## 1.1 Explain the properties of the chosen data set and plan of analysis:

The project began with an idea to predict the type of weapon used in incidents based on a range of demographic and contextual features. Initially, the weapon categories included multiple classes such as *"Firearm," "Poison," "Blunt Object,"* and others. The predictors encompassed parameters like *"Victim Age," "Perpetrator Gender,"* and various relational and geographic attributes. The main goal was to develop a robust and efficient model capable of accurately classifying the weapon category.

Several machine learning algorithms were evaluated for this task, starting with the **Random Forest Classifier**, **XGBoost**, and **CatBoost**. While XGBoost and CatBoost offered faster execution times, Random Forest consistently produced the best results in terms of accuracy and balanced performance across categories. However, its longer training time was a challenge, taking several minutes to complete the process. To address this, the trained model was saved to a file, allowing it to be reused instantly in tools like Gradio for real-time predictions.

To ensure the Random Forest model could handle the categorical data effectively, **OneHotEncoding** was applied to all categorical features. This transformation converted each category into a set of binary features, making the data compatible with the Random Forest algorithm. Features like *State, Victim Sex,* and *Relationship Category* were encoded to ensure consistent and interpretable preprocessing.

Further performance improvements were achieved by grouping certain categorical features into broader categories. For example, months were combined into seasons, states were grouped into regions, and specific relationships like *"Brother"* and *"Mother"* were consolidated under the broader category of *Family*. These transformations simplified the dataset while retaining key patterns, improving both computational efficiency and model accuracy.

Despite these improvements, the dataset's inherent class imbalance emerged as a critical issue. For instance, *"Firearm"* records significantly outnumbered others like *"Poison,"* which skewed the model’s ability to generalize across classes. To address this, weapon categories were consolidated into binary outputs: **Firearm and Non-Firearm**. While this reduced the granularity of predictions, it allowed for more balanced and meaningful outputs.

However, even after binarization, there was still an imbalance—approximately 80,000 records for *"Firearm"* compared to approximately 40,000 for *"Non-Firearm."* To address this, the **SMOTE (Synthetic Minority Oversampling Technique)** was applied. SMOTE generates synthetic examples of the minority class by interpolating between existing samples, thus creating a balanced dataset. This not only improved the model's recall for *"Non-Firearm"* but also ensured fairer performance across both classes.

## 1.2 Mention the machine-learning techniques that you will be using:

2.1 Mechanics of Selected Machine Learning Techniques

•

Random Forest: Builds multiple decision trees during training and combines their outputs to improve prediction accuracy. It mitigates overfitting by averaging predictions and supports both categorical and numerical data. For this project,

OneHotEncoding

was applied to categorical features to ensure compatibility.

•

XGBoost: A boosting algorithm that sequentially builds weak learners to correct errors from previous iterations. While it can handle categorical features natively, we maintained OneHotEncoding for consistency with other models.

•

Logistic Regression:   
  
ANN - TODO

>>>>> TODO : FIX HERE

## 

## 2. BACKGROUND

2.1 Demonstrate and describe the mechanics of the selected machine learning techniques.

**2.2.1 Logistic Regression**

As a starting point, we trained a **logistic regression model**, a basic model for classification problems. Eventually, this basic model was fine-tuned using **regularization techniques**, namely Lasso and Ridge. Preparing data for training machine learning models is a crucial step, and this also applies to the logistic regression case. This model requires that each categorical feature is transformed into two sets, where one set, composed of one variable, serves as the reference category and the other set of variables serves as the dummies and which are collectively referred to as **dummy variables**.

The appropriate treatment of situations modeled by a binary output and a set of features composed of a combination of categorical and numerical features is to use the **logistic regression model** and not **linear regression**. There are various technical reasons for this:

* Since the distribution of the disturbance term is not normal, standard errors are invalid.
* Hence, the F-statistic, t-statistics, and p-values are invalid.
* The model is subject to heteroskedasticity, i.e., variance of the residuals is unequal because it is dependent on xxx.
* In addition, nonsensical predictions are a concern because for high values of xxx, predicted values may be greater than 1, and for negative values of xxx, predicted values will be less than 0. Probabilities less than 0 and greater than 1 are nonsensical.

Mathematically, the logistic function is represented by **Equation 1** below:

P(y=1∣X)=11+e−βXP(y=1|X) = \frac{1}{1 + e^{-\beta X}}P(y=1∣X)=1+e−βX1​

where:

* XXX represents each feature
* β\betaβ is the estimated weight for each feature
* P(y)P(y)P(y) is the output of the estimated probability which lies between 0 and 1. In our case, the output is the probability of a firearm being used (represented by 1) against a non-firearm being used (represented by 0).

By manipulating the logistic function, Equation 1 can be represented by the **log-odds**, i.e., **Equation 2**, which is linear in XXX:

log(P(y=1)1−P(y=1))=βX\text{log} \left(\frac{P(y=1)}{1-P(y=1)}\right) = \beta Xlog(1−P(y=1)P(y=1)​)=βX

where:

* P(y=1)P(y=1)P(y=1) can take a value between 0 and 1. Values close to 0 imply very low probabilities, and values close to 1 imply very high probabilities.

**Interpreting the Output**

Example:

* **Numerical variable:** Victim Age: A one-unit increase in ‘Victim Age’ decreases the log-odds of a firearm being used for the crime to be carried out by 0.19.
* **Categorical variables:** Two classes case – Victim Sex: Where the crime involves male victims, the log-odds of a firearm being used for the crime is 0.6869 higher compared to the reference category, i.e., when the crime involves female victims.

o More than two classes – The same logic for the two classes case applies.

It is important to note that **log-odds** can be converted to **odds** and to **probabilities**. Converting log odds to odds involves simply the following manipulation, for example for Victim\_Sex\_Male:

Hence, in terms of the odds the output for Victim Sex is interpreted as follows: where the crime involves male victims the odds of a firearm being used is almost two times higher than when the crime involves female victims.

Converting this to **probabilities** takes us back from Equation 2 to Equation 1. The computation of probabilities requires the collective consideration of all features and the calculation of Equation 1. It is conventional to use a probability of 0.5 as the decision boundary, such that in our case, where the probability is higher than 0.5, it implies that the use of a firearm is ‘Yes’ and where the probability is lower than 0.5, it implies that the use of a firearm is ‘No’.

**2.2.2 Random Forest Classifier**

The Random Forest Classifier is an ensemble machine learning algorithm that operates by constructing multiple decision trees during training and combining their outputs to make robust and accurate predictions. Each tree in the forest is trained on a random subset of the dataset, and the algorithm aggregates the predictions of all trees, effectively reducing overfitting and increasing generalization. This approach is particularly effective for both classification and regression problems, as it can handle numerical and categorical data without extensive preprocessing.

In this project, the Random Forest Classifier was chosen for its ability to manage class imbalances through its class\_weight='balanced' parameter, which adjusts weights inversely proportional to class frequencies. This is critical when dealing with skewed datasets, such as firearm versus non-firearm usage, ensuring fair performance across classes. The Python implementation, shown in the project, specifies 100 estimators (n\_estimators=100) to create a sufficiently large forest for stable predictions. It also allows for deep trees (max\_depth=None) and flexible splitting (min\_samples\_split=2), enabling the model to fully explore complex patterns in the data. The use of Random Forest in this project provided a balance of interpretability, computational efficiency, and high classification accuracy, making it an ideal choice for the problem at hand.

**2.2.3 XGBoost**

XGBoost, or Extreme Gradient Boosting, is a high-performance implementation of the gradient boosting algorithm, optimized for speed and accuracy. It builds an ensemble of decision trees sequentially, where each tree corrects the errors of its predecessors by focusing on misclassified instances. For binary classification tasks, such as firearm versus non-firearm predictions, XGBoost offers advanced features like class weighting and evaluation metrics that handle imbalanced datasets effectively.

In this project, the XGBoost model was implemented using the XGBClassifier class. The scale\_pos\_weight parameter was dynamically calculated as the ratio of negative to positive class samples, ensuring that the model appropriately addressed the imbalance between firearm and non-firearm records. The eval\_metric='aucpr' parameter optimized the model for the Area Under the Precision-Recall Curve, which is especially useful when dealing with imbalanced datasets. Additionally, the random\_state=42 parameter ensured reproducibility of results. By leveraging these configurations, XGBoost provided robust performance, balancing precision and recall for the minority class, and offered a computationally efficient alternative to other classifiers used in the project.

**2.2.4 Artificial Neural Networks (TODO)**

**TODO.**

### **2.2 Describe what rescaling and normalisation are and why they are important.**

Rescaling and normalization are pre-processing steps which ensure that the features are on the same scale i.e. having the same mean and variance, ensuring comparability between features. For the logistic regression, rescaling and normalisation are not strictly required. However, since Age (in years) is an integer usually between 0 and 100, and the dummy categorical features having a value of either 1 or 0, scaling Age may improve the logistic regression’s iterative optimization process. However, interpreting Age after the rescaling process reduces interpretability because rather than being interpreted as the effect of a “1-unit increase in Age”, it has to be interpreted as the effect of a “1-standard-deviation increase in Age”.

For the purpose of the use of the logistic regression age was scaled, however for the purpose of the Web Portal, it was decided to use the original Age variable which was mapped to the Age\_Scaled variable, since it is more intuitive for the user.

### **2.3 Describe what cross-validation is and how (if applicable) it was used.**

The machine learning process involves several steps which include:

- Randomly splitting the dataset between training data and test data,

- using an algorithm to train the model using the training dataset which in the logistic regression case is represented by Equation 1 above, where each reprents the estimates of the coefficients of each feature, .

- On this basis use the test dataset to predict probabilities , which determine which class the output belongs to. In our case firearm vs non-firearm.

- Subsequently, the predicted output is compared to the true output and using a confusion matrix, one uses metrics to for example determine how much of the output is correctly classified.

Carrying out this process only once will be highly dependent on the split of the dataset and hence, cross-validation could be deployed to randomly splitting the dataset into k groups or folds of approximately equal size. In turn, each group serves as the test dataset and the remaining groups as the training datasets. In each of the five cases, the test error is calculated typically using the misclassification rate or log-loss (cross-entropy loss). Then results are aggregated using average error rate or average log-loss. Metrics such as accuracy, precision and recall can be calculated similarly for each fold and averaging across all folds.[1], [2]

**TODO: Paste the image here <<<<<<**

In the logistic regression case, cross-validation was not directly used, but was applied when regularization techniques i.e. lasso and ridge were applied.

### **2.4 Describe what dimensionality reduction and feature selection methods are**

In situations where the number of observations, n, is extremely large, relative to the number of features, p, traditional machine techniques used to address regression and classification problems work fine. However, in situations where p is very large, dimensionality reduction techniques such as lasso and ridge which penalize those features which do not contribute much to the model, forcing their coefficient to move either closer to zero or even forcing it to zero, essentially making that specific feature redundant and effectively reducing the number of dimensions.[3] Since in this case p is much smaller than n, the results from lasso and ridge are not extremely different from that of the logistic regression.

### **2.5 Explain the quantitative measurements that you will be using to quantify the results; e.g. accuracy rate**

**TODO: PATE FROM GIT**

## **Performance Metrics**

When dealing with imbalanced datasets, accuracy can be misleading as it may disproportionately reflect the performance on the majority class. Instead, we should evaluate model performance using the following metrics:

* Precision: Measures the accuracy of positive predictions.
  + Formula: Precision = TP / (TP + FP)
* Recall: Also known as sensitivity, it focuses on identifying all positive cases.
  + Formula: Recall = TP / (TP + FN)
* F1-Score: The harmonic mean of precision and recall, balancing the trade-off between the two.
* Precision-Recall (P-R) Curve: A graphical representation of the trade-off between precision and recall across different thresholds.

These metrics provide better insights into model performance, especially for the minority classes.

## **Imbalance Mitigation Techniques**

To address dataset imbalance, the following techniques can be employed:

1. Resampling:
   * Oversampling the Minority Classes:
     + Use methods like SMOTE (Synthetic Minority Oversampling Technique) to create synthetic samples for minority classes, effectively balancing the dataset.
   * Undersampling the Majority Class:
     + Reduce the size of the majority class to match the minority classes. While this can help balance the data, it risks losing valuable information from the majority class.
     + Recommendation for Our Use Case: Since "Firearm" is the majority class and contains important information, oversampling the minority classes is preferred over undersampling "Firearm" as not to lose information in it.
2. Increasing Class Weights:
   * Assign higher weights to minority classes in the model's loss function to penalize misclassifications for these classes more heavily.

## **Way Forward**

To determine the best approach, we propose the following steps:

1. Baseline Training:
   * Train the model using the dataset in its current form (without applying any resampling techniques).
2. Resampling and Comparison:
   * Apply oversampling (e.g., SMOTE) to balance the dataset and train the model on the modified data. This should be done on the training set only.
   * Compare the results of the baseline model and the resampled model using performance metrics such as precision, recall, F1-score, and the P-R curve.

This comparison will provide insights into the effectiveness of imbalance mitigation techniques for our specific use case.

**TODO: REFERENCES**

[1] Introduction to Econometrics, Chapter 10.

[2] Introduction to Statistical Learning, Block 6.

**[1] Take note of input from https://chatgpt.com/c/67765b10-d43c-8003-a6f8-fa936d823af7** [**https://notebooklm.google.com/notebook/0427e3af-8cea-4498-bdd0-daa69e101d80?\_gl=1\*si44rf\*\_ga\*MzQ5MzA0MjIxLjE3MzM5OTIyNjQ.\*\_ga\_W0LDH41ZCB\*MTczNTgxODY5MC4xMC4wLjE3MzU4MTg2OTAuMC4wLjA**](https://notebooklm.google.com/notebook/0427e3af-8cea-4498-bdd0-daa69e101d80?_gl=1*si44rf*_ga*MzQ5MzA0MjIxLjE3MzM5OTIyNjQ.*_ga_W0LDH41ZCB*MTczNTgxODY5MC4xMC4wLjE3MzU4MTg2OTAuMC4wLjA)**.**

**[2]** [**https://notebooklm.google.com/notebook/0427e3af-8cea-4498-bdd0-daa69e101d80?\_gl=1\*si44rf\*\_ga\*MzQ5MzA0MjIxLjE3MzM5OTIyNjQ.\*\_ga\_W0LDH41ZCB\*MTczNTgxODY5MC4xMC4wLjE3MzU4MTg2OTAuMC4wLjA**](https://notebooklm.google.com/notebook/0427e3af-8cea-4498-bdd0-daa69e101d80?_gl=1*si44rf*_ga*MzQ5MzA0MjIxLjE3MzM5OTIyNjQ.*_ga_W0LDH41ZCB*MTczNTgxODY5MC4xMC4wLjE3MzU4MTg2OTAuMC4wLjA)**.**

**[3] James et al, 6.4.1 High-dimensional data, pg 238.**