Multivariate Time Series Anomaly Detection Pipeline

A complete and modular Python pipeline for detecting anomalies in multivariate time series data. This project was developed as a submission for a hackathon, and it fulfills all specified requirements, including data processing, model training, anomaly scoring, and feature attribution.

Prerequisites

This project requires Python 3.9+ and the following libraries:

- pandas
- numpy
- scikit-learn

These can be installed using the provided requirements.txt file.

Installation

To set up the project environment, run the following command:

pip install -r requirements.txt

Usage

The script can be executed from the command line. It requires an input CSV file and specifies an output path for the annotated data.

Command Line Arguments

- --input_csv (required): Path to the input CSV file.
- --output_csv (required): Path to the output CSV file.
- --timestamp_col (optional): Name of the timestamp column. The script will attempt to infer this if not provided.
- --train_start, --train_end, --analysis_start, --analysis_end: Timestamps defining the training and analysis windows. Default values are set to match the hackathon's problem statement.

Example Execution

Use the following command to run the script (paste the following command as a single line):

python mycode.py

--input_csv "data.csv"

--output_csv "data_with_scores.csv"

--train_start "2004-01-01 00:00"

--train_end "2004-01-05 23:59"

--analysis_start "2004-01-01 00:00"

--analysis_end "2004-01-19 07:59"

```
PS D:\anamoly_pipeline> python3 mycode.py   --input_csv data.csv   --output_csv data_with_scores.csv   --timestamp_col Time   --train_start "2004-01-01 00:00"   --train_end   "2004-01-10 23:59"
 --analysis_start "2004-01-01 00:00"   --analysis_end   "2004-01-19 07:59"
✅ Completed. Wrote annotated CSV with 8 new columns to: data_with_scores.csv
   - Rows in analysis window: 26400
   - PCA components kept: 22
   - Numeric feature count: 52
PS D:\anamoly_pipeline>
```

Solution Overview

The pipeline is designed to be modular and robust, following three main steps:

1. Data Processing

The script first loads the raw data, identifies and parses the timestamp column, and handles common data quality issues. It uses interpolation and forward/backward filling to manage missing values and removes any constant (zero-variance) features to prevent model errors.

2. Anomaly Detection Model

A **PCA (Principal Component Analysis)** model is used for anomaly detection. The model is trained on the "normal" data period to learn the primary correlations and patterns. Anomalies are then detected as data points with high **reconstruction error**, meaning the model cannot accurately represent them using the learned "normal" patterns.

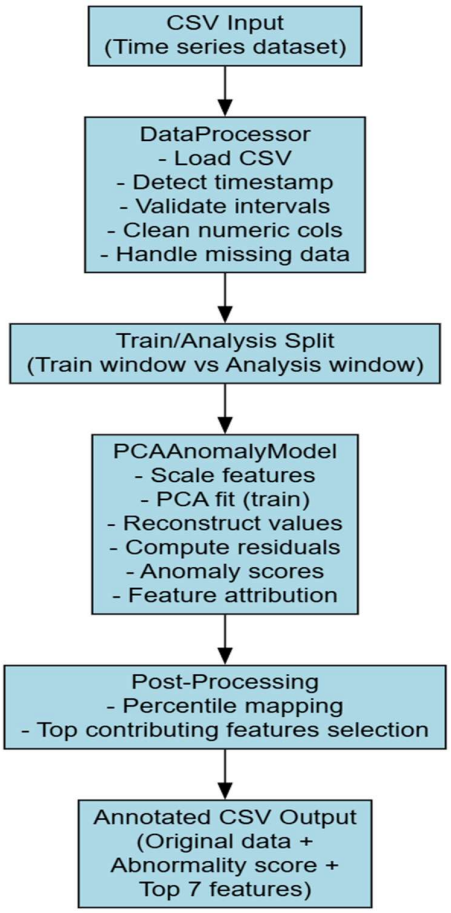
3. Scoring and Attribution

The final output is generated with two key additions:

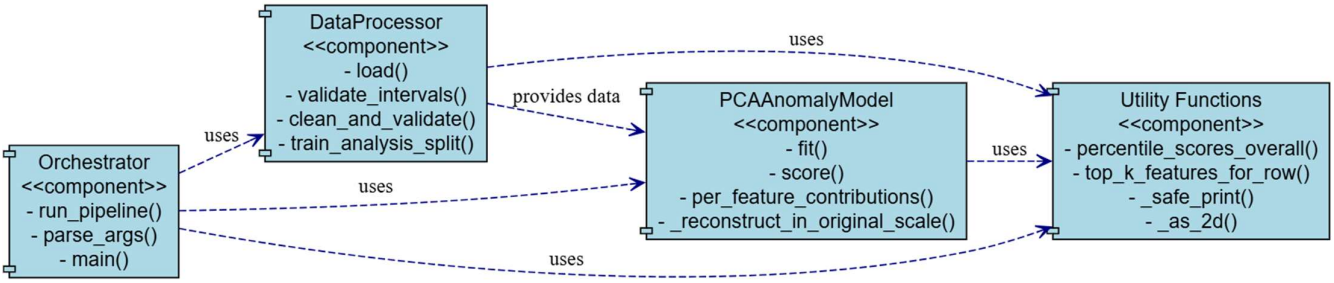
- **Abnormality Score**: The raw reconstruction error is scaled to a **0-100 score** using percentiles calculated across the entire analysis dataset. A higher score indicates a more severe anomaly.
- **Feature Attribution**: The model attributes the anomaly to the **top 7 most contributing features** by measuring the individual squared reconstruction error for each feature. This provides valuable insights into the root cause of the detected abnormality.

High-Level System Architecture

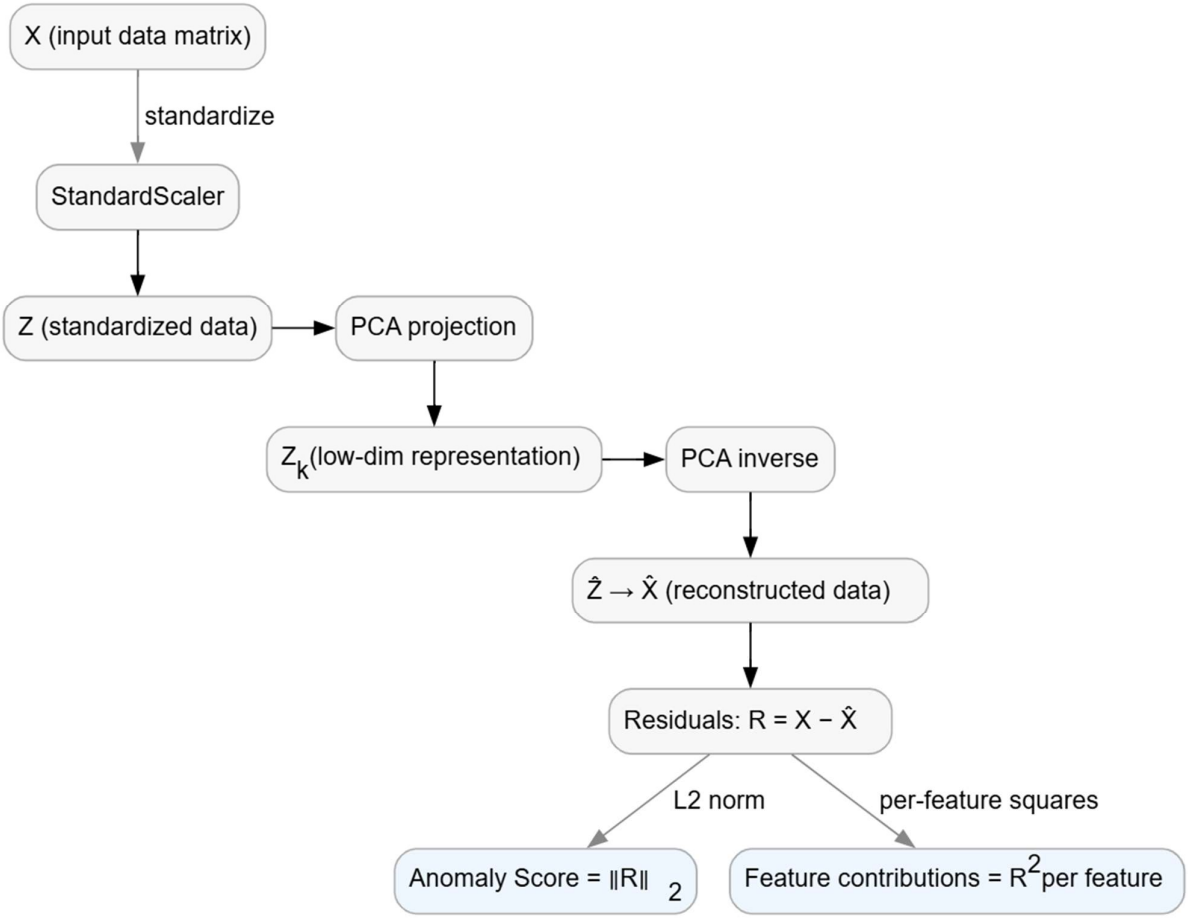
This shows the **overall flow of data** from input to output.



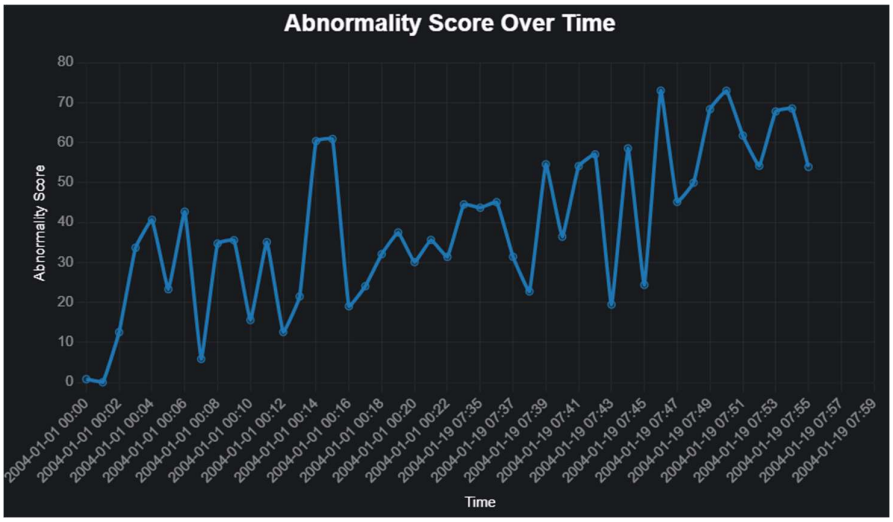
Component Diagram:



Mathematical view of PCA Model:



Time-Series Plot of Abnormality Score



Link to project folder:

https://drive.google.com/drive/folders/1JOUmcJSQhZar1mzuVlSfk7LYscVV6f8N?usp=sharing

Output:

Due to the large size of the processed results file, it has been hosted externally.

Please use the link below to access and download the output dataset:

https://drive.google.com/file/d/1ceTH9AcsaNTS1Pj_PQ8YQqQbQJ9ViNf1/view?usp=sharing