**Machine Learning Classification Models with Hyperparameter Tuning**

This project implements various machine learning classification models with hyperparameter tuning using GridSearchCV. The objective is to evaluate multiple models and identify the best-performing one for a given dataset. The project also handles data preprocessing, class imbalance, and model evaluation.

**Project Overview**

The project demonstrates a machine learning workflow, starting from data preprocessing to hyperparameter tuning and model evaluation. The key features include:

* **Preprocessing Pipeline**: Automated preprocessing for scaling, encoding, and handling missing values.
* **Class Imbalance Handling**: Use of SMOTE to oversample the minority class.
* **Model Tuning**: Hyperparameter optimization using GridSearchCV.
* **Evaluation**: Metrics such as accuracy, precision, recall, F1-score, and ROC AUC are calculated for model comparison.
* **Model Persistence**: Selected models are saved as .pkl files for later use.

**Features**

1. **Dataset Preprocessing**:
   * Missing value imputation.
   * Feature scaling using StandardScaler.
   * One-hot encoding for categorical variables.
2. **Class Imbalance Resolution**:
   * Oversampling with SMOTE to address imbalance in the target variable.
3. **Model Training**:
   * Models included:
     + K-Nearest Neighbors (KNN)
     + Logistic Regression (LR)
     + Random Forest (RF)
     + Support Vector Machine (SVC)
     + Gradient Boosting (GB)
     + Extreme Gradient Boosting (XGBoost)
     + Decision Tree (DT)
   * Hyperparameter tuning with GridSearchCV.
4. **Model Evaluation**:
   * Metrics:
     + Accuracy
     + Precision
     + Recall
     + F1-Score
     + ROC AUC
   * Results stored in a structured table for comparison.
5. **Model Saving**:
   * Best-performing models saved as.pkl files using joblib.

**Usage**

1. **Preprocess the Data**:
   * Load the dataset and preprocess using the defined pipeline.
2. **Train and Tune Models**:
   * Run the script main.py or the notebook training.ipynb to train and tune models.
3. **Evaluate Models**:
   * Evaluate models using predefined metrics and compare results.
4. **Save Models**:
   * Save the best-performing models for future use.
5. **Predict on New Data**:
   * Load the saved models to make predictions on unseen data.

**Contact**

For questions or collaboration opportunities, reach out to:

* **Name**: Timothy Morenikeji Akinremi
* **Email**: timothyakinremi@gmail.com
* **GitHub**: timothyakinremi