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

**1. Introduction**

**Problem Statement**

This project aims to explore advanced machine learning techniques to solve a complex classification problem. The objective is to independently research, implement, and optimize multiple classification algorithms on a real-world dataset, demonstrating a comprehensive understanding of preprocessing, model training, and evaluation.

**Dataset Description**

* **Dataset Name**: Adult Dataset
* **Source**: OpenML
* **Dataset Characteristics**:
  + **Samples**: 48,842
  + **Features**: 14 (categorical and numerical)
  + **Target Variable**: Income (binary classification: <=50K or >50K)
  + **Complexity**:
    - Missing values in some features.
    - Imbalanced target classes.

**2. Methodology**

**Preprocessing**

1. **Handling Missing Values**:
   * Replaced missing values with the most frequent category for categorical features.
2. **Feature Encoding**:
   * Applied LabelEncoder to convert categorical features into numerical values.
3. **Feature Scaling**:
   * Used StandardScaler to standardize numerical features for algorithms sensitive to feature scales.
4. **Dimensionality Reduction**:
   * Reduced the dataset size by 50% for faster hyperparameter tuning.

**Algorithms Applied**

1. **Random Forest**:
   * Ensemble-based algorithm that combines multiple decision trees.
   * Optimized using Grid Search and Randomized Search.
2. **Support Vector Machine (SVM)**:
   * Works by finding a hyperplane that maximizes margin.
   * Optimized using Grid Search and Randomized Search.
3. **Logistic Regression**:
   * Linear model for binary classification.
   * Optimized using Grid Search and Randomized Search.

**Optimization Techniques**

1. **Grid Search**:
   * Systematically searched hyperparameters for Random Forest and Logistic Regression.
2. **Randomized Search**:
   * Randomly sampled hyperparameters for SVM to reduce computational overhead.

**Hyperparameters Tuned**

* **Random Forest**: n\_estimators, max\_depth, min\_samples\_split, min\_samples\_leaf, max\_features
* **SVM**: kernel, C, gamma
* **Logistic Regression**: penalty, C, solver

**3. Results**

**Evaluation Metrics**

* **Accuracy**: Overall correctness.
* **Precision**: True positive predictions divided by all positive predictions.
* **Recall**: True positive predictions divided by actual positives.
* **F1-Score**: Harmonic mean of precision and recall.
* **ROC-AUC**: Area under the Receiver Operating Characteristic curve.

**Comparison Table**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Algorithm** | **Accuracy** | **Precision** | **Recall** | **F1-Score** | **Best Hyperparameters** | **Execution Time (s)** | **Remarks** |
| Random Forest | **0.8633** | **0.7768** | **0.6014** | **0.6779** | n\_estimators=100, max\_depth=10 | More than 1.5 hours | Performed well with imbalanced data. |
| Support Vector Machine | **0.8504** | **0.7483** | **0.5646** | **0.6436** | kernel='rbf', C=1.0, gamma=0.1 | More than 2 hours | Slower due to large dataset size. |
| Logistic Regression | **0.8244** | **0.7054** | **0.4568** | **0.554** | penalty='l2', C=0.5, solver='lbfgs' | 20 minutes | Efficient but lower performance. |

**Visualizations**

1. **Confusion Matrices**:
   * Visualized the performance of each algorithm in distinguishing between classes.
2. **Bar Chart**:
   * Compared Accuracy, Precision, Recall, and F1-Score across models.

**4. Analysis**

**Insights**

* **Random Forest** achieved the best overall performance, excelling in handling imbalanced data.
* **SVM** struggled with execution time due to the dataset size but provided competitive metrics with proper tuning.
* **Logistic Regression** was computationally efficient but exhibited lower performance compared to ensemble methods.

**Challenges**

1. **Hyperparameter Conflicts**:
   * Some Logistic Regression configurations caused errors (e.g., unsupported penalties for specific solvers).
2. **Time taking** :
   * Grid Search was time taking as well as random search and of all svm takes too much time.

**5. Submission Details**

**Code :**

<https://github.com/Wajeeha-Aslam/ML_Project>

WAJEEHA ASLAM SP23-BAI-054

MUTAWIFFAH MUDASSAR KHAN SP23-BAI-044