**Project: Exploring Advanced Machine Learning Techniques for Classification Problems**

**Problem Statement**

You are tasked with applying advanced machine learning techniques to solve a complex classification problem. The project involves using at least **three different machine learning algorithms**, which were not covered in the class, and optimizing their performance to achieve the best possible results. The objective is to demonstrate your ability to independently research, implement, and compare machine learning algorithms on a real-world dataset.

**Project Requirements**

1. **Dataset Selection**:
   * Choose a **real-world dataset** from repositories like [Kaggle](https://www.kaggle.com/), [UCI Machine Learning Repository](https://archive.ics.uci.edu/ml/index.php), or any other credible source.
   * The dataset must be sufficiently **complex** (e.g., imbalanced classes, high dimensionality, missing values, etc.) and contain at least **5,000 samples** and **10 features**.
   * Ensure the dataset is suited for **classification tasks**.
2. **Algorithms**:
   * Select at least **three different machine learning algorithms** for classification that were not taught in class. Examples include:
     + Random Forest
     + Support Vector Machines (SVM)
     + XGBoost
     + CatBoost
     + LightGBM
     + K-Nearest Neighbors (KNN)
     + Logistic Regression (if not covered previously)
     + Any other algoirhtm of your choice

**You must understand these algorithms to the extent that you can apply them by spcifying their different hyperparameters. The viva will focus on conceptual understanding of these algorithms as well.**

* + Implement and optimize these algorithms on your chosen dataset.

1. **Preprocessing**:
   * Perform **feature scaling** (e.g., standardization or normalization) where necessary.
   * Handle **missing data** using appropriate techniques.
   * Bonus: Perform additional pre-processing as required (detailed understanding is not required), e.g.,
     + Use **feature selection or dimensionality reduction** techniques (e.g., PCA, mutual information) to improve performance.
     + Address any **class imbalance** using techniques like SMOTE, ADASYN, or undersampling.
2. **Optimization**:
   * Tune hyperparameters for each algorithm using at least 2 methods like:
     + Grid Search
     + Random Search
     + Any other methods of your choice
   * Optimize the algorithms to achieve the best possible evaluation metric scores.
3. **Comparison and Insights**:
   * **Evaluation metrics:** Evaluate the models using multiple metrics:
     + Accuracy
     + Precision
     + Recall
     + F1-Score
     + Confusion Matrix
   * Compare the performance of the algorithms based on evaluation metrics. A table like below must be added to the report.

**Table: Comparison of Machine Learning Algorithms on [Dataset Name]**

| **Algorithm** | **Accuracy** | **Precision** | **Recall** | **F1-Score** | **ROC-AUC** | **Best Hyperparameters** | **Execution Time (s)** | **Remarks** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Random Forest | 0.85 | 0.88 | 0.84 | 0.86 | 0.92 | n\_estimators=100, max\_depth=10, min\_samples\_split=5 | 120 | Performed well with imbalanced data. |
| Support Vector Machine (SVM) | 0.80 | 0.83 | 0.76 | 0.79 | 0.87 | kernel='rbf', C=1.0, gamma=0.1 | 180 | Slower due to large dataset size. |
| XGBoost | 0.89 | 0.91 | 0.87 | 0.89 | 0.94 | learning\_rate=0.1, n\_estimators=150, max\_depth=8 | 150 | Best overall performance. |

1. **Visualization**:
   * Use data visualization libraries (e.g., Matplotlib, Seaborn) to present:
     + Plot different performance metrics in the form of a bar chart.
     + Draw Confusion matrices.
2. **Report**:
   * A concise 2-3 page report containing the following:
     + Provide a comprehensive report in a well-structured document (PDF or Jupyter Notebook format).
     + Include the following sections:
       1. **Introduction**: Problem and dataset description.
       2. **Methodology**: Preprocessing steps, algorithms applied, and optimization techniques.
       3. **Results**: Metrics and visualizations.
       4. **Analysis**: Insights, algorithm comparison, and challenges faced.

**Submission Guidelines**

* Submit the following:
  + **Code**: Ensure the code is clean, well-documented, and modular.Include the dataset or provide a clear link at the top of the code.
  + **Report**: Submit the report summarizing your work.

**Notes**

* Collaborate in teams of **2-3 members** or work individually.
* Create a GitHub repository for your project.
  + Decide team members roles. Ideally, each team member must completely take change of 1 or more classification algorithms included in the report.
  + The code must be pushed by individual team members to show individual contributions.