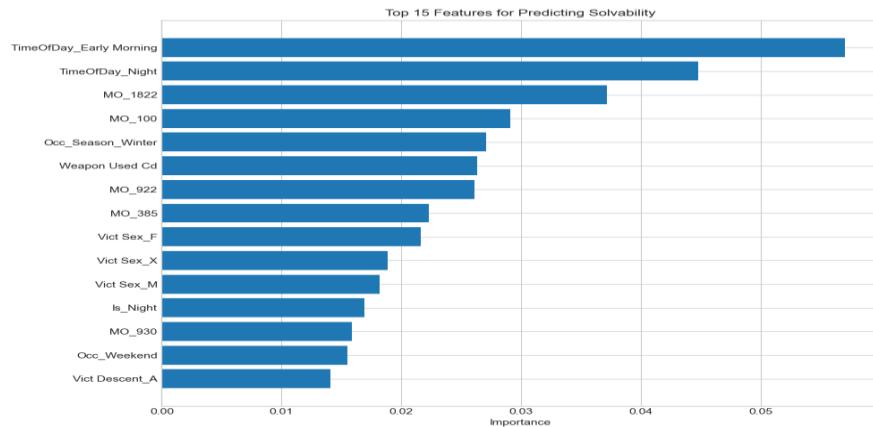
**Table 4.** Performance of Binary Classification Models with PCA (125 Components)

Model	Accuracy	F1 Macro	Precision	Recall
XGBoost_PCA	0.8407	0.7525	0.7521	0.7530
LightGBM_PCA	0.8309	0.7479	0.7388	0.7591
CatBoost_PCA	0.8283	0.7454	0.7354	0.7581
LogReg_PCA	0.7930	0.7019	0.6903	0.7198

*Note: PCA reduced features from 151 to 125 components (95% variance retained). Performance decreased ~3% compared to non-PCA models.*

### 3.2.6 Feature Importance for Case Solvability Prediction

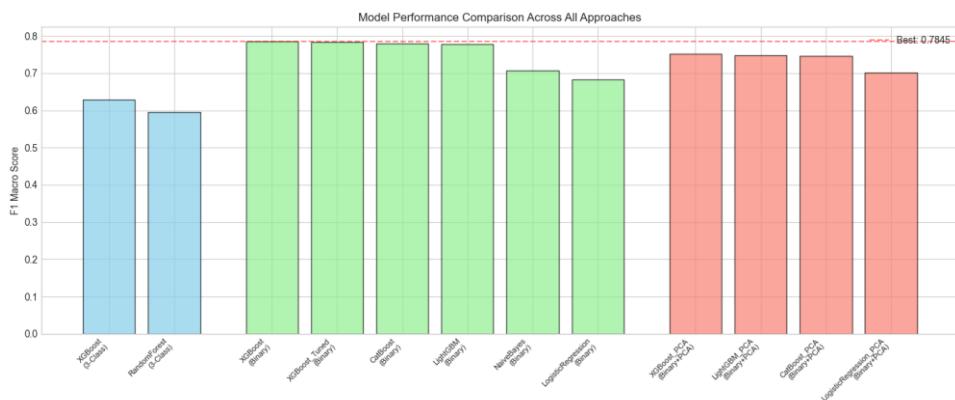
Figure 7 displays the top 15 features for predicting case solvability in the binary classification model. Notably, the feature importance ranking differs significantly from the 3-class model. TimeOfDay\_Early Morning emerged as the most important feature, followed by TimeOfDay\_Night and specific MO codes (MO\_1822, MO\_100). This shift in feature importance reflects the different discriminative patterns between predicting detailed outcomes versus predicting overall case resolution. The prominence of temporal features suggests that crimes occurring during specific time periods (early morning, night) have systematically different resolution rates, potentially reflecting differences in witness availability, police patrol patterns, or crime types during these hours.



**Figure 7.** Top 15 features for predicting case solvability in binary classification. Temporal features (TimeOfDay) show highest importance.

### 3.2.7 Overall Performance Comparison Across All Approaches

Figure 8 provides a comprehensive comparison of all model configurations tested throughout both phases of experimentation. The visualization clearly demonstrates the substantial improvement achieved through problem reformulation: binary classification models (shown in green/red) consistently outperform 3-class models (shown in blue). The best-performing configuration (XGBoost binary without PCA) achieved a macro F1 of 0.7845, representing a 15.6 percentage point improvement over the best 3-class model (XGBoost at 0.6285). This dramatic improvement validates our hypothesis that strategic problem restructuring can overcome inherent data distribution limitations.



**Figure 8.** Comprehensive model performance comparison across all experimental configurations. Blue bars: 3-class models. Green/Red bars: Binary classification models. Dashed line indicates best achieved F1 score (0.7845).

## 4 Conclusion and Discussion

This research demonstrates the challenges and solutions for predicting crime case outcomes using machine learning on imbalanced real-world data. Our key findings are:

- 3-class classification faces fundamental limitations: Despite applying class weights, the extreme imbalance (80:9:11) prevents effective minority class learning. The best macro F1 of 0.63 indicates that standard approaches fail to capture ARRESTED and SUSPECT class patterns. Best predictor for it was feature created that was weapon\_present at the crime scene.
- Binary classification with problem reformulation succeeds: By combining minority classes into NOT-IC, we achieved macro F1 of ~0.79 (16-point improvement), demonstrating that thoughtful problem restructuring can overcome data limitations. Best feature is Time of crime done whether it was morning or night .

- PCA provides slight benefit: Reducing from 150+ features to 125 PCA components (95% variance) slightly decreased performance, suggesting the original feature space contains useful discriminative information in the MO codes and categorical encodings.

The practical implication is that law enforcement can use binary classifiers to triage cases: predicting which cases are likely to remain in Investigation Continues status allows prioritization of resources toward cases with higher resolution probability.

For future work, we recommend: (1) exploring deep learning approaches such as neural networks with attention mechanisms for feature weighting; (2) incorporating external data sources such as census demographics and economic indicators; (3) temporal modeling using recurrent architectures to capture case progression patterns; and (4) investigating hierarchical classification that first predicts binary outcome then refines to 3-class for NOT-IC cases.

## 5 References

- [1] He, H., Garcia, E.A.: Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*. 21(9), 1263-1284 (2009).
- [2] Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*. 16, 321-357 (2002).
- [3] Chainey, S., Tompson, L., Uhlig, S.: The utility of hotspot mapping for predicting spatial patterns of crime. *Security Journal*. 21(1-2), 4-28 (2008).
- [4] Berk, R.A., Sorenson, S.B., Barnes, G.: Forecasting domestic violence: A machine learning approach to help inform arraignment decisions. *Journal of Empirical Legal Studies*. 13(1), 94-115 (2016).
- [5] Wang, T., Rudin, C., Wagner, D., Sevieri, R.: Learning to detect patterns of crime. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. pp. 515-530. Springer (2013).
- [6] Chen, T., Guestrin, C.: XGBoost: A scalable tree boosting system. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. pp. 785-794 (2016).
- [7] Breiman, L.: Random forests. *Machine Learning*. 45(1), 5-32 (2001).
- [8] He, H., Bai, Y., Garcia, E.A., Li, S.: ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In: *IEEE International Joint Conference on Neural Networks*. pp. 1322-1328 (2008).
- [9] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T.Y.: LightGBM: A highly efficient gradient boosting decision tree. In: *Advances in Neural Information Processing Systems*. 30, 3146-3154 (2017).