ASSIGNMENT

1. Objective: The provided Python code is tailored for performing Exploratory Data Analysis (EDA) and anomaly detection on time series data. It integrates various data visualization techniques and machine learning algorithms to extract insights from the dataset and pinpoint potential anomalies.

2. Libraries and Packages:

- pandas: Used for data manipulation and analysis, especially for reading CSV files containing time series data and associated labels.
- numpy: Provides support for large, multi-dimensional arrays and mathematical functions for numerical operations and data manipulation.
- matplotlib: A popular data visualization library for creating diverse plots such as time series plots, histograms, and scatter plots.
- seaborn: Built on top of matplotlib, it offers an intuitive interface for statistical graphics, utilized here for correlation heatmaps.
- scikit-learn: A comprehensive machine learning library for classification, regression, clustering, and model evaluation. Specific components include:
  - RandomForestRegressor: Identifies important features contributing to anomalies.
  - permutation_importance: Calculates feature importance.
  - PCA (Principal Component Analysis): Reduces dimensionality for visualization.
  - KMeans: Clusters data points.
  - LocalOutlierFactor: Detects anomalies.

3. Code Structure:

- read_files(test_file, label_file=None): Reads time series and label data from CSV files.
- plot_time_series(test_data, label_data=None, anomaly_indices=None, time_column='timestamp_(min)', downsample_factor=2000): Generates time series plots, highlighting anomalies if labels are available.

- detect_anomalies(test_data, method='local_outlier_factor'): Implements anomaly detection using Local Outlier Factor algorithm.

- perform_eda(test_data): Conducts Exploratory Data Analysis including histograms, correlation heatmap, PCA visualization, and K-Means clustering.

- find_root_cause(test_data, anomaly_indices): Uses RandomForestRegressor to identify key features contributing to anomalies via permutation importance.

- process_file(test_file, label_file): Integrates all components for analysis.

4. Usage: The code is intended to be executed from cells in a Jupyter Notebook or a Python script. The process_file function is invoked with appropriate file names for time series and label data, orchestrating data loading, anomaly detection, visualization, EDA, and root cause analysis.

5. Purpose:

- Primary Purpose: To offer a comprehensive toolkit for analyzing time series data, detecting anomalies, and understanding their root causes.

- EDA Component: Explores data distributions, feature correlations, and potential clusters or patterns.

- Anomaly Detection Component: Identifies deviations from normal behavior, crucial for system monitoring, quality control, or anomaly detection.

- Root Cause Analysis: Identifies key features contributing to anomalies, aiding further investigation or corrective actions.

Conclusion: In summary, the provided code serves as a robust tool for time series data analysis, anomaly detection, and root cause identification. It empowers users to gain deeper insights into their data, facilitating informed decision-making based on analysis results.