Project Title: Anomaly Detection for Smartphone Dataset using Isolation Forest

- Hyperparameter Tuning for Isolation Forest in Anomaly Detection of Smartphone Dataset

Introduction

The rapid growth of smartphone usage generates vast amounts of data, including app usage, sensor data, and network activity. In this project, the goal is to detect anomalies in smartphone data, which may indicate unusual user behavior, software bugs, or potential security threats.

Anomaly Detection plays a critical role in identifying such abnormal patterns in data. For this task, we are using the **Isolation Forest (IF)** algorithm, which is widely used in unsupervised anomaly detection due to its efficiency in handling high-dimensional datasets and scalability for large datasets like those generated by smartphones.

Isolation Forest Algorithm Overview

Isolation Forest is a tree-based ensemble learning algorithm that isolates anomalies by randomly selecting a feature and then randomly selecting a split value between the minimum and maximum of the selected feature. The rationale is that anomalies are more susceptible to isolation since they are few and different. The algorithm isolates these points more quickly than normal observations.

Key benefits include:

- **Efficiency**: Works well with large datasets due to its sub-sampling technique.
- **Scalability**: Can handle high-dimensional data, common in smartphone datasets.

Hyperparameter Tuning for Isolation Forest

To optimize the performance of Isolation Forest for our specific dataset, several hyperparameters need to be tuned. These hyperparameters directly influence the model's ability to identify anomalies accurately and efficiently.

Key Hyperparameters

1. **n_estimators**:
    - **Description**: Number of trees (base estimators) in the forest.
    - **Typical Range**: 50 to 200.

    - **Impact**: More trees improve the model's robustness but increase computational cost.
2. **max_samples**:
    - **Description**: Number or fraction of samples to be drawn to train each tree.
    - **Typical Range**: 0.1 to 1.0 (fraction) or 128 to 512 (absolute value).
    - **Impact**: Small values lead to high variance; large values improve robustness but increase training time.
3. **contamination**:
    - **Description**: Proportion of the dataset expected to be anomalous.
    - **Typical Range**: 0.01 to 0.1.
    - **Impact**: Helps adjust the threshold for anomaly detection; incorrect values can lead to misclassification of normal points as anomalies and vice versa.
4. **max_features**:
    - **Description**: Number of features used to build each tree.
    - **Typical Range**: 0.5 to 1.0 (fraction of total features).
    - **Impact**: Determines the complexity of the trees; more features can lead to more accurate models but increase computational costs.

Tuning Methods

We use various hyperparameter tuning strategies to find the optimal set of parameters for our smartphone dataset anomaly detection project:

1. **Manual Search**: Initial estimates based on empirical results from literature on Isolation Forest performance in similar anomaly detection tasks.
2. **Random Search**: A more efficient search method that randomly samples a subset of the hyperparameter space. This helps in quickly identifying promising ranges for hyperparameters.
3. **Grid Search**: An exhaustive search across predefined hyperparameter values once the promising hyperparameter range is identified. Though computationally expensive, it ensures we do not miss any optimal configuration.
4. **Bayesian Optimization**: Advanced tuning technique that models the objective function as a probabilistic model, optimizing hyperparameters based on the uncertainty in previous searches.

Evaluation Metrics

The performance of the Isolation Forest in detecting anomalies is evaluated using the following metrics:

1. **Precision**:
    - The proportion of true positives (correctly identified anomalies) to the total number of predicted anomalies.
    - **Importance**: High precision ensures that most flagged points are truly anomalies, which is important in cases where false positives (e.g., marking normal app usage as suspicious) need to be minimized.
2. **Recall**:
    - The proportion of true positives to the total number of actual anomalies.

    - **Importance**: Recall measures the model's ability to identify actual anomalies. This is crucial in scenarios where missing an anomaly (e.g., security threat) is costly.
3. **F1 Score**:
    - The harmonic mean of Precision and Recall.
    - **Importance**: Balances the trade-off between Precision and Recall. A high F1 score indicates the model is both accurate in its anomaly detection and comprehensive in identifying all anomalous events.
4. **ROC AUC (Area Under the Curve)**:
    - Evaluates the model's ability to distinguish between anomalies and normal data points across different thresholds.
    - **Importance**: A good AUC score indicates that the model has a high capability of detecting true anomalies while minimizing false positives.
5. **Execution Time & Memory Usage**:
    - While optimizing for accuracy, it is also important to monitor the computational cost of the model. Faster models with lower memory requirements are preferred for real-time smartphone anomaly detection tasks.

Tuning Process for Smartphone Dataset

1. **Data Preprocessing**:
   The smartphone dataset is cleaned and standardized. Features include app usage patterns, sensor readings (like accelerometer, GPS), and network data. Missing values are handled using interpolation, and outliers are retained as the focus is on anomaly detection.
2. **Initial Validation Setup**:
    - The dataset is split into training and validation sets (80/20 split).
    - A 5-fold cross-validation is employed to ensure the model is robust across different data splits.
    - **Metric**: F1 Score is chosen as the primary evaluation metric, as both false positives and false negatives are important to balance in smartphone anomaly detection.
3. **Random Search & Grid Search**:
    - Initially, Random Search is employed to identify broad hyperparameter ranges. After identifying promising ranges, Grid Search is applied to fine-tune the hyperparameters.
4. **Bayesian Optimization**:
    - Finally, Bayesian Optimization is used for precise hyperparameter tuning. This method is more computationally efficient and provides optimal results based on a probabilistic model of the objective function.

Results

The best hyperparameters found using Random Search and Bayesian Optimization are:

- **n_estimators**: 150

- **max_samples**: 0.8 (80% of the dataset)
- **contamination**: 0.05 (5% of the data is assumed anomalous)
- **max_features**: 0.75

Performance:

- **Precision**: 0.92
- **Recall**: 0.88
- **F1 Score**: 0.90
- **ROC AUC**: 0.94

The results demonstrate that Isolation Forest, when properly tuned, can accurately and efficiently detect anomalies in the smartphone dataset, identifying unusual app usage patterns or device activities.

Conclusion

In this project, Isolation Forest proved to be a highly effective algorithm for detecting anomalies in smartphone datasets. By tuning key hyperparameters such as `n_estimators`, `max_samples`, `contamination`, and `max_features`, we were able to achieve high precision, recall, and F1 scores. The model is now ready for deployment in real-time anomaly detection for smartphone data, offering a robust solution for identifying suspicious or abnormal behavior.

Further improvements could be made by integrating deep learning-based anomaly detection methods or using a hybrid approach combining supervised and unsupervised models to enhance performance in future iterations.