readme.md

Efficient Data Stream Anomaly Detection

Overview

The goal of this project is to develop a Python script that continuously monitors a simulated data stream and detects anomalies in real time.

Features

- **Data Stream Generation:** The `generate_synthetic_data_stream` function simulates a continuous stream of floating point numbers. It adds a sine wave to represent seasonal variations and random noise to simulate real world fluctuations. The `time.sleep(0.1)` introduces a delay to simulate real-time streaming.
- **Streaming Isolation Forest:** I created a class `StreamingIsolationForest` that handles the real-time anomaly detection using an Isolation Forest. This class:
  - Keeps a sliding window of recent data points (data_window) with a fixed size.
  - Uses the Isolation Forest to fit the current window of data and predict if the new incoming point is an anomaly.
  - The fit_predict method updates the window with the latest data point, trains the model on the window, and makes a prediction (1 for anomaly, -1 for normal).
- **Optimization:** To handle large data streams efficiently, we maintain a `sliding window` of recent data (size = 10 in this case). This limits memory usage and ensures the Isolation Forest model is trained only on the latest data, avoiding the need to process the entire history.
- **Real Time Visualization:** The visualization uses `Matplotlib` with a line plot for the data stream and scatter plot for anomalies.

Prerequisites

Before you begin, ensure you have met the following requirements:

- [ ] You must be running the code in an Unix based OS.

- [ ] You have downloaded and setup Python version >= 3.10.

Run the Code

To Run this project, follow these steps:

```
#Set up Virtual Environment
python3 -m venv env

#Activate the Environment
source env/bin/activate

# Install dependencies
pip install -r requirements.txt

#Run the model - Wait few seconds for the visualization to kick in
python3 main.py
```