MLCrpyt Challenge

Data preparation – The dataset consists of 2000 samples, each containing a 1024-length bitstring, representing two classes. The bitstrings are converted into integer arrays to ensure compatibility with machine learning and deep learning models, facilitating efficient data processing and model training for classification tasks.

Methodology –

1. Models Applied:
   - LGBM (LightGBM): A gradient boosting algorithm applied to classify the bitstrings.
   - Gaussian Naive Bayes (GaussianNB): A probabilistic classifier used for bitstring classification.
   - Random Forest (RandomForestClassifier): Applied as an ensemble method to handle the classification task.
   - Artificial Neural Network (ANN): A deep learning model to explore non-linear relationships within the bitstring data.

2. Cross-validation:
   - Due to the limited number of data samples (2000 samples, each with 1024-bit length), cross-validation was used to ensure reliable model performance and to prevent overfitting.

3. Alternative Approaches Tested:
   - Hamming Distance with K-Nearest Neighbors (KNN): Utilized to capture the similarity between bitstrings. This approach works well for categorical data, identifying patterns based on bitstring proximity.
   - Hidden Markov Model (HMM): Used to capture probabilistic bit patterns, assuming that the bitstring follows an underlying probabilistic process.

4. Statistical Feature Engineering:
   - Descriptive Statistics: Computed features such as mean, median, skewness, and kurtosis to summarize the distribution of the bitstring data.
   - Fourier Transform Features: Applied to capture frequency-domain features of the bitstrings, useful in detecting periodic patterns.
   - N-gram Count (up to 5 grams): Features based on the frequency of contiguous sequences of bits, aimed at identifying bitstring patterns over different lengths.
   - Entropy: Calculated to measure the randomness or uncertainty within the bitstring data.
   - Autocorrelation: Explored for any repetitive patterns or relationships between bits across different positions.

5. Data Manipulation Techniques:
    - Bit Flipping: Applied to invert bits at random positions, creating variations in the data.
    - Circular Bit Shifting: Shifted the bitstring circularly, helping to explore cyclic patterns in the bit data.

6. Results and Conclusion:
    - Despite extensive feature engineering and model experimentation, no consistent classification patterns or results emerged.
    - Accuracy fluctuated around 52%, likely due to the small sample size, indicating that with the current dataset, there were no strong distinguishing features to classify the bitstrings effectively.
    - The lack of significant results leads to the conclusion that the available data may not provide sufficient information to establish reliable classification patterns.