Real Time Monitoring of Water Distribution System

DA331 Big Data Analytics: Course Project by
*Ishan Gupta

I. Introduction

Water is an essential resource in every society, and although renewable there is still a huge problem of water scarcity in various parts of the world. Moreover, the effect of availability for fresh water resources are increasingly felt as a result of competing demands, due by climatic change and environmental pollution.

The primary objective of this project is to develop a machine learning model to monitor water distribution by analyzing consumption patterns. Through identifying abnormal or suspicious consumption behaviors, this system aims to achieve the following:

1) **Classify water consumption readings as normal or anomalous.**
   The model will analyze water usage data and classify consumption patterns, distinguishing between typical (normal) and atypical (anomalous) readings.
2) **Detect patterns associated with water leakages or unusual activities.**
   By recognizing consumption behaviors indicative of leakages or altered usage, the system can identify potential instances of water wastage.

II. Dataset Generation

The water consumption dataset used for this project contains several key features that assist in detecting anomalous behaviors associated with water leakages. The primary features include:

- **Sensor ID**: A unique identifier for each sensor, used to track individual consumption patterns from each house.
- **Timestamp**: The date and time of each reading, enabling temporal analysis of consumption trends.
- **Flow Rate**: The recorded water flow rate for each timestamp. This value is subject to fluctuations based on external factors such as leakages.
- **Pressure**: The recorded incoming pressure of water at every house to monitor any anomalies.
- **Turbidity**: Turbidity refers to the cloudiness of water. Similar to pressure, turbidity can also give us various about the distribution system.
- **Temperature (°C)**: External temperature in a water distribution system might fluctuate due to changes in pressure and flow rate.
- **pH**: Ideally, the pH of water should be close to neutral (pH = 7). Abnormal pH can indicate the extent of water pollution due to leakages.

In this section, we generate synthetic data to simulate real-world events. This data will be sent to a Kafka topic, allowing us to test the Kafka producer and consumer functionality. By generating data with random values, timestamps, or other features, we can simulate scenarios like sensor readings, financial transactions, or user activities.

Algorithm 1 Generate Water Distribution Data

Require: (Number of Sensors), (Number of readings per sensor) , (Start Time)

- **Flow Rate**: We have chosen Normal Distribution because flow rate typically fluctuates around an average value, with occasional spikes or dips. The normal distribution captures this behavior with a bell-shaped curve where most values fall near the mean and fewer occur at the extremes.
- **Pressure**: Pressure in a water distribution system is unlikely to be negative, but it can vary significantly. The log-normal distribution is skewed towards positive values and has a long tail, making it suitable for modeling pressure readings where most values are positive and some can be much higher.
- **Temperature**: Similar to flow rate, temperature in a water distribution system might fluctuate around an average value with occasional deviations
- **pH**: Ideally, the pH of water should be close to neutral (pH = 7). The normal distribution with a small standard deviation (sigma) allows for slight variations around this ideal value while keeping most readings within a narrow acceptable range.
- **Turbidity**: turbidity readings are unlikely to be negative and can vary considerably. The log-normal distribution with a low mean reflects this, with most values being low (clear water) and some potential for higher turbidity levels.

This data structure will be published to Kafka as individual JSON messages.

III. Data Visualization

In this section, we present various visualizations to better understand the water distribution network, particularly focusing on both normal and anomalous data points. We will use Dashboards for Data visualizations. Dashboards are powerful tools in data visualization, offering a consolidated view of data for monitoring, analysis, and decision-making.

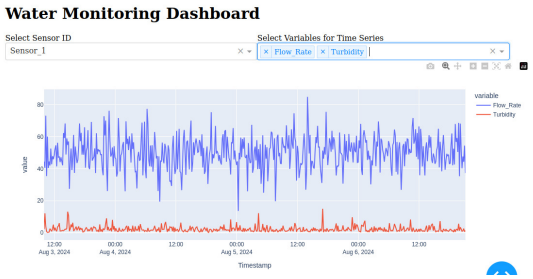

Fig. 1. You can plot time series of different features over a specified period for different sensors. This plot illustrates the trends in consumption and highlights periods of anomalous behavior.

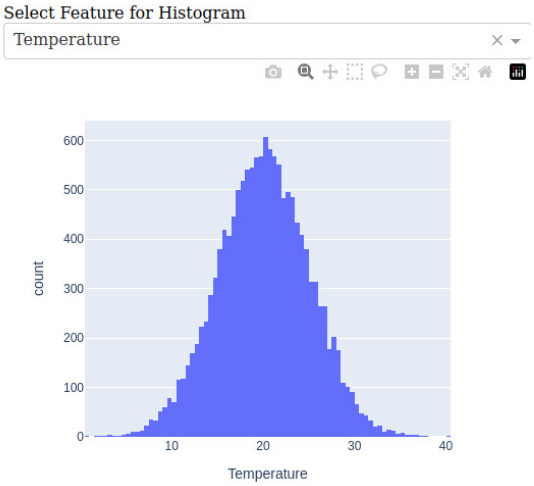

Fig. 2. Histograms help reveal the distribution shape of different features, whether it's normal, skewed, bimodal, or has any other pattern. This insight is essential for deciding on further statistical analyses.

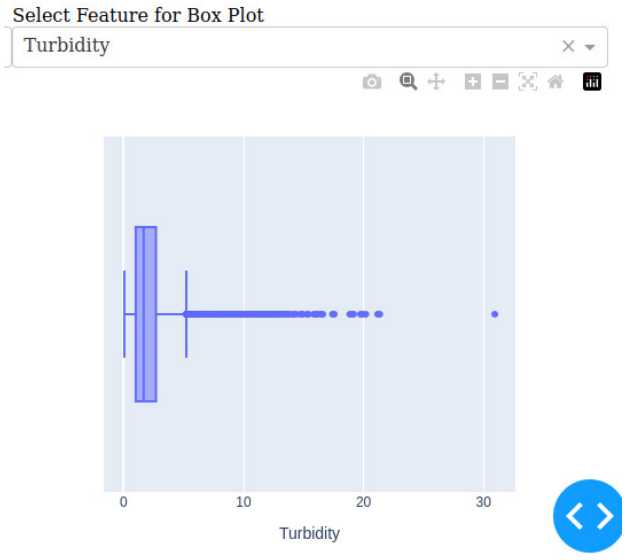

Fig. 3. Box plot of turbidity showing the distribution of normal and anomalous data. The anomalies are marked separately, providing insights into the variance and outliers in the consumption patterns.

A. *Insights from Visualizations*

From the box plot (Figure 3), we can observe the distribution of consumption values. The presence of outliers indicates periods of abnormal behavior that may warrant further investigation. The time series plot (Figure 1) provides insights into how feature characteristics changes over time, with clear peaks during specific hours. Finally, the histogram (Figure 2) helps identify potential relationships among different variables, guiding feature selection for predictive modeling.

IV. Machine Learning Models for Anomaly Detection

Several machine learning models were evaluated for detecting anomalous water consumption patterns, each with different strengths in handling this type of classification task. The following models were tested:

A. *Anomaly Detection Models*

1) Isolation Forest: Label Most Data as Normal: Assign a label (e.g., label = 0) to the majority of the data, assuming most of it is normal. Randomly Sample and Label Anomalies: Randomly select a small subset of the data and label them as 1 (anomalies). The exact proportion can be adjusted based on your expectations for anomaly prevalence.

`Accuracy: 0.8825`

While the Isolation Forest model achieved an accuracy of 88.25%, it showed slightly lower overall performance in comparison to other models.

2) Clustering Using K-means: Cluster the data into K clusters using KMeans. Calculate distance of each point from the nearest cluster center. Flag outliers as those points whose distance from the centroid is greater than a specified threshold.

`Accuracy: 0.8625`

Selecting K and threshold is another problem with this approach and is highly domain-dependent.

3) Mahalanobis Distance for Multivariate Outlier Detection: Using Mahalanobis distance on multiple features can identify anomalies, especially when features are correlated. It measures the distance from the center (mean vector) scaled by the covariance.

Compute the mean vector and covariance matrix of the features. Calculate Mahalanobis distance** for each point. Set a threshold (such as the Chi-square critical value) to label points as anomalies.

`Accuracy: 0.9115`

This approach can be useful when features are dependent and single feature might not be enough to detect an anamoly. In conclusion, this study explored various machine learning models for real-time water monitoring system. Clustering using Mahalanobis distance showed the best performance, making it the model of choice.

The implementation of a Kafka-based streaming solution provided real-time monitoring, and further work will focus on improving latency and extending the system to larger datasets.

V. Real-Time Data Streaming and Classification with Apache Kafka

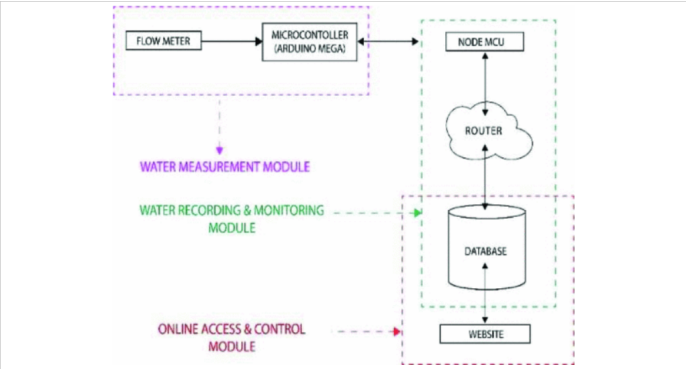

Fig. 4. These modules include the water measurement module, water recording and monitoring module, online access and control module.

For our real-time water distribution monitoring system,Apache Kafka was configured to handle continuous streaming data from multiple data sources. The setup includes Kafka topics, producers, and consumers designed to capture, process, and classify data with minimal latency. This section provides an overview of the setup, latency considerations, and a visual monitoring system implemented for live anomaly tracking. Plotly was used for dashboard and monitoring.

A. Setup and Configuration

- **Kafka Architecture**: Kafka's architecture was established with distinct topics representing data streams from various sensors across different locations. Producers continuously send data to these topics, simulating energy consumption and other relevant metrics.
- **Consumers and Real-Time Processing**: A consumer subscribes to Kafka topics and feeds the incoming data into a deployed anomaly detection model, which classifies each entry as normal or anomalous based on consumption patterns. The model was hosted to work in a low-latency, high-throughput environment to handle real-time demands.

B. Real-Time Classification Workflow

Once data is published to Kafka, it goes through the following steps:

1) **Data Aggregation**: Kafka was tuned to simulate each message with a typical latency of **200 ms**. Data from multiple sensors was send to a centralized node which we used to do aggregate for classification. The idea was to see whether the input was equivalent to output or not. The ideal working of our system can be understood from (Figure 4)
2) **Classification**: The data is then passed to the anomaly detection model for classification. The model processes each input with an inference latency of approximately **135 ms**.
3) **Output Publishing**: The result, a label indicating normal or anomalous consumption, is published back to Kafka for further monitoring or storage.

C. Live Anomaly Tracking

To visually monitor the anomalies detected in real-time, a dashboard was created to plot the incoming data. Anomalies are marked as high spikes or low spikes in consumption patterns, as illustrated in Figure 5.

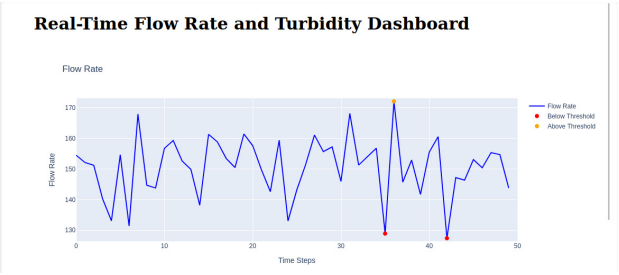

Fig. 5. Flow rate Monitoring. Min and Max threshold is selected based on domain knowledge.

Fig. 6. Monitoring of Turbidity based on incoming inputs.

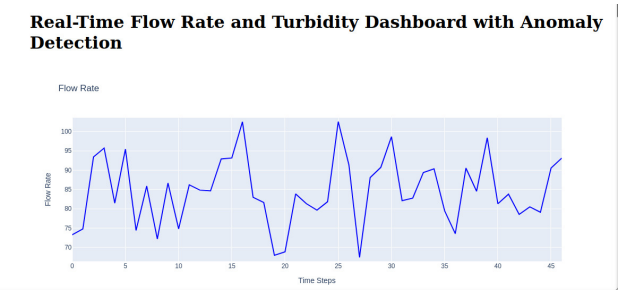

Fig. 7. Live Anomaly Tracking: Based on all features using Mahalanobis distance.

VI. References

1) Apache Kafka Official Documentation. Available at: https://kafka.apache.org/documentation/
2) Confluent Kafka Documentation. Available at: https://docs.confluent.io/platform/current/overview.html
3) Neha Narkhede, Gwen Shapira, and Todd Palino, *Kafka: The Definitive Guide*. O'Reilly Media, 2017.
4) Jay Kreps, *Kafka: The Definitive Guide* (2nd Edition). O'Reilly Media, 2021.
5) Synthetic Data Generation with Python. Available at: https://towardsdatascience.com/synthetic-data-generation-using-python-5c9c2c63d6cd
6) How to Generate Synthetic Data for Machine Learning. Available at: https://www.kdnuggets.com/2021/09/generate-synthetic-data-machine-learning.html
7) G. M. Diamos et al., *Synthetic Data Generation: A Survey*. arXiv:2007.03377.
8) Chandola, V., Banerjee, A., & Kumar, V. (2009). "Anomaly Detection: A Survey." *ACM Computing Surveys (CSUR)*, 41(3), 1-58. Available at: https://doi.org/10.1145/1541880.1541882
9) Iglewicz, B., & Hoaglin, D. C. (1993). *How to Detect and Handle Outliers*. SAGE Publications.
10) Anomaly Detection Techniques in Machine Learning with Python. Available at: https://www.datacamp.com/community/tutorials/anomaly-detection-machine-learning-python
11) Hodge, V. J., & Austin, J. (2004). "A Survey of Outlier Detection Methodologies." *Artificial Intelligence Review*, 22(2), 85-126.
12) Ahmed, M., Mahmood, A. N., & Hu, J. (2016). "A survey of network anomaly detection techniques." *Journal of Network and Computer Applications*, 60, 19-31. Available at: https://doi.org/10.1016/j.jnca.2015.11.016